Technical Integration - Front page

Digital Post

Exported on 2023-11-14 14:21:25

Table of Contents

1 Versioning 8

2 Introduction and overview 27

2.1 Roadmap for new functionality 27

2.2 Status on interfaces 27

2.3 Background 28

2.4 Memo 29

2.4.1 Information 29

2.4.2 MeMo library 29

2.5 Size requirements for messages and attachements 29

2.6 Target group 30

2.6.1 For test authorities 30

2.6.2 For businesses 30

2.7 List of terms 30

2.8 Minimum requirements for authorities 33

2.8.1 Sender systems 33

2.8.2 Recipient systems 34

2.9 More information 34

2.10 Overview of DP 34

2.10.1 At least once-principle 35

2.10.2 Mutual SSL authentication using API key 35

2.10.3 Open API description 40

2.10.4 REST Implementation 41

3 Querying and searching 48

3.1 Domain names 48

3.1.1 QA environment 48

3.1.2 Test environment 48

3.1.3 Production 49

3.2 Outgoing IP 49

3.3 Querying and searching resources 49

3.3.1 Eventually consistent 49

3.3.2 SearchResult 50

3.3.3 Paging 50

3.3.4 Sorting 51

3.3.5 Filtering 52

3.3.6 Searching 53

4 Contact registry services - TI 56

4.1 Contact registry data model 58

4.1.1 Physical data model for the contact registry 58

4.1.2 Querying Contacts 64

4.2 Searching 64

4.2.1 Examples 64

4.3 Using the body of the GET request 65

4.3.1 Subscribing to NemSMS 66

4.3.2 Querying the contact 67

4.3.3 Fetching the contact 67

4.3.4 Updating the contact 68

4.3.5 Contact Subscription 70

4.3.6 Using the contact subscriptions 71

4.3.7 Contact Subscription 71

4.3.8 Endpoint exposed in the contact-subscription-store 72

4.3.9 Notifications 73

5 System registry services - TI 74

5.1 Organisation 74

5.2 Systems 75

5.3 Contact structure 79

5.4 System registry data model 83

5.4.1 Organisation Model 83

5.4.2 System Model 86

5.4.3 Contact Structure Model 92

5.5 Querying in System registry APIs 96

5.5.1 Fetching organisations on ID 97

5.5.2 Searching for ContactPoints across all organisations or within an Organisation 97

5.5.3 Searching for Systems within an organisation 99

5.5.4 Searching for Contact Groups across all organisations or within an organisation 99

5.6 Fetching hidden contact points and contact groups 100

5.7 Updating items in system registry 101

5.8 Exporting certificate for upload to Administrative Access 104

5.9 NAME\_CHAINING -error code 105

5.9.1 Subscribing to changes in the system registry: 105

5.9.2 Organisation contact-group/point Subscription 107

5.9.3 Endpoint exposed in the system-subscription-store 108

5.9.4 Notifications 109

6 Mailbox services - TI 110

6.1 Mailbox 110

6.2 Accesses 111

6.3 Messages 112

6.4 Documents 115

6.5 Files 116

6.6 Folders 118

6.7 System fetches 119

6.8 Mailbox persistence entity model 120

6.8.1 Mailbox model 120

6.8.2 Access model 122

6.8.3 Message model 126

6.8.4 Folder model 139

6.8.5 SystemFetch model 140

6.9 Querying for Messages 142

6.9.1 Searching 143

6.10 Common use case examples 143

6.10.1 Update access 143

6.10.2 Create folder 144

6.10.3 Create draft message 145

6.10.4 Update draft message 146

6.10.5 Upload content to a file resource 149

6.10.6 Send message 151

6.10.7 Reply to message 153

6.10.8 Update one or more of certain predetermined fields of message (PATCH) 156

6.10.9 Move message between folders 157

6.10.10 Forward message to e-mail address 158

6.10.11 Forward message to trusted recipient and authority 161

6.10.12 ReplyData mail threads 164

6.10.13 Write to the authorities 164

6.10.14 Examples of error messages 165

6.11 Uploading invalid html to a file 165

6.12 Trying to send a message without content in the main document 167

6.13 Replying to an unreplyable message (reply = false) 167

6.13.1 Message state action matrix 167

7 Event log services - TI 169

7.1 Querying the event-log 170

7.1.1 Querying 170

7.2 Event Log Index Events 173

7.2.1 Fields of interest on events grouped on subject. 176

8 Push notification integration - TI 183

8.1 Some general info about push notifications via DP 183

8.2 Registering as a push notification tenant (aka “I want to send push notifications”) 184

8.3 Creating FCM service account + private key 185

8.4 Interaction with “push notifications” 188

8.5 Subscription to push notification 192

8.5.1 Example 193

8.6 Visual guide for push notification flows 195

9 Identity registry services - TI 196

9.1 IDENTITIES 196

9.2 DIRECT PRIVILEGES 197

9.3 GRANTEE 198

9.4 IDENTITY GROUP 198

9.5 PRIVILEGE TYPE 199

9.6 Querying Identities, Direct privileges, Privilege Type 200

9.7 IDENTITIES 200

9.7.1 Fetching identities on ID 200

9.8 DIRECT PRIVILEGES 201

9.8.1 Fetching Direct privileges on ID 201

9.9 PRIVILEGE TYPE 201

9.10 Direct privilege 201

9.11 Creating Direct privilege 202

9.12 Privilege group 202

9.13 Querying the Privilege group 203

9.13.1 Fetching 203

9.13.2 Searching 204

9.13.3 Creating 205

9.13.4 Updating 205

9.13.5 Adding Privilege 206

9.13.6 Updating Privilege 207

9.13.7 Adding Grantee 208

9.13.8 Updating Grantee 208

10 Distribution - TI 210

10.1 Distribution Services 210

10.1.1 Legacy services: Sending DP/DP2 messages via REST 210

10.1.2 Asynchronous endpoint 214

10.1.3 Legacy services: Sending DP/DP2 messages via SFTP 218

10.1.4 Legacy services: Sending DP/DP2 messages via SMTP 223

10.1.5 Legacy error codes 230

10.1.6 Fetching registration status for contact 232

10.1.7 Fetching registration status list 233

10.1.8 Fetching contact registration status part list 233

10.1.9 Bulk receipt list (massekvitteringsliste) 235

10.1.10 Fetch bulk receipt list 235

10.1.11 Delete bulk receipt list 236

10.1.12 Handling of reply destination in Digital Post contact structure when sending DP/ DP2 237

10.1.13 Contact-registry in Record format 239

10.1.14 Contact in legacy csv 241

10.1.15 Distribution REST services 242

10.2 Inbound services 242

10.2.1 Send MeMo messages 244

10.3 Outbound services 245

10.3.1 Sender system respond to received Business Receipt 246

10.4 Inbound services 249

10.4.1 Send message to Digital Post 249

10.5 Outbound services 249

10.5.1 Send MeMos to SMTP Recipient Systems 249

10.5.2 Send technical receipts to senders 250

10.5.3 Digital Post Receipt domain model 252

10.5.4 REST receipt procedure 253

10.5.5 SMTP receipt procedure 254

10.5.6 SFTP receipt procedure 255

10.5.7 JSON examples 256

10.5.8 Bulk MeMo SFTP Service 257

10.5.9 Distribution use case examples 262

10.5.10 Fetching business receipts 264

10.5.11 Fetching receipts for a REST\_PULL sender system 264

10.5.12 Bulk-fetching receipts 265

10.5.13 Sending business receipts as a recipient system 266

10.6 Flow for resending messages 266

10.7 Flow for resending business receipts - REST Push protocol 268

10.8 HTML whitelist for document validation 268

10.8.1 Introduction 268

10.8.2 Strict 269

10.8.3 Lenient 277

10.8.4 Testing 279

11 Access request registry 280

11.1 Access request registry - introduction 282

11.2 Purpose of the registry 282

11.3 Privilege requests 283

11.3.1 Journalized: False 283

11.4 Delegation requests 283

11.4.1 Journalized: False 284

11.5 Appointed delegation requests 284

11.5.1 Journalized: False 284

11.6 Connection agreement requests 284

11.6.1 Journalized: True 284

11.7 Terms approval requests 284

11.7.1 Journalized: False 284

11.8 User administrator’s statement of truth privilege requests 284

11.8.1 Journalized: True 284

11.9 Lost user administrator privilege requests 284

11.9.1 Journalized: True 285

11.10 Special privilege requests 285

11.10.1 Journalized: True 285

11.11 Delegated support admin privilege request 285

11.11.1 Journalized: False 285

11.12 Concepts 285

11.13 Access request registry - common use case examples 286

11.14 Introduction 286

11.15 User administrator delegates privilege to employee identified with an e-mail address 287

11.15.1 Request 287

11.15.2 Response 287

11.16 Usage of generic identity id in access requests 288

11.16.1 Request 290

11.16.2 Response 290

11.16.3 Attaching documents to request 291

11.17 1. Add documentation element 292

11.17.1 Request 292

11.17.2 Response 293

11.18 2. Upload byte content to documentation 295

11.18.1 Request 295

11.18.2 Response 296

11.18.3 Citizen requesting access to other citizens mailbox 298

11.19 1. Citizen A submits request 298

11.19.1 Request 298

11.19.2 Response 299

11.20 2. Citizen B queries to see incoming requests 301

11.20.1 Request 301

11.20.2 Response 302

11.21 3. Citizen B approves request 304

11.21.1 Request 304

11.21.2 Response 305

12 Sender-/Receiver Systems 307

12.1 Rate-limiting 307

12.2 Patterns for integration to Digital Post 307

12.3 Sender system 308

12.3.1 REST Protocol 308

12.3.2 SFTP Protocol 308

12.3.3 SMTP/IMAP Protocol 309

12.4 Receiver system 310

12.4.1 REST Protocol 310

12.4.2 SMTP/IMAP Protocol 310

12.4.3 Sending Memo 311

12.4.4 Receiving Memo 313

13 Encoding formats, Environments and Error codes 316

13.1 Encoding format whitelist for files of documents 316

13.1.1 Main document 316

13.1.2 Additional documents 316

13.1.3 Technical documents 317

13.2 Access to environments 317

13.2.1 Making requests - important! 317

13.2.2 TEST environment 317

13.2.3 PROD Environment 317

13.2.4 Guidelines for downtime 318

13.3 Error codes 318

13.3.1 Front-end validation and error codes in the Viewclient 319

13.3.2 Back-end validation and error codes in distribution 334

13.3.3 Business receipt error codes 334

13.3.4 Technical receipt error codes 337

13.3.5 Tracing requests using W3C headers 338

14 Java/.Net Core, Security perspective, MeMo-lib and Test 340

14.1 Reference Systems for Java and .Net Core 340

14.1.1 Overall description and purpose 340

14.1.2 Supported platforms 340

14.1.3 Application architecture 340

14.1.4 Resources and webinars 341

14.1.5 REST protocol examples 341

14.1.6 SMTP protocol examples 346

14.1.7 SFTP protocol examples 351

14.2 Security Perspective 352

14.2.1 Authentication 352

14.2.2 Allowed certificate cipher suites 353

14.3 MeMo-lib 353

14.3.1 Purpose 353

14.3.2 Repository 353

15 Access to Test environments 354

15.1 Test 354

15.1.1 Prerequisites 354

15.1.2 Acquisition of an OCES test certificate 354

15.2 Access to the administration portal on the test environment 354

15.2.1 Step 1 - Create a MitID simulator identity 354

15.2.2 Step 2 - Get the new identity enrolled in DP test environment 354

15.2.3 Step 3 - Test login for a test user 356

15.2.4 Links 357

15.2.5 Points of attention for suppliers 357

15.3 Access to the Test Portal on the test environment 357

15.4 How to login to Administrative Access with test-identities 358

16 Troubleshooting, SFTP server, SDLC, OpenID Connect, Connect 360

16.1 Troubleshooting 360

16.1.1 Certificate policies 360

16.1.2 Allowed ciphers 374

16.1.3 Rating 374

16.1.4 Troubleshooting 375

16.1.5 Ensuring the authenticity of Digital Posts through the OCES certificate 375

16.1.6 Point of attention regarding certificates 376

16.1.7 DP Service Desk 379

16.2 SFTP server 379

16.2.1 Generate an SSH-key to DP 379

16.2.2 Access the SFTP server 383

16.3 Software Development Life Cycle (SDLC) for the API 384

16.3.1 Overall strategy 384

16.3.2 Versioning semantics (change category) 384

16.4 OIO OpenID Connect to Digital Post 385

16.4.1 OIO OIDC profile - Digital Post subset 385

16.4.2 OIDC client enrollment in Digital Post 387

16.4.3 RP-initiated Logout 388

16.4.4 NemLog-in Realms 388

16.4.5 MitID AppSwitch support 389

16.4.6 eIDAS eID Gateway 389

16.5 Connect to Digital Post: Test and Prod 389

16.5.1 Roles in the administration portal “Administrativ Adgang”. 391

16.6 Additional configuration support 391

# Versioning

|  |  |
| --- | --- |
| **Version:** | 1.41 |
| **Status:** | Final |
| **Author:** | Netcompany |

**Document history**

|  |  |  |
| --- | --- | --- |
| **Version** | **Date** | **Comments** |
| 0.1 | 30-05-2020 |  |
| 0.11 | 03-06-2020 | Added additional examples to “Distribution Services”. |
| 0.12 | 11-06-2020 | Added ‘Allowed certificate cipher suites’.  Added ‘Roles in the solution’  Added ‘Bulk MeMo SFTP service’ |
| 0.13 | 26-06-2020 | Added the section ‘More information’  Added additional terms in the section ‘List of Terms’  Updated the following sections:   * Services exposed from Digital Post * Domain names, components and services * Contact registry services * Querying Contacts * System registry services * Querying in System registry APIs * Mailbox services * Querying for Messages * Event log services * Distribution REST services * Receipt domain model * Bulk Memo SFTP service * Verification Registry Services * Subscription services   Added sections:   * Errorcodes * Test * Creating and updating items in system registry |
| 0.9 | 03-07-2020 | Added the following sections:   * ‘Service Desk’. * ‘Updating items in the contact registry’ * ‘Updating items in system registry’ * 'Update mailbox example' * 'Create Folder example'   Updated the following sections:   * Services exposed from Digital Post with a change log function. * Domain names, components and services * Mailbox services * Sender- and receiver systems * Distribution REST services * Distribution SMTP services * SMTP MeMo example * Receipt domain model * Businessreceipt example (REST) * Pull notification domain model * Bulk Memo SFTP service * Verification Registry Services * Access to environments * Subscription services * Error codes |
| 1.0 | 05-08-2020 | Removed internal links.  Removed To do comments.  Added the following sections:   * Memo * Common use case examples * domain models for contact and system registry * Sending and receiving memo * Distribution use case examples   Updated the following sections:   * Minimum requirements. * Mailbox services * Receipt domain model * Error codes |
| 1.01 | 24-08-2020 | General   * Removed section Domain names, components and services   Added the following sections:   * Create folder * Reply to message * Update only flags of message * Tracing requests using Zipkin   Updated the following sections:   * Error codes for front-end * Services exposed from Digital Post * Domain names, components and services * Contact registry services * System registry services * Querying in System registry APIs * Mailbox services * Mailbox domain model * Querying for Messages * Update mailbox * Create folder * Create draft message * Forward message to e-mail address * Forward message to trusted recipient and authority * ReplyData mail threads * Event log services * Querying the event-log * Distribution REST services * Receipt domain model * Back-end validation and errorcodes in distribution |
| 1.02 | 02-09-2020 | Updated the following sections:   * Receiver systems * Prerequisites |
| 1.03 | 18-09-2020 | Added the following sections:   * Access to the administration portal on the test environment   Updated the following sections:   * Bulk Memo SFTP service * Encoding format whitelist for files of documents * Back-end validation and errorcodes in distribution * Domain names, components and services * Access to environments * Back-end validation and errorcodes in distribution |
| 1.04 | 01-10-2020 | Updated the following sections:   * Distribution SMTP services * Receipt domain model * Services exposed from Digital Post * Querying for Messages * Event reporting * SMTP MeMo example * Businessreceipt example (REST) * REST PULL notification example * Bulk MeMo SFTP service * Fetching a single MeMo over REST PULL * Sending MeMo * Receiving MeMo * Recipient-system error codes |
| 1.05 | 14-10-2020 | Updated the following sections:   * Recipient-system error codes * MeMo * Querying the event-log * Updating items in system registry * Update only flags of massage (PATCH) * Receipt domain model * Business receipt example (REST) * Access Agreement |
| 1.06 | 29-10-2020 | Updated the following sections:   * Introduction * Repository   + Acces to the administration portal on the test environment * Services exposed from Digital Post * DMZ API Gateway * Mailbox services * Update one or more of certain predeterminded fields of message (PATCH) * Bulk MeMo SFTP service * Recipient-system error codes * SMS Gateaway |
| 1.07 | 13-11-2020 | Updated the following sections:   * Contact registry services * Querying Contacts * Updating items in the contact registry * Digital Post- and NemSMS Terms * System registry services * Querying in System registry APIs * Exporting certificate for upload to AA * Mailbox domain model * Receipt domain model * Distribution REST services * backBusiness receipt example (REST) * Bulk Memo SFTP service * Sending memo message over REST PUSH * Recipient-system error codes * Printservice * List of terms   + Added a description of Service Level Agreement(SLA) + Time To Responed * Corrected SLA IDs M3 and M4 to MA3 and MA4 in table: Distribution REST services * Access to the administration portal on the test environment |
| 1.08 | 27-11-2020 | Updated the following sections:   * Roles in the solution * Integration components overview * Mailbox domain model * Querying the event-log * SMTP MeMo example * SMS Gateway * Access to the administration portal on the test environment   + Troubleshooting   + Points of attention regarding certificates |
| 1.09 | 09-12-2020 | Updated the following sections:   * List of terms: UUID definition * Exporting certificate for upload to AA * Roles in the solution * Integration components overview * Port numbers for local development * Generate an SSH-key for DP * Legacy services * SMS gateway * Access to the administration portal on the test environment   + Points of attention for suppliers |
| 1.10 | 22-12-2020 | Updated the following sections:   * Introduction * Roles in the Solution * Integration components overview * Contact registry services * System registry services * Updating items in system registry * Mailbox services * Mailbox domain model * Event log services * Distribution REST services * Receipt domain model * SMS gateway   Added the following section:   * Access to the Test Portal on the test environment * Authentication error “Der kan ikke ” |
| 1.11 | 12-01-2021 | Updated the following sections:   * Roles in the solution * Domain names, components and services * Mailbox services   + Mailbox domain model   + Forward message to e-mail address |
| 1.12 | 01-02-2021 | Added the following sections:   * Overall status for implementation * Stay updated * Check validity of a OCES certificate for test and production * Web certificates policy * Guidelines for downtime * Legacy Services   + Sending DP/DP2 messages via REST   + Sending DP/DP2 messages via SFTP   + Sending DP/DP2 messages via SMTP   Updated the following sections:   * Querying Contacts * Querying in System registry APIs * Updating items in system registry * Mailbox services   + Mailbox domain model   + Update mailbox   + Create draft message * Distribution Services   + Outbound MeMo REST Push example request   + Distribution SMTP services * REST PUBLISH SUBSCRIBE notification example * Bulk Memo SFTP service * Patterns for integration to Digital Post   + Receiving MeMo * Encoding format whitelist for files of documents * Error codes   + Recipient-system error codes |
| 1.13 | 24.02.2021 | Added the following section:   * Test Portal: Create a test-identity and login to Administrative Access * Event Log: Index Events * Reference Systems for Java and .Net Core   + REST protocol examples   + SMTP protocol examples   + SFTP protocol examples   Updated the following section:   * Services exposed from Digital Post * Domain names, components and services * Event log services * Access to environments * Web certificates policy |
| 1.14 | 10.03.2021 | Updated the following section:   * Update mailbox * Event Log Index Events * Inbound MeMo REST Push examples * Distribution SMTP services * Sending DP/DP2 messages via SMTP * Back-end validation and errorcodes in distribution * Recipient-system error codes |
| 1.15 | 17.03.2021 | Updated the following section:   * List of terms * Exporting certificate for upload to Administrative Access * Access to environments (NB: New DP IP’s) * Overall status for implementation |
| 1.16 | 25.03.2021 | Updated the following section:   * Access to the Administration Portal on the test environment * Generate an SSH-key to DP * Domain names, components and services * Querying Contacts * Querying in System Registry APIs * Exporting certificate for upload to Administrative Access * Querying for Messages * Distribution REST services * Sending DP/DP2 messages via SMTP * Access to environments   Added the following section:   * Roadmap: Overall status for Implementation of DP * SFTP server   + Access the SFTP server * REST Implementation |
| 1.17 | 22.04.2021 | Updated the following sections:   * Domain names, components and services * Updating items in system registry * Mailbox services   + Mailbox domain model * Querying messages * Querying the event-log * Distribution REST services   + Inbound MeMo REST Push request   + Outbound MeMo REST Push request * Distribution SMTP services * DP Receipt domain model * Bulk MeMo SFTP service * Legacy services: Sending DP/DP2 messages via SFTP * HTML whitelist for document validation * Front-end validation and error codes in the viewclient * Back-end validation and error codes in distribution * Recipient-system error codes   Included the following sections:   * Querying and searching resources |
| 1.18 | 06.05.2021 | Updated the following sections:   * Roadmap: Overall status for Implementation of DP * Domain names, components and services * Querying contacts * Mailbox services   + Mailbox domain model * Distribution REST services   + Inbound MeMo REST Push request |
| 1.19 | 19.05.2021 | Updated the following sections:   * Mailbox services   + Mailbox domain model * Distribution REST services   + Inbound MeMo REST Push request * Distribution SMTP services * DP Receipt Domain model * Bulk Memo SFTP service * Sending MeMo * Back-end validation and errorcodes in distribution   Included the following sections:   * Move message between folders * Push notification integrations |
| 1.20 | 10.06.2021 | Updated the following sections:   * REST Implementation * Services exposed from Digital Post * Querying and searching resources * Contact registry services   + Querying Contacts * System registry services * Exporting certificate for upload to Administrative Access * Mailbox services * Eventlog services * Querying the event-log * Event Log Index Events * Push notification integrations * Sending MeMo * Front-end validation and errorcodes in the Viewclient * Back-end validation and errorcodes in distribution * Web certificates policy   Included the following sections   * Roadmap for new functionality * Mutual SSL authentication using API key * Open API description * Domain names in Digital Post * Tracing requests using W3C headers * Ensuring the authenticity of Digital Post through the OCES certificate   Removed the following sections   * Domain names, components and services |
| 1.21 | 20.06.2021 | Updated the following sections:   * MeMo * Contact registry services * Updating items in System registry * Mailbox domain model * Push notification integrations * Event Log Index Events * Distribution REST services * Bulk MeMo SFTP service * Fetching a single MeMo over REST PULL * HTML whitelist for document validation * Front-end validation and errorcodes in the Viewclient * Reference Systems for Java and .Net Core   + REST protocol examples   + SMTP protocol examples   + SFTP protocol examples   Included the following sections:   * Configuring mutual SSL for REST flow * Access request registry services   + Access request domain model   + Querying access requests * Create test messages   + OIO OpenID Connect in Digital Post |
| 1.22 | 15.07.2021 | Updated the following sections:   * Services exposed from Digital Post * Querying and searching resources * Contact registry data model * System registry services   + System registry data model   + Updating items in the system registry * Mailbox services   + Reply to message   + Update one or more of certain predetermined fields of message   + Move message between folders   + Forward message to e-mail address   + Forward message to trusted recipient and authority * Eventlog services   + Eventlog index events * Distribution REST services   + Inbound MeMo REST Push request   + Outbound MeMo REST Push request   + Inbound business receipt request * Fetching a single MeMo over REST PULL * Bulk MeMo SFTP services * Push notification integrations * Access request registry services * HTML whitelist for document validation   Included the following sections:   * REST\_PULL service protocol * Fetching business receipts |
| 1.23 | 06.08.2021 | **Added the following sections**   * Demo viewclients * Services exposed from Digital Post   + Quality of service when searching   **Updated the following sections**   * Roadmap for new functionality * Introduction - D0180 * REST Implementation * Querying and searching resources * Patterns for integration to Digital Post   + Sending MeMo   + Receiving MeMo * Contact registry services   + Querying Contacts * Event log services   + Querying the event-log   + Event Log Index Event * Distribution Services   + Distribution REST services     - Inbound MeMo REST Push request     - Outbound MeMo REST Push request   + Distribution Receipt Domain Model   + Bulk MeMo SFTP service * Access request registry services * Error codes   + Front-end validation and errorcodes in the Viewclient   + Back-end validation and error codes in distribution * Reference Systems for Java and .Net Core   + Reference architecture of Sender and Receiver Systems   + REST protocol examples     - Configuring mutual SSL for REST flow   **Removed/rearranged the following sections**   * Domain names in Digital Post * Quality of service when searching * Contact registry services   + Update mailboxSubscription registrationStatus   + Subscribing to NemSMS * System registry services   + Querying Organisations, Contact points and Contact groups * Distribution REST services   + Inbound business receipt request   + Overview protocols for sender and recipient systems * Identity registry services * Auth Server service * Access request registry services   + Access request registry - introduction   + Access request domain model   + Access request registry - commom use case examples * Test-IDP (test-identity-provider) * Pull notification Domain Model * Reference Systems for Java and .Net Core   + Reference architecture of Sender and Receiver Systems * Software Development Life Cycle (SDLC) for the API * Versioned API per component type * Services Digital Post integrates to |
| 1.24 | 17.08.21 | **Added the following sections**   * Identity registry services * Access request registry services |
| 1.25 | 15.09.21 | **Updated the following sections**   * Querying the event log * ReplyData mail threads * Bulk MeMo SFTP service * Identity registry services * Back-end validation and error codes in distribution * SMTP protocol examples * OIO OpenID Connect in Digital Post * Connect to DP: Test and prod   **Included the following sections**   * Flow for resending messages * PROD environment * Legacy error codes * Fetching registration status for contact * Handling of reply destination in DP contact structure when sending DP/DP2 * Software Development Life Cycle (SDLC) for the API   **Removed/rearranged the following sections**   * Demo view clients |
| 1.26 | 24.09.21 | **Updated the following**   * Access to environments   **Included the following**   * Making requests - important! |
| 1.27 | 06.10.21 | **Updated the following**   * Event Log Index Events * Fetching business receipts * HTML whitelist for document validation * OIO OpenID Connect in Digital Post   **Included the following**   * Message state action matrix   + Rate-limiting |
| 1.28 | 26.10.21 | **Updated the following**   * Inbound MeMo Rest Push request * Access request registry - introduction * Legacy services: Sending DP/DP2 messages via SFTP   **Included the following**   * Contact-registry in Record format   + Flow for resending business receipts - REST Push protocol |
| 1.29 | 04.11.21 | **Updated the following**   * Legacy services: Fetching registration status for contact |
| 1.30 | 08.11.2021 | **Updated the following**   * Event Log Index Events   **Included the following**   * Sending business receipts as a recipient system |
| 1.31 | 23.11.2021 | **Included the following**   * Contact in legacy csv   **Updated the following**   * Domain names in Digital Post * Querying messages * Exporting certificate for upload to Administrative Access * Legacy services: Sending DP/DP2 messages via REST * Handling of reply destination in DP contact structure when sending DP/DP2 * HTML whitelist for document validation * Back-end validation and error codes in distribution |
| 1.32 | 15.12.2021 | **Included the following**   * Querying all contacts by the usages of search after and the parameter ‘next’ * Fetching contact registration status part list   **Updated the following**   * Querying contacts * HTML whitelist for document validation * Allowed certificate cipher suites |
| 1.33 | 28.01.2022 | **Updated the following**   * Event Log Index Events * Digital Post Receipt domain model   + Sending MeMo message over REST PUSH * REST protocol examples |
| 1.34 | 22.02.2022 | **Included the following**   * At least once-principle   + Specifies that Digital Post operates with a principle which ensures that all messages are delivered at least once * Requirements for messages and attachments   + Specifies the allowed sizes of attachments and messages * Fetching registration status list   + Specifies that the interface defaults to JSON. Use the accept header if XML is preferred. * Bulk receipt list (massekvitteringsliste)   + Specifies that only XML return values are supported for this endpoint   **Updated the following**   * List of terms   + Specifies that DP uses UUID version 4 * Rate-limiting   + Minor update * Distribution REST services   + Specifies that the interface defaults to JSON. Use the accept header if XML is preferred. * DP receipt domain model   + Specifies that COMPLETED means that the message has been validated * Sending DP/DP2 messages via REST   + Specifies that the interface defaults to JSON. Use the accept header if XML is preferred. * Event Log Index Events   + New event with ID 81 added to show events in the event log in the Rights Portal regarding access request acitivities. * Fetching registration status for contact   + accept header added to the documentation * Querying identities, Direct privileges, Privilege Type   + Specifies that you can also revisit the OpenAPI specification for more background * Back-end validation and error codes in distribution   + Specifies that TransmissionID must be unique to be valid |
| 1.35 | 17.06.2022 | **Updated the following**   * Querying and searching resources   + Specifies the use of quotations * Querying the event-log   + Description of Querying for more than 10k using “next” * Flow for resending messages   + Description of retrying to default recipient system and system deactivation |
| 1.36 | 14.09.2022 | **Added the following**   * Certificates   + Information about how Digital Post currently supports OCES2 and OCES3 * Push Notification Integration: Creating FCM service account + private key   + Guide to how view clients can offer push notifications   **Updated the following**   * 14: Access to the Test Portal on the test environment   + Content has been deleted as DIGST has created an external guide for the Test Portal of Digital Post   **Removed/rearranged the following sections**   * “15: Test Portal: Create a test-identity and login to Administrative Access” has been changed to “15: How to login to Administrative Access with test-identities” as the previous content can be seen in above-mentioned guide |
| 1.37 | 21.11.2022 | **Updated the following**   * 3.3 Size requirements   + Added word to clarify paragraph |
| 1.38 | 30.05.2023 | **Updated the following**   * Domain names   + Updated External IP for the domain to access the REST API by sender and receiver systems using mutual SSL * Querying and searching resources * Querying Contacts   + Details about the query parameter isBulkLookup * Encoding format whitelist for files of documents   + Added additional documents * Exporting certificates for upload to Administrative Access   + Sending messages using the SMTP protocol will be faced out 16 August2023 * Event log services   + Link to a full comprehensive list of all events stored in the event log * Creating and working with drafts   + Usage of generic identity id in access requests * Front-end validation and error codes in the Viewclient * Back-end validation and error codes in distribution * Software Development Life Cycle (SDLC) for the API * HTML whitelist for document validation   + Updated elements and attributes |
| 1.39 | 20.06.2023 | **Updated the following**   * Contact Status   + Added the column statusDate * Messages   + Added the columns Dele messages and Fetch unread status * Afsendelse (resource)   + Added the columns VedhaeftningSamlingKvantitet, MeddelelseKvitteringsTypeNavn, MeddelelseKvitteringPostkasseIdentifikator, MeddelelseFESDmetadata, MeddelelseTidsfristDato and MeddelelseTidsfristTeks * HTML whitelist for document validation   + Updated the Styles table * Overall strategy for the Software Development Life Cycle (SDLC) for the API   **Included the following**   * Comment: PublicRegistrationStatus * Contact Subscription   + The purpose of the contact subscription is to persist subscriptions for sendersystems, on either a set of specific citizens, organisations or all changes fitting a category. * Subscribing to changes in the system registry   + The purpose of the system subscription is to persist subscriptions for sendersystems, on either a set of specific organisations, or all changes fitting a category. When a change is made to a contact point is made or a new contactpoint is created, the subscription components finds all matching subscriptions, and notifiesthe sendersystem on all the matching subscriptions. * Error codes   + System Fetch services   **Removed/rearranged the following sections**   * Structure of the API |
| 1.40 | 24.07.2023 | * **Introduction**   + Added additional terms to the terms list   + Expanded the description of the usage of mutual SSL and API token   + Updated the list of trusted certificate authorities * **Access to Test environment**   + Updated onboarding flow to start with log-in to Testportalen * **Encoding formats**   + Updated E-mails from Digital Post * **Event Log**   + Updated Event Log Index Events with more elements and examples of events. * **OIO OpenID Connect to Digital Post**   + Added details regarding logout in Digital Post and NemLog-in   + Added details for MitID AppSwitch and eID-Gateway support   + Updated client enrollment section with more detailed descriptions and any relevant restrictions where applicable * **Querying and searching:**   + Added limitation of 100 characters for each search field and added examples |
| 1.41 | 18.10.2023 | * **Contact registry services**   + Full Access added as a role that can use contact registry   + PublicRegistrationstatus updated with status = CLOSED * **Distribution**   + Added section “Should I send single messages or bulks?” * **Encoding formats, Environments and Error codes**   + Added new error codes related to full access * **Event log services**   + Updated documentation on querying the event log.     - A default time interval on 3 weeks has been added     - Wildcard searches are no longer supported   + contentResposible added to example of element in event log * **Mailbox services**   + Added full access as a new role that can use mailbox services * **Troubleshooting, SFTP server, SDLC, OpenID Connect**   + Updated fingerprints for accepted issuer CAs   + Updated link to OCES certificates   + Updated section “OIO OpenID Connect to Digital Post” |
| 1.42 | 14.11.2023 | * **Access to Test environments**   + Updated steps for creating test users with MitID Simulator. |

**References**

|  |  |  |  |
| --- | --- | --- | --- |
| **Reference** | **Title** | **Author** | **Version** |
| User Manual for Administration Portal  <https://digitaliser.dk/digital-post/vejledninger/administrativ-adgang> | 'Administrative Access User Guide' | Netcompany and DIGST | Latest |
| User Manual for Rights Management Portal  <https://digitaliser.dk/digital-post/vejledninger/rettighedsportalen> | ‘Rights Management Portal User Guide’ | Netcompany and DIGST | Latest |

# Introduction and overview

## Roadmap for new functionality

This page presents an overview over planned releases to test.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Component** | **Feature** | **Target release** | **Details** | **Required action** |
| Mailbox, distribution | Full access | Q3 2023 on TEST  Expected to be enabled in production in 2024 | Digital Post will implement a new kind of power of attorney called ‘full access’ which enables Users to manage the mailbox of an individual including being able to send messages of others behalf | More details will be published later |
| Test portal | Improved test data | Release 66 | Creating of test data in the test portal will be changed to imitate real data from CPR and CVR to enable better testing capabilities | None |

## Status on interfaces

|  |  |  |
| --- | --- | --- |
| **Distribution - Sendersystem** | | |
| **MeMo services** | **Status** | **Comment** |
| REST PUSH (sender) | Done |  |
| REST PULL (sender) | Done |  |
| SMTP | Done |  |
| SFTP | Done |  |
| **Legacy** | **Status** | **Comment** |
| REST | Deprecated | End of life November 2023 |
| SMTP | Deprecated | End of life 16th of August 2023 |
| SFTP | Deprecated | End of life November 2023 |

|  |  |  |
| --- | --- | --- |
| **Distribution - Recipient system** | | |
| **MeMo services** | **Status** | **Comment** |
| REST PUSH | Done |  |
| REST Publish/Subscribe | Done |  |
| REST PULL | Done |  |
| SMTP | Deprecated | End of life 16th of August 2023 |

|  |  |  |
| --- | --- | --- |
| **Contact registry** | | |
| **DP services** | **Status** | **Comment** |
| Download registration status | Done |  |
| Set up/edit/delete subscriber to changes | Done |  |
| Update NemSMS connection | Done |  |
| Download local registration file from SFTP | Done |  |
| **Legacy** | **Status** | **Comment** |
| Download local registration file from SFTP in e-Boks format | Deprecated | End of life November 2023 |
| Fetch registration status in e-Boks format | Deprecated | End of life November 2023 |

|  |  |  |
| --- | --- | --- |
| **System registry** | | |
| **DP services** | **Status** | **Comment** |
| Download contact groups | Done |  |
| Download organizations | Done |  |
| Subscribe to changes | Done |  |
| Download contact points | Done |  |

|  |  |  |
| --- | --- | --- |
| **Event log** | | |
| **DP services** | **Status** | **Comment** |
| Download events | Done |  |

## Background

This document contains the technical system documentation in order for authorities to integrate their systems to Digital Post. Such systems are referred to as sender- and recipient systems. With this generation of Digital Post comes a set of new features and changes, such as:

* New message format (MeMo)
* Improved administration portal
* Easier access to data in the log files

Note: Setting up sender- and receiver systems and accessing log files etc. is handled in the new administration portal ‘Administrativ Adgang’. Follow the user guide (See ‘Reference’) for more information on how to navigate in the portal and how to set up your systems once they are ready.

Disclaimer: This is not a complete guide in building or adjusting sender and receiver systems. It does however provide all the technical information of the Digital Post solution needed to build or adjust your own sender and receiver systems for the new infrastructure.

“DP” refer to “Digital Post”.

## Memo

MeMo (abbreviation for **me**ssage **mo**del) is the new message model for exchanging messages in Digital Post. It is developed and maintained by The agency of Digitalization or in Danish “Digitaliseringsstyrelsen”. With the goal of increasing a widespread use of an intelligent and automated message format.

### Information

The detailed information about the format can be found here; <https://digst.dk/it-loesninger/naeste-generation-digital-post/for-myndigheder-og-it-leverandoerer/det-nye-meddelelsesformat/>

Further documentation, schemas and examples for MeMo:

* MeMo version 1.1 <https://digitaliser.dk/digital-post/vejledninger/memo>

### MeMo library

As MeMo is utilized by DP, senders and recipients; handling (de-/serialization) from XML is encapsulated in a common library that is available for public use. As well as compression utilities for the TAR+LZMA for bulk shipments.

This ensures compatibility, intended to lower the need for support when developing integration with DP. And is intended to encourage switching from the deprecated DP2-format to MeMo for existing integrations.

The current version of the library is implemented according to [MeMo version 1.1](https://www.digitaliser.dk/resource/5248921) (specification and examples in Danish) and can be found here: [Referenceimplementeringer og MeMo-lib (digst.dk)](https://digst.dk/it-loesninger/digital-post/anvendelse/forstaa-digital-post/memo/referenceimplementeringer-og-memo-lib/)

## Size requirements for messages and attachements

Different restrictions apply to a message depending on whether it is sent from a view client or from a sender system. However, it always applies that a message cannot contain more than 1 MainDocument and 10 AdditionalDocuments/TechnicalDocuments (combined), and each of these documents cannot contain more than 10 files.

**Requirements for messages sent from a view client**

A message cannot exceed 70MB when it is sent from a view client. This limit is set to ensure that the entirety of the message does not exceed 99,5MB after the message is encoded. Furthermore, a single attachment cannot exceed 10MB.

**Requirements for messages sent from a sender system**

A message cannot exceed 99,5 MB when sent from a sender system. There is no limit on a single attachment as long as the entirety of the message does not exceed 99,5 MB.

## Target group

This guide is primarily intended for developers to understand the design, technical implementation and integrations. As well as for architects and business analysts to ensure that technical integrations are documented and that access has been established. Readers are assumed to have knowledge of technical terms (e.g. web certificates, IDP), common protocols (e.g. HTTP, SFTP) and industry standards (e.g. RESTful services, TLS, JSON).

In addition, this guide describe the high-level processes involved in adjusting sender- and receiver systems, which may be relevant for project planning.

Note: Only integrations from the Digital Post perspective are documented and not the detailed implementation.

For the process of integration, activating, and establishing as well as deactivating sender- and receiver systems in the administration portal see ‘References’.

If additional help is needed to adjust or build sender- and/or receiver systems see ‘Additional configuration support’.

### For test authorities

This guide is intended for authorities with the need to adjust, adapt or build their receiver- and sender systems integrating to DP.

### For businesses

This guide is intended for businesses with their own recipient - and sender systems. For businesses that do not have a dedicated receiver system it is still possible to read digital post at <http://virk.dk> .



Whenever something technical in this guide is not relevant for you as a business the section has been marked with “Not relevant for businesses”.

## List of terms

|  |  |  |
| --- | --- | --- |
| **Term / English** | **Danish** | **Description** |
| **Authority** | Myndighed | An authority is official institutions, ATP and systems administered by ATP, all municipalities and regions and some independent institutions funded by public finance. An authority is identified by the CVR-number. Every authority can have multiple sender- and recipient systems, registers and portals. |
| **Business** | Virksomhed | A business is a private corporation able to receive and send Digital Post. A business can only send Digital Post to an authority. Most businesses use their own mailbox at <http://virk.dk> but some have their own recipient (and sender) systems in order to distribute Digital Post in the organization . |
| **Organisation** | Organisation | A collective term used for both companies and authorities. |
| **Contact group** | Kontaktgruppe | Contact groups are used to organize contact points in folder-like entities. Internally, the purpose is to provide an option of establishing an overview of the total of the contact points. Externally, the groups containing related contact points (“subjects”) can be presented as either a ‘subject group’ or an organizational unit. |
| **Contact point** | Kontaktpunkt | The contact point is a unique entity in the authority’s contact structure used to indicate where mails need to be distributed to. Externally, these are presented as “subjects”. |
| **Contact registry** | Kontaktregister | The contact registry supports the look up of information about registration of citizens and businesses to Digital Post and NemSMS. Registration status is used for distribution of messages and for authorities to look if a citizen is exempted or not. |
| **Contact structure** | Kontaktstruktur | The contact structure is the authority’s structure of how the authority receives and distributes mails for the receiver systems from DP. The contact structure may also be used as a link to a self-service solution (selvbetjeningsløsning). |
| **Classification** | Klassifikation | Classification is a structured way of listing terms and subjects in relation to common public administration tasks. Here, they can be linked to contact points to further earmark incoming mail towards the correct receiver within an authority. |
| **Classification code** | Klassifikationskode | Classification code is a unique identification number for a common public administration task. |
| **Classification name** | Klassifikationsnavn | Classification name is the name of a common public administration task. |
| **Classification type** | Klassifikationstype | Classification types indicate which public administration classification system is in use e.g. FORM or KLE. |
| **CVR** | CVR nummer | CVR-nr. is a business' identification number in Denmark. The CVR number is fetched from the Contact Registry. |
| **CPR** | CPR nummer | The CPR number is a unique and universal identifier of a physical person registeret in the states central person registry “Det Centrale Personregister” |
| **End-point** | Endepunkt | End-point is a technical address for a system connected to DP. |
| **Legal person** | Juridisk person | Contact information on a legal person in the organization. |
| **Location** | Lokationskode | The location code is used to indicate what physical device or location the contact point is connected to e.g. SOR or GLN |
| **Master data** | Master data | Master data is information or sensitive information about a business or an authority for a registered system that is connected to or integrated with DP. |
| **DP** | Digital Post | The name of the project which is developing the Digital Post solution. Often used interchangeably as a reference to the Digital Post solution |
| **Report Link** | Rapportlink | ReportLinks are links connected to a contact point with a description. When a contact point containing a ReportLink is used, the user will be presented with the link before proceeding to send mail via the contact point. E.g. this can be used to inform the user that a self-service solution should be used instead of sending digital post. |
| **Search word** | Søgeord | Search words are used as ‘tags’ to further describe with alternative words the subject or related subjects of a contact point, making the contact point easier to find through search. |
| **Sender system** | Afsendersystem | Sender system is a system which is used for sending mails via DP. |
| **Recipient system** | Modtagersystem | Recipient system is a system which is able to receive mails via Digital Post. Can also be called receiver system |
| **System** | System | A common term for sender- and recipient systems connected to Digital Post. |
| **System registry** | Systemregister | The system registry is responsible for gathering information about the authorities and businesses that are sending messages via the solution.  It also contains the Contact Structure as well as all registered sender and recipient systems. |
| **Authority registry** | Myndigheds register | The authority registry is part of the system registry |
| **Target group** | Målgruppe | Indicates which target group will be exposed to a contact point. |
| **Technical person** | Teknisk person | In the administration portal every organization has to write contact information on a technical resource e.g. a system administrator. |
| **UUID** | UUID | UUID is short for "Universal Unique Identifier" and is a 128-bit number used to identify digital objects in Digital Post.  Version 4 is used to generate UUIDs. |
| **View client** | Visningsklient | A view client displays the messages and make them accessible for end users. |
| **Public View client** | Offentlig visningsklient | A view client provided by the Danish public sector. [Borger.dk](http://Borger.dk) for citizens and [virk.dk](http://virk.dk) for businesses |
| **Virk** | Virk  [Virk.dk](http://virk.dk) | The public view client for business |
| **Borger.dk** | Borger DK  [Borger.dk](http://Borger.dk)  BDK | The public view client for citizens |
| **Notification** | Advisering | A notification is information which is triggered when information is received, In the context of Digital Post often related to a message received in a users mailbox where a notification can be send thought PUSH, email or SMS. |
| **Legal notification** | Forkyndelse | Legal notifications issued by a danish court |
| **Bulk transaction** | Masseforsendelse | A transaction of sending multiple single messages at once |
| **Exemption** | Fritagelse | Legally exempt from receiving digital post. |

## Minimum requirements for authorities

The following are the minimum requirements in terms of sender- and receiver systems in order to integration with Digital Post.

### Sender systems

* Sending: There is a transition period of two years (November 2023) until sending messages via the new MeMo-format is a requirement
* Has to use the new interface (API) for the contract structure
* Must be able to receive business receipts in the new format - also when sending DP2
* Has to use the new contact registry
* Has the same restrictions in terms of file size to send post as the current solution

### Recipient systems

* Has to point to the new DP solution. Configuration is done via the new administration portal (See ‘References’)
* Has to be adjusted in order to receive messages in the new MeMo-format at Go Live August 2021.
* Businesses may have one receiver system.
* Authorities may have 1-multiple receiver systems (no cap: Five is recommended).
  + At least one receiver system for messages sent to you as a business.
    - At least one of your receiver systems has to be able to receive 99,5 mb sized messages
* All messages related to tasks as an authority has to be collected and authorities have to store the messages in its’ own systems.
* Has the same legal requirements in terms of file size to send post as the current solution.

## More information

For a comprehensive overview of guides in relation to the new Digital Post solution please go to: <https://digst.dk/it-loesninger/digital-post/vejledninger/>

## Overview of DP

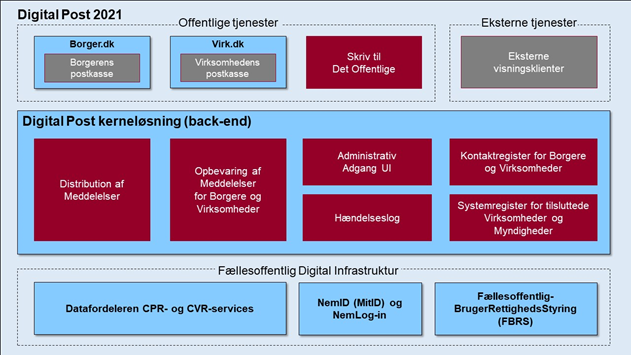


Figure 1 Overall architecture in Digital Post (In Danish)

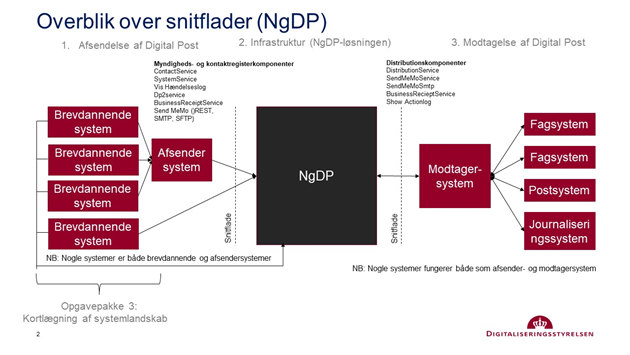


Figure 2 Overview of interfaces (In Danish)

### At least once-principle

Since the delivery of messages to recipients is the number one priority of Digital Post, the solution adheres to an “At least once”-principle when sending out requests to sender systems and recipient systems. Under normal operation, Digital Post will send messages through integrations only once, of course. But if there’s a fatal infrastructure disaster, there is a risk that the Solution will resend out any messages that haven’t been processed completely in the near real-time backup. If this situation occurs, the volume is expected to be a few seconds (<20 seconds) worth of processing that would be resent. This holds true for both MeMo-messages, Receipts and Publishing of changes (for systems subscribing on changes of Contacts, for instance).

When a recipient system receives an already received MeMo, it is expected to send a business receipt in return, although one might already have been sent, when the MeMo was first received. Otherwise Digital Post might see consider the recipient system failing and will deactivate it after a number of attempts.

### Mutual SSL authentication using API key

This section aims to give a concise overview of how sender- and recipient-systems are expected to interact with the REST API of Digital Post. This section does not go into details on how mutual SSL/TLS works or how OCES certificates are obtained. Additionally, concrete *example* implementations of how to integrate with Digital Post can be found in the reference systems which are provided in both [Java](https://bitbucket.org/nc-dp/reference-systems-for-java/src/master/) and [dotNET](https://bitbucket.org/nc-dp/reference-systems-for-dotnet/src/master/).

**Background**

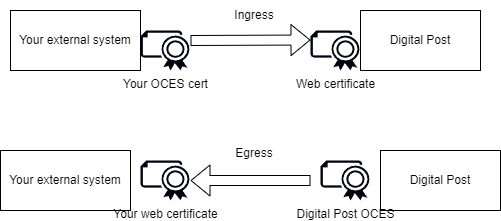
For sender- and recipient-systems to communicate securely with Digital Post they are expected to utilize mutual SSL where both the initiator and the responder are presenting a certificate. That way, both parties can create a secure channel of communication and verify the identity of each other.

The reasoning behind why Digital Post requires sender- and recipient-systems to authenticate using mutual SSL is because it is an industry standard for two parties to establish a secure connection and exchange sensitive information where both parties can verify the identity of the other party. *Additionally*, Digital Post requires an API key used **together** with a mutual SSL connection to avoid binding the client certificates to specific systems.

**Certificates used in the http traffic**

When performing API calls not just any certificate can be used. Digital Post has strict requirements for the format and use of certificates. There is always one web certificate and one OCES certificate in motion as part of the mutual SSL. The pattern is the same, however, it depends on who is initiating the communication. This can be described as follows, where ingress is a sender- or recipient system connecting to Digital Post and egress where Digital Post connects to a external system

* **Ingress**: Sender systems present their **OCES certificate** and Digital Post presents a valid **web certificate**
* **Egress**: Digital Post presents its valid **OCES certificate**, while recipient system presents a valid **web certificate.**



**Certificates used to authenticate**

As shown in the above figure, the mutual SSL authentication uses both an OCES certificate and a web certificate. The OCES certificate is a certificate as defined by the [OCES specification](https://digst.dk/it-loesninger/nemid/om-loesningen/oces-standarden/) and are issued through MitID Erhverv. In the specification there are a couple of different certificate types. Digital Post supports the following types

* Organisationscertifikater (organisation certificates)
* Systemcertifikater (System certificates)

During ingress calls you, as the client, is expected to present a certificate which is issued to your organisation. This certificate is not issued in the context of Digital Post but is something which you need your MitID administration to issue for you. You can find more information about how to obtain an OCES certificate [here](https://www.mitid-erhverv.dk/avanceret/certifikater/).

During ingress Digital Post, as the server, exposes a web certificate. Web certificates are certificates issued by generally trusted certificate authorities, for further details see section *Web certificates policy*.

Similarly during egress, where Digital Post is the client and calling you as the server, Digital Post will present its OCES certificate and you are expected to expose your server with a valid web certificate. To increase the security during egress communication you can pin the certificate which Digital Post is using to ensure that you are only accepting calls from Digital Post. You can find the public certificates here: [Digitaliser.dk - OCES certifikater til Digital Post løsningen](https://digitaliser.dk/digital-post/vejledninger/oces-certifikater).

As the above diagram shows, during ingress communication you as the client is expected to present a valid OCES certificate.

**Support for OCES2**

The old version of the OCES certificates, called OCES2, was part of the NemID suite and with the introduction of MitID Erhverv OCES2 was replaced by OCES3. Digital Post, however, still supports OCES2 for the time being. The sender can thus utilize a FOCES Funktionscertifikat (function certificate) or a VOCES virksomhedscertifikat (company certificate). However, support for OCES2 will be phased out on October 31, 2023. If you are still using OCES2 we strongly recommend migrating as soon as possible.

For calls to Digital Post it is transparent if you use either OCES2 or OCES3 and does not require any setup changes in Administrative Access as long as the certificate is issued to the same company.

Outgoing calls from Digital Post to your system, Digital Post has already transitioned to OCES3.

|  |
| --- |
| Support for OCES2 will be phased out on October 31, 2023. Make sure that you have migrated your systems beforehand |

#### Before calling the API

Before you can begin calling the Digital Post REST API you must create a sender- or recipient-system in Administrative Access. Here you are presented with a range of options, that are required for the system to be operational. Refer to the dedicated guide for [Administrative Access](https://digitaliser.dk/digital-post/vejledninger/administrativ-adgang) for further details.

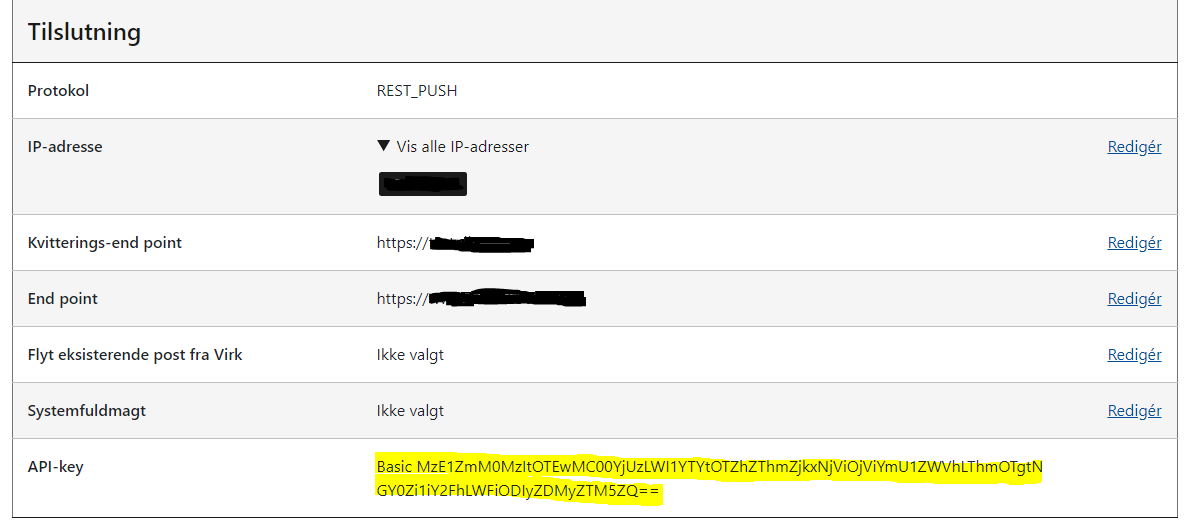
Once you have created an active system and chosen one of the available rest service protocols you are presented with an API key. You must save this value. Additionally, you can also choose to delegate the systems to a vendor, more on this later.

**API key**

The API key is an attribute on REST systems in the system-registry. It is used by Digital Post to identify which system that is calling the API, not as a way to establish a secure channel of communication (this is why mSSL is still required). The API key is a random string of text that is unique to a given system. When the system is created through Administrative Access an API key is always automatically assigned, note that the system still must be valid and active to be used.

The value that is presented in Administrative Access, and the one that the user should pay attention to is the encoded value. Where the system will perform encoding of the systems ID together with the value of the token. This is to ensure the least amount of work from the integrator since they will then be able to send the API key directly as presented in the authorization header and not having to worry about encoding.

From the Administrative Access, the API key can be retrieved from system details as the screenshot below



To avoid confusion only the encoded value is present in the UI, however taking a look at the raw response from the server, we can see the encoded key together with the raw value

|  |
| --- |
| {  ...  "apiToken" : {  "id" : "ef1ec3b0-0136-47e1-8ef2-f396d2db6e58",  "version" : 0,  "value" : "5bbe5eea-8f98-4f4f-bcaa-ab822d32e39e",  "authenticationToken" : "Basic MzE1ZmM0MzItOTEwMC00YjUzLWI1YTYtOTZhZThmZjkxNjViOjViYmU1ZWVhLThmOTgtNGY0Zi1iY2FhLWFiODIyZDMyZTM5ZQ==",  "lastUpdated" : "2021-05-10T13:45:46.155Z"  }  ...  } |

In addition to the value property, which is the persisted value of the API key, the details also return a property named authenticationToken . The encoded value (shown in the UI and in authenticationToken) is encoded using base64 encoding following the [The 'Basic' HTTP Authentication Scheme](https://datatracker.ietf.org/doc/html/rfc7617), and the “username” is the id of the system and the “password” is the value of the key.

**Mutual SSL using API key**

To call Digital Post it is not necessary to upload or otherwise register your OCES certificate to your REST System. However, it is still *mandatory* for the client to present the OCES certificate in the mutual SSL requests. In addition, the API key authenticationToken value must **always** be added in the Authorization header.

Why does Digital Post require both mSSL and API key to be present? These reasons are, but not limited to, the following

* Organisations with sender- and recipient-systems are not required to upload their OCES certificates in Digital Post, but there is still a need for unambiguously identifying the system in order to obtain authorization for the REST API
* No need for maintaining certificates if they expire or are compromised in Digital Post
* Delegation of systems can be done using vendors' certificate
* OCES certificates are not required to be unique across the entire Digital Post solution and can thus be reused for all systems an organisation has

The reason why we gain these benefits while still having both a secure and reliable solution where certificates are no longer required to be unique is that when initiating a call, Digital Post does the following checks when authenticating the sender-/recipient-system. If any of the following checks fail the call is rejected

* A valid Organisationscertifikat or Systemcertifikat is used to establish a secure mutual SSL connection
* An API key is presented in the authorization header
* The API key can be used to find the system
* The system is valid and active
* **The system belongs to the same organisation as is present in the callers OCES certificate**. This is done by matching the CVR numbers. If you need a vendor to perform an actions on your behalf, read the section [*Delegated Sender- and Receiversystems*](#scroll-bookmark-30)*.*
* The callers IP matches the IP or is within the IP range as configured in Administrative Access.

Given this setup, you cannot register your certificate preemptively in Digital Post since we do not have any direct binding to a specific certificate but rather uses the CVR number present in the OCES certificate when your system is authenticating via mutual SSL when calling Digital Post. This means that you do not upload your certificate in Administrative Access, rather you authenticate using the certificate.

|  |
| --- |
| You are required to use both mutual SSL and API token when calling Digital Posts REST API |

In this setup, Digital Post does not know your certificate, as it matches using CVR you are free to use as many and as few certificates as you desire. And you are not required eforehand to register any specific certificate in Digital Post. This method is secure for both sender- and recipient-systems and Digital Post since the caller can verify the identity of Digital Post, and Digital Post can with confidence extract the CVR from the callers OCES certificate since we verify the certificate chain.

**Examples**

See the example of using the current implementation of mutual SSL using [cURL](https://curl.se/) on the TEST environment. In this example, the certificate my\_foces\_certificate.cer is not known to Digital Post because it was not uploaded. However, as mentioned above, uploading a certificate to the system after creation is no longer required since Digital Post can authenticate using the FOCES certificate presented by the caller, and find the system based on the API key provided in the Authorization header

|  |
| --- |
| curl -v \  --http1.1 \  --key "my\_foces\_certificate.pkcs8" \  --key-type pem \  --cert-type pem \  --cert "my\_foces\_certificate.cer" \  -H "Authorization: Basic ZTU0NWIzMzktODU0Mi00YTMwLWFlNzUtYzY3ZTRkMmE3Yjk5OjE1NjMyYThkLTQwN2MtNGMzYS1iN2IxLWFlYmFhNTE0ZmNhYg==" \  "https://api.test.digitalpost.dk/apis/v1/contacts/" |

For a more detailed example see reference implementations.

#### Delegated Sender- and Recipient-systems

If you as an organisation or authority want to outsource or delegate a specific sender- or recipient-system you need to apply an additional settings during the setup of the system in Administrative Access. You need to specify the “Giv din systemleverandør fuldmagt (Valgfrit)” field with the CVR number of the *vendor* that you want to delegate to. Once this is done, Digital Post has noted this special relation, and the vendor can then use **their own OCES certificates**to act on behalf of your organization, and Digital Post then ensure that only they are allowed to use that system.

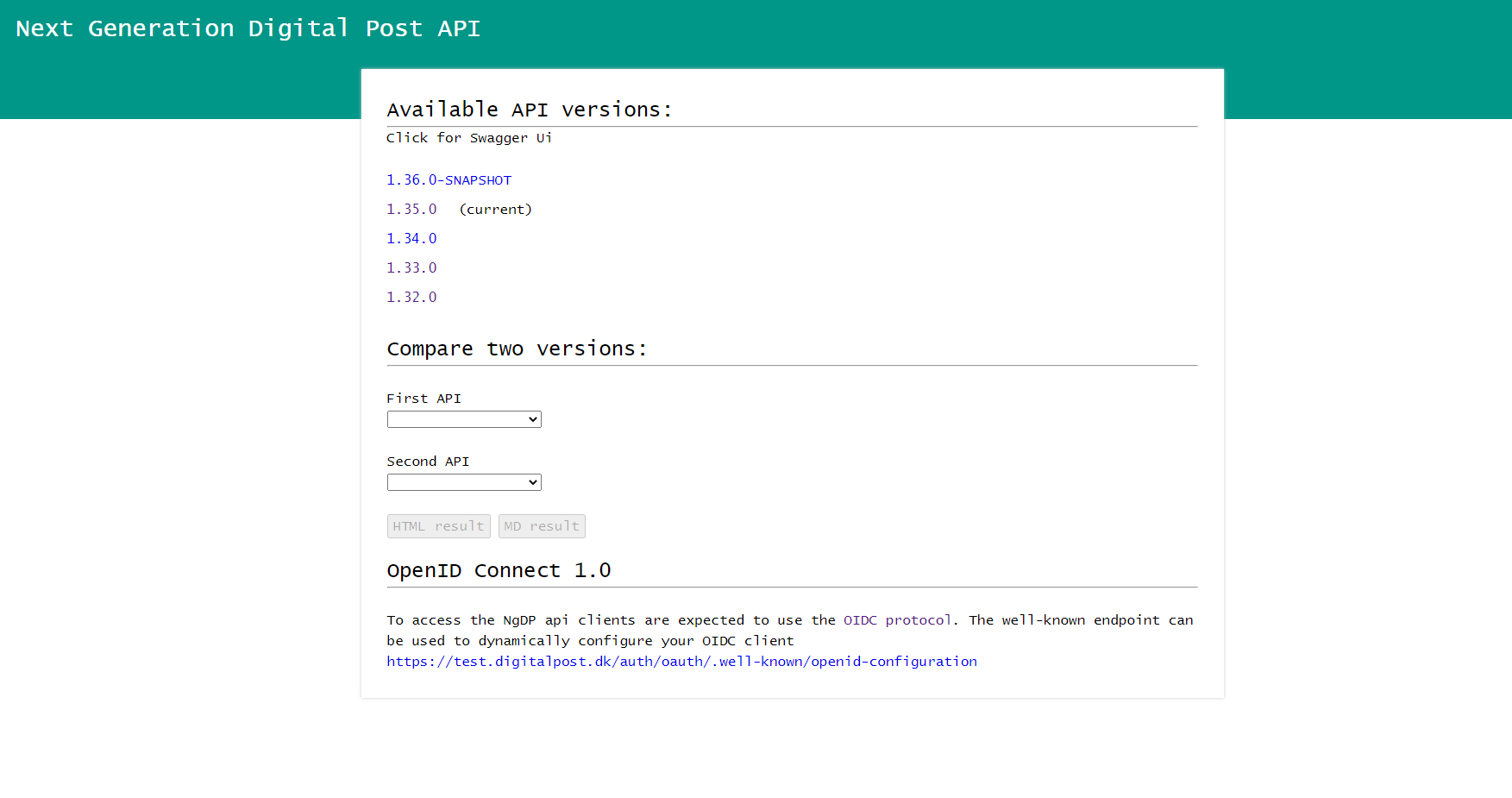
|  |
| --- |
| Delegated sender- and recipient-systems are required to use both mutual SSL and API key. |

### Open API description

Digital Post defines all of the externally exposed services in an [OpenAPI specification](https://spec.openapis.org/oas/v3.1.0). The intention is to make the API easily digestable, and to provide a programing language-agnostic description of the rest API. Eliminating any guess work when integrating with the rest API and to avoid having to rely directly on this documentation as the source of truth. Additionally this also provides versioning for the API so that you are able to see the differences between different versions of the API.

Digital Post also provides a tool that developers can use to compare different versions of the API, which should help when upgrading between the different versions. Generally Digital Post provides a new OpenAPI description after each release.

The OpenAPI definition can be found following the URL; <https://test.digitalpost.dk/api/> or <https://api.test.digitalpost.dk/api/> .



*Note that both URLs point to the same destination and provides the same information.*

When following above link you will be presented with a web-page show casing a handful of the lastest description as well as a marking for the currently used page. Additionally a snapshot version containing the upcoming changes are also present, however this version is subject to change. By clicking on one of the versions you will be directed to a “[SwaggerUI](https://swagger.io/tools/swagger-ui/)” where you are able to navigate the all the services in a user friendly way. Note that this page, contains all externally served services, which include both services for sendersystems, receiversystems, administrative access and view clients for both citizens and companies.

|  |
| --- |
| The Swagger UI does currently not support mutual SSL |

All consumers of the Digital Post API are encouraged to utilize the OpenAPI description to reduce errors and help when updating between version.

### REST Implementation

This section gives a general introduction to how Digital Post implements the HTTP based API. The general convention is to follow the REST concepts. The general concept is that data points are treated as resources, where you can fetch lists, a single resource, or creating a resource.

#### Endpoints

A resource is always referenced using the plural name in the path with lower casing. Also if the name of a resource consist of multiple words (such as contact point) they are separated with a - (dash).

|  |  |  |  |
| --- | --- | --- | --- |
| **HTTP Method** | **Endpoint** | **Term** | **Description** |
| GET | /resources/{id} | Fetch | Fetches a single resource by its ID |
| GET | /resources/ | Query | Queries all resources in a list, using query parameters to filter the results |
| POST | /resources/ | Create | Creates a new resource |
| PUT | /resources/{id} | Update | Updates a resource |
| PATCH | /resources/{id} | Patch update | Updates/modifies a resource, providing only a partial resource |
| DELETE | /resources/{id} | Delete | Deletes a resource |

This section describes a root level resource, such as Mailbox or Contact. However the same pattern also applies to sub-resources, such as Contact points, which are placed under an Organisation /organisations/{id}/contact-points/

#### Precondition headers

To ensure that updates from different sources does not accidentally overwrite each other Digital Post implements optimistic locking thought a set of the precondition headers as specified by <https://datatracker.ietf.org/doc/html/rfc7232> .

The optimistic locking is implemented by all resources having a version, that is then expected by the caller to provide as they modify a resource. Then if another client modified the resource in the meantime, the API will then reject the update since the caller is no longer working on the most recent and would potentially overwrite the changes. By applying optimistic locking, and it means that Digital Post can avoid having pessimistic locking, where the caller would have to explicitly lock a resource while editing to prevent others from editing. And required the call to both lock and unlock a resource while editing.

* [ETag](https://datatracker.ietf.org/doc/html/rfc7232#section-2.3); is present as a response header when fetching a resource
* [If-Match header](https://datatracker.ietf.org/doc/html/rfc7232#section-3.1) must always be set by the calling when updating or deleting a resource. The value is expected to be the exact version of the modified resource. Multiple values or wildcard is not accepted by the api.

#### Fetch

GET /resources/78eb6ffa-cdc4-412e-bcb3-e1e0ef552ee9

Fetch a single resource by its ID. All resources always contain both a version and an ID at the root level of the resource. Additionally sub-resources (or objects) often also contain their own ID and version. The result conforms to the following structure in JSON:

|  |
| --- |
| {  "id": "78eb6ffa-cdc4-412e-bcb3-e1e0ef552ee9",  "version": 12  ...  } |

The main purpose of having a version on resources is to have optimistic locking, where the caller provides the version that it modified. Then if another client modified the resource in the meantime, the API will then reject the update since the caller is no longer working on the most recent and would potentially overwrite the changes. By applying optimistic locking, and it means that Digital Post can avoid having pessimistic locking, where the caller would have to explicitly lock a resource while editing to prevent others from editing. And required the call to both lock and unlock a resource while editing.

##### Return codes

The HTTP response status codes indicate whether a specific request has been completed:

|  |  |  |
| --- | --- | --- |
| **HTTP Status code** | **HTTP Status** | **Description** |
| 200 | OK | The resource is returned in the body |
| 401 | UNAUTHORIZED | User is not authorized |
| 403 | FORBIDDEN | User didn’t have access to resource defined in path parameters |
| 404 | NOT FOUND | The resource from ID was not found |

##### ETag

The entity version is also always returned in the HTTP response header in the ETag:

|  |
| --- |
| ETag: "12" |

#### Query

GET /resources/

Returns a search result with a list of resources matching the query. The caller can then limit the results by providing one or more filters thought query parameters. When no parameter is provided, all resources are returned that the caller have access to. For instance, if a contact is calling the /contacts/ with no parameters only the citizens own contact is returned where as when a sender system is doing the same query all contacts are returned.

The matching resource results are always wrapped in a search result which enables pagination. The result therefore conforms to the following structure in JSON, where <resource> is the name of the specific resource.

|  |
| --- |
| {  "currentPage": 0,  "totalPages": 1,  "elementsOnPage": 6,  "totalElements": 6,  "<resources>": [  {  "id": "78eb6ffa-cdc4-412e-bcb3-e1e0ef552ee9",  "version": 12  ...  },  ...  {  "id": "c291d912-f932-46ca-b7f7-4415cabcf4a3",  "version": 1  ...  }  ...  ]  } |

##### Return codes

The HTTP response status codes indicate whether a specific request has been completed:

|  |  |  |
| --- | --- | --- |
| **HTTP Status code** | **HTTP Status** | **Description** |
| 200 | OK | The call was successful |
| 400 | BAD REQUEST | The query arguments do not match the expected formal definition |
| 401 | UNAUTHORIZED | User is not authorized |
| 403 | FORBIDDEN | User didn’t have access to resource defined in path parameters |
| 404 | NOT FOUND | Some resource defined in path parameters did not exist |

##### Specifying a query

GET /resources/?type=CITIZEN&postCode=2100

Will list all citizens living in postcode 2100. Type and postCode will be bound to the QueryCommand and resolved and formulated as a query for the implementing datastore (Elasticsearch or JPA) in the Persistent Service (with appropriate query infrastructure components) to resolve the result.

If nothing is found the service returns an empty result:

|  |
| --- |
| {  "currentPage": 0,  "totalPages": 1,  "elementsOnPage": 0,  "totalElements": 0,  "<resources>": []  } |

When querying the contacts it is possible to use the body of the GET request [Querying Contacts](https://digidp.atlassian.net/wiki/spaces/DP/pages/49941054/Querying+Contacts)

#### Create

POST /resources/

When creating a new resource the client is expected to provide the entire resource, however id and version is can either be omitted or provided with null values, since the service will always create these on the fly. If the creation is successful the service will provide the created resource as part of the response body. Where computed field such as id, version, transaction id and creating timestamps will be populated.

|  |
| --- |
| {  "id": **null**,  "version": **null**,  ...  } |

The service will return the created entity upon successful creation of the entity:

|  |
| --- |
| {  "id": "c291d912-f932-46ca-b7f7-4415cabcf4a3",  "version": 0,  ...  } |

##### Return codes

The HTTP response status codes indicate whether a specific request has been completed:

|  |  |  |
| --- | --- | --- |
| **HTTP Status code** | **HTTP Status** | **Description** |
| 201 | CREATED | Upon successful creation of the resource |
| 400 | BAD REQUEST | The JSON does not conform to the expected structure or the content of the JSON does not validate according to validation rules |
| 401 | UNAUTHORIZED | User is not authorized |
| 403 | FORBIDDEN | User didn’t have access to resource defined in path parameters |

#### Update

PUT /resources/c291d912-f932-46ca-b7f7-4415cabcf4a3

When updating a resource the calling is expected to initially fetch the specific resource, when do the necessary edits and then return the entire updated resource. Since the caller request body is treated as the new state of the resource, clients should be careful when updating, since omitting parts of the resource will be treated as a removal. Since resources are optimistically locked, the caller must also provide the If-Match precondition header with the version which was modified.

Updates an existing resource identified by ID. Input is the HTTP entity following the following JSON structure:

|  |
| --- |
| {  "id": "c291d912-f932-46ca-b7f7-4415cabcf4a3",  "version": 0,  ...  } |

The service will return the updated entity upon successful update of the entity:

|  |
| --- |
| {  "id": "c291d912-f932-46ca-b7f7-4415cabcf4a3",  "version": 1,  ...  } |

##### Return codes

The HTTP response status codes indicate whether a specific request has been completed:

|  |  |  |
| --- | --- | --- |
| **HTTP Status code** | **HTTP Status** | **Description** |
| 200 | OK | Upon successful update of the resource |
| 400 | BAD REQUEST | The JSON does not conform to the expected structure or the content of the JSON does not validate according to validation rules |
| 401 | UNAUTHORIZED | User is not authorized |
| 403 | FORBIDDEN | User did not have access to resource defined in path parameters |
| 409 | CONFLICT | The resource version mismatched. The resource was updated in mid-air |

##### Precondition headers

The exact resource version must be communicated to the service using the If-Match Request Header, for version verification.

|  |
| --- |
| If-Match: 0 |

Note that the api does accept wildcards or multiple values.

The new resource version then returned upon a successfully update in the response deader ETag:

|  |
| --- |
| ETag: "1" |

#### Patch Update

PATCH /resources/c291d912-f932-46ca-b7f7-4415cabcf4a3

Some resource are also updateable using the PATCH http method. When patch updating, unlike normal update, the client is only expected to provide the part of the resource which they have modified. This can especially be useful when editing large resource that contains lots of information. Like the normal update the caller is also expected to provide the precondition If-Match header.

Updates an existing resource identified by ID. Input is specific fields of the HTTP entity following the below shown JSON structure:

|  |
| --- |
| {  "id": "c291d912-f932-46ca-b7f7-4415cabcf4a3",  "version": 0,  ...  } |

The service will modify the entity on the exposed fields only, leaving the rest unmodified. The service will return the modified entity upon successful update:

|  |
| --- |
| {  "id": "c291d912-f932-46ca-b7f7-4415cabcf4a3",  "version": 1,  ...  } |

##### Return codes

The HTTP response status codes indicate whether a specific request has been completed:

|  |  |  |
| --- | --- | --- |
| **HTTP Status code** | **HTTP Status** | **Description** |
| 200 | OK | Upon successful update of the resource |
| 400 | BAD REQUEST | The JSON does not conform to the expected structure or the content of the JSON does not validate according to validation rules |
| 401 | UNAUTHORIZED | User is not authorized |
| 403 | FORBIDDEN | User didn’t have access to resource defined in path parameters |
| 409 | CONFLICT | The resource version mismatched. The resource was updated in mid-air |

##### Precondition headers

The exact resource version must be communicated to the service using the If-Match Request Header, for version verification.

|  |
| --- |
| If-Match: 0 |

Note that the api does accept wildcards or multiple values.

The new resource version then returned upon a successfully update in the response deader ETag:

|  |
| --- |
| ETag: "1" |

#### Delete

DELETE /resources/c291d912-f932-46ca-b7f7-4415cabcf4a3

Deletes an existing resource identified by ID. Similar to update, the caller is also expected to provide the precondition If-Match header.

##### Return codes

The HTTP response status codes indicate whether a specific request has been completed:

|  |  |  |
| --- | --- | --- |
| **HTTP Status code** | **HTTP Status** | **Description** |
| 204 | NO CONTENT | Upon successful delete |
| 401 | UNAUTHORIZED | User is not authorized |
| 403 | FORBIDDEN | User didn’t have access to resource defined in path parameters |
| 409 | CONFLICT | The resource version mismatched. The resource was updated in mid-air |

# Querying and searching

## Domain names

|  |
| --- |
| Additional information on how to access environments can be found in the section: “Access to environments”. |

This section gives an overview of the different domains that are in Digital Post as well as a description.

Recommended firewall usage, is to dynamically detect IP of the DNS name, so changes to IP are picked up automatically.

External IP are only here for information purpose, and is not expected to be a source of truth.

### QA environment

|  |  |  |
| --- | --- | --- |
| **Hostname** | **External IP** | **Description** |
| <https://admin-qa.test.digitalpost.dk> | 80.198.95.55 | The domain for the Administrative Access portal that is used to configure contact-structure, system-setup, event-log, exemption etc. |
| <https://testportal-qa.test.digitalpost.dk> | 80.198.95.55 | The portal to manage and create test data in Digital Post |
| <https://app-qa.test.digitalpost.dk> | 80.198.95.55 | The portal for managing push-notification and revoking app access |
| <https://api-qa.test.digitalpost.dk/> | 80.198.95.13 | The domain to access the REST API of Digital Post by sender and receiver systems using mutual SSL |
| <https://gateway-qa.test.digitalpost.dk/> | 80.198.95.55 | The domain to access the REST API by view clients (such as [http://borger.dk](http://Borger.dk) ) |
| <http://sftp-qa.test.digitalpost.dk> | 188.64.157.65 | The domain that exposes the SFTP server for sender and receiver systems |

### Test environment

|  |  |  |
| --- | --- | --- |
| **Hostname** | **External IP** | **Description** |
| <https://admin.test.digitalpost.dk> | 80.198.95.45 | The domain for the Administrative Access portal that is used to configure contact-structure, system-setup, event-log, exemption etc. |
| <https://testportal.test.digitalpost.dk> | 80.198.95.45 | The portal to manage and create test data in Digital Post |
| <https://app.test.digitalpost.dk> | 80.198.95.45 | The portal for managing push-notification and revoking app access |
| <https://api.test.digitalpost.dk/> | 80.198.95.10 | The domain to access the REST API of Digital Post by sender and receiver systems using mutual SSL |
| <https://test.digitalpost.dk/> | 80.198.95.45 | The domain to access the REST API by view clients (such as borger.dk) |
| <http://sftp.test.digitalpost.dk> | 80.198.95.42 | The domain that exposes the SFTP server for sender and receiver systems |

### Production

This table contains the domain and IPs for the production environment. Note that production is not yet open for business.

|  |  |  |
| --- | --- | --- |
| **Hostname** | **External API** | **Description** |
| <https://admin.digitalpost.dk> | 80.198.95.24 | The domain for the Administrative Access portal that is used to configure contact-structure, system-setup, event-log, exemption etc. |
| <https://app.digitalpost.dk> | 80.198.95.24 | The portal for managing push-notification and revoking app access |
| <https://api.digitalpost.dk> | 80.198.95.23 | The domain that exposes the REST API of Digital Post by sender and receiver systems using mutual SSL |
| <https://gateway.digitalpost.dk> | 80.198.95.24 | The domain to access the REST API by view clients (such as [http://borger.dk](http://Borger.dk)) |

## Outgoing IP

The IP address used for outgoing traffic from Digital Post til external parties is 80.198.95.62 for both QA, TEST and Production.

## Querying and searching resources

The components that expose an endpoint for querying all offer the querying/searching functionality that is described here.

### Eventually consistent

Resources received when querying (GET) the URL

|  |
| --- |
| /resource-name(in plural)/ |

are only eventually consistent with resources from unsafe methods. This means that after a POST, PUT, PATCH, or DELETE the state change might not be immediately available for this type of querying. GET using ID for fetching a single resource will always be consistent.

### SearchResult

The result of searching will be a resource specific implementation of a SearchResult containing a list of resources and paging information.

Example JSON

|  |
| --- |
| {  "next": "",    "currentPage": 0,    "totalPages": 1,    "elementsOnPage": 1,    "totalElements": 1,    "resource-name(in plural)": []  } |

### Paging

It is possible to paginate the results using the following parameters:

|  |  |
| --- | --- |
| page | Zero-based page index (0..n) |
| size | The size of the page to be returned. If not specified the default page size is 100. The upper limit is 10.000, as set by ElasticSearch |
| next | An encoded string representing the last element on the previous page. Necessary when needing to page through more than 10.000 resources. The number of resources returned on the next page can be adjusted by changing the value of size. |

#### Example

**Using ‘page’ and ‘size’**

|  |
| --- |
| /resource-name(in plural)/?page=1&size=10 |

**Using ‘next’**

In order to utilize next when paginating results it is necessary to a regular query against the endpoint which will then provide a next value that can be used for paging through all resources mathing the request. That is, first we query normally

|  |
| --- |
| /resource-name(in plural)/?page=0&size=100 |

which returns

|  |
| --- |
| {  "currentPage": 0,  "next": "nextValue1"  "totalPages": 100,  "elementsOnPage": 100,  "totalElements": 10000,  "resource-name(in plural)": [  {Resource1},  {Resource2},  ...  {Resource100}  ]  } |

then to get the next page we query using the next parameter instead of specifying page and size as follows

|  |
| --- |
| /resource-name(in plural)/?next=nextValue1 |

which returns

|  |
| --- |
| {  "currentPage": 0,  "next": "nextValue2"  "totalPages": 1000,  "elementsOnPage": 100,  "totalElements": 100000,  "resource-name(in plural)": [  {Resource101},  {Resource102},  ...  {Resource200}  ]  } |

To increase the readability of the above examples, the parameter ‘next' has been simplified.  
A real life example of the value for the parameter 'next’ could look like:  
WyAiMjAyMS0xMi0wOFQwOTo1ODo1Ni45NDkiLCAiRkRKVVg3TE5mbzFFa1ZjUFJsbmFCU3FrTUl6UHFDVXAiIF0=

The ‘next’ value can also be combined with other search parameters

|  |
| --- |
| /resource-name(in plural)/?field1=value1&next=nextValue |

**Important notes when using ‘next’**

When paging through a registry using the next parameter it is important to sort the results in ascending order such that the resources with the newest values, e.g. lastUpdated, are sorted at the very end of the results instead of at the beginning. If you sort values in a non-deterministic way and without using a tiebreaking field in case of duplicate values the returned search result may be inconsistent and resources may be skipped or missed.

Please note that when utilizing the next parameter, it is important to be aware that the value of the currentPage will always be 0, regardless of how many times the next parameter is used.

### Sorting

Sorting can be done using the following parameter

|  |  |
| --- | --- |
| sortFields | One or more fields to sort by. Each field can be appended the desired sort order by separating them using a colon. If a sort order is not provided it will default to ‘asc’ for that particular field. The sort takes place in the order given.  If no sortFields are provided it defaults to a resource specific default sort field and order.  The fields can be a nested field in the resource-structure using dot (.) between the elements. For example *field1.subField1*. |

#### Example

|  |
| --- |
| /resource-name(in plural)/?sortFields=field1.subField1:asc,field2:desc |

This example sorts on subField1 ascending first and where the subField1 of the resources is equal, it goes on to sort on field2 descending.

### Filtering

Filtering can be used to only fetch certain fields like for instance ID and version.

|  |  |
| --- | --- |
| fields | One or more name of the fields you wish to fetch. The fields can be a nested field in the resource structure using dot (.) between the elements. For example *field1.subField1* |

#### Example

|  |
| --- |
| /resource-name(in plural)/?fields=id,version,field1.subField1 |

returns only those fields of each resource:

|  |
| --- |
| {  "currentPage": 0,  "totalPages": 2,  "elementsOnPage": 100,  "totalElements": 130,  "resource-name(in plural)": [  {  "id": "5e3cd399-84df-4fc3-856f-a3bb1fb2a21f",  "version": 3,  "field1": {  "subFields1": "value"  }  },  {  "id": "0a6ea322-7ca0-49f9-93c2-fddce3de2dae",  "version": 11,  "field1": {  "subFields1": "value"  }  },  ...  ]  } |

### Searching

The individual resources may override specific search functionality for a field. In which case it will be documented under that service and have own parameter description in Open Api. Besides the specific search options these are available

|  |  |
| --- | --- |
| any | One or more search terms to search across all fields in the resource |
| Field name from resource | Specify one or more fields from resource each with one or more search terms.  A field can be a nested field in the resource-structure using dot (.) between the elements. For example *field1.subField1*.  If more values are given to a single specific field, the matches will be where either match (OR).  If more search fields are given, the matches will be where all match (AND) - see below override option. |
| Operator on search term | As mentioned above, given more than one search field will by default make sure all different fields MUST match. I.e. an AND is placed between them. This functionality can be overridden by adding the an operator prefixed to the search value. The following is supported:   * |   + OR * &   + AND * !   + NOT   Examples:  /?param=|alfa,|bravo,!charlie - equivalent to /?param=|alfa&param=|bravo&param=!charlie  This operator is currently only available on these generics search parameters, and thus not on the fixed filters the individual endpoint offers. |
| Quoted | A search term may be put in quotes, in which case the it will search for the entire term without doing any attempts to match only part of the term, fuzziness or the like.  “ and ' are both allowed.  The search is still done case insensitive. |

|  |
| --- |
| **Note that there is a limitation of 100 characters on the length of each value used for the search fields** |

#### Examples

**Using a parameter that exceeds 100 characters :**

|  |
| --- |
| /resource-name(in plural)/?message=alpha%20beta%20kappa%20zeta%20eta%20phi%20epsilon%20delta...add%20more%20until%20you%20reach%20100&subject=MESSAGE |

we get the following response

|  |
| --- |
| {  "code": "digital.post.error",  "message": "IllegalArgumentException: message length of (number of characters) exceeds limit of 100",  "fieldErrors": []  } |

**Using multiple parameters that are below 100 characters :**

|  |
| --- |
| /resource-name(in plural)/?message=value1,value&subject=MESSAGE |

**Using ‘any’ field:**

|  |
| --- |
| /resource-name(in plural)/?any=Netcompany&any=Digitaliseringsstyrelsen  /resource-name(in plural)/?any=Netcompany,Digitaliseringsstyrelsen |

searches for ‘Netcompany' OR 'Digitaliseringsstyrelsen’ across all fields in a resource.

**With specific fields:**

|  |
| --- |
| /resource-name(in plural)/?recipient.recipientId=14814833,43720082 |

returns resources where recipientId equals 14814833 OR recipientId equals 43720082.

|  |
| --- |
| /resource-name(in plural)/?recipient.recipientId=14814833&label=Fra%20Digitaliseringsstyrelsen |

returns resources where recipientId equals 14814833 AND label equals ‘Fra Digitaliseringsstyrelsen’.

**Operator:**

|  |
| --- |
| /resource-name(in plural)/?recipient.recipientId=|14814833&label=|Fra%20Digitaliseringsstyrelsen |

returns resources where recipientId equals 14814833 OR label equals ‘Fra Digitaliseringsstyrelsen’

**Using wildcard:**

|  |
| --- |
| /resource-name(in plural)/?field1.subField2=Flytterod\* |

return matches where subfield2 starts with 'Flytterod'.

**Possible wildcards:**

|  |  |
| --- | --- |
| \* | Match zero or more characters, including an empty one |
| ? | Matches any single character |

Wildcard search is only possible when specific field is provided - not using the 'any' search field.

**Using quotations:**

|  |
| --- |
| /resource-name(in plural)/?field1="Københavns Kommune - Borger Service" |

will only return with a full match and thus not return other matches such as “Københavns Kommune - Affald“.

# Contact registry services - TI

Below table shows an overview of all the services that the contact-registry exposes externally. The table also gives a small description of the common usage patterns that the APIs are intended to support as well as an overview of which roles have permissing to call. This overview does not go into details about which contacts the different roles can view or update.

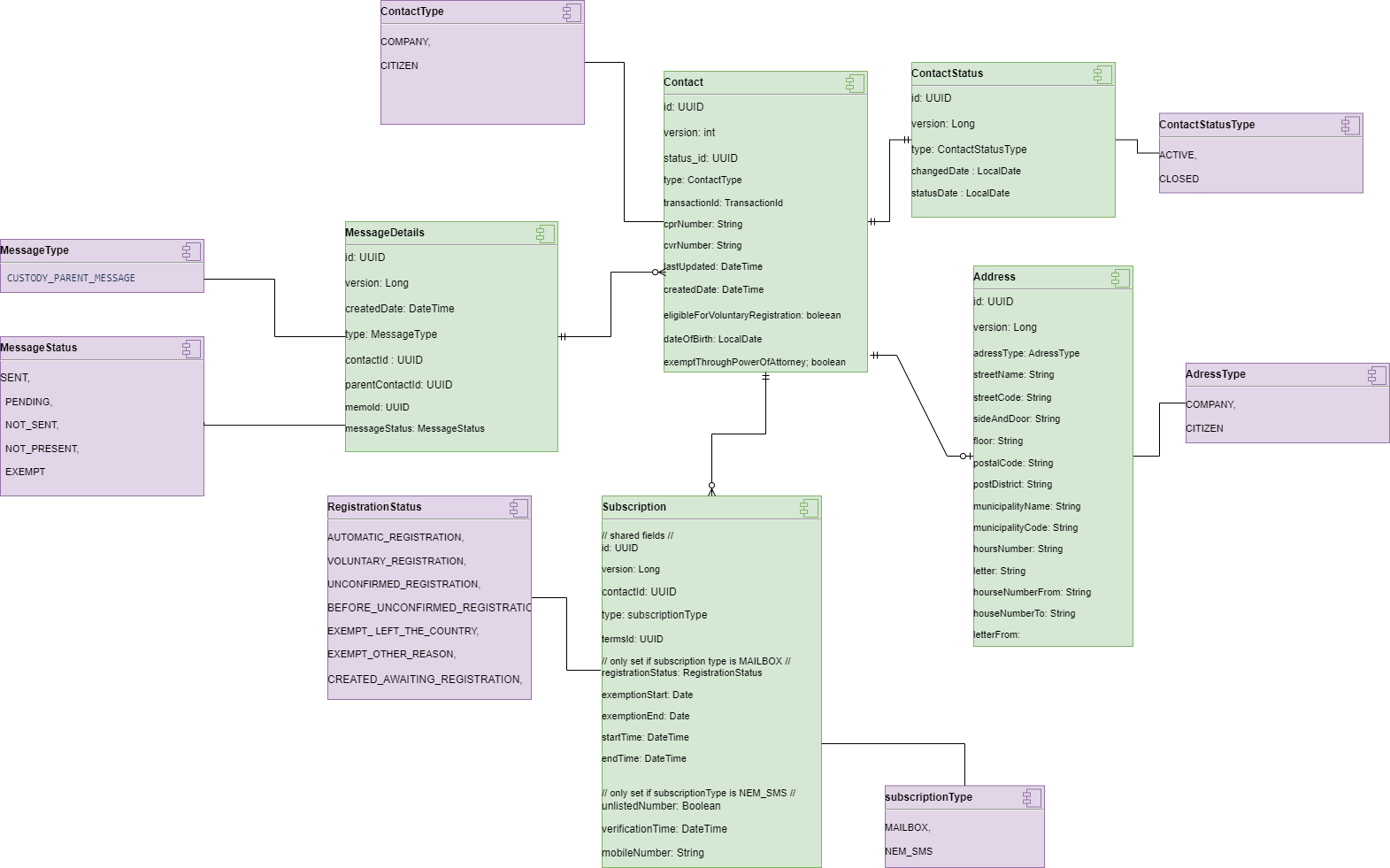
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Service** | **URL** | **Data returned** | **Usage** | **Required roles** |
| Fetch a contact | GET /contacts/{contact-id} | A single contact | Used to fetch the status of a citizens or companies Digital Post subscription and NemSMS number | * Citizen * Citizen service employee * Sender systems * Business service employee * Message employee * Organisation Administrator * DP Full access |
| Query contacts | GET /contacts/ | List of contacts | Search for contacts that matches a set of requirements | * Citizen * Citizen service employee * Sender systems * Business service employee * Message employee * Organisation Administrator * DP Full access |
| Update contacts | PUT /contacts/{contact-id} | The updated contact | Used to update a contact such as subscribing to NemSMS or changing the NemSMS number. | * Citizen * Citizen service employee * Sender systems * Business service employee * Message employee * Organisation Administrator * DP Full access |
| Verify NemSMS subscription | PUT /contacts/{contact-id}/verifications/{nemsms-subscription-id} | A single contact | Used to verify the NemSMS number by providing the received PIN code. Or ordering a new pincode. | * Citizen * Citizen service employee * Sender systems * Business service employee * Message employee * Organisation Administrator * DP Full access |
| Create contact-subscription | POST /contacts/subscriptions/ | The created contact-subscription | Used for sender-systems to create a subscription for changes to contacts | * Sender system |
| Fetch contact-subscription | GET /contact/subscriptions/ | A list of contact-subscriptions | Used by sender-systems to fetch their contact-subscription | * Sender system |
| Update contact-subscription | PUT /contact/subscriptions/{id} | The updated contact-subscription | Used to update the subscription for changes to contacts, such as adding specific contacts to the list or adding general properties for notifications | * Sender system |

## Contact registry data model

### Physical data model for the contact registry

The contact registry contains and exposes information about if and how the public authorities can contact citizens and companies. It uses standard JPA/Hibernate convention and converts camel case to snake case (underscore delimiter) so for instance the entity ContactStatus maps to a CONTACT\_STATUS table.

All tables include Id (internal ID of the resource), Version (how many times the resource have been updated), and possibly a createdDate and lastUpdated. If the table represents a REST ressource that is indexed it will also include a TransactionId (id of the transaction the resource was last updated in) . These fields will not be included in the attributes descriptions.



**Contact**

Person or company that can or could be contacted by public authorities.

|  |  |  |  |
| --- | --- | --- | --- |
| **Column** | **Description** | **Required** | **Comment** |
| id | id of entity | Yes | The primary resource identifier of the Contact |
| version | The current version of the Contact entity | Yes | Used to ensure optimistic locking and is automatically increased by one on each update |
| statusId | Foreign key for ContactStatus (see below) | Yes |  |
| Type | Type of contactable entity | Yes |  |
| TransactionId | Identifier of the transaction of which the entity was last changed | Yes |  |
| cprNumber | External identifier of Contacts of type Citizen | No | Not required, but if present must be unique |
| cvrNumber | External identifier of Contacts of type Company | No | Not required, but if present must be unique |
| lastUpdated | timestamp of when the Contact was last updated | Yes | Automatically updated on each update with the current timestamp. |
| createdDate | timestamp of when the Contact was created | Yes | Automatically set when the contact is created |
| eligibleForVoluntaryRegistration | Flag that indicates if the contact of type citizen is eligible for voluntary registration | Yes | Citizens who are not required to have Digital Post can still sign up for Digital Post on a voluntary basis. This is only relevant for citizen with an administrative cpr numbers such as foreigners working in Denmark |
| dateOfBirth | Date for when the citizen was born | No | Used to calculate when the citizen is coming of age to enter Digital Post |
| exemptThroughPowerOfAttorney | Add flag which indicates whether an exemption is made on the basis of a power of attorney. | No | This is a required field when a citizen or business is exempt. When exemption ceases, the value must not be set (null). |
| addressId | The foreign key for Address (see below) | No | The legal address of both citizens and companies, used when forwarding the mail in the mailbox during exemption |

**Contact Status**

The CONTACT\_STATUS table contains the info about the Contact and the current state. A contact is active while the citizen is alive and the company is still in operation. The status of the contact can be active even when the contact is not subscribed to Digital Post or NemSMS.

|  |  |  |  |
| --- | --- | --- | --- |
| **Column** | **Description** | **Required** | **Comment** |
| id | id of entity | Yes | always set |
| version | The current version of the ContactStatus | Yes | Used to ensure optimistic locking and is automatically increased by one on each update |
| contactStatusType | Whether the contact is active or permanently closed | Yes | A Contact is set to inactive when the citizen deceases or the company is closed |
| changedDate | Timestamp of when the ContactStatus was last updated | Yes | Automatically updated on each update with the current timestamp. |
| statusDate | Date of the latest change of contact’s status based on CVR and CPR integrations. | No | Helps to maintain the criterion of no closed Citizen contacts being closed for longer than 5 Years and Companies for 10. |

**Subscription**

Subscription entity either responsible for both NemSMS and Digital Post registration status.

|  |  |  |  |
| --- | --- | --- | --- |
| **Column** | **Description** | **Required** | **Comment** |
| id | id of subscription | Yes | The primary key of the entity |
| version | The current version of the Subscription | Yes | Used to ensure optimistic locking and is automatically increased by one on each update |
| contactId | foreign key for contact | Yes | A contact can only have one subscription of type mailbox and one of type NemSMS |
| type | The type denoting if this subscription represents a NemSMS subscription or a mailbox subscription | Yes | This value is used to determine which property the subscription is deserialized into |
| ***NemSMS Fields*** | Fields used when the subscriptionType is NemSMS | - | The fields are not required on a database level however on a business logic perspecitve |
| verificationTime | The last time the number was verified | No | The nemSms Number is only verified once |
| confirmedDateTime | The last time the number was confirmed, also updates if the verificationTime is changed, is always equal or newer than verificationTime. | No | Different from verification since confirmedDateTime only depends on the user stating if the number is correct, and not actually testing it. |
| mobileNumber | The SMS number used for receiving NemSMS’es | Yes | Only danish numbers without county code is allowed. Only number following this pattern are allowed: ^[2-9]\\d{7}$ |
| verificationId | The reference to the verification which is used to verify the number | No |  |
| ***Mailbox fields*** | Fields used when the subscriptionType is Mailbox | No | if subscriptionType is not Mailbox these are null |
| RegistrationStatus | The current status for registration of Digital Post for the related Contact. | Yes | Set to Closed if the contact is status closed. |
| exemptionStart | If exempted, when was the exemption started | No | required if registrationStatus is exempts |
| exemptionStop | If exempted, this column noted when the exemption end. This is only relevant for companies, where exemptions are only temporary. Any value for citizens with be rejected or ignored | No | Mandatory for companies and automatically set to november 1. in 2 years from the date of exception. This is always calculated when exemptionStart is changed. |
| startTime | When the mailbox subscription to Digital Post was started |  |  |
| endTime | When the mailbox subscription to Digital Post was ended |  | null if not ended |

**Comment: PublicRegistrationstatus**

PublicRegistrationstatus is a calculated field mapped from RegistrationStatus, and have the following fields;

|  |
| --- |
| PublicRegistrationStatus{  UNKNOWN,  REGISTERED,  EXEMPT,  CLOSED  } |

PublicRegistrationstatus is mapped as follows:

|  |
| --- |
| **switch** (registrationStatus) {  **case** AUTOMATIC\_REGISTRATION:  **case** VOLUNTARY\_REGISTRATION:  **return** PublicRegistrationStatus.REGISTERED;    **case** EXEMPT\_LEFT\_THE\_COUNTRY:  **case** EXEMPT\_OTHER\_REASON:  **case** UNCONFIRMED\_REGISTRATION:  **case** BEFORE\_UNCONFIRMED\_REGISTRATION:  **case** CREATED\_AWAITING\_REGISTRATION:  **return** PublicRegistrationStatus.EXEMPT;    **case** UNKNOWN:  **default**:  **return** PublicRegistrationStatus.UNKNOWN; |

|  |
| --- |
| If the contact Status is set to closed, the public registration status is set to CLOSED always. |

**Address**

The table which contains the legal address for citizens and companies. The address is synchronized with data from Datafordeleren which provides the legal address of both companies and citizens. The address is an internally used datapoint and currently only used during the forwarding of exiting mail during exemption and not exposed externally.

|  |  |  |  |
| --- | --- | --- | --- |
| **Column** | **Description** | **Required** | **Comment** |
| id | id of entity | Yes | The primary key |
| version | version | Yes | 0 or above |
| addressType | The type of address, can be COMPANY or CITIZEN | Yes |  |
| streetName | name of street | No | The name of the street as defined by Dawa |
| streetCode | The street code that unique identifies street in the DAWA. | No | Also know as vejkode |
| sideAndDoor | The side and door of the address | No |  |
| floor | The floor of address | No |  |
| postalCode | The post district code | No | If present is always 4 numbers |
| postDistrict | The name of the post district | No |  |
| municipalityName | name of municipality | No | Example: Københavns Kommune |
| municipalityCode | code of municipality | No | Always 4 numbers. Example 0101 for The municipality of Copenhagen |
| **Contact.*Address* (CitizenAddress)** | class that extends abstract class Address, responsible for citizen address |  |  |
| houseNumber | number of house | No |  |
| letter | letter assigned to house | No |  |
| **Contact.*Address* (CompanyAddress)** | class that extends abstract class Address, responsible for company address |  |  |
| houseNumberFrom | from house number | No |  |
| houseNumberTo | to house number | No |  |
| letterFrom | from letter | No |  |
| letterTo | to letter | No |  |
| careOfName | C/O name | No |  |

**MessageDetails**

The table with information about the messages that should be sent to the parents of children soon to be turning 15.

|  |  |  |  |
| --- | --- | --- | --- |
| **Column** | **Description** | **Required** | **Comment** |
| id | id of entity | Yes | The primary key |
| version | version | Yes | 0 or above |
| createdDate | The date when the Custody Parent message was created | Yes |  |
| type | type of the message |  |  |
| contactId | the id of the child that the parent is receiving a message about | Yes |  |
| parentContactId | the id of the parent | No |  |
| memoId | the id of a memo that should be sent | No |  |
| messageStatus | status of the message, can be : sent , not\_sent or exempt in case the parent is not registered in Digital Post | No |  |

### Querying Contacts

For description of common search functionality in Digital Post, please revisit the section [**Querying and searching resources**](https://digidp.atlassian.net/wiki/spaces/DP/pages/685670548/Querying+and+searching+resources).

Querying contacts are done using a GET request to the /contacts/ endpoint.

The result is a ContactSearchResult, which looks like this in JSON

|  |
| --- |
| {    "currentPage": 0,  "next": "string",    "totalPages": 1,    "elementsOnPage": 1,    "totalElements": 1,    "contacts": [...]  } |

Fetching a specific contact can be done as a GET request to the /contacts/{id} endpoint. The result is the contact with the specified ID. The definition of a Contact can be found the OpenAPI specification, which also includes a short description of all fields and types used in the Contact registry. Please consult the [OpenAPI description](https://digidp.atlassian.net/wiki/spaces/DP/pages/757497917/OpenAPI+description) section for details.

## Searching

Besides the general functionality described above, the Contact registry overrides and offers search using the following parameters

|  |  |
| --- | --- |
| **Field** | **Description** |
| lastUpdated | Get all contacts that have been updated between the provided timestamp and now. Note that the timestamp should be provided in UTC time following ISO8601. Note that unlike the other parameters, only **one** lastUpdated parameter can be searched for |
| createdDate | Get all contacts that have been created after the provided timestamp. |
| isBulkLookup | Indicator used to specify the search parameters in the request body instead of query parameters |

### Examples

Generally the format is

|  |
| --- |
| /contacts/?<parameter>=<value>  /contacts/?<parameter>=<value>,<value>,<value>  /contacts/?<parameter>=<value>&<parameter>=<value>&<parameter>=<value> |

It’s possible to mix-and-match different parameters. For lines 2 and 3 it’s possible to add as many parameters/values as desired.

#### Searching with CPR

A common use case of the contact registry is searching for a Contact with a specific CPR number. This can be done using the following method

|  |
| --- |
| GET https://api.digitalpost.dk/apis/v1/contacts/?cprNumber=1234567890 |

Which will return a search result which contains all Contacts with that cprNumber. This will of course always be either 0 or 1 results (note that 1234567890 is a fictive cpr number). It is also possible to search after contacts using multiple CPR numbers

|  |
| --- |
| /contacts/?cprNumber=1234567890,0987654321  /contacts/?cprNumber=1234567890&cprNumber=0987654321 |

return matches where cprNumber equals 1234567890 or 0987654321

#### With searchField lastUpdated

|  |
| --- |
| /contacts/?lastUpdated=2020-04-27T11:01:43.652Z |

return matches where lastUpdated is later than the 11:01:43 the 27th April 2020 (UTC timezone).

## Using the body of the GET request

It is possible to query contacts while providing the parameters inside a request body. This functionality is provided to support filtering for a larger number of contacts in a single request. In order to provide a uniform API there is a limit to the number of combined cvrNumbers and cprNumbers which can be provided by the caller. The limit is currently 10.000 to match the number of search results provided, however this value can be changed to ensure performance in the API. It is currently only cvrNumbers and cprNumbers which are counted towards the request limit.

To query the contact registry using the request body the query parameter isBulkLookup must be used and set to true as shown

|  |
| --- |
| GET https://api.digitalpost.dk/apis/v1/contacts/?isBulkLookup=true |

and the remaining query parameters are stored in the body

|  |
| --- |
| {  "cvrNumber": ["43585118","43585045","30826191"]  } |

Note that parameter value is an array of strings and not a comma-separated string as is the case when querying using query parameters. Using the bulk query functionality supports the same filtering functionality as when using query parameters, however the structure is slightly different. For example, say you want to search for a list of CPR numbers, but only include those that are registered for Digital Post using the public registration status the request body should be

|  |
| --- |
| {  "cprNumber": ["cpr1", "cpr2"],  "mailboxSubscription": {  "publicRegistrationStatus": ["REGISTERED"]  }  } |

That is, the structure of the request body follows the structure of the Contact model, but all values for fields from the Contact model should be given as an array of string values. It is also possible to page through the results by specifying page, size, and next in the request body

|  |
| --- |
| {  "page": 3,  "size": 10,  "cprNumber": ["cpr1", "cpr2", ..., "cpr1000"],  "mailboxSubscription": {  "publicRegistrationStatus": ["REGISTERED"]  }  } |

When the isBulkLookup query parameter is equal to anything other than true it will be considered false and the request body ignored. Furthermore, when performing a bulk query you cannot specify any other query parameters and if done a HTTP 400 bad request will be returned

|  |
| --- |
| -- request  GET https://api.digitalpost.dk/apis/v1/contacts/?isBulkLookup=true&cprNumber=1234567890    -- response  {  "code": "digital.post.error",  "message": "ValidationException: Validation failed",  "fieldErrors": [  {  "resource": "target",  "field": "bulkLookup",  "code": "invalid.bulk.search",  "rejectedValue": **true**  }  ]  } |

Setting the bulk parameter to true and not providing the body will also be regarded as a bad request.

### Subscribing to NemSMS

One of the common use cases for both citizens, sender-systems and citizen service employees is subscribing a Contact to NemSMS. To perform this action we first have to find the relevant contact, depending on the user this can be done multiple ways. To simplify this example we assume the caller is a sender-system in an authority, which have access to the entire dataset of contacts. Remember to always consult the OpenAPI definition to see endpoints, parameters and resource types.

### Querying the contact

Let us assume that we want to find the contact for a citizen, since we as the sender-system know that his cpr number is 1111111234 we can go ahead and search for that cprNumber;

|  |
| --- |
| GET https://api.digitalpost.dk/apis/v1/contacts/?cprNumber=1111111234 |

Which gives us the following response;

|  |
| --- |
| {  "currentPage": 0,  "totalPages": 1,  "elementsOnPage": 1,  "totalElements": 1,  "contacts": [  {  "id": "c69912c4-11d5-4c62-97e6-79fcb1d7b99d",  "version": 9,  "type": "CITIZEN",  "transactionId": "F640l9yALrjfHfASwcvYC0o5TUP7hjNZ",  "cprNumber": "1111111234",  "mailboxSubscription": {  "id": "e19bbf71-587a-46a5-8392-dfaf7fd50a6a",  "version": 9,  "publicRegistrationStatus": "EXEMPT",  "exemptionStart": "2021-07-15",  "startTime": "2021-05-21T07:01:48.370Z"  },  "lastUpdated": "2021-07-15T15:06:50.004Z",  "status": {  "id": "7abe9012-435c-457c-84aa-170ec83d275f",  "version": 0,  "type": "ACTIVE",  "changedDate": "2021-05-21"  },  "eligibleForVoluntaryRegistration": **false**  }  ]  } |

### Fetching the contact

Since Digital Post only promises eventual consistency, the search index and the persistence store might be out of sync when we query. We therefore must also fetch the Contact before editing;

|  |
| --- |
| GET https://api.digitalpost.dk/apis/v1/contacts/c69912c4-11d5-4c62-97e6-79fcb1d7b99d |

Which then gives the following result;

|  |
| --- |
| {  "id": "c69912c4-11d5-4c62-97e6-79fcb1d7b99d",  "version": 9,  "type": "CITIZEN",  "transactionId": "F640l9yALrjfHfASwcvYC0o5TUP7hjNZ",  "cprNumber": "1111111234",  "mailboxSubscription": {  "id": "e19bbf71-587a-46a5-8392-dfaf7fd50a6a",  "version": 9,  "publicRegistrationStatus": "EXEMPT",  "exemptionStart": "2021-07-15",  "startTime": "2021-05-21T07:01:48.370Z"  },  "lastUpdated": "2021-07-15T15:06:50.004Z",  "status": {  "id": "7abe9012-435c-457c-84aa-170ec83d275f",  "version": 0,  "type": "ACTIVE",  "changedDate": "2021-05-21"  },  "eligibleForVoluntaryRegistration": **false**  } |

### Updating the contact

Now that we have fetched the Contact we can go a head and subscribe him to NemSMS. We do that by adding the nemSmsSubscription to the JSON structure, along with the phone number that should be subscribed. For the sake of the example we use 20202020;

|  |
| --- |
| {  "id": "c69912c4-11d5-4c62-97e6-79fcb1d7b99d",  "version": 9,  "type": "CITIZEN",  "transactionId": "F640l9yALrjfHfASwcvYC0o5TUP7hjNZ",  "cprNumber": "1111111234",  "mailboxSubscription": {  "id": "e19bbf71-587a-46a5-8392-dfaf7fd50a6a",  "version": 9,  "publicRegistrationStatus": "EXEMPT",  "exemptionStart": "2021-07-15",  "startTime": "2021-05-21T07:01:48.370Z"  },  "nemSmsSubscription": {  "mobileNumber": "20202020"  },  "lastUpdated": "2021-07-15T15:06:50.004Z",  "status": {  "id": "7abe9012-435c-457c-84aa-170ec83d275f",  "version": 0,  "type": "ACTIVE",  "changedDate": "2021-05-21"  },  "eligibleForVoluntaryRegistration": **false**  } |

We can then update the Contact with above context using this request;

|  |
| --- |
| PUT https://api.digitalpost.dk/apis/v1/contacts/c69912c4-11d5-4c62-97e6-79fcb1d7b99d  If-Match: 9  Contect-Type: application/json |

Note that we must always provide the precondition header If-Match to ensure that no-one made any changes to the Contact while we where editing. Since the API compares the current state to our request and figures out what changed.

If the request completes successfully we get the updated Contact in the response body;

|  |
| --- |
| {  "id": "c69912c4-11d5-4c62-97e6-79fcb1d7b99d",  "version": 10,  "type": "CITIZEN",  "transactionId": "F640l9yALrjfHfASwcvYC0o5TUP7hjNZ",  "cprNumber": "1111111234",  "mailboxSubscription": {  "id": "e19bbf71-587a-46a5-8392-dfaf7fd50a6a",  "version": 9,  "publicRegistrationStatus": "EXEMPT",  "exemptionStart": "2021-07-15",  "startTime": "2021-05-21T07:01:48.370Z"  },  "nemSmsSubscription": {  "id": "21bc5b76-d914-4da0-8895-ab3d725183af",  "version": 0,  "verificationTime": **null**,  "confirmedDateTime": **null**,  "mobileNumber": "20202020"  },  "lastUpdated": "2021-07-20T10:00:50.004Z",  "status": {  "id": "7abe9012-435c-457c-84aa-170ec83d275f",  "version": 0,  "type": "ACTIVE",  "changedDate": "2021-05-21"  },  "eligibleForVoluntaryRegistration": **false**  } |

Note how the version of the Contact have now increased by one, and the nemSmsSubscription have been assigned an id and a version. However, before the Contact can be contacted using the NemSMS he first must verify that he owns the number. We can see that currently both the confirmedDateTime and the verificationTime are null, meaning that he have yet to perform the verification.

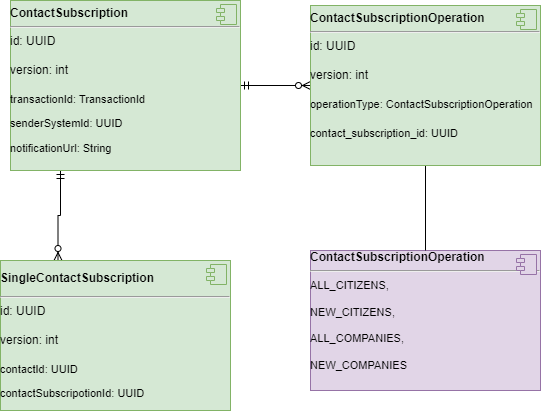
The contact registry will send an SMS to the number that was used, with either a pin code or a link. The pin code is used when it is the citizen or an employee in the company is doing the registration. Whereas the link will be used when assisted signup is done (so when a citizen service employee or a sender system is doing the signup on behalf of the citizen or company).

If the contact is already signed up for NemSMS but we want to change the number, we must simply replace the entire nemSmsSubscription similar to how the initial signup is done. And the verification process will again start over.

Additionally, if the Contact is already using the same number elsewhere in Digital Post, e.g. for notification when he receives new messages in the mailbox, he will not have to verify the number again. Instead, when we do the signup both the verificationTime and confirmedDateTime will be filled with the same data indicating that the number was already verified.

### Contact Subscription

The purpose of the contact subscription is to persist subscriptions for sendersystems, on either a set of specific citizens, organisations or all changes fitting a category. When a change to the Contact is made the subscription components identifies all the matching criteria of a subscription, and notifies the sendersystem on all the matching subscriptions.



**Contact Subscription**

The ContactSubscription is the anchor of the subscription for any given sendersystem. It contains the notification endpoint and the reference to the sendersystem.

|  |  |  |  |
| --- | --- | --- | --- |
| **Column** | **Description** | **Required** | **Comment** |
| id | The identifier of the subscription | Yes |  |
| version | The current version of the subscription resource | Yes | Incremented on each update |
| transactionId | The identifier of the transaction of which the subscription was either created or last updated | Yes | Added or updated by the application on create or update |
| senderSystemId | The uuid of the sendersystem taken from the access token which for calling sendersystem | Yes | The access token replaces the mutual SSL connection during successful authentication in the API gateway |
| notificationUrl | The URL of which the sendersystem is notified on every relevant change | No | The URL must be https. The subscription is deemed inactive if the URL is not present |

**Contact Subscription Operation**

The ContactSubscriptionOperation table contains all the operations which the sendersystem is subscribed to. The operations are bulk subscription to enable the sendersystem to get notified for all changes or any new entities. These can be used in tandem with explicit CPR and CVR subscriptions.

|  |  |  |  |
| --- | --- | --- | --- |
| **Column** | **Description** | **Required** | **Comment** |
| id | The identifier of the subscription | Yes |  |
| version | The current version of the subscription operation resource | Yes | Incremented on each update |
| operationType | The type of bulk operations that are contained in the subscription | Yes | The values which the column can be;   |  | | --- | | ALL\_CITIZENS  NEW\_CITIZENS  ALL\_COMPANIES  NEW\_COMPANIES | |
| contactSubscriptionId | The foreign key to [Contact Subscription](#scroll-bookmark-87) | Yes |  |

**Single Contact Subscription**

The SingleContactSubscription table contains all the contact ids which are explicitly defined by the sendersystem on its Contact Subscription.

|  |  |  |  |
| --- | --- | --- | --- |
| **Column** | **Description** | **Required** | **Comment** |
| id | The identifier of the subscription | Yes |  |
| version | The current version of the subscription resource | Yes | Incremented on each update |
| contactId | The contact id that the sendersystem has an explicit subscription for | Yes | The format should always follow the pattern: ^\d{10}$ |
| contactSubscriptionId | The foreign key to [Contact Subscription](#scroll-bookmark-87) | Yes |  |

### Using the contact subscriptions

### Contact Subscription

In order to maintain local copy of the contact-registry or have own registration lists updated with the newest contact data, the authorities can subscribe to changes to contacts registrations.

Authority sender system will as a part of the subscription specify an endpoint were they will get notified every time there is an update to a contact that are included in their subscription.

An authority can only have one subscription per sender system.

The subscription can either be made on individual contacts or by the usages of subscriptionOperators.

A subscription on individual contacts will contain contactIDs of all the specific contact the subscriber want to be notified about. The amount of contactIDs should be kept on an minimum, however there is no maximum limit. If the authority sender system tries to provide a contactID that do not exist in Digital Post contact registry, the contact will be filtered out of the subscription (Nb. No error will be thrown).

SubscriptionOperators are some pre-defined filters for common use cases. There is a total of four operator:

* ALL\_CITIZENS - The operator includes all updates to already existing citizens in the contact-registry.
* NEW\_CITIZENS - The operator includes all citizens created in the contact-registry.
* ALL\_COMPANIES - The operator includes all updates to already existing companies in the contact-registry.
* NEW\_COMPANIES - The operator includes all citizens created in the contact-registry.

The operators can be used individually or several in the same subscription.

### Endpoint exposed in the contact-subscription-store

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Service** | **URL** | **Data returned** | **Usage** | **Required roles** |
| Create subscription | POST  /contacts/subscriptions/ | created subscription | Creating a new subscription to get notified when organisations changed. Only one subscription per sender system. | AUTHORITY\_SENDER\_SYSTEM |
| List subscriptions | GET  /contacts/subscriptions/ | List of all subscriptions | Listing all subscriptions | AUTHORITY\_SENDER\_SYSTEM |
| Update subscription | PUT  /contacts/subscriptions/{subscription-id} | Updated subscription | Updating a subscription | AUTHORITY\_SENDER\_SYSTEM |
| Delete subscription | DELETE  /contacts/subscriptions/{subscription-id} |  | Deleting a subscription on the subscription ID with an “if-match” header matching the version | AUTHORITY\_SENDER\_SYSTEM |

Example of a subscription creation with usage of individual contactIDs

|  |
| --- |
| {  "singleContactSubscriptions":[  "1e7ad5a8-f1a9-454a-a171-947b56737bd7",  "fab79010-ab67-41e2-8936-d3b9c726cc84",  "193e93d7-9d64-4826-85e4-9851e3b45664",  "a9accf48-3d7a-4057-9c71-387df6d79000",  "1100f22e-0efc-4b97-8948-6588112179f9",  "1ce50ba0-f290-40fd-90b6-3c1e9717b3f7",  "521f3356-d1c5-4812-bc69-59423fce52b7"  ],  "notificationUrl": "https://postman-echo.com/post"  } |

Example of a subscription creation with usage of subscription operators:

|  |
| --- |
| {  "subscriptionOperations": [  "ALL\_CITIZENS"  ],  "notificationUrl": "https://postman-echo.com/post"  } |

### Notifications

A push notification will be sent to the specified endpoint every time a contact is updated or created, and which matches the criteria from the subscription. The notification contain the id and the version of the contact.

Example of Notification:

|  |
| --- |
| {  "id": "521f3356-d1c5-4812-bc69-59423fce52b7"  "version": "1"  } |

# System registry services - TI

The systems registry is responsible for four logical domains in Digital Post:

* Organisation: Used to keep track of which companies are authorities, their authority type, their logo as well as who is allowed to send legal and mandatory messages
* System: The system resource is a representation of connected sender and receiver systems, their technical details such as IP’s service protocol and API token
* Contact structure: The contact structure is the structure used to define how the authority can be contacted and how the messages are directed to different departments inside the authority. It consist of the ContactPoint and ContactGroup resources, which together define the contact structure
* Contact structure subscription: A possibility for sender systems to subscript to changes to all or some contact structures

These logical domains are editable accessible through the following services which all are exposed by the system registry.

## Organisation

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Service** | **URL** | **Data returned** | **Usage** | **Required roles** | **Consumer** |
| Query organisations | GET  /organisations/ | List of organisations | Fetching one or multiple organisations by CVR number, name, type and searchTerm | 1. System administrator 2. Contact administrator 3. System distribution 4. Message write 5. System Subscription 6. Delegated Support Administrator | 1. Distribution 2. View client 3. Administrative Access 4. Sender systems |
| Fetch organisation | GET  /organisations/{organisation-id} | Organisation | Fetching a single organisation by organisationId | 1. Citizen 2. System administrator 3. Contact administrator 4. System distribution 5. Message write 6. System Subscription 7. System Registry 8. System Transformation 9. Delegated Support Administrator | 1. Administrative Access 2. View client |
| Update organisation | PUT  /organisations/{organisation-id} | Organisation | Updating an organisation | 1. System administrator 2. Contact administrator | Administrative Access |
| Delete organisation | *Internal* | Return code | Deleting an organisation | 1. A system role | Administrative Access |
| Add logo content to organisation | PUT /organisations/{id}/logo | Id and version of logo | Adding logo to authorities | 1. DP Virksomhedsadministrator 2. System manager | Administrative Access |
| Get logo content | GET /organisations/{id}/logo | PNG bytes | Getting logo byte content | 1. Any role | 1. Administrative access 2. View client |
| Delete logo | DELETE /organisations/{id}/logo | 204 | Removed logo from organisation | 1. DP Virksomhedsadministrator 2. System manager | Administrative Access |

## Systems

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Service** | **URL** | **Data returned** | **Usage** | **Required roles** | **Consumer** |
| List Systems | GET  /organisations/{organisation-id}/systems/ | List of systems | Fetching all systems from an organisation, that the user is allowed to view | 1. System administrator 2. Contact administrator 3. System distribution 4. System Subscription 5. Delegated Support Administrator | 1. Distribution 2. Administrative Access |
| Add a system | POST  /organisations/{organisation-id}/systems/ | System | Add a system to an organisation | 1. Organisation Administrator 2. System Manager 3. System System Registry | 1. Administrative Access |
| Remove a system | DELETE  /organisations/{organisation-id}/systems/{id} | Void | Remove a system from an organisation | 1. Organisation Administrator 2. System Manager 3. System System Registry | 1. Administrative Access |
| Change default Recipient | POST  /organisations/{organisation-id}/systems/{id}/makedefaultrecipient | System | Change Default recipient system, this action will mark the target system as RECIPIENT\_DEFAULT and remove the marking on the existing | 1. Organisation Administrator 2. System Manager 3. System System Registry | 1. Administrative Access |
| Renew API Token | POST  /organisations/{organisation-id}/systems/{id}/renewapitoken | System | Renew API token. Create new API token if not already existed or Update API token value if not existed. Existing OCES certificate will be removed | 1. Organisation Administrator 2. System Manager 3. System System Registry | 1. Administrative Access |
| Fetch system | GET  /organisations/{organisation-id}/systems/{system-id} | System | Fetching system by ID | 1. Kontakt Administrator 2. DP Virksomhedsadministrator 3. DP Skriver 4. System Distribution 5. System Mailbox 6. System Subscription 7. System Systemregister 8. System transformation 9. Delegated Support Administrator | Administrative Access |
| Update system | PUT  /organisations/{organisation-id}/systems/{system-id} | System | Updating a system by Id | 1. Kontakt Administrator 2. DP Virksomhedsadministrator 3. DP Skriver 4. System Distribution 5. System Mailbox 6. System Subscription 7. System Systemregister 8. System transformation | Administrative Access |
| Upload certificate for system | POST  /organisations/{organisation-id}/systems/{system-id}/certificate | System | Creating/updating certificate for system | 1. Kontakt Administrator 2. DP Virksomhedsadministrator 3. DP Skriver 4. System Distribution 5. System Mailbox 6. System Subscription 7. System Systemregister 8. System transformation | Administrative Access |
| Upload SSH-key for system | POST  /organisations/{organisation-id}/systems/{system-id}/sshkey | System | Creating/updating SSH-key for system | 1. Kontakt Administrator 2. DP Virksomhedsadministrator 3. DP Skriver 4. System Distribution 5. System Mailbox 6. System Subscription 7. System Systemregister 8. System transformation | Administrative Access |
| Check certificate | POST  /organisations/oces-public-certificate/check/{protocolType} | OcesPublicCertificateCheckResult:   * CVR number * OcesPublicCertificate | Checks that the certificate is valid | 1. Organisation Administrator 2. System Manager 3. System System Registry | 1. Administrative access |

## Contact structure

The contact structure consist of the ContactPoint and ContactGroup resource. The structure can be read anonymously since users should be able to identity where to contact an authority before sending the message, which requires the an authorized user.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Service** | **URL** | **Data returned** | **Usage** | **Required roles** | **Consumer** |
| Query contact points | GET  /organisations/{organisation-id}/contact-points/ | List of contact points | Fetching some or all contact points | 1. Any role 2. Anonymous access | 1. Administrative Access 2. View clients 3. Sender systems |
| Create a contact point | POST  /organisations/{organisation-id}/contact-points/ | Contact point | Add a contact point | 1. Contact Administrator 2. System Manager 3. System System Registry | 1. Administrative Access |
| Delete contact point | DELETE  /organisations/{organisation-id}/contact-points/{id} | Void | Delete a contact point given the ID | 1. Contact Administrator 2. System Manager 3. System System Registry | 1. Administrative Access |
| Query contact groups | GET  /organisations/{organisation-id}/contact-groups/ | List of contact groups | Fetching some or all contact groups | 1. Any role 2. Anonymous access | 1. Administrative Access 2. View clients 3. Sender systems |
| Create a contact group | POST  /organisations/{organisation-id}/contact-groups/ | Contact group | Add a contact group | 1. Contact Administrator 2. System Manager 3. System System Registry | Administrative Access |
| Delete contact group | DELETE  /organisations/{organisation-id}/contact-groups/{id} | Void | Delete a contact group given the ID | 1. Contact Administrator 2. System Manager 3. System System Registry | Administrative Access |
| Fetch contact point | GET  /organisations/{organisation-id}/contact-points/{contact-point-id} | Contact point | Fetching contact point by ID | 1. Any role 2. Anonymous access | Administrative Access |
| Update contact point | PUT  /organisations/{organisation-id}/contact-points/{contact-point-id} | Contact point | Updating a contact point by ID | 1. Kontakt Administrator 2. DP Virksomhedsadministrator 3. DP Skriver 4. System Distribution 5. System Mailbox 6. System Subscription 7. System Systemregister 8. System transformation | Administrative Access |
| Fetch contact group | GET  /organisations/{organisation-id}/contact-groups/{contact-group-id} | Contact group | Fetching contact group by ID | 1. Any role 2. Anonymous access | Administrative Access |
| Update contact group | PUT  /organisations/{organisation-id}/contact-groups/{contact-group-id} | Contact group | Updating a contact group by ID | 1. Kontakt Administrator 2. DP Virksomhedsadministrator 3. DP Skriver 4. System Distribution 5. System Mailbox 6. System Subscription 7. System Systemregister 8. System transformation | Administrative Access |
| Query contact groups across organisations | GET /contact-groups/ | List of Contact Groups | Finding a contact groups without knowing which organisation it belongs to | 1. Any role 2. Anonymous access | 1. Administrative Access 2. View clients |
| Query contact points across organisations | GET /contact-points/ | List of Contact Points | Finding a contact points without knowing which organisation it belongs to | 1. Any role 2. Anonymous access | 1. Administrative Access 2. Transformation |
| Query list of possible postkasseIds for organisation | GET /organisations/{organisation-id}/postkasse-ids/ | List of posktasseIds | To find list of possible postkasseIds per organisation | 1. System Systemregister 2. Contact Administrator 3. System manager 4. Delegated Support Administrator | 1. Administrative Access 2. System registry |
| Query list of possible postkasseEmneIds for organisation and postkasseId | GET /organisations/{organisation-id}/postkasse-ids/{postkasseId}/postkasse-emne-ids/ | List of posktasseEmneIds | To find list of possible postkasseEmneIds per organisation and postkasseId | 1. System Systemregister 2. Contact Administrator 3. System manager 4. Delegated Support Administrator | 1. Administrative Access 2. System registry |

## System registry data model

The role of the system registry is to keep data for integrated sender- and receiver-systems for Danish authorities and companies, as well as for authorities to handle their contact hierarchy.

The system registry datamodel can be abstracted into three different submodels.

* Organisation
* System
* Contact Structure

The organisation part keeps the information about what type the organisation is and who the contact person in Digital post is for it.

The system part stores the information about the system or systems connected to an organisation.

The contact structure part, stores the contact structure, what contact points are connected to what system, and how are they grouped and exposed.

### Organisation Model



**Organisation**

The companies and authorities with system integration to Digital Post.

|  |  |  |  |
| --- | --- | --- | --- |
| **Column** | **Description** | **Required** | **Comment** |
| id | primary key | Yes |  |
| version | version | Yes |  |
| TransactionId | Id of the transaction responsible for current state | Yes |  |
| createdDate | date resource was created | No |  |
| authorityId | Placeholder for authorityId when that is relevant | No |  |
| name | The name of the organisation | No |  |
| cvrNumber | The cvrNumber of the organisation | No |  |
| type | Either public authority or company | No |  |
| authorityType | STATE, MUNICIPALITY, REGION, OTHER | No |  |
| legalNotificationAllowed | If the authority is allowed to send legal notifications | No | Only Domstolsstyrelsen is currently allowed to do so |
| mandatoryPostAllowed | If the authority is allowed to send mandatory post | No |  |
| legalContactId | The person responsible for the Digital Post send by the authority | No | Foreign Key |
| logoAvailable | If a logo has been added | No |  |
| logoMediaType | Media type of logo | No |  |
| authorityTerms | If the authority has accepted the terms to act as an authority. Will change type from company to authority |  |  |
| industryCode | “branchekode” in virk registry. e.g. 841100 for generral public services or 021000 for growing trees. |  |  |

**OrganisationDbTargets**

|  |  |  |  |
| --- | --- | --- | --- |
| **Column** | **Description** | **Required** | **Comment** |
| organisation\_db\_id | foreign key to organisation | Yes |  |
| targets | TargetType | Yes |  |

**Person**

Either a juridical or technical person to contact in case of an issue.

|  |  |  |  |
| --- | --- | --- | --- |
| **Column** | **Description** | **Required** | **Comment** |
| id | primary key | Yes |  |
| version | version | Yes |  |
| name | The name of the contact person | No |  |
| email | The email used to contact the person | No |  |
| phonenumber | The phone number used to contact the person | No |  |

### System Model



**System**

The system that integrates to Digital Post. Receiving systems can be associated with contact point and will receive mails for that contact point.

|  |  |  |  |
| --- | --- | --- | --- |
| **Column** | **Description** | **Required** | **Comment** |
| id | primary key | Yes |  |
| version | version | Yes |  |
| TransactionId | Id of the transaction responsible for current state | Yes |  |
| createdDate | date resource was created | no |  |
| name | The name of the system | No |  |
| organisationId | foreign key of the organisation the system belongs to | No |  |
| endpoint | The endpoint used for exchanging messages | No |  |
| receiptEndpoint | The endpoint used for exchanging receipt | No | URL endpoint for receipts |
| activeFrom | The date where the system is/was activated | No |  |
| activeTo | The date where the system is/was deactivated | No |  |
| technicalContactId | foreign key of the person to contact in case of any technical issues | No |  |
| serviceProtocol | The protocol of the system | No |  |
| standardSystemTemplateId | foreign key of a template for systems integrating to Digital Post, e.g. a municipality system provided by an IT vendor that multiple municipalities uses | No | If a standard system is selected many fields will be preset in the frontend, and cannot be set on this object |
| receiptFormat | Field for indicating the format of the receipt (MeMo, DP, DP2) which is being sent to the system | No | MEMO, DP, DP2 |
| supplier | The supplier of the system | No |  |
| businessContactEmail | Email of the business contact of the system | No |  |
| delegatedCVR | A CVR number of an organization to which this system is delegated | No |  |
| ocesPublicCertificateId | foreign key to oces certificate. Described in own table. | no |  |
| sshId | foreign key to ssh. Described in own table. | no |  |
| apiTokenId | foreign key to apiToken. Described in own table. | No |  |
| defaultMaterialId | Only occurs for Authority systems. Filed is mapped to a material of a messages used in legacy solution for DP/DP2 format. | No | Legacy support |

**ApiToken**

Used for identifying the system in mutual SSL, to replace support for identifying system based on certificate serial number. Relates to [Ensuring the authenticity of Digital Posts through the OCES certificate.](https://digidp.atlassian.net/wiki/spaces/DP/pages/778011214)

|  |  |  |  |
| --- | --- | --- | --- |
| **Column** | **Description** | **Required** | **Comment** |
| id | primary key | Yes |  |
| version | version | Yes |  |
| lastUpdated | when the object was last changed | No |  |
| value | Value of the API Token | Yes |  |
| authenticationToken | Non-persistent field, the value is generated. | No | The format is as follows:  Basic BASE\_64\_ENCODING(<system\_id>:<apiToken\_value>) |

**OcesPublicCertificate**

The certificate linked to the system, the certificate itself is not stored, only the sha1 and md1 hash.

|  |  |  |  |
| --- | --- | --- | --- |
| **Column** | **Description** | **Required** | **Comment** |
| id | primary key | Yes |  |
| version | version | Yes |  |
| subjectDistinguishedNameSerialNumber | The distinguished name, as a string, format depends on if certificate is FOCES or VOCES | No |  |
| certificateSerialNumber | The serial number of the certificate | No |  |
| certificateAutority | The authority of the certificate, issuer of certificate chain | No |  |
| expiryDate | When the certificate expires | No |  |
| sha1 | sha1 hash calculated when consuming a x509 certificate | No |  |
| md5 | md5 hash calculated when consuming a x509 certificate | No |  |
| lastUpdated | When the object was last changed | No |  |
| certificatePublicKeyId | foreign key to certificate public key | no |  |
| subjectKeyIdentifier | The key identifier of the certificate | no |  |

**CertificatePublicKey**

The list of system types the system has.

|  |  |  |  |
| --- | --- | --- | --- |
| id | primary key | Yes |  |
| version | version | Yes |  |
| algorithm | the certificate algorithm | No |  |
| encoded | the encoded public key | No |  |
| format | the format | No |  |

**MailSystemDbSystemTypes**

The list of system types the system has.

|  |  |  |  |
| --- | --- | --- | --- |
| **Column** | **Description** | **Required** | **Comment** |
| mail\_system\_db\_id | the foreign key of the system the type belongs to | Yes | Foreign key to system |
| system\_type | The type | No | RECIPIENT/RECIPIENT\_DEFAULT/SENDER |

**AllowedIP**

The allowed IPs for a system to communicate with Digital Post from.

|  |  |  |  |
| --- | --- | --- | --- |
| **Column** | **Description** | **Required** | **Comment** |
| id | primary key | Yes |  |
| version | version | Yes |  |
| ip | The allowed ip | No |  |
| range | if the given ip is a range of ip’s | No |  |

**SystemAllowedIp**

A relation table between the System-table and AllowedIp-table, which keeps tract of which whitelisted IPs and IP ranges a givens system have.

|  |  |  |  |
| --- | --- | --- | --- |
| **Column** | **Description** | **Required** | **Comment** |
| systemId | the foreign key of the system the allowed IP belongs to | Yes | Foreign key to system |
| allowedIpId | the foreign key of the allowed IP | Yes | Foreign key to allowedIp |

**StandardSystemTemplateAllowedIp**

A relation table between the StandardSystemTemplate-table and AllowedIp-table, which keeps tract of which whitelisted IPs and IP ranges a givens system have.

|  |  |  |  |
| --- | --- | --- | --- |
| **Column** | **Description** | **Required** | **Comment** |
| standardSystemTemplateId | the foreign key of the system template the allowed ip belongs to | Yes | foreign key to system template |
| allowedIpId | the foreign key of the allowed ip | Yes | foreign key to allowed ip |

**StandardSystemTemplate**

Template for systems integrating to Digital Post, e.g. a municipality system provided by an IT vendor that multiple municipalities uses. A standard system template cannot in it self send or receive Digital Post, it has to be instantiated for each of the using public authorities or companies.

|  |  |  |  |
| --- | --- | --- | --- |
| **Column** | **Description** | **Required** | **Comment** |
| id | primary key | Yes |  |
| version | version | Yes |  |
| createdDateTime | The time and date the template was created | No |  |
| lastUpdated | The time and date the template was last updated | No |  |
| deletedDateTime | The time and date the template was marked deleted | No | A deleted template must not be attached to new systems |
| name | The name of the system | No | Should not propagate to MailSystems created using standardSystem template |
| systemTypes | The type of the system | No | SENDER/RECIPIENT/RECIPIENT\_DEFAULT |
| endpoint | The endpoint used for exchanging messages | No |  |
| receiptEndpoint | The endpoint used for exchanging receipt | No |  |
| technicalContactId | The foreign key of Person to contact in case of any technical issues | No |  |
| serviceProtocol | The protocol of the system | No |  |
| receiptFormat | Field for indicating the format of the receipt (MeMo, DP, DP2) | No |  |
| ocesPublicCertificateId | foreign key to ocersPublicCertificate |  |  |
| sshId | foreign key to sshId |  |  |

**Ssh**

The Ssh table contains the ssh-credentials used for systems with the SSH service protocol.

|  |  |  |  |
| --- | --- | --- | --- |
| **Column** | **Description** | **Required** | **Comment** |
| id | primary key | Yes |  |
| version | version | Yes |  |
| sshUsername | Username as string | No | The ssh username is generated by the application |
| sshPublicKey | The public key as a clob (character large object) | No |  |
| lastUpdated | Last time the ssh was changed | No |  |

### Contact Structure Model



**ContactPoint**

A point of contact. Only for public authorities.

|  |  |  |  |
| --- | --- | --- | --- |
| **Column** | **Description** | **Required** | **Comment** |
| id | primary key | Yes |  |
| version | version | Yes |  |
| name | The name displayed in view client when writing to the contact point | No | Only for public authorities |
| TransactionId | id of transaction responsible for current state | Yes |  |
| createdDate | date resource was created | No |  |
| description | A short description of the contact point | No | Only for public authorities |
| internalDescription | The internal describtion of the contact point, not show externally | No | For company to add non public information |
| organisationId | the foreign key of the organisation the contact point belongs to | No |  |
| systemId | foreign key of the system the contact point belongs to | No |  |
| memoSizeThresholdMB | The maximum size of a message for the contact point | No | Only for public authorities |
| allowedNumberOfAttachments | The maximum number of attachments in any message for this contact point | No | Only for public authorities  Message with more attachments will be sent to default system for the authority |
| active | Only active contact point receive Digital Post | No | Only for public authorities |
| visible | Only visible contact points will be shown in view client | No | Only for public authorities |
| reportLinkId | the foreign key of the relalated report link | No |  |
| legacyRecipientSystem | The name of the legacy system that this contact point was migrated from | No | Only written by migration. |

**ContactGroup**

A grouping for contact points. Only for public authorities.

|  |  |  |  |
| --- | --- | --- | --- |
| **Column** | **Description** | **Required** | **Comment** |
| id | primary key | Yes |  |
| version | version | Yes |  |
| TransactionId | id of transaction responsible for current state | Yes |  |
| createdDate | date resouce was created |  |  |
| name | The name of the contact group | No | Only for public authorities |
| organisationId | foreign key of the organisation owning the contact group | No |  |
| description | A textual description of the contact group to be displayed in view clients | No | Only for public authorities |
| parentId | foreign key of the parent contact group | No |  |
| organisationalUnit | A boolean signifying that the contact group is an organisational unit, opposed to a subject based group | No | Only for public authorities |
| targets | Gross list of contact points' target types belonging to organisation |  | Not updatable |
| postkasseIds | List of postkasseIds | No |  |

**ContactGroupDbTargets**

|  |  |  |  |
| --- | --- | --- | --- |
| **Column** | **Description** | **Required** | **Comment** |
| contact\_group\_db\_id | foreign key to contact group | Yes |  |
| targets | TargetType | Yes |  |

**ContactGroupDbPostkasseIds**

|  |  |  |  |
| --- | --- | --- | --- |
| **Column** | **Description** | **Required** | **Comment** |
| contact\_group\_db\_id | foreign key to contact group | Yes |  |
| postkasse\_ids | integer | Yes |  |

**ContactPointContactGroup**

table facilitating the many to many relationship of contactpoints and contact groups

|  |  |  |  |
| --- | --- | --- | --- |
| **Column** | **Description** | **Required** | **Comment** |
| contact\_point\_id | foreign key to contact point | Yes |  |
| contact\_group\_id | foreign key to contact group | Yes |  |

**ReportLink**

A contact point can have a link outside of Digital Post. The purpose of this is to be able to guide users to handle the contact outside Digtal Post, e.g. a self-service portal, if available. Only for public authorities.

|  |  |  |  |
| --- | --- | --- | --- |
| **Column** | **Description** | **Required** | **Comment** |
| id | primary key | Yes |  |
| version | version | Yes |  |
| externalLinkText | The text to be displayed to the user in view clients | No | Only for public authorities |
| externalLink | The webpage to be linked to | No | Only for public authorities |
| description | A description of the webpage behind the link | No | Only for public authorities |
| label | The label of the link | No | Only for public authorities |

**RecommendedAttribute**

Attribute the view client should nudge the user to provide when writing to this specific contact point. Only for public authorities.

|  |  |  |  |
| --- | --- | --- | --- |
| **Column** | **Description** | **Required** | **Comment** |
| id | primary key | Yes |  |
| version | version | Yes |  |
| name | A descriptive name of the attribute, e.g. 'CPR-number' | No | Only for public authorities |
| contactPointId | foreign key to the contact point the attribute belongs to | Yes |  |

**SearchTerm**

The terms, which when searched for should make sure the contact point is found. Only for public authorities.

|  |  |  |  |
| --- | --- | --- | --- |
| **Column** | **Description** | **Required** | **Comment** |
| id | primary key | Yes |  |
| version | version | Yes |  |
| contactPointId | foreign key to the contact point the searchterm belongs to | Yes |  |
| term | The searchTerm name, which is also the term searched for | No | Only for public authorities |

**ContactPointCode**

If a contact point is part of a classification or are associated with a location, e.g. the contact point is a unit in a hospital, the location can be specified. Only for public authorities.

|  |  |  |  |
| --- | --- | --- | --- |
| **Column** | **Description** | **Required** | **Comment** |
| id | primary key | Yes |  |
| version | version | Yes |  |
| name | The user friendly name of the contact point code, to help users distinguish on more than ID | No | Only for public authorities |
| type | The type of the contact point code | No | Only for public authorities |
| code | If the contact point code has a code | No | Only for public authorities |
| codeVersion | The version of the contact point code for external use. Not the same as the technical version used in the code | No | Only for public authorities |
| contactPointCodeTypeName | An extra name for code type. Mainly used when type is put as custom | No | Only for public authorities |
| contactPointId | foreign key to the contact point the searchterm belongs to | Yes |  |

**Translation**

Messages sent to the transformation component in DP1 / DP2 format, which can be answered, contain references to postkasseId and postkasseEmendId. When the message is transformed into MeMo format, these should be translated into a contact point.

|  |  |  |  |
| --- | --- | --- | --- |
| **Column** | **Description** | **Required** | **Comment** |
| id | primary key | Yes |  |
| version | version | Yes |  |
| postkasseId | First field used to translate E-boks contact hierarchy to DigitalPost contact structure. | No | Only for public authorities |
| postkasseEmneId | Second field used to translate E-boks contact hierarchy to DigitalPost contact structure. | No | Only for public authorities |

## Querying in System registry APIs

For description of common search functionality, please revisit the section **Querying and searching resources**.

The system registry exposes endpoint to fetch organisation. An organisation in the system registry contains multiple sub-resources, these are sender and receiver systems (called systems), contact point and contact groups. Each have their own set of points that can be queried. This is done to accommodate multiple users editing simultaneously in both the contact structure (contact point and groups) and systems of a single organisation.

The following five endpoints have been exposed externally from System Registry:

* /organisations/
  + Queries all organisations the user is allowed to see. A system manager can see all organisations, while an anonymous user can only see authorities. Likewise is it with regards to how much is returned. An organisation administrator can see all fields of own organisation but only certain fields of authorities.
* /contact-groups/
  + Queries all contact groups across all organisations. A contact group can have a parent contact group or no parent if the group is placed at the top of the hierarchy.
* /organisations/{organisation-id}/contact-groups/
  + Queries contact groups under specified organisation.
* /organisations/{organisation-id}/contact-points/
  + Returns list of contact-points under specified organisation. Each contact-point has a list of contact-groups (it can now belong to more than one group). If the contact-point is at the top at the hierarchy, the contact-group list is empty. A contact-point **cannot** both exist in a group **and** also at the root-level.
* /organisations/{organisation-id}/systems/
  + Returns a list of sender and receiver systems that the given organisation currently has created.

The result is an OrganisationSearchResult, which looks like this in JSON:

|  |
| --- |
| {    "currentPage": 0,    "totalPages": 1,    "elementsOnPage": 1,    "totalElements": 1,    "organisations": []  } |

### Fetching organisations on ID

|  |
| --- |
| /organisations/70b29451-a265-427c-ac7b-cda01413a080 |

will fetch the organisation with id 70b29451-a265-427c-ac7b-cda01413a080 (Random UUID with no actual organisation behind it).

**Searching for organisations**

Searching can be done using the generic search mentioned above and/or using the following parameters:

|  |  |
| --- | --- |
| id | Organisation ID(s) to search for. Used to fetch many organisations with known IDs at the same time |
| name | Name(s) of organisation(s) to search for |
| cvrNumber | One or more CVR numbers to search for |
| systemId | One or more systemIds to search for |
| type | A organisation type to search for. One of AUTHORITY COMPANY. Also requires a cvrNumber parameter. |

**Examples**

Generally the format is:

|  |
| --- |
| /organisations/?<parameter>=<value>  /organisations/?<parameter>=<value>,<value>,<value>  /organisations/?<parameter>=<value>&<parameter>=<value>&<parameter>=<value> |

It’s possible to mix-and-match different parameters. For lines 2 and 3 it’s possible to add as many parameters/values as desired.  
Example:

|  |
| --- |
| /organisations/?cvrNumber=1234567890 |

returns the organisation with CVR number 1234567890

|  |
| --- |
| /organisations/?cvrNumber=1234567890,2345678901 |

would get us the two organisations with the given CVR numbers.

### Searching for ContactPoints across all organisations or within an Organisation

Querying for ContactPoints within an organisation is done using a GET request to either of the endpoints:

*/organisations/*id/contact-points/

*/*contact-points/

**Where id is the id of the organisation**.

The result is a page of contactPointSearchResult, which looks like this in JSON:

|  |
| --- |
| {    "currentPage": 0,    "totalPages": 1,    "elementsOnPage": 1,    "totalElements": 1,    "contactPoints": []  } |

It can be paged, just like the organisationSearchResult and the structure of a contactpoint is part of the organisation

/organisations/{id}/contact-points/

The endpoint supports each searching against all properties on the ContactPoint. Additionally the endpoint also has the following search options:

|  |  |
| --- | --- |
| isRoot | boolean. If true, the only contact-point that is not related to any contact-groups is returned. If provided as false, only contact-point associated to any contact-group is returned |
| searchTerm | One or more search terms to search for |
| searchTermOrName | One or or more search words that can be found in either search terms OR name fields |
| postkasseId | One id to search for |
| postkasseEmneId | One id to search for |

**Examples**

Generally the format is:

|  |
| --- |
| /organisations/id/contact-points/?<parameter>=<value>  /organisations/id/contact-points/?<parameter>=<value>,<value>,<value>  /organisations/id/contact-points/?<parameter>=<value>&<parameter>=<value>&<parameter>=<value> |

It’s possible to mix-and-match different parameters. For lines 2 and 3 it’s possible to add as many parameters/values as desired.

|  |
| --- |
| /organisations/70b29451-a265-427c-ac7b-cda01413a080/contact-points/? |

returns all the contact points for the organisation with id 70b29451-a265-427c-ac7b-cda01413a080

|  |
| --- |
| /organisations/70b29451-a265-427c-ac7b-cda01413a080/contact-points/?searchTerm=Folkeskole |

returns all the contact points with search term “Folkeskole” belonging to the organisation with ID 70b29451-a265-427c-ac7b-cda01413a080

|  |
| --- |
| /organisations/70b29451-a265-427c-ac7b-cda01413a080/contact-points/?searchTerm=Folkeskole&contactGroupId=456,789 |

returns all the contact points that are in contact groups with either ID 456 or 789 that also include search term “Folkeskole” belonging to the organisation with ID 70b29451-a265-427c-ac7b-cda01413a080

|  |
| --- |
| /organisations/70b29451-a265-427c-ac7b-cda01413a080/contact-points/?searchTermOrName=Røntgen\*&active=**true**&visible=**true** |

returns all the contact points where either search terms starts with “Røntgen” OR name starts with “Røntgen“ AND is active AND visible, belonging to the organisation with ID 70b29451-a265-427c-ac7b-cda01413a080

### Searching for Systems within an organisation

Searching for systems is done using a GET request to the */organisations/id/systems/* endpoint.

The result is a SystemSearchResult, which looks like this in JSON:

|  |
| --- |
| {    "currentPage": 0,    "totalPages": 1,    "elementsOnPage": 1,    "totalElements": 1,    "systems": []  } |

**Example**

|  |
| --- |
| /organisations/70b29451-a265-427c-ac7b-cda01413a080/systems/ |

returns all systems belonging to organisation with ID 70b29451-a265-427c-ac7b-cda01413a080. You can search for any property on a System.

### Searching for Contact Groups across all organisations or within an organisation

Querying contact-groups is done using a GET request to either of the endpoints

* */contact-groups/*
* */organisations/id/contact-groups/*

The result is a ContactGroupSearchResult, which looks like this in JSON:

|  |
| --- |
| {    "currentPage": 0,    "totalPages": 1,    "elementsOnPage": 1,    "totalElements": 1,    "contactGroups": []  } |

**Examples**

|  |
| --- |
| /organisations/70b29451-a265-427c-ac7b-cda01413a080/contact-groups/? |

returns all contact groups belonging to organisation with id 70b29451-a265-427c-ac7b-cda01413a080. There are not parameters to narrow down the results.

|  |
| --- |
| /organisations/70b29451-a265-427c-ac7b-cda01413a080/contact-groups/?name=pas |

returns all contact groups that have “pas” in their name somewhere. This includes examples like: “Bestil nyt pas” and just “Pas”.  
These parameters are available for use when searching for contact groups:

|  |  |
| --- | --- |
| id | ID of the organisation to search for contact points in |
| name | name of the contact group |

## Fetching hidden contact points and contact groups

Contact points and contact groups both have the boolean field “visible” that determines if a point or group is hidden. The need for this is to support deep-links in the contact point structure. Given the necessary privileges a query with the parameter “visibility“ can be made. The visibility parameter has three different values:

* VISIBLE - Shows only points and groups with visible = true
* HIDDEN - Shows only points and groups with visible = false
* ALL - Shows all points and groups regardless of visibility

**Examples**

/organisations/70b29451-a265-427c-ac7b-cda01413a080/contact-groups/?visibility=HIDDEN

Returns all contact groups belonging to organization with id 70b29451-a265-427c-ac7b-cda01413a080, that have visible = false

/organisations/70b29451-a265-427c-ac7b-cda01413a080/contact-points/?visibility=VISIBLE

Returns all contact points belonging to organization with id 70b29451-a265-427c-ac7b-cda01413a080, that have visible = true

/organisations/70b29451-a265-427c-ac7b-cda01413a080/contact-groups/?id=fd8ca657-157c-3ffe-0000-000000000e96&visibility=HIDDEN

Returns a single contact group with id fd8ca657-157c-3ffe-0000-000000000e96 belonging to organization with id 70b29451-a265-427c-ac7b-cda01413a080, that have visible = false

/organisations/70b29451-a265-427c-ac7b-cda01413a080/contact-points/?id=163710ae-5078-350c-0000-00000000306a&visibility=ALL

Returns a single contact point with id 163710ae-5078-350c-0000-00000000306a and organization id 70b29451-a265-427c-ac7b-cda01413a080 regardless of visibility.

## Updating items in system registry

This page describes how to, and what to include when updating items in the System registry.

**Update an organisation information**

Updating an organisation is done using a PUT request to the following endpoints:  
 */organisations/{orgId}*  
 */organisations/{orgId}/contact-points/{id}*  
 */organisations/ {orgId}/contact-groups/{id}*  
 */organisations/ {orgId}/systems/{id}*

With an PUT request the organisationId needs to be set. If putting on a contact-points, contact-groups or systems their ID needs to be set too.

When updating a full organisation only a few fields are optional. The optional fields are shown in the table below. If a field is not shown in the table, it means that the field is mandatory.

|  |  |
| --- | --- |
| **Organisation** Entity | **Optional Fields** |
| **Organisation** | * logo |

|  |  |
| --- | --- |
| **Contact-Point** Entity | **Optional Fields** |
| ContactPoint | * memoSizeThresholdMB (defaults to 95.5 MB) * allowedNumberOfAttachements (defaults to 10) * ContactPointCode * ContactGroup ((id of multiple or no groups) * ReportLink * RecommededAttributes * internalDescription |
| ReportLink | * description * label |
| recommendedAttributes | * name |
| ContactPointCode | * codeVersion (not optional for type: FORM and KLE) |

|  |  |
| --- | --- |
| **Contact-Group** Entity | **Optional Fields** |
| ContactGroup | * parentId (id of parent group, null if top) |

|  |  |
| --- | --- |
| **System** Entity | **Optional Fields** |
| System | * activeTo (only available if activeFrom is set) * activeFrom * supplier * businessContactEmail * standardSystemTemplate (are allowed to be empty) * contactPoints (id of none to multiple contact points ) |
| technicalContact | optional |
| standardSystemTemplateId | optional |
| allowedIps | optional |
| receiptFormat | optional |
| receiptEndpoint | Should be present for sender system and be empty for recipient systems |
| endpoint | Should be present for all recipient systems but REST\_PULL and be empty for sender systems |

**Examples**

IDs in examples will have to be fitted to the current environment.

Example of a PUT request for organisations:

|  |
| --- |
| {  "authorityId": "string",  "name": "string",  "cvrNumber": "string",  "type": "AUTHORITY",  "logo": "string",  "legalNotificationAllowed": **true**,  "mandatoryPostAllowed": **true**,  "systemFetch": **true**,  "legalContact": {  "id": "3fa85f64-5717-4562-b3fc-2c963f66afa6",  "version": 0,  "name": "string",  "email": "string",  "phoneNumber": "string"  }  } |

Example of a PUT request for contactGroups

|  |
| --- |
| {  "name": "string",  "description": "string",  "parentId": "3fa85f64-5717-4562-b3fc-2c963f66afa6",  "organisationalUnit": **true**,  "contactPoints": [  "3fa85f64-5717-4562-b3fc-2c963f66afa6"  ]  } |

Example of a PUT request for contactPoints

|  |
| --- |
| {  "name": "string",  "description": "string",  "internalDescription": "string",  "targets": [  "UNKNOWN"  ],  "memoSizeThresholdMB": 5,  "allowedNumberOfAttachments": 0,  "active": **true**,  "visible": **true**,  "systemId": "3fa85f64-5717-4562-b3fc-2c963f66afa6",  "reportLink": {  "id": "3fa85f64-5717-4562-b3fc-2c963f66afa6",  "version": 0,  "externalLink": "https://digidp.atlassian.net/wiki",  "externalLinkText": "https://digidp.atlassian.net/wiki",  "description": "string",  "label": "string"  },  "contactGroups": [  "3fa85f64-5717-4562-b3fc-2c963f66afa6"  ],  "contactPointCode": [  {  "id": "3fa85f64-5717-4562-b3fc-2c963f66afa6",  "version": 0,  "name": "string",  "type": "CUSTOM",  "codeVersion": "string",  "contactPointCodeTypeName": "string",  "code": "string"  }  ],  "recommendedAttributes": [  {  "id": "3fa85f64-5717-4562-b3fc-2c963f66afa6",  "version": 0,  "name": "string"  }  ],  "searchTerms": [  {  "id": "3fa85f64-5717-4562-b3fc-2c963f66afa6",  "version": 0,  "term": "string"  }  ]  } |

Example of a PUT request for systems

|  |
| --- |
| {  "name": "string",  "endpoint": "string",  "receiptEndpoint": "string",  "certificateSerialNumber": "string",  "activeFrom": "2019-01-19T07:57:05.294Z",  "activeTo": "2021-01-19T07:57:05.294Z",  "technicalContact": {  "id": "3fa85f64-5717-4562-b3fc-2c963f66afa6",  "version": 0,  "name": "string",  "email": "string",  "phoneNumber": "string"  },  "serviceProtocol": "REST\_PUSH",  "standardSystemTemplateId": **null**,  "systemTypes": [  "RECIPIENT\_DEFAULT"  ],  "allowedIps": [  {  "id": "3fa85f64-5717-4562-b3fc-2c963f66afa6",  "version": 0,  "ip": "string"  }  ],  "contactPoints": [  "3fa85f64-5717-4562-b3fc-2c963f66afa6",  "3fa85f64-5717-4562-b3fc-2c963f66afa6"  ],  "receiptFormat": "DP2",  "supplier": "string",  "businessContactEmail": "test@test.com"  } |

Please note that neither the receiptEndpoint nor the endpoint are allow to be a .local domain.

**Note:** If “standardSystemTemplateId” is specified, common system fields after an update have to remain consistent with related StandardSystemTemplate. Otherwise mismatch exception is thrown and response states BAD\_REQUEST (400).

Only the first default system created can be inactive ( activeFrom set in the future), the next created default systems cannot be innactive

## Exporting certificate for upload to Administrative Access

|  |
| --- |
| Sending messages using the SMTP protocol will be faced out 16 August2023 |

When setting up a system using the SMTP service protocol you are expected to upload the public part of the OCES certificate to the system-registry thought Administrative Access. The certificate is required to have the the entire trust chain, for Digital Post to ensure the validity of the certificate.

To export a useful certificate from the pkcs12 certificate you have received from Nets use the following command (Linux):

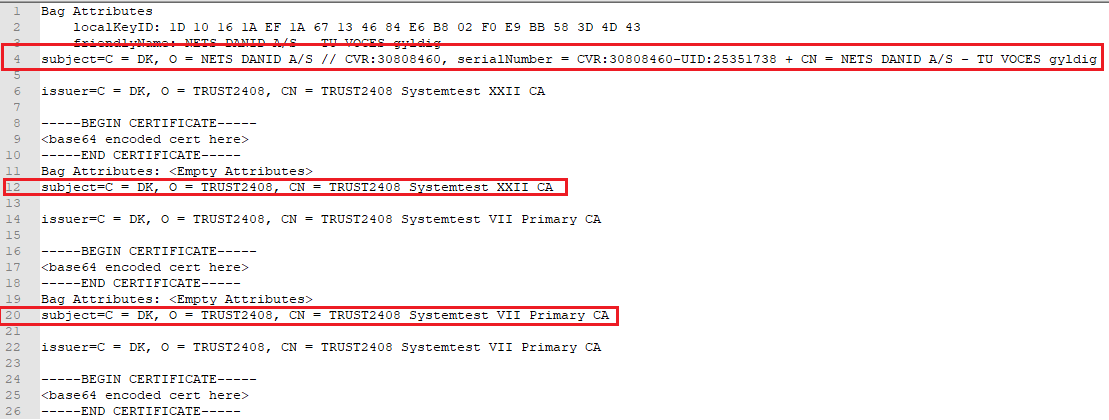
|  |
| --- |
| openssl pkcs12 -in <cert\_from\_nets>.p12 -out valid\_certificate.cer -nokeys |

|  |
| --- |
| For **Windows** users we recommend getting Windows Subsystem for Linux, which is what we use to run the above command with. Alternatively you can use [git for windows](https://git-scm.com/download/win) to get [openssl.exe](https://stackoverflow.com/questions/50625283/how-to-install-openssl-in-windows-10) runnable directly on windows. |

Now you should have a valid certificate.

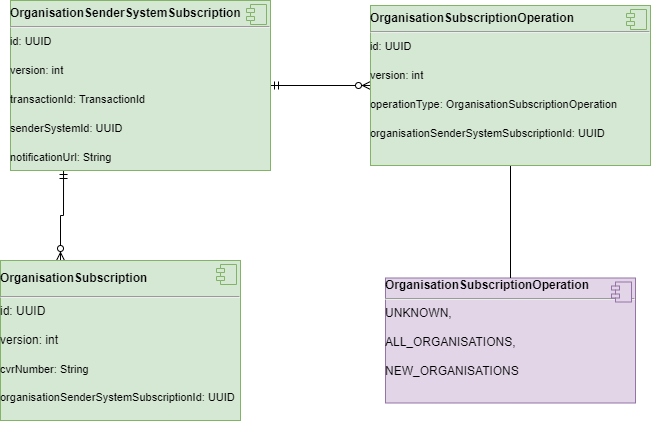
## NAME\_CHAINING -error code

If you do what is described above and still get an error that says: “NAME\_CHAINING“, it probably means that **the order of the certificate chain is messed up**. In the picture below, note the CA on lines 4,12,20. The order of the certificates should be *Your Cert → Intermediate* → *Root*. If *Systemtest VII* comes in the middle, the certificates should be swapped (you can simply do this in a text editor - but avoid notepad.exe).



### Subscribing to changes in the system registry:

The purpose of the system subscription is to persist subscriptions for sendersystems, on either a set of specific organisations, or all changes fitting a category. When a change is made to a contact point is made or a new contactpoint is created, the subscription components finds all matching subscriptions, and notifies the sendersystem on all the matching subscriptions.



**OrganisationSenderSystemSubscription**

The OrganisationSenderSystemSubscription is the anchor of the subscription for any given sendersystem. It contains the notification endpoint and the reference to the sendersystem.

|  |  |  |  |
| --- | --- | --- | --- |
| **Column** | **Description** | **Required** | **Comment** |
| id | The identifier of the subscription | Yes |  |
| version | The current version of the subscription resource | Yes | Incremented on each update |
| transactionId | The identifier of the transaction of which the subscription was either created or last updated | Yes | Added or updated by the application on create or update |
| senderSystemId | The uuid of the sendersystem taken from the access token which for calling sendersystem | Yes | The access token replaces the mutual SSL connection during successful authentication in the API gateway |
| notificationUrl | The URL of which the sendersystem is notified on every relevant change | No | The URL must be https. The subscription is deemed inactive if the URL is not present |

**OrganisationSubscriptionOperation**

The OrganisationSubscriptionOperation table contains all the operations which the sendersystem is subscribed to. The operations are bulk subscription to enable the sendersystem to get notified for all changes or any new entities. These can be used in tandem with explicit organisation subscriptions.

|  |  |  |  |
| --- | --- | --- | --- |
| **Column** | **Description** | **Required** | **Comment** |
| id | The identifier of the subscription | Yes |  |
| version | The current version of the subscription operation resource | Yes | Incremented on each update |
| operationType | The type of bulk operations that are contained in the subscription | Yes | The values which the column can be;   |  | | --- | | ALL\_ORGANISATIONS  NEW\_ORGANISATIONS | |
| organisationSenderSystemSubscriptionId | The foreign key to [system Subscription](https://digidp.atlassian.net/wiki/spaces/DP/pages/813138223/Contact+subscription+data+model#Contact-Subscription) | Yes |  |

**OrganisationSubscription**

The OrganisationSubscription table contains the explicit subscriptions on cvr numbers.

|  |  |  |  |
| --- | --- | --- | --- |
| **Column** | **Description** | **Required** | **Comment** |
| id | The identifier of the subscription | Yes |  |
| version | The current version of the subscription resource | Yes | Incremented on each update |
| cvrNumber | The cvr number the system has an explicit subscription for | Yes |  |
| systemSubscriptionId | The foreign key to [system Subscription](https://digidp.atlassian.net/wiki/spaces/DP/pages/813138223/Contact+subscription+data+model#Contact-Subscription) | Yes |  |

### Organisation contact-group/point Subscription

In order to have a local registration lists updated with the newest contact-point data, the authorities can subscribe to changes to contact-points and contact-groups.

Authority sender system will as a part of the subscription specify an endpoint were they will get notified every time there is an update to a contact-point/group that is included in their subscription.

An authority can only have one subscription per sender system.

The subscription can either be made on individual organisations or by the usages of subscriptionOperators.

A subscription on individual organisation will contain the cvr numbers of all the specific organisations the subscriber want to be notified about. The amount of cvr numbers should be kept to a minimum, however there is no maximum limit. If the authority sender system tries to provide a cvr that does not exist in Digital Post system registry, the contact will be filtered out of the subscription (Nb. No error will be thrown).

SubscriptionOperators are some pre-defined filters for common use cases. There is a total of two operator:

* ALL\_ORGANISATIONS - The operator includes only updates on updated contact-groups/points.
* NEW\_ORGANISATIONS - The operator includes only updates on created contact-groups/points.

The operators can be used individually or several in the same subscription.

### Endpoint exposed in the system-subscription-store

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Service** | **URL** | **Data returned** | **Usage** | **Required roles** |
| Create subscription | POST  /organisations/subscriptions/ | created subscription | Creating a new subscription to get notified when organisations changed. Only one subscription per sender system. | AUTHORITY\_SENDER\_SYSTEM |
| List subscriptions | GET  /organisations/subscriptions/ | List of all subscriptions | Listing all subscriptions | AUTHORITY\_SENDER\_SYSTEM |
| Update subscription | PUT  /organisations/subscriptions/{subscription-id} | Updated subscription | Updating a subscription | AUTHORITY\_SENDER\_SYSTEM |
| Delete subscription | DELETE  /organisations/subscriptions/{subscription-id} |  | Deleting a subscription on the subscription ID with an “if-match” header matching the version | AUTHORITY\_SENDER\_SYSTEM |

Example of a subscription creation with usage of individual contactIDs

|  |
| --- |
| {  "organisationSubscriptions":[  "12345678",  "12345679",  "12345671",  "12345672",  "12345673"  ],  "notificationUrl": "https://postman-echo.com/post"  } |

Example of a subscription creation with usage of subscription operators:

|  |
| --- |
| {  "subscriptionOperations": [  "ALL\_ORGANISATIONS"  ],  "notificationUrl": "https://postman-echo.com/post"  } |

### Notifications

A push notification will be sent to the specified endpoint every time a contact-point/group is updated or created, and which matches the criteria from the subscription. The notification contain the type of resource CONTACT\_POINT/CONTACT\_GROUP, uuid of resource, version of resource, uuid of organisation the resource belongs to.

Example of Notification:

|  |
| --- |
| {  "resourceId": "521f3356-d1c5-4812-bc69-59423fce52b7",  "resourceVersion": 1,  "type": "CONTACT\_POINT", (can be CONTACT\_POINT/CONTACT\_GROUP)  "organisationId": "521f3356-d1c5-4812-bc69-59423fce52bb"  } |

# Mailbox services - TI

The following services are exposed from the mailbox.

## Mailbox

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Service** | **URL** | **Data returned** | **Usage** | **Required roles** |
| Create mailbox | *internal* | Mailbox and owner Access | Creating a mailbox | * System access (system\_mailbox) |
| Delete mailbox | *internal* |  | Deleting a mailbox including all accesses, folders and messages. | * System access (system\_mailbox) |
| Query mailboxes | GET  /mailboxes/ | List of Mailbox | Fetching all mailboxes user has access to with optional paging. | * Borger * DP Skriver * DP Basis * DP Medarbejder * DP Admin * Kurator / bobestyrer * Partsrepræsentant (læser) * DP Full access |
| Fetch mailbox | GET  /mailboxes/{mailbox-id} | Mailbox | Fetching a single mailbox. | * Borger * DP Skriver * DP Basis * DP Medarbejder * DP Admin * Kurator / bobestyrer * Partsrepræsentant (læser) * DP Full access |
| Update mailbox | PUT  /mailboxes/{mailbox-id} | Mailbox | Creating and updating email- sms - and push-notification subscriptions.  Setting introduction completed flag. | * Borger * DP Admin * Kurator / bobestyrer * Partsrepræsentant (læser) * DP Full access |
| List trustees | GET  /mailboxes/trustees | List of trustees | Fetching all trustees of the mailbox owner, with optional sorting. | * Borger * DP Basis * DP Medarbejder * Kurator / bobestyrer * DP Full access |
| Fetch sender information | GET  /mailboxes/{id}/sender-information/ | List of SenderInformation | Fetch a list of senders for messages to a specific mailbox. Each element of the list contains information about most recently received message, number of messages and unread messages. | * Borger * DP Skriver * DP Basis * DP Medarbejder * DP Admin * Kurator / bobestyrer * Partsrepræsentant (læser) * DP Full access |

## Accesses

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Service** | **URL** | **Data returned** | **Usage** | **Required roles** |
| Create access | POST  /mailboxes/{mailbox-id}/accesses/ | Access | Creating Access.  Creating email- sms - and push-notification subscriptions.  Setting introduction completed flag. | * Borger * DP Admin * Kurator / bobestyrer * Partsrepræsentant (læser) * DP Full access |
| Update access | PUT  /mailboxes/{mailbox-id}/accesses/{access-id} | Access | Creating and updating email- sms - and push-notification subscriptions.  Setting introduction completed flag. | * Borger * DP Admin * Kurator / bobestyrer * Partsrepræsentant (læser) * DP Full access |
| Verify Access | PUT  /mailboxes/{mailbox-id}/accesses/{access-id}/subscriptions/{subscription-id}/verification | Access | Updates the verification time on an Access' subscription | * Borger * DP Admin * Kurator / bobestyrer * Partsrepræsentant (læser) * DP Full access |
| Verify Access | PUT  /mailboxes/{mailbox-id}/accesses/{access-id}/subscriptions/{subscription-id}/verification | Access | Updates the verification time on an Access' subscription | * Borger * DP Admin * Kurator / bobestyrer * Partsrepræsentant (læser) * DP Full access |

## Messages

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Service** | **URL** | **Data returned** | **Usage** | **Required roles** |
| Create draft message | POST  /mailboxes/{mailbox-id}/messages/ | Message | Creating a new draft message. Will be located in DRAFTS folder. | * Borger * DP Skriver * DP Basis * DP Full access |
| Create reply | POST  /mailboxes/{mailbox-id}/messages/{message-id}/reply | Message | Creating a reply template from a message. Will be a new draft message located in DRAFTS folder. | * Borger * DP Skriver * DP Basis * DP Full access |
| Update draft message | PUT  /mailboxes/{mailbox-id}/messages/{message-id} | Message | Updating a message. | * Borger * DP Skriver * DP Basis * DP Full access |
| Send draft message | POST  /mailboxes/{mailbox-id}/messages/{message-id}/send | Message | Sending the message. Will move message to SENT folder. | * Borger * DP Skriver * DP Basis * DP Full access |
| Forward message to e-mail address | POST  /mailboxes/{mailbox-id}/messages/{message-id}/forward | Message | Forwarding a received message to an e-mail address. | * Borger * DP Basis * DP Medarbejder * Kurator / bobestyrer * Partsrepræsentant (læser) * DP Full access |
| Forward message to trusted recipient or authority | POST  /mailboxes/{mailbox-id}/messages/{message-id}/forward | Message | Forwarding a received message to a trusted recipient (present in users saml token).  Forwarding a received message to an authority using either CVR or conntact point. | * Borger * DP Basis * DP Medarbejder * Kurator / bobestyrer * DP Full access |
| Query messages | GET  /mailboxes/{mailbox-id}/messages/ | List of Message | Fetching multiple messages in a mailbox with optional paging, searching, filtering and sorting. | * Borger * DP Skriver * DP Basis * DP Medarbejder * DP Admin * Kurator / bobestyrer * Partsrepræsentant (læser) * DP Full access |
| Fetch message | GET  /mailboxes/{mailbox-id}/messages/{message-id} | Message | Fetching a single message. | * Borger * DP Skriver * DP Basis * DP Medarbejder * DP Admin * Kurator / bobestyrer * Partsrepræsentant (læser) * DP Full access |
| Update message | PUT  /mailboxes/{mailbox-id}/messages/{message-id} | Message | Updating a message's flags, folder and note | * Borger * DP Skriver * DP Basis * DP Medarbejder * Kurator / bobestyrer * Partsrepræsentant (læser) * DP Full access |
| Patch message | PATCH  /mailboxes/{mailbox-id}/messages/{message-id} | Message | Updating one or more of certain predetermined fields of a message such as folderId, flags, recipient etc. | * Borger * DP Skriver * DP Basis * DP Medarbejder * Kurator / bobestyrer * Partsrepræsentant (læser) * DP Full access |
| Delete message | DELETE  /mailboxes/{mailbox-id}/messages/{message-id} |  | Deleting a message | * Borger * DP Medarbejder * Tilbagekaldsadministrator * DP Full access (only for drafts) |
| Fetch unread status | GET  /mailboxes/messages/unread/exists | Unread | Returns true/false of whether the mailbox of a given CPR/CVR contains unread REGULAR messages from within the last 6 months | * Borgerservice * Erhvervsservice * DP Full access |

## Documents

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Service** | **URL** | **Data returned** | **Usage** | **Required roles** |
| List documents | GET  /mailboxes/{mailbox-id}/messages/{message-id}/documents/ | List of Document | Fetching all documents of a message | * Borger * DP Skriver * DP Basis * DP Medarbejder * DP Admin * Kurator / bobestyrer * Partsrepræsentant (læser) * DP Full access |
| Fetch document | GET  /mailboxes/{mailbox-id}/messages/{message-id}/documents/{document-id} | Document | Fetching a single document | * Borger * DP Skriver * DP Basis * DP Medarbejder * DP Admin * Kurator / bobestyrer * Partsrepræsentant (læser) * DP Full access |
| Create document | POST  /mailboxes/{mailbox-id}/messages/{message-id}/documents/ |  | Creating a document | * Borger * DP Skriver * DP Basis * DP Full access |
| Update document | PUT  /mailboxes/{mailbox-id}/messages/{message-id}/documents/{document-id} |  | Updating a document | * Borger * DP Skriver * DP Basis * DP Full access |
| Delete document | DELETE  /mailboxes/{mailbox-id}/messages/{message-id}/documents/{document-id} |  | Deleting a single document | * Borger * DP Skriver * DP Basis * DP Full access (not allowed unless draft) |

## Files

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Service** | **URL** | **Data returned** | **Usage** | **Required roles** |
| List files | GET  /mailboxes/{mailbox-id}/messages/{message-id}/documents/{document-id}/files/ | List of File | Fetching all files of a document | * Borger * DP Skriver * DP Basis * DP Medarbejder * DP Admin * Kurator / bobestyrer * Partsrepræsentant (læser) * DP Full access |
| Fetch file | GET  /mailboxes/{mailbox-id}/messages/{message-id}/documents/{document-id}/files/{file-id} | File | Fetching a single file | * Borger * DP Skriver * DP Basis * DP Medarbejder * DP Admin * Kurator / bobestyrer * Partsrepræsentant (læser) * DP Full access |
| Fetch file content | GET  /mailboxes/{mailbox-id}/messages/{message-id}/documents/{document-id}/files/{file-id}/content | Raw bytes | Fetching file content bytes | * Borger * DP Skriver * DP Basis * DP Medarbejder * DP Admin * Kurator / bobestyrer * Partsrepræsentant (læser) * DP Full access |
| Create file | POST  /mailboxes/{mailbox-id}/messages/{message-id}/documents/{document-id}/files/ |  | Creating a file | * Borger * DP Skriver * DP Basis * DP Full access |
| Update file | PUT  /mailboxes/{mailbox-id}/messages/{message-id}/documents/{document-id}/files/{file-id} |  | Updating a file | * Borger * DP Skriver * DP Basis * DP Full access |
| Delete file | DELETE  /mailboxes/{mailbox-id}/messages/{message-id}/documents/{document-id}/files/{file-id} |  | Deleting a single file from a document | * Borger * DP Skriver * DP Basis * DP Full access (only for drafts) |
| Update file content | PUT  /mailboxes/{mailbox-id}/messages/{message-id}/documents/{document-id}/files/{file-id}/content | FileContent | Update the contents of a file | * Borger * DP Skriver * DP Basis * DP Full access |

## Folders

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Service** | **URL** | **Data returned** | **Usage** | **Required roles** |
| Create folder | POST  /mailboxes/{mailbox-id}/folders/ | Folder | Creating a folder. | * Borger * DP Medarbejder * Kurator / bobestyrer * Partsrepræsentant (læser) * DP Full access |
| Fetch folder | GET  /mailboxes/{mailbox-id}/folders/{folder-id} | Folder | Fetching a single folder. | * Borger * DP Skriver * DP Basis * DP Medarbejder * DP Admin * Kurator / bobestyrer * Partsrepræsentant (læser) * DP Full access |
| Query folders | GET  /mailboxes/{mailbox-id}/folders/ | List of Folder | Fetching folders in a mailbox with optional paging. | * Borger * DP Skriver * DP Basis * DP Medarbejder * DP Admin * Kurator / bobestyrer * Partsrepræsentant (læser) * DP Full access |
| Update folder | PUT  /mailboxes/{mailbox-id}/folders/{folder-id} | Folder | Updating a folder. | * Borger * DP Medarbejder * Kurator / bobestyrer * Partsrepræsentant (læser) * DP Full access |
| Delete folder | DELETE  /mailboxes/{mailbox-id}/folders/{folder-id} |  | Deleting a folder. | * Borger * DP Medarbejder * DP Full access |
| Fetch folder summary | GET  /mailboxes/{id}/folders/summary/ | List of FolderMessageSummary | Fetches a list of folders and a summary of total and unread amount of messages for each folder | * Borger * DP Skriver * DP Basis * DP Medarbejder * DP Admin * Kurator / bobestyrer * Partsrepræsentant (læser) * DP Full access |

## System fetches

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Service** | **URL** | **Data returned** | **Usage** | **Required roles** |
| Create SystemFetch | POST /mailboxes/{mailbox-id}/system-fetches/ | SystemFetch | Creates a system fetch job that starts emptying a mailbox into a recipient system | * DP Admin |
| Fetch SystemFetch | GET /mailboxes/{mailbox-id}/system-fetches/{system-fetch-id} | SystemFetch | Fetches existing system fetch. | * DP Admin |
| Update SystemFetch | PUT /mailboxes/{mailbox-id}/system-fetches/{system-fetch-id} | SystemFetch | Updates a system fetch. Only option is to request it STOPPED using the status type. Stops the running job that fetches | * DP Admin |
| Delete SystemFetch | DELETE /mailboxes/{mailbox-id}/system-fetches/{system-fetch-id} |  | Stops running job and deletes SystemFetch resource | * DP Admin |
| List SystemFetch | GET /mailboxes/{mailbox-id}/system-fetches/ | List of SystemFetch | Returns all SystemFetch for the mailbox | * DP Admin |

## Mailbox persistence entity model

The role of the mailbox is to keep data for mailboxes, accesses, messages and folders for users. It uses standard JPA/Hibernate convention and converts camel case to snake case (underscore delimiter) so for instance the entity SmsNotificationSubscription maps to a sms\_notification\_subscription table.

All tables include Id (internal ID of the resource), Version (how many times the resource has been updated), and possibly a createdDateTime/lastUpdated. If the table represents a REST resource that is indexed it will also include a TransactionId (id of the transaction the resource was last updated in). These fields will not be included in the attributes descriptions.

For all one-to-many relations the type on the many side of the relationship has a foreign key to the other component. The field will always be named foreigntablenameId, so for example there is a one-to-many between mailbox and folder, so folder will have a field called mailboxId.

### Mailbox model



**Mailbox**

The Mailbox entity contains the mailbox root resource for citizens and companies.

|  |  |  |  |
| --- | --- | --- | --- |
| **Column** | **Description** | **Required** | **Comment** |
| id | id of entity | Yes | The primary key |
| version | The current version | Yes | Used to ensure optimistic locking and is automatically increased by one on each update |
| transactionId | The id of the transaction the resource was updated in. | Yes |  |
| createdDateTime | The time of creation | Yes |  |
| lastUpdated | The time of last updating | Yes |  |
| statusDate | Date the status was updated | Yes | Used to clean up closed mailbox 5/10 years after they are closed for a citizen/company |
| access | All the accesses for the given mailbox | Yes |  |
| recipientSystemAvailable | Marks if the owner of the mailbox has a recipient system | Yes | Only messages to organisations without a recipient system should be stored in the mailbox |
| exempt | Marks if the owner of the mailbox is exempt from receiving Digital Post messages | Yes | If the owner is exempt, they can only receive Mandatory Digital Post messages. |
| ownerType | Type of the owner of the mailbox (citizen/organisation) | Yes |  |
| ownerIdentityId | Id from identity registry. Citizen or Organisation | Yes |  |
| ownerName | Name of citizen or company | No | Maintained via identity-registry |
| ownerExternalId | CPR/CVR | No | Maintained via identity-registry |
| statusType | Status of the mailbox | Yes |  |
| lastAccessedDate | Specifies the last date this mailbox was opened (either by the owner or a Power of attorney). If not specified the mailbox has never been opened | No | Default Null. |
| reminded | True - A Reminder Letter event has been generated from the Mailbox.  False - This mailbox has not been part of a Reminder Letter sendout flow. | Yes | Default False |
| activationDate | The first date a messages has been delivered to the mailbox. | No | Default Null |

### Access model



**Access**

The Access entity is used to indicate a user's access to the specific mailbox, and to store notification subscription information for the user.

|  |  |  |  |
| --- | --- | --- | --- |
| **Column** | **Description** | **Required** | **Comment** |
| id | id of entity | Yes | The primary key |
| version | The current version | Yes | Used to ensure optimistic locking and is automatically increased by one on each update |
| mailboxId | Id of the mailbox the access belongs to | No | Foreign key to mailbox |
| transactionId | The id of the transaction the resource was updated in. | Yes |  |
| createdDateTime | The time of creation | Yes |  |
| lastUpdated | The time of last update | Yes |  |
| optedOutOfNotificationDateTime | If the access have resigned from all nortifications | Yes |  |
| accessType | Type of access | Yes | E.g. a citizen will be owner of their own mailbox, but may have a party representative that can also read the mailbox |
| identityId | The identity the access belongs to | Yes | The identifier of the identity owned by the identity-registry |
| smsNotificationSubscriptionId | Id of the SMS notification subscription on the mailbox, if any | No | Foreign key |
| introductionCompleted | Marks if the user has completed the introduction | Yes |  |

**SMS Notification Subscription**

The SmsNotificationSubscription entity contains a subscription for SMS notifications for whenever the mailbox receives new messages.

|  |  |  |  |
| --- | --- | --- | --- |
| **Column** | **Description** | **Required** | **Comment** |
| id | id of entity | Yes | The primary key |
| version | The current version | Yes | Used to ensure optimistic locking and is automatically increased by one on each update |
| number | Number to send SMS notifications to | Yes | Must be a valid Danish phone number without country code |
|  |  |  |  |
| unlistedNumber | Marks if the number is unlisted | Yes | \*Deprecated, waiting for removal |
| verificationTime | Marks when the subscription was verified via SMS | No | Empty if the subscription has not been verified.  The verification-store is master of this data, and the number can also be verified thought NemSMS when using the same number |
| confirmationTime | Marks when the subscription was last confirmed not via SMS | No | This is used by view client to give reminders to check if the user still uses the mobile number |

**Email Notification Subscription**

The EmailNotificationSubscription entity contains a subscription for email notifications for whenever the mailbox receives a new messages.

|  |  |  |  |
| --- | --- | --- | --- |
| **Column** | **Description** | **Required** | **Comment** |
| id | id of entity | Yes | The primary key |
| version | The current version | Yes | Used to ensure optimistic locking and is automatically increased by one on each update |
| accessId | Id of the access the subscription belongs to | Yes | Foreign key to the Access table |
| email | Email address to send email to | Yes | Must be a valid email address |
| verificationTime | Marks when the subscription was verified | No | Empty if the subscription has not been verified |
| confirmationTime | Marks when the subscription was last confirmed (not via email) | No | This is used by view client to give reminders to check if the user still uses the mobile number |

**Push Notification Subscription**

The PushNotificationSubscription entity contains the subscription for push notification on devices for the mailbox. When a user has a push notification subscription native notification is delivered to the view clients smartphone application.

|  |  |  |  |
| --- | --- | --- | --- |
| **Column** | **Description** | **Required** | **Comment** |
| id | id of entity | Yes | The primary key |
| version | The current version | Yes | Used to ensure optimistic locking and is automatically increased by one on each update |
| deviceId | The id of the device | Yes | This and deviceToken fields will be collapsed into a single field in the future) For now just put some dummy info. \*under clarification |
| deviceToken | DeviceToken should hold the FCM/APNs deviceToken to receive push notifications | Yes | Obtained by the app on the device, the deviceId \*under clarification |
| instanceId | InstanceId from the Settings in the push-notification-settings-store | Yes | A tenant can have multiple apps. With instanceId, the settings for the specific app are checked.  See “Push notification integrations” in the documentation for more info and endpoints to find this value |
| tenantId | TenantId from push-notification-settings-store - Identifier for the specific push notification sender | Yes | See “Push notification integrations” in the documentation for more info and endpoints to find this value |
| providerType | The push notification provider type. Either FCM or APN | Yes | Current Digital Post supports Apple Push Notification Service (APN) for iOS notifications and Firebase Cloud Messaging (FCM) for Android notification |

### Message model



**Message**

The Message entity contains a message for a given mailbox.

|  |  |  |  |
| --- | --- | --- | --- |
| **Column** | **Description** | **Required** | **Comment** |
| id | id of entity | Yes | The primary key |
| version | The current version | Yes | Used to ensure optimistic locking and is automatically increased by one on each update |
| transactionId | The id of the transaction the resource was updated in. | Yes |  |
| createdDateTime | The time of creation | Yes |  |
| lastUpdated | The time of last update | Yes |  |
| mailboxId | Id of the mailbox the message belongs to | Yes | Foreign key |
| folderId | Id of the folder the message is placed in | Yes | Foreign key |
| memoCreatedDateTime | Date the MeMo was created by the sender | No | This is when the MeMo that was sent to the mailbox was created - Not the message object in the database |
| receivedDateTime | When the message was received by the mailbox | No | Null if the message has not been received (e.g. sent messages and drafts) |
| messageId | The senders identifier for the message | No |  |
| messageType | Type of the message | Yes |  |
| messageState | State of the message | Yes | DRAFT/SENT/RECEIVED |
| memoId | messageUUID from Memo | No |  |
| senderId | Id of the sender of the message | Yes | Foreign key |
| recipientId | Id of the recipient of the message | Yes | Foreign key |
| forwardDataId | Id of the forward data in the message | No | Foreign key |
| contentDataId | Id of the content data in the message | No | Foreign key |
| forward | Marks if the message can be forwarded | Yes |  |
| reply | Marks if the message can be replied to | Yes |  |
| flag | Used by view clients | Yes | Used to mark the message with a flag.  Like the flag option in Outlook. There is no connected business logic - it is up to the clients to use for their purpose |
| legallyNotified | Marks if the message has been legally notified | Yes | Only relevant for messages with message type LEGAL\_NOTIFICATION. Attempts to update field on irrelevant types will be rejected with BAD REQUEST |
| legallyNotifiedDateTime | Stamped when legallyNotified is set true | No | Only relevant for messages with message type LEGAL\_NOTIFICATION |
| read | Marks if the message has been read | Yes |  |
| welcomeMessage | Marks if the message is a welcome message | Yes | A welcome message is created when the mailbox is created |
| errorMessage | Marks if the message is an error message | Yes | When a user sends a MeMo from the mailbox, if it is rejected by the validator component, an error message will be returned to the mailbox |
| label | Title of the message | No |  |
| messageCode | Code for the message | No | There are no specific rules for messageCode, but it can be used to indicate the types of messages, e.g. all messages transformed by transformation-processor has code Transformed |
| sendDateTime | The date the message was sent, if it was sent from this mailbox | No | Received messages does not have a value in this field |
| note | Field the user can use to write a note to the message | No |  |
| postType | Type of post - Company/Authority | No |  |

**Document**

The Document entity contains the attachments/files of MeMo messages. Three types of documents can be added to a DIGITALPOST type message. Messages of the type NEMSMS, has no body and therefore no added documents.

|  |  |  |  |
| --- | --- | --- | --- |
| **Column** | **Description** | **Required** | **Comment** |
| id | id of entity | Yes | The primary key |
| version | The current version | Yes | Used to ensure optimistic locking and is automatically increased by one on each update |
| message\_id | Id of the message the document belongs to | Yes | Foreign key |
| documentId | Identifier of the document | No | Custom identifier, not the database id |
| label | Name of the document | No |  |
| document\_type | Type of the document (MAIN/ADDITIONAL/TECHNICAL) | Yes |  |

**File**

The File entity is the File in the document. Multiple files in the same document can be used e.g. to include the same document in different language.

|  |  |  |  |
| --- | --- | --- | --- |
| **Column** | **Description** | **Required** | **Comment** |
| id | id of entity | Yes | The primary key |
| version | The current version | Yes | Used to ensure optimistic locking and is automatically increased by one on each update |
| documentId | Id of the document the file belongs to | Yes | Foreign key |
| encodingFormat | Encoding format of the attached file | No | application/pdf etc. |
| filename | Name of the file | No |  |
| language | The language of the file content | No | Must follow ISO 639-1. If not specified the language is considered to be Danish |
| fileSize | Number of bytes | No | Automatically filled out when file is added |

**Action**

The Action entity is for actions the sender can supply, which makes it possible for the recipient to act directly on the message. It could be creating a calendar appointment, pay an invoice in the sender’s payment solution, signing a document or other interactions with external systems based on an external link.

|  |  |  |  |
| --- | --- | --- | --- |
| **Column** | **Description** | **Required** | **Comment** |
| id | id of entity | Yes | The primary key |
| version | The current version | Yes | Used to ensure optimistic locking and is automatically increased by one on each update |
| documentId | Id of the document the action belongs to | Yes | Foreign key |
| label | Title of the action | No |  |
| actionCode | Indicates the type of action | No | e.g. BETALING, SIGNERING |
| actionStartTime | The start date and time from when an action can be carried out | No |  |
| actionEndTime | The deadline to carry out the specified action if relevant | No |  |
| entryPointUrl | URI to the service/action made available by the sender of the message e.g. a payment solution or a self-service solution | No |  |
| description | Further description of the appointment | No |  |
| reservationUUID | This property defines the persistent, unique identifier for the calendar component | No |  |
| reservationAbstract | Short text/headline for the calendar appointment | No |  |
| location | Location | No |  |
| reservationStartTime | Calendar start time of the reservation | No |  |
| reservationEndTime | Calendar end time of the reservation | No |  |
| organizerMail | The calendar organizers email | No |  |
| organizerName | The calendar organizers name | No |  |

**ForwardData**

The ForwardData entity is for data used in forwarded messages to reference the original message.

|  |  |  |  |
| --- | --- | --- | --- |
| **Column** | **Description** | **Required** | **Comment** |
| id | id of entity | Yes | The primary key |
| version | The current version | Yes | Used to ensure optimistic locking and is automatically increased by one on each update |
| originalMessageId | id of the message that is forwarded | No |  |
| originalMemoId | messageUUID of the message that is forwarded | No |  |
| originalSender | Sender of the message the original/forwarded message | No | CPR/CVR of original sender |
| originalContentResponsible | Identification of the original content responsible of the forwarded message | No |  |
| comment | Any comment from the original recipient of the message to the new receiver of the forwarded message | No |  |
| contactPointId | Identification of the original contact point id of the forwarded message | No |  |
| newMemoId | ID of the new MeMo (the forward) | No |  |
| originalMessageDateTime | Send date of the original message | No |  |

**ReplyData**

The ReplyData entity contains data that allows the sender to specify data which must be returned in an answer.

|  |  |  |  |
| --- | --- | --- | --- |
| **Column** | **Description** | **Required** | **Comment** |
| id | id of entity | Yes | The primary key |
| version | The current version | Yes | Used to ensure optimistic locking and is automatically increased by one on each update |
| childMessageId | Message id of this reply data structure | Yes |  |
| parentMessageId | Message id of the parent message of this reply data structure. Ie. the message this was a reply to | Yes |  |
| childMemoId | MessageUUID of this reply data structure | No |  |
| parentMemoId | MessageUUID of the parent message of this reply data structure. Ie. the message this was a reply to | No |  |
| messageIdentifier | Messageidentifier of replying message | No |  |
| senderId | SenderID that must be returned in a reply, if specified by the sender | No | CVR/CPR |
| recipientId | RecipientID that must be returned in a reply, if specified by the sender | No | CVR/CPR |
| caseId | CaseID that must be returned in a reply, if specified by the sender | No |  |
| contactPointId | ContactPointID that must be returned in a reply, if specified by the sender | No |  |
| generatingSystemId | GeneratingSystem that must be returned in a reply, if specified by the sender | No |  |
| comment | Comment that must be returned in a reply, if specified by the sender | No |  |
| messageId | ID of the message the replydata belongs to | Yes | Foreign key |

**AdditionalReplyData**

The AdditionalReplyData entity contains up to four subclasses of additional information that must be returned in a reply if specified by the sender, can be added to the ReplyData entity.

|  |  |  |  |
| --- | --- | --- | --- |
| **Column** | **Description** | **Required** | **Comment** |
| id | id of entity | Yes | The primary key |
| version | The current version | Yes | Used to ensure optimistic locking and is automatically increased by one on each update |
| replyDataId | ID of the replydata the additionalreplydata belongs to | Yes | Foreign key |
| label | Label for additional information that must be returned in a reply, if specified by the sender | No |  |
| value | Value for the additional information that must be returned in a reply, if specified by the sender | No |  |

**ContentData**

The ContentData entity contains content data that is specified as data containing the relevant properties. The purpose is to help the recipient sort and distribute received messages internally.

|  |  |  |  |
| --- | --- | --- | --- |
| **Column** | **Description** | **Required** | **Comment** |
| id | id of entity | Yes | The primary key |
| version | The current version | Yes | Used to ensure optimistic locking and is automatically increased by one on each update |
| addressId | Id of the address belonging to the content data | No | Foreign key |
| cprDataCprNumber | Personal registration number | No |  |
| cprDataName | The person name | No |  |
| cvrDataCvrNumber | Company identification (CVR-number) | No |  |
| cvrDataCompanyName | The company name | No |  |
| motorVehicleLicenseNumber | The vehicles license number | No |  |
| motorVehicleChassisNumber | The vehicles chassis number | No |  |
| propertyNumber | Property number | No |  |
| caseId | Case number | No |  |
| caseSystem | The system where the case number is identified | No |  |
| kleDataSubjectKey | Identification of the KLE subject key | No |  |
| kleDataVersion | The KLE version | No |  |
| kleDataActivityFacet | Identification of the KLE activity facet | No |  |
| kleDataLabel | Text or label for the KLE code | No |  |
| formDataTaskKey | Identification of the FORM subject key | No |  |
| formDataVersion | The FORM version | No |  |
| formDataActivityFacet | Identification of the FORM activity facet | No |  |
| formDataLabel | Text or label for the FORM code | No |  |
| productionUnitNumber | Production unit code (“P-nummer”) | Yes |  |
| productionUnitName | Text or label for the production unit | No |  |
| educationCode | The code for that specific education | No |  |
| educitationName | Text or label describing the education | No |  |
| unstructuredAddress | Unstructured address specification | No |  |

**AdditionalContentData**

The AdditionalContentData entity contains additional content data for a message. A maximum of 10 AdditionalContentData pr. message is allowed.

|  |  |  |  |
| --- | --- | --- | --- |
| **Column** | **Description** | **Required** | **Comment** |
| id | id of entity | Yes | The primary key |
| version | The current version | Yes | Used to ensure optimistic locking and is automatically increased by one on each update |
| contentDataId | ID of the content data the additional content data belongs to | Yes | Foreign key |
| contentDataType | The type of additional content data | No |  |
| contentDataName | The name of the additional content data element | No |  |
| contentDataValue | The value of the additional content data element | No |  |

**Address**

The Address entity is for additional information on the ContentData or AttentionData entities used to specify an address.

|  |  |  |  |
| --- | --- | --- | --- |
| **Column** | **Description** | **Required** | **Comment** |
| id | id of entity | Yes | The primary key |
| version | The current version | Yes | Used to ensure optimistic locking and is automatically increased by one on each update |
| addressId | Unique identifier for the address | No |  |
| addressLabel | Street name | No |  |
| houseNumber | House number | No |  |
| door | Identifier for the door | No |  |
| floor | Floor | No |  |
| co | Care of/attention | No |  |
| zipCode | Zip code | No |  |
| city | City name | No |  |
| country | Country code | No |  |
| geographicEastingMeasure | Degree of latitude | No |  |
| geographicNorthingMeasure | Degree of longitude | No |  |
| geographicHeighMeasure | Altitude | No |  |

**AttentionData**

The AttentionData entity is for supplementary data describing the sender or the recipient.

|  |  |  |  |
| --- | --- | --- | --- |
| **Column** | **Description** | **Required** | **Comment** |
| id | id of entity | Yes | The primary key |
| version | The current version | Yes | Used to ensure optimistic locking and is automatically increased by one on each update |
| addressId | ID of the address on the attention data | No | Foreign key |
| personId | Identifier for the person e.g. employee number | No |  |
| personLabel | Text or label for the ID e.g. person name | No |  |
| productionUnitNumber | Production unit code (“P-nummer”) | No |  |
| productionUnitName | Text or label for the production unit | No |  |
| globalLocationNumber | The Global Location Number of the respective organization, person, or place | No |  |
| location | Text or label for the GLN-number | No |  |
| emailAddress | Email address | No |  |
| relatedEmailAgent | The owner/agent related to the email address | No |  |
| seNumber | VAT-number (“SE-nummer”) | No |  |
| seCompanyName | Text or label for the owner of the VAT-number | No |  |
| telephoneNumber | Phone number | No |  |
| relatedTelephoneAgent | The owner/agent related to the phone number | No |  |
| eId | eID identifier | No |  |
| eIdLabel | The name of the owner of the eID/certificate | No |  |
| contentResponsibleId | The identifier for the content responsible | No |  |
| contentResponsibleLabel | Text for label for the content responsible | No |  |
| generatingSystemId | Identifier for the generating system | No |  |
| generatingSystemLabel | Text or label for the generating system | No |  |
| unstructuredAddress | Unstructured address specification | No |  |
| sorDataSorIdentifier | SOR identification | No |  |
| sorDataEntryName | Name related to SOR identifier | No |  |

**Sender**

The Senderentity contains information about the sender of the message.

|  |  |  |  |
| --- | --- | --- | --- |
| **Column** | **Description** | **Required** | **Comment** |
| id | id of entity | Yes | The primary key |
| version | The current version | Yes | Used to ensure optimistic locking and is automatically increased by one on each update |
| attentionDataId | An id to the related attentionData | No |  |
| contactPointId | An id to the related contact point of the sender | No |  |
| senderId | An id of the sender of the message | No | cvr or cpr of the sender |
| senderIdType | The type of sender. Can be CPR or CVR | No |  |
| senderIdTypeLabel | Name of the used idType | No |  |
| label | Name of the sender | No |  |

**Recipient**

The Recipient entity contains information about the recipient of the message.

|  |  |  |  |
| --- | --- | --- | --- |
| **Column** | **Description** | **Required** | **Comment** |
| id | id of entity | Yes | The primary key |
| version | The current version | Yes | Used to ensure optimistic locking and is automatically increased by one on each update |
| attentionDataId | Attention data on the recipient | No | Foreign key |
| contactPointId | Contact point on the recipient | No | Foreign key |
| recipientId | The identification of the recipient e.g. by CVR-number, CPR-number, or e-mail address in case of forwarding | No |  |
| recipientIdType | Specifies the type of the ID, e.g. “CVR”, “CPR” or “EMAIL“ | No |  |
| recipientIdTypeLabel | Name of the used idType | No |  |
| label | Name of the recipient | No | Recipient label is NOT used internally by DP. When recipient’s name is needed, e.g.. for notification emails, it is looked up in the Identity Registry. |

**Contact point**

The ContactPoint entity is used to identify different areas within a public authorities that it makes sense for eg. a citizen to send Digital Post to. Each ContactPoint has an identified receiving system setup in the Digital Post solution.

|  |  |  |  |
| --- | --- | --- | --- |
| **Column** | **Description** | **Required** | **Comment** |
| id | id of entity | Yes | The primary key |
| version | The current version | Yes | Used to ensure optimistic locking and is automatically increased by one on each update |
| contactGroup | A contact group defined by the authority | No | Name of the contact group of the contact point in the system registry |
| contactPointId | The authority’s contact point Id | No | Id of the contact point in the system registry |
| label | Text or label describing the contact point | No |  |

**Contact info**

The ContactInfo entity contains supplementary contact information requested by the authority. When specifying the contact point, the authority can add up to two additional fields, which must be filled out before a message is sent to that particular contact point.

|  |  |  |  |
| --- | --- | --- | --- |
| **Column** | **Description** | **Required** | **Comment** |
| id | id of entity | Yes | The primary key |
| version | The current version | Yes | Used to ensure optimistic locking and is automatically increased by one on each update |
| contactPointId | ID of the contact point the contact info belongs to | Yes | Foreign key |
| label | Name for the contact info field | No |  |
| value | Value for the contact info field | No |  |

### Folder model



**Folder**

A Folder entity to place messages in.

|  |  |  |  |
| --- | --- | --- | --- |
| **Column** | **Description** | **Required** | **Comment** |
| id | id of entity | Yes | The primary key |
| version | The current version | Yes | Used to ensure optimistic locking and is automatically increased by one on each update |
| transactionId |  |  |  |
| createdCateTime |  |  |  |
| lastUpdated |  |  |  |
| folderType | Type of the folder | Yes | Standard folders that exist for all mailboxes (INBOX/DRAFTS/SENT/DELETED), and a custom folder type USER\_DEFINED |
| parentFolderId | ID of the parent folder | No | Folders can be placed inside another folder.  The max depth of folders in folders is 10 folders |
| name | Name of the folder | No | Standard folders have predefined names that cannot be updated |
| mailboxId | ID of the mailbox the folder belongs to | No | Foreign key |

### SystemFetch model



**SystemFetch**

The SystemFetch entity is used in the operation of system fetch. Used when a recipient system is connected instead of a mailbox. All messages in the mailbox is fetched and send to the recipient system.

|  |  |  |  |
| --- | --- | --- | --- |
| **Column** | **Description** | **Required** | **Comment** |
| id | id of entity | Yes | The primary key |
| version | The current version | Yes | Used to ensure optimistic locking and is automatically increased by one on each update |
| createdDateTime |  |  |  |
| lastUpdated |  |  |  |
| finishedDateTime | Type of the folder | No | Stamped when system fetch has finished |
| systemFetchStatusType | Current status | Yes | RUNNING FINISHED STOPPED (manually stopped before being finished) |
| mailboxId | ID of the mailbox that is being system fetched | Yes |  |
| organisationId | ID of the organisation that is fetching | Yes | From system-registry |
| systemId | ID of the system that is being fetched to | Yes | From system-registry |
| contactPointId | ID of the contact point that is being fetched to | No | From system-registry |
| totalMessages | Total messages available for fetch | Yes |  |
| fetchedMessages | Number of messages (currently) fetched | No |  |
| failedMesssaged | Number of messages that failed | No |  |

**ActionStatus**

An ActionStatus entity for supplementary data describing the status of an Action in a document.

|  |  |  |  |
| --- | --- | --- | --- |
| **Column** | **Description** | **Required** | **Comment** |
| id | id of entity | Yes | The primary key |
| version | The current version | Yes | Used to ensure optimistic locking and is automatically increased by one on each update |
| createdDateTime |  |  |  |
| lastUpdated |  |  |  |
| statusCodeLastUpdated | Timestamp to track when the status code last updated. | No |  |
| statusCode | Status of the action | Yes | RECEIVED  STARTED  COMPLETED  FAILED\_WITH\_ERROR  CANCELLED |
| note | Text to describe the action status | No |  |
| additionalActionStatusData | A list describing additional status data e.g. earlier states. | No |  |

**AdditionalActionStatusData**

An AdditionalActionStatusData list for supplementary data additonaly describing status data e.g. earlier states.

|  |  |  |  |
| --- | --- | --- | --- |
| **Column** | **Description** | **Required** | **Comment** |
| id | id of entity | Yes | The primary key |
| version | The current version | Yes | Used to ensure optimistic locking and is automatically increased by one on each update |
| createdDateTime |  |  |  |
| lastUpdated |  |  |  |
| label | A label to describe the additional description e.g title | No |  |
| value | Value of the label | No |  |
| additionalActionStatusDataType | Text to describe the action status | Yes | TECHNICAL  END\_USER\_PRESENTABLE |

## Querying for Messages

For description of common search functionality, please revisit the section **Querying and searching resources**.

Querying messages is done using a GET request to the */mailboxes/{id}/messages/* endpoint.

The result is a messageSearchResult, which looks like this in JSON:

|  |
| --- |
| {    "currentPage": 0,    "totalPages": 1,    "elementsOnPage": 1,    "totalElements": 1,    "messages": []  } |

### Searching

Besides, and in combination with, the general searching described above, the following parameters can be used:

|  |  |
| --- | --- |
| folderId | One or more UUIDs of the folder(s) to filter by. Either separated by comma of by repeating the request parameter.  Folders can be found using GET /mailboxes/{id}/folders/ |
| dateFrom | Used to search for messages created on or after this date. Format: yyyy-MM-dd. |
| dateTo | Used to search for messages created on or before this date. Format: yyyy-MM-dd. |

**Examples**

**Folder:**

|  |
| --- |
| /mailboxes/{id}/messages/?folderId=5d2e4e6a-40dc-477a-ade9-5bd53afd1d3e,5e8c5915-3433-4adf-b80c-c8f94e198284  /mailboxes/{id}/messages/?folderId=5d2e4e6a-40dc-477a-ade9-5bd53afd1d3e&folderId=5e8c5915-3433-4adf-b80c-c8f94e198284 |

returns messages located in in either of the two given folders.

**Date range:**

|  |
| --- |
| /mailboxes/{id}/messages/?dateFrom=2020-04-01&dateTo=2020-04-15 |

searches for messages created between 2020-04-01 and 2020-04-15 - both included.

## Common use case examples

### Update access

The If-Match header must be set to the version of the Access ressource. The latest can be fetched using

*GET /mailboxes/8deb553b-0536-4671-9c9e-239f202d56e0/accesses/* ce65def2-eb5b-4d4d-86da-4f70ffe9f6e1 *-* in this case 3 as shown in response:

|  |
| --- |
| {  "id": "ce65def2-eb5b-4d4d-86da-4f70ffe9f6e1",  "version": 3,  "transactionId": "EnIowtKlvFhKtjc5UJkmsoqiphRf2jyO",  "createdDateTime": "2020-07-03T08:24:15.230Z",  "lastUpdated": "2020-07-03T08:32:33.173Z",  "accessType": "OWNER",  "mailboxId": "8deb553b-0536-4671-9c9e-239f202d56e0",  "introductionCompleted": **false**,  "emailNotificationSubscriptions": []  } |

Adding smsNotificationSubscription and emailNotificationSubscriptions:

PUT */mailboxes/8deb553b-0536-4671-9c9e-239f202d56e0/accesses/ce65def2-eb5b-4d4d-86da-4f70ffe9f6e1*

|  |
| --- |
| {  "introductionCompleted": **false**,  "smsNotificationSubscription": {  "mobileNumber": "29892630"  },  "emailNotificationSubscriptions": [  {  "email": "test@nc.dk"  }  ]  } |

Updating the email-address:

PUT */mailboxes/8deb553b-0536-4671-9c9e-239f202d56e0/accesses/ce65def2-eb5b-4d4d-86da-4f70ffe9f6e1*

|  |
| --- |
| {  "introductionCompleted": **false**,  "smsNotificationSubscription": {  "id": "8e2b1cf0-1b3c-4db6-950a-663f26209f3d",  "version": 0,  "unlistedNumber": **false**,  "mobileNumber": "29892630"  },  "emailNotificationSubscriptions": [  {  "id": "87052e65-67da-4bbc-bbd7-ecd78cfa1928",  "version": 0,  "email": "test2@nc.dk"  }  ]  } |

### Create folder

Endpoint for creation of folders:

|  |
| --- |
| /mailboxes/{mailboxId}/folders/ |

Example:

POST */mailboxes/8deb553b-0536-4671-9c9e-239f202d56e0/folders/*

|  |
| --- |
| {  "folderType": "USER\_DEFINED",  "name": "My archive"  } |

Create a folder with a parent folder:

|  |
| --- |
| {  "parentFolderId" : "0cd94f1a-4e6b-4065-8f1c-ac95371df03b",  "folderType": "USER\_DEFINED",  "name": "Sub folder"  } |

### Create draft message

Endpoint for creation of a draft message:

|  |
| --- |
| POST  /mailboxes/{mailboxId}/messages/ |

**Example:**  
POST */mailboxes/8deb553b-0536-4671-9c9e-239f202d56e0/messages/*

Create a draft message with default values.

This *empty* request will construct a message, placed in the default folder DRAFTS, with one main html document ready for content:

|  |
| --- |
| {  } |

Returns:

|  |
| --- |
| {  "id": "a00aca0a-920c-4a81-945a-1ecddca52964",  "version": 0,  "mailboxId": "8deb553b-0536-4671-9c9e-239f202d56e0",  "folderId": "767442c8-9b93-41f0-8ced-509c397ab934",  "transactionId": "EpYjn8kP0N7wVASwtRTEWYCgrbjyM0US",  "createdDateTime": "2020-08-17T17:41:26.634Z",  "lastUpdated": "2020-08-17T17:41:26.634Z",  "messageType": "REGULAR",  "forward": **false**,  "reply": **false**,  "flag": **false**,  "legallyNotified": **false**,  "read": **false**,  "welcomeMessage": **false**,  "errorMessage": **false**,  "sender": {  "id": "98776603-058d-46aa-afaf-8b3b7381ca47",  "version": 0,  "senderId": "0103785457",  "senderIdType": "CPR"  },  "replyData": [],  "documents": [  {  "id": "f63fc5b8-9651-4aa9-b131-8eeb05d85ee5",  "version": 0,  "documentType": "MAIN",  "label": "Hoveddokument",  "files": [  {  "id": "cbd20048-194c-42fa-aa3a-fa38191047cd",  "version": 0,  "encodingFormat": "text/html",  "filename": "hoveddokument.html",  "language": "da"  }  ],  "actions": []  }  ]  } |

### Update draft message

This example shows how to add recipient data and label to a draft message.

A draft message can be created as described in the “Create draft message”.  
You add the recipient data below the “sender” data. Below you can see an example on how the recipient data looks with its data fields and then the draft with the recipient data added to it. Please note if the recipient contains recipientIdtype: CVR this recipient must be present in Digital Post, otherwise the update will fail in validation.  
Above the “sender” you can add the label, which also is shown before added to draft message.

|  |
| --- |
| "recipient": {  "recipientId": "44556679",  "recipientIdType": "CVR"  }, |

|  |
| --- |
| "label": "Hejsa Lone Defaultssen", |

|  |
| --- |
| {  "id": "2ea949fe-ebf3-4035-926c-7dea3dc0c6b5",  "version": 0,  "mailboxId": "e000a1d2-2933-44a6-a126-071ccdb4e090",  "folderId": "a3fe6924-aac1-46e9-bcb4-1138b14a19a0",  "transactionId": "EyLyCQBRxwxyFsfb4bzJrSXeDUiLCsvO",  "createdDateTime": "2021-02-10T13:55:06.914Z",  "lastUpdated": "2021-02-10T13:55:06.914Z",  "messageType": "REGULAR",  "forward": **false**,  "reply": **false**,  "flag": **false**,  "legallyNotified": **false**,  "read": **true**,  "welcomeMessage": **false**,  "errorMessage": **false**,  "label": "Hejsa Lone Defaultssen",  "sender": {  "id": "64bad6cd-9d26-4b6d-866d-4f689c0e0107",  "version": 0,  "senderId": "2307921515",  "senderIdType": "CPR",  "label": "Lone 2076524521 Defaultssen"  },  "recipient": {  "id": "34abe116-424f-478c-ac21-e583c7415945",  "version": 0,  "recipientId": "44556679",  "recipientIdType": "CVR"  },  "replyData": [],  "documents": [  {  "id": "4d253d55-ac00-4332-9b2f-fd4b269a164b",  "version": 0,  "documentType": "MAIN",  "label": "Hoveddokument",  "files": [  {  "id": "b8509a8f-d454-4cb2-b9de-b6d86061278b",  "version": 0,  "encodingFormat": "text/html",  "filename": "hoveddokument.html",  "language": "da"  }  ],  "actions": []  }  ]  } |

You will need to send the draft with the recipient data and label as a put request to:

|  |
| --- |
| PUT  /mailboxes/{mailboxId}/messages/{messageId}    example:  /mailboxes/e000a1d2-2933-44a6-a126-071ccdb4e090/messages/2ea949fe-ebf3-4035-926c-7dea3dc0c6b5 |

The last part of the URL needs to match your message ID on your draft you are updating and the If-Match header must be set to the draft version.  
  
Then the put request is done you have added recipient data and label to your draft and the response will look like this with the version number of the draft updated:

|  |
| --- |
| {  "id": "2ea949fe-ebf3-4035-926c-7dea3dc0c6b5",  "version": 1,  "mailboxId": "e000a1d2-2933-44a6-a126-071ccdb4e090",  "folderId": "a3fe6924-aac1-46e9-bcb4-1138b14a19a0",  "transactionId": "EyM2UOwkrnoeJtc2DHMBvCMqx3GGEnG0",  "createdDateTime": "2021-02-10T13:55:06.914Z",  "lastUpdated": "2021-02-10T14:27:25.962Z",  "messageType": "REGULAR",  "label": "Hejsa Lone Defaultssen",  "forward": **false**,  "reply": **false**,  "flag": **false**,  "legallyNotified": **false**,  "read": **true**,  "welcomeMessage": **false**,  "errorMessage": **false**,  "sender": {  "id": "64bad6cd-9d26-4b6d-866d-4f689c0e0107",  "version": 0,  "senderId": "2307921515",  "senderIdType": "CPR",  "label": "Lone 2076524521 Defaultssen"  },  "recipient": {  "id": "34abe116-424f-478c-ac21-e583c7415945",  "version": 1,  "recipientId": "44556679",  "recipientIdType": "CVR"  },  "replyData": [],  "documents": [  {  "id": "4d253d55-ac00-4332-9b2f-fd4b269a164b",  "version": 0,  "documentType": "MAIN",  "label": "Hoveddokument",  "files": [  {  "id": "b8509a8f-d454-4cb2-b9de-b6d86061278b",  "version": 0,  "encodingFormat": "text/html",  "filename": "hoveddokument.html",  "language": "da"  }  ],  "actions": []  }  ]  } |

### Upload content to a file resource

When writing a draft or replying to a Message you must add the file content to a File resource present in a Document as part of your Message.

It is done using a PUT multipart request. In the below example using the endpoint:

|  |
| --- |
| PUT  /mailboxes/{mailboxId}/messages/{messageId}/documents/{documentId}/files/{fileId}/content    Example:  PUT  /mailboxes/e7c96848-2e39-4cfa-b10e-3347a68ba022/messages/7ebd24ea-78e8-443d-ad1c-8289d3f50c99/documents/5f74111f-e189-4955-99ec-f44e1a60ed6a/files/27337891-de83-4f59-812c-3964c248a14e/content |

With the content fil as body

|  |
| --- |
| {  "id": "7ebd24ea-78e8-443d-ad1c-8289d3f50c99",  "version": 1,  "mailboxId": "e7c96848-2e39-4cfa-b10e-3347a68ba022",  "folderId": "f9d69b81-e0ba-4c5c-b959-f9241677b888",  "transactionId": "Ex3rp8PRW3oMY4RKE8McXiUF1AT8oFqj",  "createdDateTime": "2021-01-15T14:17:31.579Z",  "lastUpdated": "2021-01-15T14:22:39.818Z",  "messageType": "REGULAR",  "forward": **false**,  "reply": **false**,  "flag": **false**,  "legallyNotified": **false**,  "read": **true**,  "welcomeMessage": **false**,  "errorMessage": **false**,  "sender": {  "id": "51d1b490-ec9f-4cf1-98ab-f36927aa42ee",  "version": 0,  "senderId": "69832541",  "senderIdType": "CVR",  "label": "Mejerby Standardbank"  },  "replyData": [],  "documents": [  {  "id": "5f74111f-e189-4955-99ec-f44e1a60ed6a",  "version": 0,  "documentType": "MAIN",  "label": "Hoveddokument",  "files": [  {  "id": "27337891-de83-4f59-812c-3964c248a14e",  "version": 1,  "encodingFormat": "text/html",  "filename": "hoveddokument.html",  "language": "da"  }  ],  "actions": []  }  ]  } |

The If-Match header must be set to the version of the File resource.

The name of the form element must be ‘file’. Here is a Curl example:

|  |
| --- |
| curl --location --request PUT 'https://test.digitalpost.dk/apis/v1/mailboxes/e7c96848-2e39-4cfa-b10e-3347a68ba022/messages/7ebd24ea-78e8-443d-ad1c-8289d3f50c99/documents/5f74111f-e189-4955-99ec-f44e1a60ed6a/files/27337891-de83-4f59-812c-3964c248a14e/content' \  --header 'If-Match: 1' \  --header 'Authorization: Bearer eyJhbG...xmoiA' \  --form 'file=@"/data/documents/hoveddokument.html"' |

The response is a structure containing the new version, to be used for further PUTs - for instance for use in auto save.

|  |
| --- |
| {  "id": "27337891-de83-4f59-812c-3964c248a14e",  "version": 2,  "fileSize": 415  } |

The client must NOT base64 encode the file content. The maximum size of the file is 10 MB.

### Send message

When you want to send a message, you need to send a post request to:

|  |
| --- |
| POST /mailboxes/{mailboxId}/messages/{messageId}/send    Example:  /mailboxes/e000a1d2-2933-44a6-a126-071ccdb4e090/messages/2ea949fe-ebf3-4035-926c-7dea3dc0c6b5/send |

The first UUID after */mailboxes/* should be your mailboxs ID and the UUID after */messagers/* should be the ID of the message you want to send.  
  
In this example we want to send this message:

|  |
| --- |
| {  "id": "2ea949fe-ebf3-4035-926c-7dea3dc0c6b5",  "version": 3,  "mailboxId": "e000a1d2-2933-44a6-a126-071ccdb4e090",  "folderId": "a3fe6924-aac1-46e9-bcb4-1138b14a19a0",  "transactionId": "EyMBTksE80UnyX61UQMsN1CoJ8TDn7x7",  "createdDateTime": "2021-02-10T13:55:06.914Z",  "lastUpdated": "2021-02-10T15:35:09.140Z",  "messageType": "REGULAR",  "label": "Hejsa Lone Defaultssen",  "forward": **false**,  "reply": **false**,  "flag": **false**,  "legallyNotified": **false**,  "read": **true**,  "welcomeMessage": **false**,  "errorMessage": **false**,  "sender": {  "id": "64bad6cd-9d26-4b6d-866d-4f689c0e0107",  "version": 0,  "senderId": "2307921515",  "senderIdType": "CPR",  "label": "Lone 2076524521 Defaultssen"  },  "recipient": {  "id": "34abe116-424f-478c-ac21-e583c7415945",  "version": 1,  "recipientId": "44556679",  "recipientIdType": "CVR"  },  "replyData": [],  "documents": [  {  "id": "4d253d55-ac00-4332-9b2f-fd4b269a164b",  "version": 0,  "documentType": "MAIN",  "label": "Hoveddokument",  "files": [  {  "id": "b8509a8f-d454-4cb2-b9de-b6d86061278b",  "version": 2,  "encodingFormat": "text/html",  "filename": "hoveddokument.html",  "language": "da"  }  ],  "actions": []  }  ]  } |

And the response we got looks like this:

|  |
| --- |
| {  "id": "2ea949fe-ebf3-4035-926c-7dea3dc0c6b5",  "version": 4,  "mailboxId": "e000a1d2-2933-44a6-a126-071ccdb4e090",  "folderId": "6e6654b7-3494-4c94-8b82-6905ff601957",  "transactionId": "EyMFJModgMQZRievm6gd9vVR5CF5usgJ",  "createdDateTime": "2021-02-10T13:55:06.914Z",  "lastUpdated": "2021-02-10T16:04:01.369Z",  "messageType": "REGULAR",  "label": "Hejsa Lone Defaultssen",  "forward": **false**,  "reply": **false**,  "flag": **false**,  "legallyNotified": **false**,  "read": **true**,  "welcomeMessage": **false**,  "errorMessage": **false**,  "sender": {  "id": "64bad6cd-9d26-4b6d-866d-4f689c0e0107",  "version": 0,  "senderId": "2307921515",  "senderIdType": "CPR",  "label": "Lone 2076524521 Defaultssen"  },  "recipient": {  "id": "34abe116-424f-478c-ac21-e583c7415945",  "version": 1,  "recipientId": "44556679",  "recipientIdType": "CVR"  },  "replyData": [],  "sendDateTime": "2021-02-10T16:04:01.329Z",  "documents": [  {  "id": "4d253d55-ac00-4332-9b2f-fd4b269a164b",  "version": 0,  "documentType": "MAIN",  "label": "Hoveddokument",  "files": [  {  "id": "b8509a8f-d454-4cb2-b9de-b6d86061278b",  "version": 2,  "encodingFormat": "text/html",  "filename": "hoveddokument.html",  "language": "da"  }  ],  "actions": []  }  ]  } |

### Reply to message

Endpoint for replying to a message:

|  |
| --- |
| POST  /mailboxes/{mailboxId}/messages/{messageId}/reply    Example:  POST /mailboxes/8deb553b-0536-4671-9c9e-239f202d56e0/messages/8deb553b-0536-4671-9c9e-239f202d56e0/reply |

Will create a draft message with original sender as recipient, filled replyData, and a main document ready for content. Returns:

|  |
| --- |
| {  "id": "912a097a-1556-433b-8a2a-19e2f0c4c514",  "version": 0,  "mailboxId": "8deb553b-0536-4671-9c9e-239f202d56e0",  "folderId": "767442c8-9b93-41f0-8ced-509c397ab934",  "transactionId": "EpYk1qI5rYy74mjDKtBoQpclcBBMP7ik",  "createdDateTime": "2020-08-17T17:43:13.799Z",  "lastUpdated": "2020-08-17T17:43:13.799Z",  "messageType": "REGULAR",  "label": "Sv: Pladsanvisning",  "forward": **false**,  "reply": **false**,  "flag": **false**,  "legallyNotified": **false**,  "read": **false**,  "welcomeMessage": **false**,  "errorMessage": **false**,  "sender": {  "id": "ab7f2b42-1fb6-408d-b582-67c7a14eb96c",  "version": 0,  "senderId": "0103785457",  "senderIdType": "CPR"  },  "recipient": {  "id": "294edf0a-d4c2-438f-b6d4-679cd689950e",  "version": 0,  "recipientId": "24586369",  "recipientIdType": "CVR",  "label": "Kommunen",  "attentionData": {  "id": "e5fc3c88-9c87-4723-8e57-cb8c0d40c2db",  "version": 0,  "personId": "9000001234",  "personLabel": "Hans Hansen",  "productionUnitNumber": "1234567890",  "productionUnitName": "Produktionsenhed A",  "globalLocationNumber": "5798000012345",  "location": "Kommune A",  "emailAddress": "info@bornehaven.dk",  "relatedEmailAgent": "Hans Jensen",  "seNumber": "24586369",  "seCompanyName": "Kommune",  "telephoneNumber": "12345678",  "relatedTelephoneAgent": "Ib Jensen",  "ridNumber": "CVR:24586369-RID:1234567890123",  "ridCompanyName": "Virksomhed",  "contentResponsibleId": "22334455",  "contentResponsibleLabel": "Børnehaven, rød stue",  "generatingSystemId": "Sys-1234",  "generatingSystemLabel": "KommunaltPostSystem",  "address": {  "id": "80c31b0f-1369-474c-8b2f-36a148f89a27",  "version": 0,  "addressId": "8c2ea15d-61fb-4ba9-9366-42f8b194c852",  "addressLabel": "Gaden",  "houseNumber": "7A",  "door": "th",  "floor": "3",  "co": "C/O",  "zipCode": "9000",  "city": "Aalborg",  "country": "DK",  "crsIdentifier": "EPSG:25832",  "geographicEastingMeasure": "557501.23",  "geographicNorthingMeasure": "6336248.89",  "geographicHeightMeasure": "0.0"  }  }  },  "replyData": [  {  "id": "e4128215-19aa-4f24-ae8d-c1a91ecccef6",  "version": 0,  "childMessageId": "912a097a-1556-433b-8a2a-19e2f0c4c514",  "parentMessageId": "8deb553b-0536-4671-9c9e-239f202d56e0"  }  ],  "documents": [  {  "id": "f4abea17-ef6c-443d-96f9-928a5ee87a36",  "version": 0,  "documentType": "MAIN",  "label": "Hoveddokument",  "files": [  {  "id": "3efe49c3-28b8-4e01-a41f-d84291b25580",  "version": 0,  "encodingFormat": "text/html",  "filename": "hoveddokument.html",  "language": "da"  }  ],  "actions": []  }  ]  } |

### Update one or more of certain predetermined fields of message (PATCH)

The If-Match header must be set to the version of the Message ressource.

Please note if the recipient contains recipientIdtype: CVR this recipient must be present in Digital Post.

|  |
| --- |
| PATCH  /mailboxes/{mailboxId}/messages/{messageId}    Example:  PATCH /mailboxes/8deb553b-0536-4671-9c9e-239f202d56e0/messages/2ea949fe-ebf3-4035-926c-7dea3dc0c6b5 |

Example on body:

|  |
| --- |
| {  "folderId": "3fa85f64-5717-4562-b3fc-2c963f66afa6",  "messageCode": "string",  "label": "string",  "flag": **true**,  "legallyNotified": **false**,  "read": **true**,  "recipient": {  "id": "3fa85f64-5717-4562-b3fc-2c963f66afa6",  "version": 0,  .... full recipient with attentiondata, contactpoint etc.  }  },  "note": "string"  } |

If an element is left out or set to null that element is not used in the patch.

|  |
| --- |
| {  "flag": **true**  } |

The above only sets flag and does not alter any of the other possible values.

|  |
| --- |
| {  "flag": **true**  "legallyNotified": **null**,  "read": **null**  } |

The above does the exact same thing - only sets flag does not alter any of the other possible values.

### Move message between folders

Lets say we have a message in a mailbox with values:

|  |  |
| --- | --- |
| Mailbox id | 8deb553b-0536-4671-9c9e-239f202d56e0 |
| Message id | 61768cb7-7b46-4f4f-a980-fc1c8206a6ef |
| Folder id | 3fa85f64-5717-4562-b3fc-2c963f66afa6 |
| New folder id | 9f790fc1-7d58-4d5e-9af3-11bce02b2308 |

The message is moved between folders by updating and switching folderId. Example using PATCH:

|  |
| --- |
| PATCH  /mailboxes/{mailboxId}/messages/{messageId}    Example:  PATCH /mailboxes/8deb553b-0536-4671-9c9e-239f202d56e0/messages/61768cb7-7b46-4f4f-a980-fc1c8206a6ef |

With body

|  |
| --- |
| {  "folderId": "9f790fc1-7d58-4d5e-9af3-11bce02b2308"  } |

There a some rules for moving a message based on the messageState:

* A RECEIVED message cannot be moved to SENT, DRAFTS
* A RECEIVED can be moved to INBOX, DELETED, USER\_DEFINED
* A DRAFT message cannot be moved to SENT, INBOX
* A DRAFT can be moved to DRAFTS, DELETED, USER\_DEFINED
* A SENT message cannot be moved to INBOX, DRAFTS
* A SENT can be moved to SENT, DELETED, USER\_DEFINED

If an invalid move is attempted, the request is considered BAD (400) and the following error code is returned:

|  |
| --- |
| message.update.folderType.not.allowed |

### Forward message to e-mail address

Forwarding is done POSTing a forward command to:

|  |
| --- |
| POST  /mailboxes/{mailboxId}/messages/{messageId}/forward    Example:  POST /mailboxes/69a35cf0-01bc-4e2b-849b-08e0d059692b/messages/cb36a59f-ffef-41e7-b2d9-df136b56f07a/forward |

With body:

|  |
| --- |
| {  "recipientId": "hans@netcompany.com",  "recipientIdType": "EMAIL",  "comment": "Hi Hans. Check out this message I got from Børneforvaltningen",  "senderLabel": "Flemming Jensen"  } |

The response is a 201 Created with the forwarded message:

|  |
| --- |
| {  "id": "81dc6d2d-c622-4218-8af8-73c520996ca3",  "version": 0,  "mailboxId": "69a35cf0-01bc-4e2b-849b-08e0d059692b",  "folderId": "1705e88c-c147-4e2c-a49e-ff30ad51ca73",  "transactionId": "EnYGjNfvVYEYuYoXkiNVexOshxPjaf6c",  "createdDateTime": "2020-07-08T08:47:40.135Z",  "lastUpdated": "2020-07-08T08:47:40.135Z",  "messageType": "REGULAR",  "messageIdentifier": "MSG-12345",  "messageCode": "Besked fra Børneforvaltningen",  "memoCreatedDateTime": "2020-04-23T12:00:00.000Z",  "receivedDateTime": "2020-07-08T08:40:39.453Z",  "label": "Pladsanvisning",  "forward": **true**,  "reply": **true**,  "flag": **false**,  "legallyNotified": **false**,  "read": **false**,  "welcomeMessage": **false**,  "errorMessage": **false**,  "sender": {  "id": "43b3942e-1548-4b97-b7ba-bc9b69879f38",  "version": 0,  "senderId": "0103785457",  "senderIdType": "CPR",  "label": "Flemming Jensen"  },  "recipient": {  "id": "9a82031d-4ef4-4e3f-8068-6eccbed0a5e7",  "version": 0,  "recipientId": "hans@netcompany.com",  "recipientIdType": "EMAIL"  },  "replyData": [],  "forwardData": {  "id": "9b6170cf-b90c-46a9-9f8d-bde375730879",  "version": 0,  "originalMessageId": "cb36a59f-ffef-41e7-b2d9-df136b56f07a",  "originalSender": "24586369",  "originalContentResponsible": "Kommunen",  "comment": "Hi Hans. Check out this message I got from Børneforvaltningen",  "newMemoId": "08d958d5-7e02-4d6b-b6a4-6363a8a78828"  },  "contentData": {  "id": "cf0cda8f-96e6-4312-824b-23cbc0aae4d3",  "version": 0,  "cprDataCprNumber": "2512169991",  "cprDataName": "Emilie",  "cvrDataCvrNumber": "5493833",  "cvrDataCompanyName": "Hansen A/S",  "motorVehicleLicenseNumber": "HJ93851",  "motorVehicleChassisNumber": "HJ93851HJ93851HJ93851",  "propertyNumber": "58J",  "caseId": "TY-3473652222",  "caseSystem": "E-journal",  "kleDataSubjectKey": "874kj7d6d82a",  "kleDataVersion": "12",  "kleDataActivityFacet": "Visitation",  "kleDataLabel": "Pladshenvisning",  "formDataTaskKey": "83798972311d",  "formDataVersion": "3",  "formDataActivityFacet": "B",  "formDataLabel": "Visitation",  "productionUnitNumber": 10,  "productionUnitName": "Afdeling A",  "educationCode": "AB47236",  "educationName": "RUC",  "address": {  "id": "742a5a26-5190-4559-b5b0-7f5301dfb188",  "version": 0,  "addressId": "da1c15bb-f74d-4a26-8617-4fbb5ac4f063",  "addressLabel": "Søen",  "houseNumber": "45",  "door": "tv",  "floor": "0",  "co": "AB1",  "zipCode": "5000",  "city": "Odense",  "country": "DK"  },  "additionalContentData": [  {  "id": "3ddd2c4d-c958-429b-9a92-66eed80b62ec",  "version": 0,  "contentDataType": "En type",  "contentDataName": "Afdeling",  "contentDataValue": "Inddrivelse"  }  ]  },  "sendDateTime": "2020-07-08T08:47:40.133Z",  "documents": [  {  "id": "105498e1-5f71-4255-bf6c-d822b1e52e60",  "version": 0,  "documentType": "MAIN",  "documentId": "456",  "label": "Tilbud om børnehaveplads",  "files": [  {  "id": "1f9dd3db-3beb-4c88-94b1-1f1aa66a2b12",  "version": 0,  "encodingFormat": "application/pdf",  "filename": "Pladsanvisning.pdf",  "language": "da"  },  {  "id": "2d2cb4c7-257a-440c-932e-7ba4b9897f4c",  "version": 0,  "encodingFormat": "text/plain",  "filename": "Pladsanvisning.txt",  "language": "da"  }  ],  "actions": [  {  "id": "f5f796ea-ac10-4232-b92f-fea216bb062f",  "version": 0,  "label": "Spørgeskema",  "actionStartTime": "2018-11-09T12:00:00.000Z",  "actionEndTime": "2018-12-09T12:00:00.000Z",  "entryPointUrl": "http://www.fstyr.dk/DA/Syn-af-Koretojer/Find-synshal.asp"  }  ]  }  ]  } |

### Forward message to trusted recipient and authority

Forwarding is done POSTing a forward command to:

|  |
| --- |
| POST  /mailboxes/{mailboxId}/messages/{messageId}/forward    Example:  POST /mailboxes/69a35cf0-01bc-4e2b-849b-08e0d059692b/messages/cb36a59f-ffef-41e7-b2d9-df136b56f07a/forward |

With body:

|  |
| --- |
| {  "recipientId": "2008083560",  "recipientIdType": "CPR",  "comment": "Hi Hans. Check out this message I got from Børneforvaltningen",  "senderLabel": "Flemming Jensen"  } |

recipientId can be a CVR of a trusted recipient or a Danish authority, in which case recipientIdType must be set to CVR.

The response is a 201 Created with the forwarded message in the body:

|  |
| --- |
| {  "id": "b78e8467-9703-44f7-8569-fa6027853625",  "version": 0,  "mailboxId": "69a35cf0-01bc-4e2b-849b-08e0d059692b",  "folderId": "1705e88c-c147-4e2c-a49e-ff30ad51ca73",  "transactionId": "EnYIiV4aAZHLIqUXZ9xK3sshq07b2mk3",  "createdDateTime": "2020-07-08T09:02:37.693Z",  "lastUpdated": "2020-07-08T09:02:37.693Z",  "messageType": "REGULAR",  "messageIdentifier": "MSG-12345",  "messageCode": "Besked fra Børneforvaltningen",  "memoCreatedDateTime": "2020-04-23T12:00:00.000Z",  "receivedDateTime": "2020-07-08T08:40:39.453Z",  "label": "Pladsanvisning",  "forward": **true**,  "reply": **true**,  "flag": **false**,  "legallyNotified": **false**,  "read": **false**,  "welcomeMessage": **false**,  "errorMessage": **false**,  "sender": {  "id": "d047ec77-4725-422c-b2eb-367450e82fd7",  "version": 0,  "senderId": "0103785457",  "senderIdType": "CPR",  "label": "Flemming Jensen"  },  "recipient": {  "id": "f31b8913-d59e-46b4-8349-294a034803d1",  "version": 0,  "recipientId": "2008083560",  "recipientIdType": "CPR"  },  "replyData": [],  "forwardData": {  "id": "2e366731-b478-475d-856d-67551605f433",  "version": 0,  "originalMessageId": "cb36a59f-ffef-41e7-b2d9-df136b56f07a",  "originalSender": "24586369",  "originalContentResponsible": "Kommunen",  "comment": "Hi Hans. Check out this message I got from Børneforvaltningen",  "newMemoId": "a855a07d-6c2a-4725-9962-8d1d6235103e"  },  "contentData": {  "id": "ceec9eca-e06c-4e0c-8621-7f140b4e087a",  "version": 0,  "cprDataCprNumber": "2512169991",  "cprDataName": "Emilie",  "cvrDataCvrNumber": "5493833",  "cvrDataCompanyName": "Hansen A/S",  "motorVehicleLicenseNumber": "HJ93851",  "motorVehicleChassisNumber": "HJ93851HJ93851HJ93851",  "propertyNumber": "58J",  "caseId": "TY-3473652222",  "caseSystem": "E-journal",  "kleDataSubjectKey": "874kj7d6d82a",  "kleDataVersion": "12",  "kleDataActivityFacet": "Visitation",  "kleDataLabel": "Pladshenvisning",  "formDataTaskKey": "83798972311d",  "formDataVersion": "3",  "formDataActivityFacet": "B",  "formDataLabel": "Visitation",  "productionUnitNumber": 10,  "productionUnitName": "Afdeling A",  "educationCode": "AB47236",  "educationName": "RUC",  "address": {  "id": "6be967af-8535-4396-b395-beb824e41ecd",  "version": 0,  "addressId": "da1c15bb-f74d-4a26-8617-4fbb5ac4f063",  "addressLabel": "Søen",  "houseNumber": "45",  "door": "tv",  "floor": "0",  "co": "AB1",  "zipCode": "5000",  "city": "Odense",  "country": "DK"  },  "additionalContentData": [  {  "id": "916a9be0-cfe9-41d5-b591-0727199a50a2",  "version": 0,  "contentDataType": "En type",  "contentDataName": "Afdeling",  "contentDataValue": "Inddrivelse"  }  ]  },  "sendDateTime": "2020-07-08T09:02:37.690Z",  "documents": [  {  "id": "a2e5de45-5a85-4059-8f0c-4af1c29e84ca",  "version": 0,  "documentType": "MAIN",  "documentId": "456",  "label": "Tilbud om børnehaveplads",  "files": [  {  "id": "ae332d9f-30b6-4db4-bd95-f61392096641",  "version": 0,  "encodingFormat": "text/plain",  "filename": "Pladsanvisning.txt",  "language": "da"  },  {  "id": "027c03ee-524b-4350-ba99-59f416e93d48",  "version": 0,  "encodingFormat": "application/pdf",  "filename": "Pladsanvisning.pdf",  "language": "da"  }  ],  "actions": [  {  "id": "43656262-cec3-4130-8c16-2243354e77f9",  "version": 0,  "label": "Spørgeskema",  "actionStartTime": "2018-11-09T12:00:00.000Z",  "actionEndTime": "2018-12-09T12:00:00.000Z",  "entryPointUrl": "http://www.fstyr.dk/DA/Syn-af-Koretojer/Find-synshal.asp"  }  ]  }  ]  } |

### ReplyData mail threads

The replyData of a message contains the entire reply history of the message.

In these examples the UUID is replaced with a letter to better illustrate the structure.

Say we have a message A:

|  |
| --- |
| {  "id": "A",  "replyData": [],  } |

When A is replied to, the reply, B, looks like this:

|  |
| --- |
| {  "id": "B",  "replyData": [  {  "childMessageId": "B",  "parentMessageId": "A"  }  ]  } |

When B is replied to, that reply, C, looks like this:

|  |
| --- |
| {  "id": "C",  "replyData": [  {  "childMessageId": "C",  "parentMessageId": "B"  },  {  "childMessageId": "B",  "parentMessageId": "A"  }  ]  } |

From this last message C we can traverse the child/parent relationships fra C to B and again from B to A, thus ending up with the root message A. If you want to find all messages of A’s mail thread you can query like this:

|  |
| --- |
| /mailboxes/{mailboxId}/messages/?replyData.parentMessageId=A |

### Write to the authorities

When you need to write a message to the authorities you need to take following steps:

* Create draft message
* Update draft with recipient data(CVR number)
* Upload content to a file resource
* Send message

The first step to take is creating a draft message, this can be done by following the example “create draft message” under “Common use case examples”.

The second step is the update the newly created draft with recipient data. There is an example called “Update draft”, which can be found under “Common use case examples”. The draft should be updated with recipient data, such as CVR number and label.

The third step is to upload content file to a file resource as part of your message. This can be done by following the example “Upload content to a file resource”, under “Common use case examples”. Beware that the maximum file size of the file may not be greater than 10 MB.

The last step is to send the message. Here too you can find an example and its named “send message” and can be found under “Common use case examples”.

### Examples of error messages

The error codes and error messages that can be returned from the mailbox are documented here: [Front-end validation and errorcodes in the Viewclient](https://digidp.atlassian.net/wiki/spaces/DP/pages/216989759/Front-end+validation+and+errorcodes+in+the+Viewclient). This page shows a few examples of some of these error messages.

## Uploading invalid html to a file

See [Upload content to a file resource](https://digidp.atlassian.net/wiki/spaces/DP/pages/482511480/Upload+content+to+a+file+resource) for description of how to upload bytes.

When adding content to a file the content is validated as html if the file’s encodingFormat is text/html. The html must adhere to a narrow whitelist of allowed html. See [HTML whitelist for document validation](https://digidp.atlassian.net/wiki/spaces/DP/pages/79331853/HTML+whitelist+for+document+validation) for description. If the html added is invalid the response to the PUT request will be 400 BAD REQUEST with a body like in the example below:

|  |
| --- |
| {  "code": "digital.post.error",  "message": "ValidationException: 61d4c728-353f-4b49-bee8-c26d8fe7addd can not be updated with html content",  "fieldErrors": [  {  "resource": "target",  "code": "html.sanitizer.rejected.element.attributes",  "message": "Elementet \"meta\" indeholder attribut \"http-equiv\" der ikke accepteres af Digital Post."  },  {  "resource": "target",  "code": "html.sanitizer.rejected.element",  "message": "Elementet \"a\" indeholder html der ikke accepteres af Digital Post."  }  ]  } |

The html file looked like this:

|  |
| --- |
| <**html**>  <**head**>  <**meta** charset="utf-8"/>  <**meta** http-equiv="content-type" content="text/html; charset=UTF-8"/>  <**meta** http-equiv="refresh" content="5"/>  <**title**>Test page</**title**>  </**head**>  <**body**>  <**h1**>Hello Digst</**h1**>  <**img** src="data:image/png;base64,iVBORw0KGgoAAAANSUhEUgAAA" alt="" />  <**h1** style="background-image:url(&#39;data:image/png;base64,cwAADsMAAA&#39;)">Hello</**h1**>  <**a** href="http://www.dr.dk">Danmarks Radio</**a**>  </**body**>  </**html**> |

Error codes beginning with “html.sanitizer.“ are all from the html-validation. The error message

"message": "Elementet \"meta\" indeholder attribut \"http-equiv\" der ikke accepteres af Digital Post."

tells us that <meta http-equiv="refresh" content="5"/> is not allowed.

The error message

"message": "Elementet \"a\" indeholder html der ikke accepteres af Digital Post."

is not as revealing but as the whitelist states:



“http“ is not allowed. It must be “https” in this case.

## Trying to send a message without content in the main document

|  |
| --- |
| {  "code": "digital.post.error",  "message": "ValidationException: Unable to send message",  "fieldErrors": [  {  "resource": "message",  "field": "documents",  "code": "message.send.documents.files.content.required",  "message": "Der skal tilføjes indhold til filen ba9d5d8c-a12b-4a78-b670-d5cbb2d938f8.",  "rejectedValue": [  {  "id": "b4403d29-e2d7-4e4c-9cea-7e5edc7bd624",  "version": 0,  "documentType": "MAIN",  "label": "Hoveddokument",  "files": [  {  "id": "ba9d5d8c-a12b-4a78-b670-d5cbb2d938f8",  "version": 0,  "encodingFormat": "text/html",  "filename": "hoveddokument.html",  "language": "da"  }  ],  "actions": []  }  ]  }  ]  } |

## Replying to an unreplyable message (reply = false)

|  |
| --- |
| {  "code": "digital.post.error",  "message": "ValidationException: A reply draft can not be created for message with id 5ef34405-60c4-4995-81c5-7cc3956fa165",  "fieldErrors": [  {  "resource": "message",  "field": "reply",  "code": "message.create.reply.notAllowed",  "message": "Denne meddelse kan ikke besvares.",  "rejectedValue": **false**  }  ]  } |

### Message state action matrix

The following describes the message state and the actions that are possible within that state.

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Message state** | | |
| **Action** | **RECEIVED** | **DRAFT** | **SENT** |
| Send | (minus) | (tick) | (minus) |
| Forward1 | (tick) | (minus) | (minus) |
| Reply2 | (tick) | (minus) | (minus) |
| Move3 | (tick) | (tick) | (tick) |
| Delete | (tick) | (tick) | (tick) |
| Update read/flag | (tick) | (tick) | (tick) |
| Add or update note | (tick) | (tick) | (tick) |
| Mark legally notified4 | (tick) | (minus) | (minus) |
| Add, update or remove  sender, recipient, documents, files and file content | (minus) | (tick) | (minus) |

1. Forward is only allowed if message.forward is true.
2. Reply is only allowed if message.reply is true.
3. A RECEIVED message, a DRAFT and a SENT message are all able to be moved to other folders. However, there are exceptions, as to which folders each type can be moved to. See the rules on the page [Move message between folders](https://digidp.atlassian.net/wiki/spaces/DP/pages/739705044/Move+message+between+folders) .
4. message.legallyNotified Cannot be changed once switched true.

# Event log services - TI

From the event log the following services are exposed. Only the index exposes data directly, the index is planned to only contain data from the last 3 month, currently the index support data back to when the solution went live. Due to lack of a way to access the individual events in the database.  
The database contains events from the last 5 years. Currently to access data from the database a report needs be be created through the usage statistics API. Where aggregated statistical data can be given.

The types of events exposed in the index are listed in “Event Log Index Events” and the rights needed to access those events are listed in the “Required roles“-column.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Service** | **URL** | **Data returned** | **Usage** | **Required roles** |
| Query event log | GET /events/ | List of Events | Used to find information changes related to the caller (sender-system or citizen). Common use cases are;   * Finding messages where the delivery failed * Finding information about business or technical receipts * Finding info on who changed contact-structure * Seeing what changes to have made to a resource   And many more. The events that are exposed in the event-log are filtered by context so what you can only see relevant events. | * Citizen * Organisation administrator * Legal Representative * Curator * Message employee * Message write * Message basic * Message log administrator * Action log administrator * Search log administrator * Citizen service employee * Erhvervsstyrelsen service employee * Contact administrator * Statistics administrator * System manager * Sender system * Delegated sender system * Recipient system * Delegated recipient system |

|  |
| --- |
| The Event log by default limits your query to the past 3 week. If you need to alter this behavior you must define a custom range. See section “Default and configurable search interval” for details. |

## Querying the event-log

For description of common search functionality, please revisit the section **Querying and searching resources**.

Unlike most of the others services in Digital Post, the event-log only exposes a service that is used to search for event.

The result is a paginated EventSearchResult, on the form as shown below. Default results per page is 100.

|  |
| --- |
| {  "currentPage": 0,  "next": "WyAxNjUxMTQ3MjA0NjgzNDkyLCAiRktLVFR0Y2hzdzViY2tXR24zMXJMWm9TeG1sUXo1Um0iIF0=",  "totalPages": 100,  "elementsOnPage": 100,  "totalElements": 10000,  "events": [...]  } |

### Querying

**Format**

The format for querying the event log followed the common format in Digital Post as shown below:

|  |
| --- |
| https://api.test.digitalpost.dk/apis/v1/events/?<parameter>=<value>  https://api.test.digitalpost.dk/apis/v1/events/?<parameter>=<value>,<value2>,<value3>  https://api.test.digitalpost.dk/apis/v1/events/?<parameter>=<value>&<parameter2>=<value2> |

All of the event-properties are searchable, and can therefore be combined as needed. However since the event-log does not contain CPR or CVR for events from before March 2023, it will convert these back and forth between the identity registry as needed, to avoid the user having to deal with identities. This results in the event-log being less robust for searches using a larger than default pagesize from before that date.

**Default and configurable search interval**

As default events created over the past 3 weeks will be returned when querying the event log.

If another time interval is needed there can be configured a specific interval using the 'dateFrom' and 'dateTo' parameters with a maximum range of 3 months.

**Exampel:**

|  |
| --- |
| /events/?dateFrom=2023-05-11T13:52:16.713Z&dateTo=2023-08-11T13:52:16.713Z |

**Using wildcard**

Querying the event log using the wildcard functionality is not supported. Instead, the wildcard characters, \* and ?, are interpreted as normal text characters and ss such it is still possible to find events containing these characters. For example, the query

|  |
| --- |
| /events/?field1.subField2=6c908813\* |

will only return events where the subField2 value matches 6c908813\*.

**Examples**

Example where we specify type=BUSINESS\_RECEIPTand only 2 results:

|  |
| --- |
| https://api.test.digitalpost.dk/apis/v1/events/?size=2&subject=BUSINESS\_RECEIPT |

Which results in an example output like this:

|  |
| --- |
| {  "currentPage": 0,  "next": "WyAxNjUxMTQ3MjA0NjgzNDkyLCAiRktLVFR0Y2hzdzViY2tXR24zMXJMWm9TeG1sUXo1Um0iIF0=",  "totalPages": 806,  "elementsOnPage": 2,  "totalElements": 1611,  "events": [  {  "id": "be015432-78e8-4eaf-9bf2-c1eb8ed255d7",  "eventId": "d4fc6350-741b-4894-8b07-b8fdab26cfc4",  "version": 0,  "transactionId": "F7Uwy9Erm6DTZbm70Bh4n5yEVXQ6k2KB",  "channel": "output",  "channels": [  "output"  ],  "subject": "BUSINESS\_RECEIPT",  "type": "FAILED\_REST",  "rootId": "665311dd-4fad-40f4-8911-78aa79989a97",  "eventTime": "2021-08-13T22:52:37.143770Z",  "created": "2021-08-13T22:52:37.706Z",  "owner": "44556682",  "actor": "eac1856b-d338-450b-91a5-4fba16fcc893",  "parentId": "7a08a78e-d51e-4c52-ba8b-b98dcdf6a3a0",  "system": {  "id": "02f51669-b18b-4c50-a1a9-2f8d7be8770b",  "name": "distribution-sender-rest",  "user": "eac1856b-d338-450b-91a5-4fba16fcc893"  },  "message": "2dcc1262-13b0-49eb-aaa9-c9dbb152b56b",  "eventProperties": {  "owner": "adfe1341-70da-46dd-8efb-4510be6de280",  "error-message": "I/O error on POST request for \"https://test.digitalpost.dk\": Connect to test.digitalpost.dk:443 [test.digitalpost.dk/10.1.77.102] failed: connect timed out; nested exception is org.apache.http.conn.ConnectTimeoutException: Connect to test.digitalpost.dk:443 [test.digitalpost.dk/10.1.77.102] failed: connect timed out",  "recipient-system-endpoint": "https://test.digitalpost.dk"  },  "metaProperties": {  "transmissionId": "2964ef1c-27da-416c-bb8c-3a2dd75957f6",  "size": "2542",  "messageType": "DIGITALPOST",  "senderSystem": "7b8323c5-2883-4623-80d4-57f7bb91d181",  "sender": "44556682",  "legalNotification": "false",  "messageUUID": "2dcc1262-13b0-49eb-aaa9-c9dbb152b56b",  "recipient": "99881128",  "title": "Test MeMo built with properties: longMessage: false, numberOfAdditionalAttachments: null, reply: false, contactPointId: null, replyByDateTime: null, action: false, legalNotification: false, mandatory: false",  "mandatory": "false"  },  "eventTag": "BUSINESS\_RECEIPT\_SENT"  },  {  "id": "09d014db-ccd8-47a6-a8b1-dace1b458f92",  "eventId": "f5d9e052-68a9-4b7b-955c-6b3e59cdd226",  "version": 0,  "transactionId": "F7Uwx8yFgvsjVteuFmDI4F9CEBQlbeIa",  "channel": "output",  "channels": [  "output"  ],  "subject": "BUSINESS\_RECEIPT",  "type": "FAILED\_REST",  "rootId": "4a1ce65f-1d74-48cf-a046-d42a9a41113a",  "eventTime": "2021-08-13T22:51:36.745203Z",  "created": "2021-08-13T22:51:37.374Z",  "owner": "44556682",  "actor": "eac1856b-d338-450b-91a5-4fba16fcc893",  "parentId": "1565d9e3-584e-4a24-b395-52410dd821cb",  "system": {  "id": "02f51669-b18b-4c50-a1a9-2f8d7be8770b",  "name": "distribution-sender-rest",  "user": "eac1856b-d338-450b-91a5-4fba16fcc893"  },  "message": "9cce7b32-362a-4b72-a2dd-d42af0f6adba",  "eventProperties": {  "owner": "adfe1341-70da-46dd-8efb-4510be6de280",  "error-message": "I/O error on POST request for \"https://test.digitalpost.dk\": Connect to test.digitalpost.dk:443 [test.digitalpost.dk/10.1.77.102] failed: connect timed out; nested exception is org.apache.http.conn.ConnectTimeoutException: Connect to test.digitalpost.dk:443 [test.digitalpost.dk/10.1.77.102] failed: connect timed out",  "recipient-system-endpoint": "https://test.digitalpost.dk"  },  "metaProperties": {  "transmissionId": "d2289476-3333-4cd8-a048-309330146b81",  "size": "2542",  "messageType": "DIGITALPOST",  "senderSystem": "7b8323c5-2883-4623-80d4-57f7bb91d181",  "sender": "44556682",  "legalNotification": "false",  "messageUUID": "9cce7b32-362a-4b72-a2dd-d42af0f6adba",  "recipient": "99881128",  "title": "Test MeMo built with properties: longMessage: false, numberOfAdditionalAttachments: null, reply: false, contactPointId: null, replyByDateTime: null, action: false, legalNotification: false, mandatory: false",  "mandatory": "false"  },  "eventTag": "BUSINESS\_RECEIPT\_SENT"  }  ]  } |

## Event Log Index Events

The event log persists and enriches events, send from the other components of Digital Post. These event ranges from simply describing an update of an object in one of the store components, to more business related cases such as if a legal message has been opened for the first time and therefore is “forkyndt”.

Events from different components looks slightly different, they are however always composed of certain elements:

* id: The ID of the event resource as it is stored in the database
* eventId: The id of the Event
* version: The version of the event, this is currently always 0 as events are immutable
* transactionId: The transaction ID linked to the transaction
* channel(s): currently all channels are output due to current implementation, this is an internal data point not relevant for external parties
* subject: The subject of the event, these are specific keys used for the logic on whether to persist the event.
* type: The type of the event, these are specific keys used for the logic on whether to persist the event.
* organisationIdentityId: The id of the owning organisation if the actor is an employee/system
* eventTime:Time stamp for when the event happened, always in UTC.
* created: Time stamp for when the event was saved to the eventlog, always in UTC.
* owner: Owner of the event (or resource that the event references), if the event was done by an employee or a system, the owner will be the CVR of the linked organisation or company, or the CPR of the citizen.
* ownerIdentityId: Identity Id of the Owner
* actor: Who is responsible for the event. In the example below, an employee sends a message, therefore he is the actor. If the actor is an employee in another organisation, then the CVR returned instead. E.g. if a citizen service employee modifies data that belongs to a citizen, the citizen cannot identify which exact employee did the change only the municipality that the employee is employed by.
* actorIdentityIdIdentity Id of the Actor
* system: Systeminformation about the component responsible for sending the event.
* message: The message is most often an ID. In this case it is an ID of the message being updated.
* eventProperties: The event properties are additional context supplied by the creator of the event, in this case a changelog and the ID of the mailbox that the message is in.
  + always contains a version property as shown in the example (subject to change).
* metaProperties: metaProperties are additional information that the event log tries to gather when it persists an event. As a default no addational information is added.
* searchEventProperties: Field mostly used to store information about
* eventTag: A tag describing both the subject and type in a single atribute eg. DRAFT\_SAVED

Examples of element in event log:

|  |
| --- |
| {  "id": "[UUID]",  "eventId": "[UUID]",  "version": 0,  "transactionId": "Fcj9pmGuNI4d6xuO5KXzirQEQNUKMOPQ",  "channel": "output",  "channels": [  "output"  ],  "subject": "MEMO",  "type": "VALIDATED",  "organisationIdentityId": "[UUID]",  "eventTime": "2023-05-03T11:22:10.744051Z",  "created": "2023-05-03T11:22:11.032Z",  "owner": "[CVR/CPR/UUID/RID]",  "ownerIdentityId": "[UUID]",  "actor": "[CVR/CPR/UUID/RID]",  "actorIdentityId": "[UUID]",  "system": {  "id": "[UUID]",  "name": "distribution-validator-single",  "user": "[UUID]"  },  "message": "[UUID]",  "eventProperties": {  "owner": "[UUID]",  "parent-event-message": "[UUID]#[UUID]"  },  "metaProperties": {  "recipientIdentityId": "[UUID]",  "numberOfAttachments": "2",  "messageId": "[STRING]",  "title": "[STRING]",  "mandatory": "false",  "senderSystemName": "[STRING]",  "senderIdentityId": "[UUID]",  "transmissionId": "[UUID]",  "recipientType": "CITIZEN",  "senderContactPointId": "[UUID]",  "size": "244380",  "messageType": "DIGITALPOST",  "senderSystem": "[UUID]",  "sender": "[CVR/CPR/UUID/RID]",  "legalNotification": "false",  "messageUUID": "[UUID]",  "recipient": "[CVR/CPR/UUID/RID]",  "senderType": "AUTHORITY",  "contentResponsible": "[STRING/CPR/CVR]"  "contentResponsibleIdentityId": "[UUID]""  },  "searchEventProperties": {},  "eventTag": "MEMO\_SEND\_VALIDATED"  } |

|  |
| --- |
| {  "id": "[UUID]",  "eventId": "[UUID]",  "version": 0,  "transactionId": "Fcj09sKuUILiOsgoPBgBsMANSpimyJzr",  "channel": "output",  "channels": [  "output"  ],  "subject": "MESSAGE",  "type": "UPDATED\_DRAFT",  "eventTime": "2023-05-03T10:09:16.572694Z",  "created": "2023-05-03T10:09:16.763Z",  "owner": "[CVR/CPR/UUID/RID]",  "ownerIdentityId": "[UUID]",  "actor": "[CVR/CPR/UUID/RID]",  "actorIdentityId": "[UUID]",  "system": {  "id": "[UUID]",  "name": "mailbox-store",  "user": "[UUID]"  },  "message": "[UUID]",  "eventProperties": {  "owner": "[UUID]",  "mailboxId": "[UUID]",  "message-id": "[STRING]",  "memo-id": "[UUID]",  "version": "5",  "client\_id": "borger-dk-web-post-visningsklient-oidc-demo-id"  },  "metaProperties": {  "recipientIdentityId": "[UUID]",  "messageType": "DIGITALPOST",  "sender": "[CVR/CPR/UUID/RID]",  "legalNotification": "false",  "messageUUID": "[UUID]",  "recipient": "[CVR/CPR/UUID/RID]",  "messageId": "[UUID]",  "title": "[STRING]",  "mandatory": "false",  "senderIdentityId": "[UUID]"  },  "searchEventProperties": {},  "eventTag": "DRAFT\_SAVED"  } |

### Fields of interest on events grouped on subject.

If the subject of the event matches, the following properties are attempted added to the metaProperties/eventProperties/searchEventProperties.

**Subject: MEMO:**

* eventProperties:
  + parent-event-message
  + nem-sms-recipient (IF NEMsms)
  + recipient (if forwarded to email)
* metaProperties
  + senderContactPointId: ID of sender contact point
  + recipientContactPointId: ID of recipient contact point
  + recipientContactPointName: Name of recipient contact point
  + legalNotification: If the receipt is about a “forkyndelse”
  + mandatory: If the related message was mandatory
  + messageId: MeMo message ID
  + messageUUID: Full ID of the MeMo
  + numberOfAttachments: number of attachments
  + recipient: ID of final recipient (fx a citizen)
  + sender: ID of the sender (fx. an organisation)
  + size: In bytes
  + title: Title of the message
  + senderSystemId: ID of the sender system
  + senderSystemName: Name of the sender system
  + recipientSystemId: ID of the recipient system
  + recipientSystemName: Name of the recipient system
  + contentResponsibleId: Content responsible in sender in memo
  + contentResponsibleIdentityId: Identity Id of content responsible if cpr/cvr

**Subject: MESSAGE:**

* eventProperties:
  + nem-sms-recipient (IF NEMsms)
  + recipient
  + mailboxId
  + message-id
  + client\_id
* metaProperties
  + senderContactPointId: ID of sender contact point
  + recipientContactPointId: ID of recipient contact point
  + recipientContactPointName: Name of recipient contact point
  + legalNotification: If the receipt is about a “forkyndelse”
  + mandatory: If the related message was mandatory
  + messageId: MeMo message ID
  + messageUUID: Full ID of the MeMo
  + numberOfAttachments: number of attachments
  + recipient: ID of final recipient (fx a citizen)
  + sender: ID of the sender (fx. an organisation)
  + size: In bytes
  + title: Title of the message
  + senderSystemId: ID of the sender system
  + senderSystemName: Name of the sender system
  + recipientSystemId: ID of the recipient system
  + recipientSystemName: Name of the recipient system
  + contentResponsibleId: Content responsible in sender in memo
  + contentResponsibleIdentityId: Identity Id of content responsible if cpr/cvr

**Subject: TECHNICAL\_RECEIPT:**

Technical Receipts are only sent for SFTP and SMTP memo messages

* eventProperties:
  + receipt-status
  + mime-subject
  + sender-system-smtp-endpoint
* metaProperties
  + senderSystemSmtpEndpoint
  + messageId
  + title
  + receiptStatus

**Subject: BUSINESS\_RECEIPT:**

* eventProperties:
  + error-message
  + sender-system-id
  + receipt-status
  + failure-id
  + parent-event-type
  + parent-event-subject
  + parent-event-message
  + sender-system-receipt-endpoint
* metaProperties
  + numberOfAttachments
  + legalNotification
  + mandatory
  + messageUUID: Full ID of the MeMo
  + receiptStatus: Status of the receipt
  + recipient: ID of final recipient
  + sender: ID of the sender (fx. an organisation)
  + senderSytemId: ID of the sender system
  + senderSystemName: Name of the sender system
  + senderSystemReceiptEndpoint
  + size: In bytes
  + title: Title of the message
  + transmissionId: generated uuid, is the same for both technical and business receipt

**Subject: MAILBOX :**

* eventProperties:
  + mailboxId

**Subject: ACCESS :**

* eventProperties:
  + mailboxId
  + clientId

**Subject: ACCESS\_REQUEST :**

* eventProperties:
  + privileges : "[ORGANISATION\_USER\_ADMINISTRATOR]"
  + access-to : "d19c076b-71f8-4394-b530-8040362d7688"
  + version : "0"
  + access-request-type: "USER\_ADMIN\_LOST\_PRIVILEGE\_REQUEST"

**Subject: FOLDER :**

* eventProperties:
  + mailboxId
  + clientId

**Subject: NOTIFICATION :**

eventProperties:

* eventProperties:
  + mailboxId
  + recipient
  + messageId

**Subject: REGISTRATION\_STATUS :**

* eventProperties
  + registration-status:
  + previous-registration-status

**Subject: EMAIL\_NOTIFICATION\_SUBSCRIPTION/**  
**SMS\_NOTIFICATION\_SUBSCRIPTION/**  
**PUSH\_NOTIFICATION\_SUBSCRIPTION:**

* eventProperties
  + mailboxId
  + channel (EMAIL/PHONE OF SUBSCRIPTION)
  + changelog (if changed)

**Subject: NEM\_SMS:**

* eventProperties
  + changelog (if changed)

**Subject: CONTACT:**

* eventProperties
  + changelog (if changed)

**Subject: SYSTEM:**

* eventProperties
  + changelog (if changed)
  + resourceName
  + organisation (id)

**Subject: CONTACT\_GROUP:**

* eventProperties
  + changelog (if changed)
  + resourceName (name of group)
  + organisation (id)
  + parentGroupId

**Subject: CONTACT\_POINT:**

* eventProperties
  + changelog (if changed)
  + contactGroups (list of id’s)
  + organisation (id)
  + resourceName (name of point)

**Subject: EVENT\_LOG/CONTACT/PRIVILEGE\_GROUP/DIRECT\_PRIVILEGE:**

searchEventProperties contains the search parameters in a more readable format

* eventProperties
  + query (if search)
  + result (if search)
* searchEventProperties
  + query (if search)
  + result (if search)

**Subject: CPR/CVR:**

* eventProperties
  + firstname-provided (if cpr)
  + lastname-provided (if cpr)

**Subject: IDENTITY:**

* eventProperties
  + query (if search)
  + result (if search)
  + cpr-changed
  + name-changed
* searchEventProperties
  + query (if search)
  + result (if search)

**Subject: EXEMPTION:**

* eventProperties
  + zipCode
  + address1Text
  + address2Text
  + address3Text
  + countryCode
  + name

**Subject: CONSENT:**

* eventProperties
  + deviceId

**Subject:PRIVILEGES/DELEGATED\_SUPPORT\_ADMIN\_PRIVILEGE:**

The changedPrivileges field is a list of the privileges granted/revoked in a readable format

* eventProperties
  + changelog (if changed)
  + granteeId
  + identityGroupType
* searchEventProperties
  + changedPrivileges

**Subject: SYSTEM\_FETCH:**

* eventProperties
  + amount-failed
  + owner
  + to-date
  + from-date
  + system-fetch
  + mailboxId
  + contact-point-id
  + amount-total
  + version
  + client\_id
  + amount-fetched

**Subject: STATISTICAL\_REPORT\_SUBSCRIPTION:**

* eventProperties
  + name

**Subject: STATISTICAL\_REPORT:**

* eventProperties
  + reportId
  + subscriptionType
  + name

For a full comprehensive list of all events stored in the event log, see the file “[Events In Event-log.xlsx](https://digitaliser.dk/Media/638199265972162947/Events%20In%20Event-log.xlsx)“ at <https://digitaliser.dk/digital-post/vejledninger/technical-integration>. The Excel sheet also includes a column with the target group for the event, this is to make it clearer which events an end user can expect to see. The schema is a separate file for readability.

# Push notification integration - TI

## Some general info about push notifications via DP

*This section is only relevant for view clients.*

There is no explicit (active) way for view clients to send push notifications on behalf of DP. In this document, the automated flow is described, as well as the push notification settings component.

* The backend/developers of a view client (from here on forward referred to as a Tenant) sets their Settings for Apple’s *Apple Push Notification service* (APNs) (see <https://developer.apple.com/library/archive/documentation/NetworkingInternet/Conceptual/RemoteNotificationsPG/PayloadKeyReference.html> ) and Google’s *Firebase Cloud Messaging* (FCM) (see <https://firebase.google.com/docs/reference/fcm/rest/v1/projects.messages> ) describing notification sounds, vibration patterns etc.
* A Tenant is a push notification provider that is tied to a Digital Post Identity. It’s mostly just used to find the correct Settings required to send a push notification.
* Settings is a store for the settings that both Google and Apple, respectively, expect *every time* a push notification is sent. Since these settings are expected to rarely change, we store them inside DP so that this info is automatically included in every push notification.
* When a message is received in a mailbox, a flow is triggered based on the message and mailbox that:
  + Fetches the Settings
  + Using messageId, mailboxId, deviceId, provider (APNS/ FCM) it then
  + Sends a push notification to the device with device ID ‘deviceId’ via provider containing:
    - Message ID
    - Mailbox ID
    - Notification title
    - Notification body
      * (at the time of writing this is assumed to be a canned message akin to “You’ve received a Digital Post message from <Sender>”)
    - Any additional OS specific settings (lights, sounds, colors, priority, etc.)

|  |
| --- |
| This means that the active step Tenants have to make, is to set their Settings for every app (both iOS and Android. In case of launching a newer app in the future, it would need its own Settings). |

## Registering as a push notification tenant (aka “I want to send push notifications”)

**To setup as a tenant, contact DP via your usual service desk**.

Include the following information:

* Who you are:
  + Either:
    - CVR number
  + or
    - Organisation ID

|  |
| --- |
| **If you as an organisation have more than a single view client we also need to know which one by providing the ClientId of the client.** |

* Which app(s) you want to support push notifications for
  + App package
    - e.g. Dk.digst.digital.post.DigDPReaderApp
  + App Name
  + We need to know if you have more than a the default of one iOS app + one Android app
  + ***Note that a single Settings object can include one configuration each for APNs and FCM!***

You will then receive:

* A Tenant
  + You will most likely not use this for much directly. As it’s just for connecting Settings to your Digital Post Identity
* One (or more) Settings
  + Use these to specify FCM and APNs settings  
    See <https://developer.apple.com/library/archive/documentation/NetworkingInternet/Conceptual/RemoteNotificationsPG/PayloadKeyReference.html>  
    and <https://firebase.google.com/docs/reference/fcm/rest/v1/projects.messages>
  + Fields also listed on page “*Push notification settings store model*” under “*Settings json*“ in DD120 document.

Example application for a setup where there is “one app” but for both Android and iOS:

|  |
| --- |
| Please setup Tenant for organisation with id: 03117712-034a-4ccd-a863-c2503304e611 We have two apps: \* *MyCoolDigitalPostApp* for android. com.appdeveloper.mycooldigitalpostapp  \* RadDigitalPostRead for iOs. com.appdeveloper.raddigitalpostread |

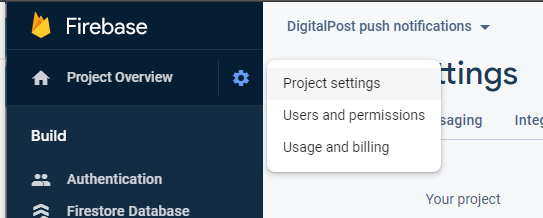
Example application for a setup where there are two APNs apps and an Android app:

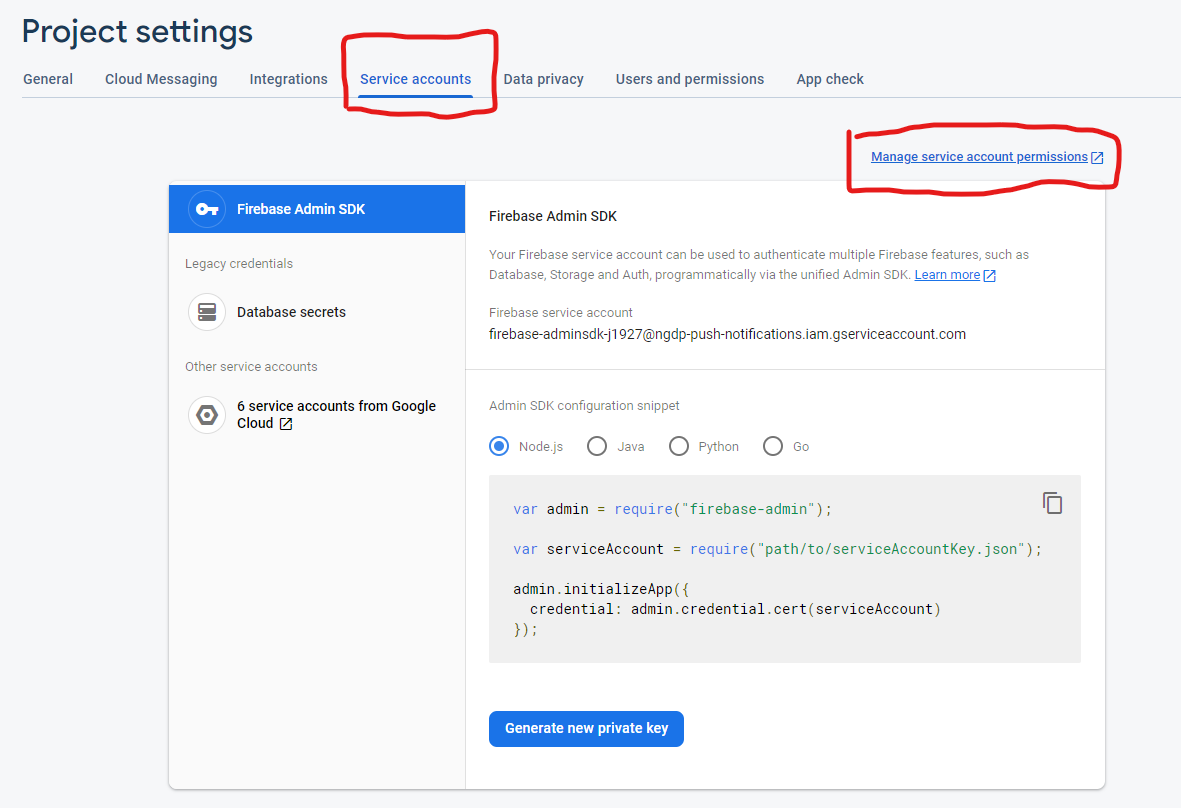
|  |
| --- |
| Please setup Tenant for organisation with id: 03117712-034a-4ccd-a863-c2503304e611 We have three apps: \* *MyCoolDigitalPostApp* for android com.appdeveloper.mycooldigitalpostapp \* RadDigitalPostRead for iOS com.appdeveloper.raddigitalpostread \* 123DigitalPost for iPadOs com.appdeveloper.123digitalpost |

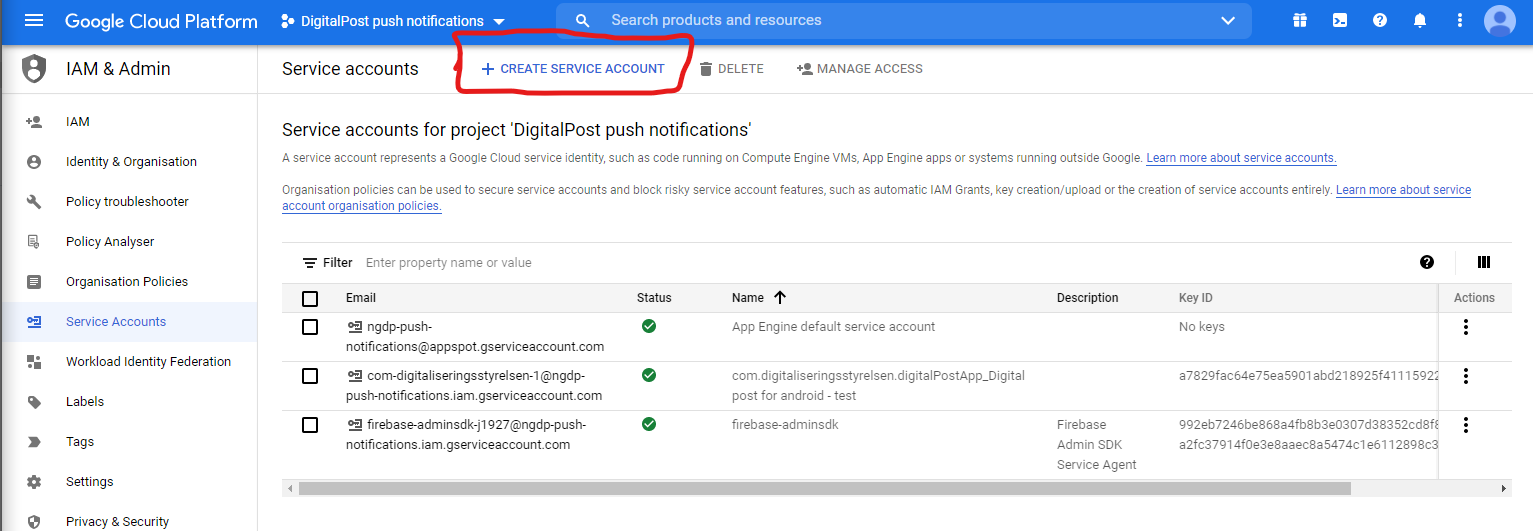
In the latter example two different Settings are required, as they would have different application identifiers for APNs.

## Creating FCM service account + private key

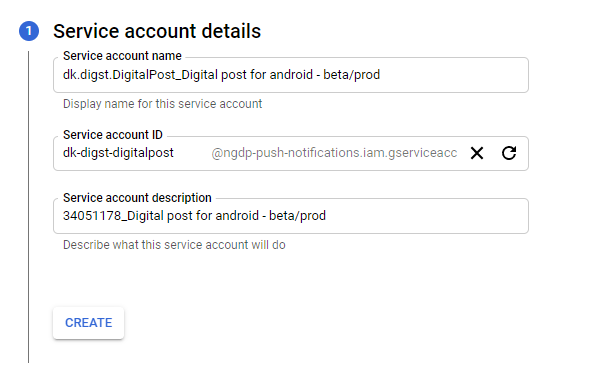
1. If you’re from DP, our project is below. If you’re a viewclient, simply start at step 2.
   1. If you do not have permissions in our DP ask dvb for them
   2. Go to <https://console.firebase.google.com/project/ngdp-push-notifications/overview>
2. Go to Project settings for your FCM app
   1. Click the little Cog and then “Project settings”

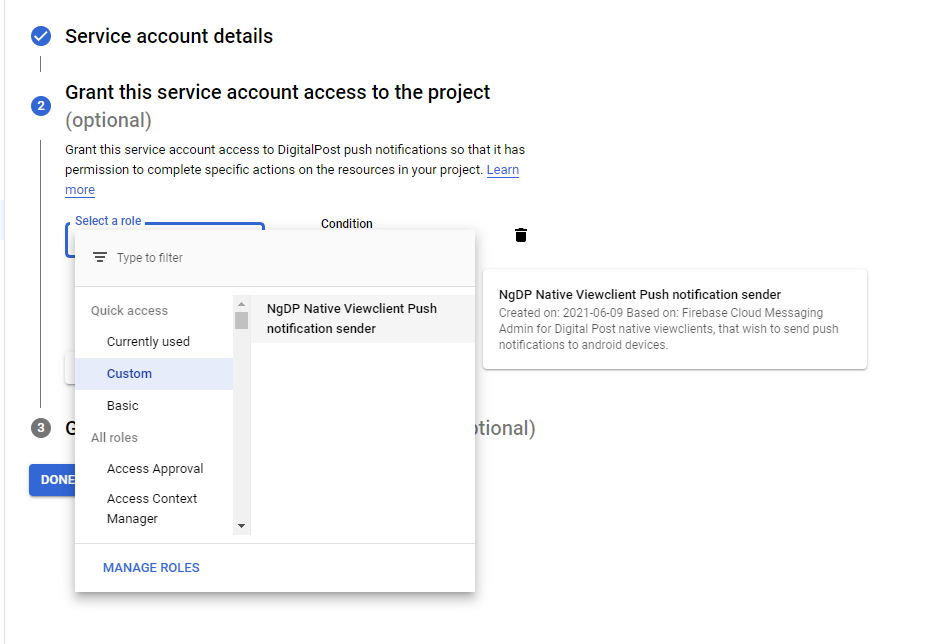


1. Click “Service accounts” and then on “Manage service account permissions”
   1. 
2. Click “Create service account”



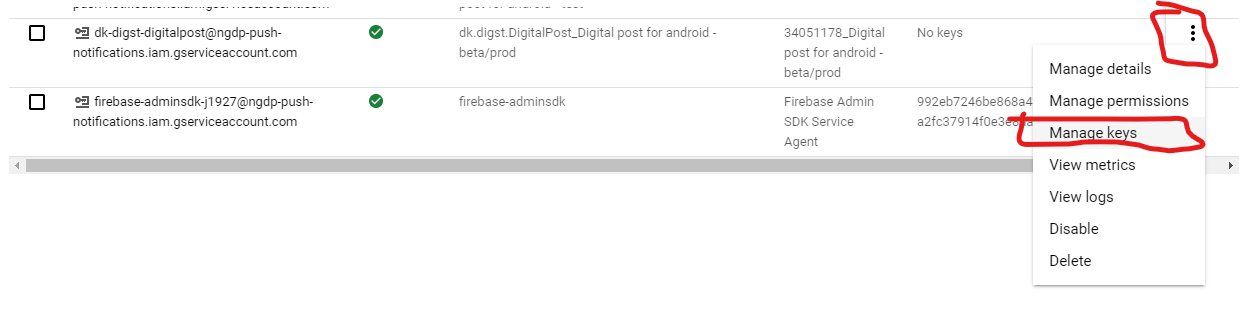
1. For step 1 Fill out the info in the following way:
   1. Service account name:
      1. <package>\_<app name>
      2. Example: dk.digst.DigitalPost\_Digital post for android
   2. Service account ID:
      1. <package> (but with dashes instead of dots)
      2. If possible, use the whole package, otherwise short down the app-specific part and use numbers
      3. Example: dk-digst-digitalpost
      4. Example: com-digitaliseringsstyrelsen-1
   3. Service account description
      1. <CVR>\_<app name>
      2. Example: 34051178\_Digital post for android - beta/prod



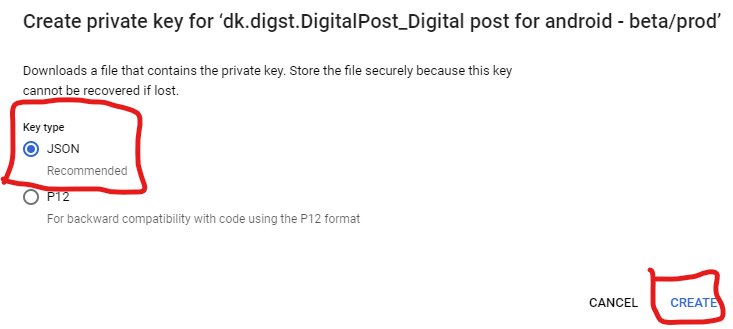
1. For step two of creating the service account, grant it privileges to send push notifications.
   1. For DP FCM project we have setup the role “NgDP Native Viewclient Push notification sender” with these rights.
2. 

For step 3 (“Grant users access to this service account (optional)”) **Skip this step**

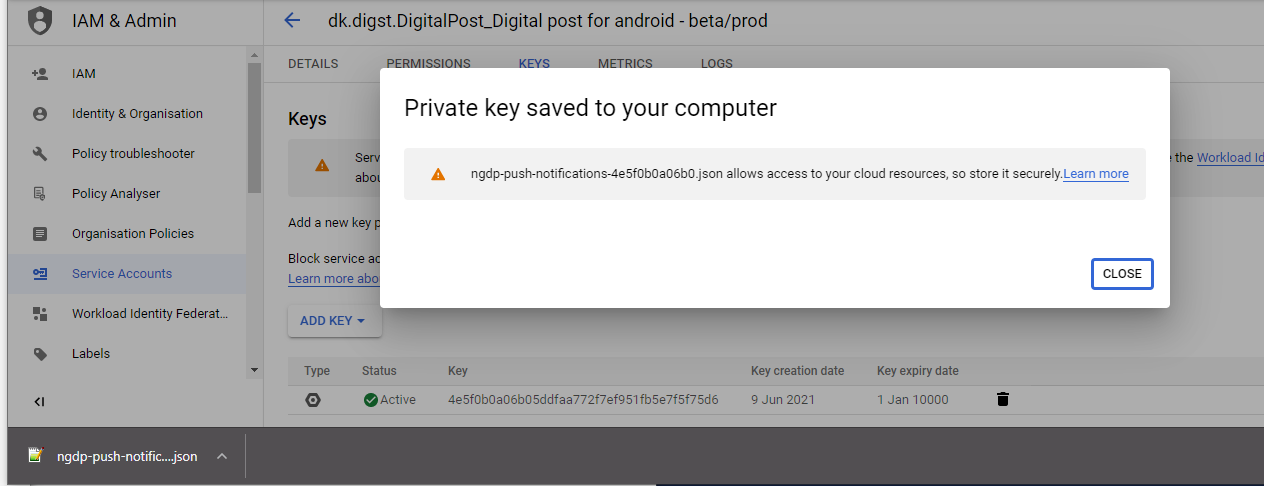
1. After the service account has been created, add a new key to the account, this key will be what you have to add to the FCM Credentials in the Settings object.
   1. Click the Kebab menu (the 3 dots)
   2. Click Manage Keys



* 1. Click Add Key
  2. Click Create new key
  3. Select JSON



1. Done!  
   The key has been downloaded to your computer. Add the content of this json to the FCM Credentials in the Settings object.



## Interaction with “push notifications”

After this setup is completed, you can proceed with the following calls.

Get your tenant ID

/apis/v1/tenants/

Response:

|  |
| --- |
| {  "content": [  {  "id": "eb1063f3-e12c-4b29-96f8-e3cb4745357e",  "version": 0  }  ],  "number": 0,  "size": 20  } |

Get full tenant info

/apis/v1/tenants/eb1063f3-e12c-4b29-96f8-e3cb4745357e

Response:

|  |
| --- |
| {      "id": "eb1063f3-e12c-4b29-96f8-e3cb4745357e",      "version": 0,      "transactionId": "F1mZpUwk6IdykTZxLXTsPs7K0FZVSUrk",      "tenantName": "Dev01 Test tenant",      "identityId": "29fb35c5-eb3e-4248-bc50-92df974bdbc6"  } |

Get list of settings for your tenant

/apis/v1/settings/

Response:

|  |
| --- |
| {  "content": [  {  "id": "6841e177-e795-4d1b-9076-7042b6f32366",  "version": 0  }  ],  "number": 0,  "size": 20  } |

Get full settings

/apis/v1/settings/6841e177-e795-4d1b-9076-7042b6f32366

Response (this specific object does not include any APNs settings)

APNs/FCM credentials should be provided by yourself, by getting an APNs/FCM project.

|  |
| --- |
| Note that encoding newlines in fcm.security.credentials as \r\n will mess with the request, which may result in push notifications not working. Instead simply use \n if needed. |

|  |
| --- |
| Note that APNs sends private keys in a different format than we expect. (For apn.apnSecurity.privateKey.) This can be fixed with openssl before submitting the private key to your settings object, for example:  openssl pkcs8 -in AuthKey\_432T42ND.p8 -out AuthKey.pem -nocrypt  Where AuthKey\_432T42ND.p8 is a cert issued by APNs, and AuthKey.pem will be a valid private key to upload. |

|  |
| --- |
| {  "id": "6841e177-e795-4d1b-9076-7042b6f32366",  "version": 0,  "transactionId": "F2ZUnp29chA6XA2ILyKktriry5lKNk9H",  "tenantId": "eb1063f3-e12c-4b29-96f8-e3cb4745357e",  "instanceId": "5ee1a8d3-c4c3-4430-8fe5-44708f78b1b7",  "apn": {  "expiration": "2021-04-20T10:46:37.361Z",  "sound": {  "name": "notificationCupcake.caf"  }  },  "fcm": {  "security": {  "credentials": "<redacted json object that describes fcm security - should be included in your original tenant+settings setup request. If not, create a ticket.>"  },  "priority": "HIGH",  "ttl": 3600000,  "notification": {  "defaultSound": **true**,  "sticky": **true**,  "localOnly": **false**,  "priority": "DEFAULT",  "defaultVibrateTimings": **true**,  "vibrateTimings": [  500,  500,  500  ],  "notificationCount": 10,  "defaultLightSettings": **true**  }  }  } |

Update settings

PUT /apis/v1/settings/6841e177-e795-4d1b-9076-7042b6f32366

Body: (change on line 26 from previous result)

|  |
| --- |
| {  "id": "6841e177-e795-4d1b-9076-7042b6f32366",  "version": 0,  "transactionId": "F2ZUnp29chA6XA2ILyKktriry5lKNk9H",  "tenantId": "eb1063f3-e12c-4b29-96f8-e3cb4745357e",  "instanceId": "5ee1a8d3-c4c3-4430-8fe5-44708f78b1b7",  "apn": {  "expiration": "2021-04-20T10:46:37.361Z",  "sound": {  "name": "notificationCupcake.caf"  }  },  "fcm": {  "security": {  "credentials": "<redacted>"  },  "priority": "HIGH",  "ttl": 3600000,  "notification": {  "defaultSound": **true**,  "sticky": **true**,  "localOnly": **false**,  "priority": "DEFAULT",  "defaultVibrateTimings": **true**,  "vibrateTimings": [  600,  500,  500  ],  "notificationCount": 10,  "defaultLightSettings": **true**  }  }  } |

Response:

|  |
| --- |
| {      "id": "6841e177-e795-4d1b-9076-7042b6f32366",      "version": 1,      "transactionId": "F2mKSNjPRrKeEBCP3HswE5PgvmSKnZop",      "tenantId": "eb1063f3-e12c-4b29-96f8-e3cb4745357e",      "instanceId": "5ee1a8d3-c4c3-4430-8fe5-44708f78b1b7",      "apn": {          "expiration": "2021-04-20T10:46:37.361Z",          "sound": {              "name": "notificationCupcake.caf"          }      },      "fcm": {          "security": {              "credentials": "<redacted>"          },          "priority": "HIGH",          "ttl": 3600000,          "notification": {              "defaultSound": **true**,              "sticky": **true**,              "localOnly": **false**,              "priority": "DEFAULT",              "defaultVibrateTimings": **true**,              "vibrateTimings": [                  600,                  500,                  500              ],              "notificationCount": 10,              "defaultLightSettings": **true**          }      }  } |

## Subscription to push notification

A push notification subscription is handled in the same way as SMS- and email- notification subscriptions.

To register a mailbox to push notification subscriptions, the app will have to provide the parameters:

* providerType  
  -~~APN for Apple~~/FCM for Google
* deviceToken  
  - which is obtained by the app on the device
* deviceId
* tenantId  
  - id of app producer
* instanceId  
  - id of the app version
* mailboxId  
  - id of the mailbox to register for push notifications

### Example

Below is an example of a list of subscriptions. This example mailbox has subscriptions for SMS, e-mail and two push notifications.

HTTP GET /mailboxes/8deb553b-0536-4671-9c9e-239f202d56e0

|  |
| --- |
| {  "id": "8deb553b-0536-4671-9c9e-239f202d56e0",  "version": 2,  "transactionId": "EnIowtB5B5OTHo0GOe5iljXgyaGfXtbO",  "createdDateTime": "2020-07-03T08:24:14.561Z",  "lastUpdated": "2020-07-03T08:32:33.155Z",  "ownerType": "CITIZEN",  "statusType": "ACTIVE",  "statusDate": "2020-07-03",  "recipientSystemAvailable": **false**,  "exempt": **false**,  "access": {  "id": "ce65def2-eb5b-4d4d-86da-4f70ffe9f6e1",  "version": 3,  "transactionId": "EnIowtKlvFhKtjc5UJkmsoqiphRf2jyO",  "createdDateTime": "2020-07-03T08:24:15.230Z",  "lastUpdated": "2020-07-03T08:32:33.173Z",  "accessType": "OWNER",  "mailboxId": "8deb553b-0536-4671-9c9e-239f202d56e0",  "introductionCompleted": **true**,  "smsNotificationSubscription": {  "id": "8e2b1cf0-1b3c-4db6-950a-663f26209f3d",  "version": 0,  "unlistedNumber": **false**,  "mobileNumber": "29892630"  },  "emailNotificationSubscriptions": [  {  "id": "87052e65-67da-4bbc-bbd7-ecd78cfa1928",  "version": 0,  "email": "test2@nc.dk"  }  ],  "pushNotificationSubscriptions": [  {  "id": "87052e65-67da-4bbc-bbd7-ecd78cfa1928",  "version": 0,  "providerType": "APN",  "deviceId": "c7135357-f27b-4c77-b87a-c81567cc4f71",  "instanceId": "8769de0a-830f-4e13-9a6f-6f757c503862",  "tenantId": "b062d3ed-a0ec-48c5-ad26-61457b9fd180",  "deviceToken": "00fc13adff785122b4ad28809a3420982341241421348097878e577c991de8f0"  },  {  "id": "e6c6ef98-0765-4723-8f98-cf1957b2a338",  "version": 0,  "providerType": "FCM"  "deviceId": "04e2278d-05c8-4346-a3cf-afee406175f3",  "instanceId": "955f7fd7-14e6-48fe-9541-49075bf25585",  "tenantId": "b6936f9a-6241-464e-a045-819e311e72cf",  "deviceToken": "654C4DB3-3F68-4969-8ED2-80EA16B46EB0"  }  ],  }  } |

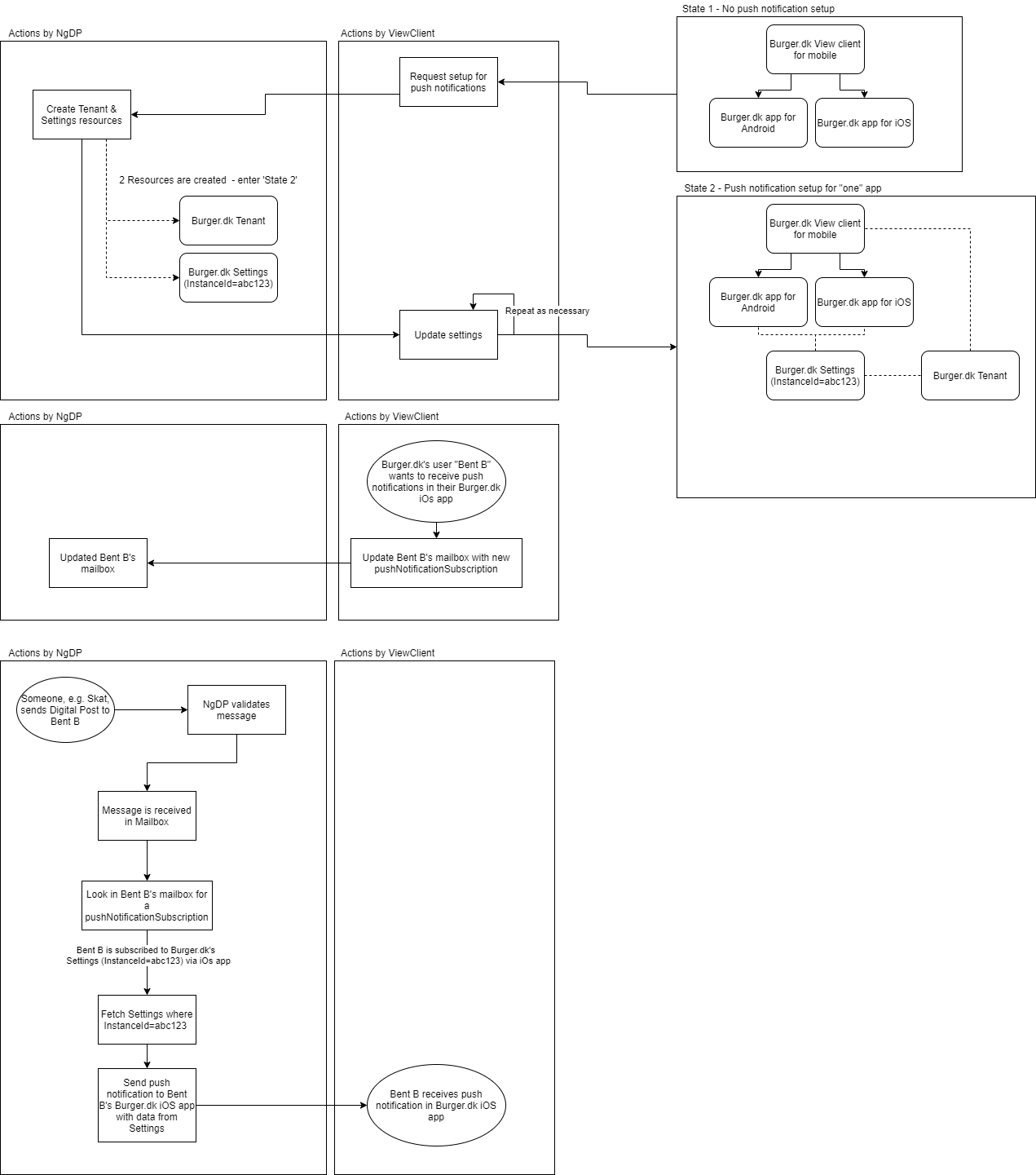
A new push notification subscription is created by adding a new one to the list. An existing is updated by modifying it in the current list. An existing is deleted by removing it from the list.

PUT /mailboxes/8deb553b-0536-4671-9c9e-239f202d56e0

## Visual guide for push notification flows

Here is a visual diagram describing the 3 different flows for push notifications:

* Register as Tenant
* Sign up users for push notifications
* Users receive push notifications when they receive message in their mailbox



# Identity registry services - TI

The below table shows an overview of all the services that the Identity-registry exposes externally. The table also gives a small description of the common usage patterns that the APIs are intended to support as well as an overview of which roles have permission to call. This overview does not go into details about which Identities the different roles can view or update.

## IDENTITIES

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Service** | **URL** | **Data returned** | **Usage** | **Required roles** |
| Query identities | GET  /identities/ | List of Identities | Fetching one or multiple Identities by CPR, CVR, type, employee’s ID, PID number, SID number, NemLogin’s ID and Client ID. | 1. Citizen 2. Citizen service employee 3. Business service employee 4. Digital Post rights administrator 5. Delegated Support Admin |
| Fetch Identity | GET /identities/{id} | Identity | Fetching a single identity | 1. Digital Post rights administrator 2. Delegated Support Admin |
| Update identity | PUT /identifies/{id} | Identity | Updating the identity, providing a e-mail to user rights administrators or adding alias to employee | 1. Digital Post rights administrator |
| CPR Validation | POST /identities/validation/cpr | CPR Validation Result | Verify the given CPR is matched with the citizen user in the token. | 1. Citizen |

## DIRECT PRIVILEGES

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Service** | **URL** | **Data returned** | **Usage** | **Required roles** |
| Query Direct Privilege | GET /privileges/direct/{grantee\_id} | List of privileges | Used to search Direct privileges of a grantee | 1. Citizen 2. Citizen service employee 3. Digital Post rights administrator 4. Delegated Support Admin |
| Fetch Direct Privilege | GET /privileges/direct/{direct\_privilege\_id} | Direct privilege. | Used to fetch the information of a Direct privilege | 1. Citizen 2. Citizen service employee 3. Business service employee 4. Digital Post rights administrator 5. Delegated Support Admin |
| Create Direct Privilege | POST /privileges/direct/ | Direct privilege | Creating Direct privilege | 1. Citizen 2. Citizen service employee 3. Business service employee 4. Digital Post rights administrator |
| Delete Direct Privilege | DELETE /privileges/direct/{direct\_privilege\_id} | Void | Revoke a Direct Privilege | 1. Citizen 2. Citizen service employee 3. Business service employee 4. Digital Post rights administrator |

## GRANTEE

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Service** | **URL** | **Data returned** | **Usage** | **Required roles** |
| Fetch Grantee | GET identity-groups/{identity\_group\_id}/grantees/{grantee\_id} | Grantee | Used to fetch the information of a Grantee | 1. Citizen 2. Citizen service employee 3. Business service employee 4. Digital Post rights administrator 5. Delegated Support Admin |

## IDENTITY GROUP

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Service** | **URL** | **Data returned** | **Usage** | **Required roles** |
| Fetch Identity group | GET /identity-groups/{identity-group-id} | Identity group | Fetch identity group information | 1. Citizen 2. Citizen service employee 3. Business service employee 4. Digital Post rights administrator 5. Delegated Support Admin |
| Create Identity group | POST /identity-groups | Identity group | Creating an Identity group | 1. Citizen 2. Citizen service employee 3. Business service employee 4. Digital Post rights administrator |
| Update Identity group | PUT /identity-groups/{identity-group-id} | Identity group | Updating an Identity group | 1. Citizen 2. Citizen service employee 3. Business service employee 4. Digital Post rights administrator |
| Delete Identity group | DELETE /identity-groups/{identity-group-id} | *internal* | Deleting an Identity group | 1. Citizen 2. Citizen service employee 3. Business service employee 4. Digital Post rights administrator |

## PRIVILEGE TYPE

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Service** | **URL** | **Data returned** | **Usage** | **Required roles** |
| Query privilege type | GET /privilege-types | List all available privilege types | Get all available privilege types which the caller can grant | 1. Citizen service employee 2. Digital Post rights administrator 3. Delegated Support Admin |

## Querying Identities, Direct privileges, Privilege Type

## IDENTITIES

For a description of common search functionality, please revisit the section **Querying and searching resources** as well as the OpenAPI specification.

The following endpoint has been exposed externally from the Identity registry:

* /identities/
  + Queries all identities the user is allowed to see. Example: Citizen service employees can see the identities of citizens and companies.

The result is an IdentitySearchResult, which looks like this in JSON:

|  |
| --- |
| {  "currentPage": 0,  "totalPages": 1,  "elementsOnPage": 1,  "totalElements": 1,  "identities": []  } |

### Fetching identities on ID

|  |
| --- |
| GET /identities/0b723a6e-c32f-42b6-a124-a79c2cb7d599 |

will fetch the identity with ID 0b723a6e-c32f-42b6-a124-a79c2cb7d599 (Random UUID with no actual identity behind it).

**Searching for Identities**

Generally, the format is:

|  |
| --- |
| GET /identities/?<parameter>=<value>  GET /identities/?<parameter>=<value>,<value>,<value>  GET /identities/?<parameter>=<value>&<parameter>=<value>&<parameter>=<value> |

It’s possible to mix and match different parameters. For lines 2 and 3, it’s possible to add as many parameters/values as desired.  
Example:

|  |
| --- |
| GET /identities/?cvrNumber=1234567890 |

returns the identity with CVR number 1234567890

|  |
| --- |
| GET /identities/?cvrNumber=1234567890,2345678901 |

would get us the two identities with the given CVR numbers.

## DIRECT PRIVILEGES

### Fetching Direct privileges on ID

|  |
| --- |
| GET /privileges/direct/c77d2e47-bb5d-410d-8ff7-e08c1f971c54 |

will fetch the direct privilege with ID c77d2e47-bb5d-410d-8ff7-e08c1f971c54 (Random UUID with no actual direct privilege behind it).

## PRIVILEGE TYPE

Endpoint for querying of privilege type:

|  |
| --- |
| GET /privilege-types |

Example:

Get privilege type for an employee with the Right Administrator role:

GET /privilege-types

|  |
| --- |
| [  "MESSAGE\_WRITE",  "MESSAGE\_EMPLOYEE",  "ACTION\_LOG\_ADMINISTRATOR",  "SEARCH\_LOG\_ADMINISTRATOR",  "MESSAGE\_LOG\_ADMINISTRATOR",  "STATISTICS\_ADMINISTRATOR",  "CITIZEN\_SERVICE\_EMPLOYEE",  "TEST\_PORTAL",  "MESSAGE\_BASIC"  ] |

## Direct privilege

A direct privilege is expected to be a unique combination of; issuer, privilege-type, scope & grantee - the before-mentioned combination is most likely enough to unambiguously identity and revoke a privilege.

A direct privilege is between the following identities:

|  |  |
| --- | --- |
| **Tildeler (Issuer)** | **Recipient (Grantee)** |
| Citizen | Company |
| Citizen | Citizen |
| Company | Company |
| Company | Employee |

A Rights Administrator in a company can see all the company's proxies assigned to other identities (other companies, employees in other companies, or citizens). Likewise, he/she can also see all the proxies that have been assigned to my company or an employee in my company.

The employee can see the privileges that are granted to and from their company.  
The employee can see only the privileges **only** of their company.

Citizens may only see themselves, as well as the power of attorney relationships. Power of attorney both ways - from and to a citizen.

## Creating Direct privilege

The creating endpoint will create a direct privilege and an identity group in which the grantee is an owner. The identity group is created only once the first time, the next direct privilege will be assigned to the existing group.

Endpoint for creation of Direct privileges:

|  |
| --- |
| POST /privileges/direct/ |

Example:

POST privileges/direct/

|  |
| --- |
| {  "granteeId": "2a1e9c38-0fbc-47ac-ab8c-a5ca213e2405",  "issuerId": "a58e5129-8c21-463f-8355-008eab2957e3",  "scopeId": "a58e5129-8c21-463f-8355-008eab2957e3",  "source": "SELF\_SERVICE",  "type": "CITIZEN"  } |

After we create the privilege, this is the response we get:

|  |
| --- |
| {  "createdDate": "2021-07-13T06:48:25.063Z",  "granteeId": "2a1e9c38-0fbc-47ac-ab8c-a5ca213e2405",  "id": "e009a294-e6cf-4b40-8b80-1eab13f42863",  "identityGroupId": "9666a80a-cb7b-4289-9de6-c71f810c46e3",  "issuerId": "a58e5129-8c21-463f-8355-008eab2957e3",  "lastUpdated": "2021-07-13T06:48:25.063Z",  "scopeId": "a58e5129-8c21-463f-8355-008eab2957e3",  "source": "SELF\_SERVICE",  "type": "CITIZEN",  "version": 0  } |

## Privilege group

With the introduction of privilege groups, the number of privileges an identity can be granted expands.  
Ie. an identity inherits privileges assigned to the privilege groups an identity is included in. Privilege groups grant a user one or more privileges. A privilege is equivalent to having a specific role in the context of a particular company / authority.

There are the following scenarios

* A user (identity) must be able to see an overview of all privileges
  + Assigned (user is *grantee* )
  + Distributed (user is *issuer* )
* An administrator must be able to view and manage privilege groups and associated members (identities) and the group's assigned privileges.

## Querying the Privilege group

Querying privilege groups are done using a GET request to the /identity-groups/ endpoint.

The result is an IdentityGroupSearchResult, which looks like this in JSON:

|  |
| --- |
| {    "currentPage": 0,    "totalPages": 1,    "elementsOnPage": 1,    "totalElements": 1,    "identityGroups": []  } |

### Fetching

Getting a privilege group can be done using a GET request to the /identity-groups/{id} endpoint.

The result is the privilege group with the specified ID.

Let us assume that we want to find privilege group that has the id is 7ee17165-c961-4b61-8212-1b980ae2294f

|  |
| --- |
| GET https://api.digitalpost.dk/apis/v1/identity-groups/7ee17165-c961-4b61-8212-1b980ae2294f |

Which gives us the following response:

|  |
| --- |
| {  "id": "7ee17165-c961-4b61-8212-1b980ae2294f",  "version": 2,  "name": "DELEGATED\_PRIVILEGE\_ed84499b-ef7b-4dc4-8bcd-e31f1c875bfd",  "transactionId": "F3f3fUzjEORcPKuOIM1rwET9OZoYEevJ",  "createdDate": "2021-05-28T07:37:30.273Z",  "lastUpdated": "2021-05-28T07:37:30.371Z",  "issuerId": "ed84499b-ef7b-4dc4-8bcd-e31f1c875bfd",  "ownerId": "29ed649b-1a62-44db-8203-9000d8d06596",  "grantees": [  {  "id": "ebb5a669-f9bd-4dfe-933b-7fd3a7be1834",  "version": 0,  "identityGroupId": "7ee17165-c961-4b61-8212-1b980ae2294f",  "identityId": "6e61749d-3b2d-4353-be57-76142ca38342",  "issuerId": "ed84499b-ef7b-4dc4-8bcd-e31f1c875bfd",  "createdDate": "2021-05-28T07:37:30.350Z",  "lastUpdated": "2021-05-28T07:37:30.350Z"  }  ],  "privileges": [  {  "id": "eb0c922a-6975-42c9-af93-bdf4ddb01622",  "version": 0,  "identityGroupId": "7ee17165-c961-4b61-8212-1b980ae2294f",  "issuerId": "ed84499b-ef7b-4dc4-8bcd-e31f1c875bfd",  "scopeId": "ed84499b-ef7b-4dc4-8bcd-e31f1c875bfd",  "type": "COMPANY\_SENDER\_SYSTEM",  "source": "SELF\_SERVICE",  "createdDate": "2021-05-28T07:37:30.367Z",  "lastUpdated": "2021-05-28T07:37:30.367Z"  }  ],  "type": "DEFAULT"  } |

### Searching

Besides the functionality described above, the Privilege group overrides and offers search using the following parameters:

|  |  |
| --- | --- |
| List of owner | List of identity IDs of owners of the privilege group. |
| List of issuer | List of identity IDs of the issuer of the privilege. |
| List of grantee | List of identity IDs of grantees of the privilege. |
| List of scope | List of identity IDs of who are delegated by the issuer. |

Generally, the format is:

|  |
| --- |
| /identity-groups/?<parameter>=<value>  /identity-groups/?<parameter>=<value>,<value>,<value>  /identity-groups/?<parameter>=<value>&<parameter>=<value>&<parameter>=<value> |

It’s possible to mix and match different parameters. For lines 2 and 3, it’s possible to add as many parameters/values as desired.

Example:

|  |
| --- |
| /identity-groups/?scope=576f90c8-aecd-492c-9d41-cae3db5c2fe7 |

returns the privilege group/s has scope ID is 576f90c8-aecd-492c-9d41-cae3db5c2fe7

|  |
| --- |
| /identity-groups/?scope=576f90c8-aecd-492c-9d41-cae3db5c2fe7,935877f7-3379-463f-9c75-8fbd715e3702 |

would get us the privilege group/s with the given scope Ids.

### Creating

Endpoint for creation of identity-group:

|  |
| --- |
| POST /identity-groups/ |

Example:

POST */identity-groups/*

|  |
| --- |
| {  "name": "TEST GROUP"  } |

Create an Identity group:

|  |
| --- |
| {  "id": "28e35a75-857f-490e-a9e3-3be807bc34fb",  "version": 0,  "name": "TEST GROUP",  "transactionId": "F5x1bes9QnqjG9LVrXLilVK7I1MJukiQ",  "createdDate": "2021-07-13T08:43:49.502Z",  "lastUpdated": "2021-07-13T08:43:49.502Z",  "issuerId": "387cdf11-59bf-4016-8afe-be0e0de7e45e",  "ownerId": "387cdf11-59bf-4016-8afe-be0e0de7e45e",  "grantees": [],  "privileges": [],  "type": "MANUAL"  } |

### Updating

When updating the IdentityGroup, as all endpoint we must first fetch the specific resource. Since the search index in only eventually consisted we first must do a fetch of the resource we want to update first.

|  |
| --- |
| GET /identity-groups/28e35a75-857f-490e-a9e3-3be807bc34fb |

Which gives

|  |
| --- |
| {  "id": "28e35a75-857f-490e-a9e3-3be807bc34fb",  "version": 0,  "name": "TEST GROUP",  "transactionId": "F5x1bes9QnqjG9LVrXLilVK7I1MJukiQ",  "createdDate": "2021-07-13T08:43:49.502Z",  "lastUpdated": "2021-07-13T08:43:49.502Z",  "issuerId": "387cdf11-59bf-4016-8afe-be0e0de7e45e",  "ownerId": "387cdf11-59bf-4016-8afe-be0e0de7e45e",  "grantees": [],  "privileges": [],  "type": "MANUAL"  } |

Now we can change the properties that we want to, note taht grantees and privileges are sub-resources, which means they have to be changed seperatly (e.g /identity-groups/{id}/grantees/. In this example we change the name of the group like so;

|  |
| --- |
| PUT /identity-groups/28e35a75-857f-490e-a9e3-3be807bc34fb |

With this request body;

|  |
| --- |
| {  "name": "GROUP\_V1"  } |

Note that we must also include the [precondition headers](https://digidp.atlassian.net/wiki/spaces/DP/pages/1771424/REST+Implementation#Precondition-headers).

If the request is successful the endpoint will return the updated group in the response body;

|  |
| --- |
| {  "id": "28e35a75-857f-490e-a9e3-3be807bc34fb",  "version": 1,  "name": "GROUP\_V1",  "transactionId": "F5x2Ytx8TYc1uwzxHrnq4Dj2kPH1ww4l",  "createdDate": "2021-07-13T08:43:49.502Z",  "lastUpdated": "2021-07-13T08:51:01.391Z",  "issuerId": "387cdf11-59bf-4016-8afe-be0e0de7e45e",  "ownerId": "387cdf11-59bf-4016-8afe-be0e0de7e45e",  "grantees": [],  "privileges": [],  "type": "MANUAL"  } |

### Adding Privilege

Adding a privilege is done using a POST request to the /identity-groups/{groupId}/privileges/ endpoint.

Now that we have fetched the Identity group we can go ahead to add privilege. Let’s add a privilege:

|  |
| --- |
| POST https://api.digitalpost.dk/apis/v1/identity-groups/2e273d28-0fb7-4797-a391-36f5e549e26c/privileges/  {  "scopeId": "35a3e43f-4506-4f74-9cc5-9421d5e80bbe",  "source": "MANUAL",  "type": "CITIZEN"  } |

After we create the privilege, this is the response we get:

|  |
| --- |
| {  "id": "a9ff0e61-5292-4eec-a906-13fc505dd43a",  "version": 0,  "identityGroupId": "2e273d28-0fb7-4797-a391-36f5e549e26c",  "issuerId": "35a3e43f-4506-4f74-9cc5-9421d5e80bbe",  "scopeId": "35a3e43f-4506-4f74-9cc5-9421d5e80bbe",  "type": "CITIZEN",  "source": "MANUAL",  "createdDate": "2021-07-26T03:38:44.279Z",  "lastUpdated": "2021-07-26T03:38:44.279Z"  } |

### Updating Privilege

Using fetched privilege we can modify certain fields. Let’s change the source.

|  |
| --- |
| PUT https://api.digitalpost.dk/apis/v1/identity-groups/2e273d28-0fb7-4797-a391-36f5e549e26c/privileges/a9ff0e61-5292-4eec-a906-13fc505dd43a |

With header corresponding to the current version of privilege (which in this case is 0):

|  |
| --- |
| If-Match: 0 |

And update body:

|  |
| --- |
| {  "id": "a9ff0e61-5292-4eec-a906-13fc505dd43a",  "version": 0,  "identityGroupId": "2e273d28-0fb7-4797-a391-36f5e549e26c",  "issuerId": "35a3e43f-4506-4f74-9cc5-9421d5e80bbe",  "scopeId": "35a3e43f-4506-4f74-9cc5-9421d5e80bbe",  "type": "CITIZEN",  "source": "SELF\_SERVICE",  "createdDate": "2021-07-26T03:38:44.279Z",  "lastUpdated": "2021-07-26T03:38:44.279Z"  } |

When we send this update request this is the result we get:

|  |
| --- |
| {      "id": "a9ff0e61-5292-4eec-a906-13fc505dd43a",      "version": 1,      "identityGroupId": "2e273d28-0fb7-4797-a391-36f5e549e26c",      "issuerId": "35a3e43f-4506-4f74-9cc5-9421d5e80bbe",      "scopeId": "35a3e43f-4506-4f74-9cc5-9421d5e80bbe",      "type": "CITIZEN",      "source": "SELF\_SERVICE",      "createdDate": "2021-07-26T03:38:44.279Z",      "lastUpdated": "2021-07-26T03:46:26.846Z"  } |

### Adding Grantee

Now that we have fetched the Identity group we can go ahead to add a grantee. Let’s add a grantee:

|  |
| --- |
| POST https://api.digitalpost.dk/apis/v1/identity-groups/77464abe-f017-42b4-a278-1f29fc97fd84/grantees/ |

And the create body:

|  |
| --- |
| {  "identityId": "2a1e9c38-0fbc-47ac-ab8c-a5ca213e2405",  "issuerId": "f31cdacc-9c37-4be2-8e78-cf03bce61ea8"  } |

After we create the grantee, this is the response we get:

|  |
| --- |
| {  "id": "a8e46333-3d9a-4aee-9773-2ac53658389e",  "version": 0,  "identityGroupId": "77464abe-f017-42b4-a278-1f29fc97fd84",  "identityId": "2a1e9c38-0fbc-47ac-ab8c-a5ca213e2405",  "issuerId": "f31cdacc-9c37-4be2-8e78-cf03bce61ea8",  "createdDate": "2021-07-13T07:28:11.448Z",  "lastUpdated": "2021-07-13T07:28:11.448Z"  } |

### Updating Grantee

Using fetched the grantee we can modify certain fields. Let’s change the identity:

|  |
| --- |
| PUT https://api.digitalpost.dk/apis/v1/identity-groups/77464abe-f017-42b4-a278-1f29fc97fd84/grantees/a8e46333-3d9a-4aee-9773-2ac53658389e |

With header corresponding to the current version of privilege (which in this case is 0):

|  |
| --- |
| If-Match: 0 |

And update body:

|  |
| --- |
| {  "identityId": "77a7475b-9f47-4a0b-a4b0-2232a5446a73",  "issuerId": "f31cdacc-9c37-4be2-8e78-cf03bce61ea8"  } |

When we send this update request this is the result we get:

|  |
| --- |
| {  "id": "a8e46333-3d9a-4aee-9773-2ac53658389e",  "version": 1,  "identityGroupId": "77464abe-f017-42b4-a278-1f29fc97fd84",  "identityId": "77a7475b-9f47-4a0b-a4b0-2232a5446a73",  "issuerId": "f31cdacc-9c37-4be2-8e78-cf03bce61ea8",  "createdDate": "2021-07-13T07:28:11.448Z",  "lastUpdated": "2021-07-13T07:57:02.277Z"  } |

# Distribution - TI

## Distribution Services

### Legacy services: Sending DP/DP2 messages via REST

This section describes the available REST legacy services for sending DP/DP2 messages. These can be used by any sender system that has been set up for DP.

For any asynchronous validation, the result of the message (if it was rejected or sent to the recipient) will be provided through a receipt. The receipt provided will depend on the set up of the sender system.

The endpoints default to return JSON data, so the accept-header is needed for XML errors.

#### Sending DP messages

DP messages can be sent to DP via REST. There are two endpoints exposed for DP messages; one that synchronously validates the message and one that asynchronously validates the message.

##### Asynchronous endpoint

|  |  |
| --- | --- |
| **/dp/afsendersystem/{systemid}/masseafsendelser/{meddelelsesid}** | |
| description | Receives version 1 Afsendelse XML’s |
| content-type | application/xml or text/xml |
| accept | application/xml |
| encoding | UTF-8 |
| request-type | PUT |
| Responses | 204, 400, 401, 403 |
| Return Data | Version 1 Fejl xml for 400 BAD REQUEST, otherwise empty |
| Input parameters | |
| systemId | UUID of the sender system in the system-registry |
| meddelelsesId | ID of the DP message. If set in Afsendelse, this value should be the same |
| Content | |
| Afsendelse | Version 1 Afsendelse XML |

The XML can only be delivered if it is a Schemavalid DP Afsendelse, and if MeddelelsesId in the URL matches the MeddelelsesId in the Afsendelse (Or meddelelsesId is empty in the Afsendelse). A successful (204) response only signifies that the message has been received by the solution and will be validated at a later point, i.e. it does not signify that the recipient has received the message.

##### Synchronous endpoint

|  |  |
| --- | --- |
| **/dp/afsendersystem/{systemid}/afsendelser/{meddelelsesid}** | |
| description | Receives version 1 Afsendelse XML’s. |
| content-type | application/xml or text/xml |
| accept | application/xml |
| encoding | UTF-8 |
| request-type | PUT |
| Responses | 204, 400, 401, 403 |
| Return Data | Version 1 Fejl xml for 400 BAD REQUEST, otherwise empty. |
| Input parameters | |
| systemId | UUID of the sender system in the system-registry |
| meddelelsesId | ID of the DP message. If set in Afsendelse, this value should be the same. |
| Content | |
| Afsendelse | Version 1 Afsendelse XML |

Any message delivered to this endpoint will be validated synchronously. If 204 is returned, it signifies that the message will be sent to the recipient.

##### Afsendelse (resource)

This section describes the fields in the Afsendelse resource. The Afsendelse resource is a Digital Post message.

|  |  |
| --- | --- |
| **Field** | **Description** |
| AfsendelseURLreference | Not used by DP |
| MeddelelseIdentifikator | The senders identifier of the message |
| AfsendelseModtager | The recipient of the message identified by a CPR or CVR number |
| MeddelelseTypeNavn | Type of the message. *‘Meddelelse’* for a normal Digital Post message, *‘ServiceBesked’* for NemSMS. This value can also be set through MeddelelseIndholdstypeIdentifikator |
| MeddelelseIndholdstypeIdentifikator | ID used to set standard values for similar messages. The following values can be set through this field:   * A title / title prefix * The message can be marked Mandatory * MeddelelseSvarTypeNavn * MeddelelseSvarPostkasseIdentifikator * MeddelelseSvarEmneIdentifikator * MeddelelseTypeNavn   If this value is not set, the required data has to be set in the Afsendelse XML |
| MeddelelseTitelTekst | Title of the message. This value will get prefixed by the value (if any) from MeddelelseIndholdstypeIdentifikator |
| MeddelelseIndholdData | Base64 data of the main attachment of the message |
| MeddelelseIndholdURLreference | Not used by DP |
| FilformatNavn | The file-type of the main attachment, e.g. *pdf, txt* or *html.*  For ServiceBesked/NemSMS this must be *txt* |
| IndholdStoerrelseMaal | Not used by DP |
| MeddelelseTraadIdentifikator | Not used by DP |
| AfsendelseDatoTid | The time at which the message should be sent to the recipient. This can be up to 5 days in the future.  Only the ‘date’ value of the field is used |
| AfsendelseTilstandNavn | Not used by DP |
| MeddelelseSvarTypeNavn | Standard: The value from MeddelelseIndholdstypeIdentifikator is used  Angivet: The recipient can reply to the message  IkkeMuligt: The recipient can not reply to the message |
| MeddelelseSvarPostkasseIdentifikator | Marks (together with MeddelelseSvarEmneIdentifikator) which contact point a reply to the message should be sent to |
| MeddelelseSvarEmneIdentifikator | Marks (together with MeddelelseSvarPostkasseIdentifikator) which contact point a reply to the message should be sent to |
| VedhaeftningSamling | Contains any additional attachments in the message. Using this is not allowed if the message is a ServiceBesked/NemSMS  See VedhaeftningSamling below |
| VedhaeftningSamlingKvantitet | Number of additional attachments (entries in VedhaeftningSamling) |
| MeddelelseFESDmetadata | The values in these fields will be returned in any reply to the message. |
| MeddelelseTidsfristDato | Marks a deadline for replying to the message |
| MeddelelsesTidsfristTekst | Not used by DP |
| MeddelelseServiceBeskedTekst | Not used by DP |

###### Example

|  |
| --- |
| <?xml version="1.0" encoding="UTF-8" standalone="yes"?>  <Afsendelse xmlns="urn:oio:dkal:1.0.0" xmlns:ns2="urn:oio:adir:dagpenge:2009.07.01">  <AfsendelseTilstandNavn>planlagt</AfsendelseTilstandNavn>  <AfsendelseModtager>  <ns2:CPRnummerIdentifikator>0000000000</ns2:CPRnummerIdentifikator>  </AfsendelseModtager>  <MeddelelseTypeNavn>meddelelse</MeddelelseTypeNavn>  <MeddelelseTitelTekst>Afsendelse eksempel</MeddelelseTitelTekst>  <MeddelelseIndholdData>YQ==</MeddelelseIndholdData>  <FilformatNavn>HTML</FilformatNavn>  <MeddelelseSvarTypeNavn>ikkeMuligt</MeddelelseSvarTypeNavn>  <MeddelelseFESDmetadata>  <FESDdokumentIdentifikator>c348339b-d2fa-4c1b-a301-9f3bf4724987</FESDdokumentIdentifikator>  <FESDaktoerIdentifikator>00000000-0000-0000-0000-000000000000</FESDaktoerIdentifikator>  <FESDsagIdentifikator>00000000-0000-0000-0000-000000000000</FESDsagIdentifikator>  <FESDsagsklassifikationIdentifikator>00000000-0000-0000-0000-000000000000</FESDsagsklassifikationIdentifikator>  </MeddelelseFESDmetadata>  <VedhaeftningSamlingKvantitet>2</VedhaeftningSamlingKvantitet>  <VedhaeftningSamling>  <Vedhaeftning>  <VedhaeftningNavn>Attachment 1</VedhaeftningNavn>  <FilformatNavn>txt</FilformatNavn>  <VedhaeftningIndholdData>U29tZSBpbmZvcm1hdGlvbg==</VedhaeftningIndholdData>  </Vedhaeftning>  <Vedhaeftning>  <VedhaeftningNavn>Atatchment 2</VedhaeftningNavn>  <FilformatNavn>txt</FilformatNavn>  <VedhaeftningIndholdData>Tm90aGluZyBpbnRlcmVzdGluZw==</VedhaeftningIndholdData>  </Vedhaeftning>  </VedhaeftningSamling>  </Afsendelse> |

##### VedhaeftningSamling (resource)

VedhæftningSamling contains any additional attachments to a DP Afsendelse. It contains a list of Vedhaeftning, which are described in the table below.

|  |  |
| --- | --- |
| **Field** | **Description** |
| VedhaeftningNavn | Name of the attachment (without file extension) |
| VedhaeftningIndholdData | Data of the attachment (in Base64) |
| FilformatNavn | File extension (pdf, docx, html, txt, etc.) |
| BilagIdentifikator | Not used by DP |
| IndholdStoerrelseMaal | Not used by DP |
| VedhaeftningIndholdURLreference | Not used by DP |

#### Sending DP2 messages

DP2 messages can also be sent to DP via REST. There are two endpoints for DP2 messages - one that synchronously validates the message and one that does it asynchronously.

### Asynchronous endpoint

|  |  |
| --- | --- |
| **/dp2/afsendersystem/{systemid}/masseafsendelser/{meddelelsesid}** | |
| description | Receives version 2 Afsendelse XML’s. |
| content-type | application/xml or text/xml |
| accept | application/xml |
| encoding | UTF-8 |
| request-type | PUT |
| Responses | 204, 400, 401, 403 |
| Return Data | Version 2 Fejl xml for 400 BAD REQUEST, otherwise empty. |
| Input parameters | |
| systemId | UUID of the sender system in the system-registry |
| meddelelsesId | ID of the DP2 message. If set in Afsendelse, this value should be the same. |
| Content | |
| Afsendelse | Version 2 Afsendelse XML |

The XML can only be delivered if it is a Schemavalid DP2 Afsendelse, and if MeddelelsesId in the URL matches the MeddelelsesId in the Afsendelse (Or meddelelsesId is empty in the Afsendelse). A successful (204) response only signifies that the message has been received by the solution and will be validated at a later point, i.e. it does not signify that the recipient has received the message.

#### Synchronous endpoint

|  |  |
| --- | --- |
| **/dp2/afsendersystem/{systemid}/afsendelser/{meddelelsesid}** | |
| description | Receives version 2 Afsendelse XML’s. |
| content-type | application/xml or text/xml |
| accept | application/xml |
| encoding | UTF-8 |
| request-type | PUT |
| Responses | 204, 400, 401, 403 |
| Return Data | Version 2 Fejl xml for 400 BAD REQUEST, otherwise empty. |
| Input parameters | |
| systemId | UUID of the sender system in the system-registry |
| meddelelsesId | ID of the DP2 message. If set in Afsendelse, this value should be the same. |
| Content | |
| Afsendelse | Version 2 Afsendelse XML |

Any DP2 message delivered to this endpoint will be validated synchronously. If 204 is returned, it signifies that the message will be sent to the recipient.

#### Afsendelse (resource)

This section describes the fields in the Afsendelse resource. The Afsendelse resource is a Digital Post message.

|  |  |
| --- | --- |
| **Field** | **Description** |
| AfsendelseURLreference | Not used by DP |
| MeddelelseIdentifikator | The senders identifier of the message |
| AfsendelseModtagerSamling | The recipients of the message identified by a CPR or CVR number. The afsendelse is treated individually for each recipient, and the sender system will receive a receipt for each |
| MeddelelseTypeNavn | Type of the message. *‘Meddelelse’* for a normal Digital Post message, *‘ServiceBesked’* for NemSMS. This value can also be set through MeddelelseIndholdstypeIdentifikator |
| MeddelelseIndholdstypeIdentifikator | ID used to set standard values for similar messages. The following values can be set through this field:   * A title / title prefix * The message can be marked Mandatory * MeddelelseSvarTypeNavn * MeddelelseSvarPostkasseIdentifikator * MeddelelseSvarEmneIdentifikator * MeddelelseTypeNavn   If this value is not set, the required data has to be set in the Afsendelse XML |
| MeddelelseTitelTekst | Title of the message. This value will get prefixed by the value (if any) from MeddelelseIndholdstypeIdentifikator |
| MeddelelseIndholdData | Base64 data of the main attachment of the message |
| MeddelelseIndholdURLreference | Not used by DP |
| FilformatNavn | The file-type of the main attachment, e.g. *pdf, txt* or *html.*  For ServiceBesked/NemSMS this must be *txt* |
| IndholdStoerrelseMaal | Not used by DP |
| AfsendelseAdviseringMailTekst | Specifies individual notification text for email notifications. |
| MeddelelseTraadIdentifikator | Not used by DP |
| AfsendelseDatoTid | The time at which the message should be sent to the recipient. This can be up to 5 days in the future.  Only the ‘date’ value of the field is used |
| AfsendelseTilstandNavn | Not used by DP |
| MeddelelseSvarTypeNavn | Standard: The value from MeddelelseIndholdstypeIdentifikator is used  Angivet: The recipient can reply to the message  IkkeMuligt: The recipient can not reply to the message |
| MeddelelseSvarPostkasseIdentifikator | Marks (together with MeddelelseSvarEmneIdentifikator) which contact point a reply to the message should be sent to |
| MeddelelseSvarEmneIdentifikator | Marks (together with MeddelelseSvarPostkasseIdentifikator) which contact point a reply to the message should be sent to |
| VedhaeftningSamling | Contains any additional attachments in the message. Using this is not allowed if the message is a ServiceBesked/NemSMS  See VedhaeftningSamling below |
| VedhaeftningSamlingKvantitet | Number of additional attachments (entries in VedhaeftningSamling) |
| MeddelelseKvitteringsTypeNavn | Not used by DP |
| MeddelelseKvitteringPostkasseIdentifikator | Not used by DP |
| MeddelelseFESDmetadata | The values in these fields will be returned in any reply to the message |
| MeddelelseTidsfristDato | Marks a deadline for replying to the message |
| MeddelelseTidsfristTekst | Not used by DP |

##### Example

|  |
| --- |
| <?xml version="1.0" encoding="UTF-8" standalone="yes"?>  <ns2:Afsendelse xmlns="urn:oio:dkal:1.0.0" xmlns:ns2="urn:oio:dkal:2.0.0" xmlns:ns5="urn:oio:adir:dagpenge:2009.07.01">  <AfsendelseTilstandNavn>planlagt</AfsendelseTilstandNavn>  <AfsendelseModtagerSamling>  <ns2:AfsendelseModtager>  <CPRnummerIdentifikator>0000000000</CPRnummerIdentifikator>  </ns2:AfsendelseModtager>  <ns2:AfsendelseModtager>  <CPRnummerIdentifikator>0000000001</CPRnummerIdentifikator>  </ns2:AfsendelseModtager>  <ns2:AfsendelseModtager>  <CPRnummerIdentifikator>0000000002</CPRnummerIdentifikator>  </ns2:AfsendelseModtager>  </AfsendelseModtagerSamling>  <ns2:MeddelelseTypeNavn>meddelelse</ns2:MeddelelseTypeNavn>  <MeddelelseIndholdstypeIdentifikator>00000</MeddelelseIndholdstypeIdentifikator>  <AfsendelseAdviseringMailTekst>DP2 specifikt felt</AfsendelseAdviseringMailTekst>  <ns2:MeddelelseTitelTekst>Testscenarie DP2 digst</ns2:MeddelelseTitelTekst>  <MeddelelseIndholdData>YQ==</MeddelelseIndholdData>  <FilformatNavn>HTML</FilformatNavn>  <MeddelelseSvarTypeNavn>angivet</MeddelelseSvarTypeNavn>  <MeddelelseSvarPostkasseIdentifikator>00000</MeddelelseSvarPostkasseIdentifikator>  <MeddelelseSvarEmneIdentifikator>00000</MeddelelseSvarEmneIdentifikator>  <MeddelelseFESDmetadata>  <FESDdokumentIdentifikator>c348339b-d2fa-4c1b-a301-9f3bf4724987</FESDdokumentIdentifikator>  <FESDaktoerIdentifikator>00000000-0000-0000-0000-000000000000</FESDaktoerIdentifikator>  <FESDsagIdentifikator>00000000-0000-0000-0000-000000000000</FESDsagIdentifikator>  <FESDsagsklassifikationIdentifikator>00000000-0000-0000-0000-000000000000</FESDsagsklassifikationIdentifikator>  </MeddelelseFESDmetadata>  <VedhaeftningSamlingKvantitet>6</VedhaeftningSamlingKvantitet>  <VedhaeftningSamling>  <Vedhaeftning>  <VedhaeftningNavn>ODK Excelark 6</VedhaeftningNavn>  <FilformatNavn>xlsx</FilformatNavn>  <IndholdStoerrelseMaal>12305</IndholdStoerrelseMaal>  <VedhaeftningIndholdData>U29tZSBpbmZvcm1hdGlvbg==</VedhaeftningIndholdData>  </Vedhaeftning>  <Vedhaeftning>  <VedhaeftningNavn>ODK Excelark</VedhaeftningNavn>  <FilformatNavn>pdf</FilformatNavn>  <IndholdStoerrelseMaal>9787</IndholdStoerrelseMaal>  <VedhaeftningIndholdData>Tm90aGluZyBpbnRlcmVzdGluZw==</VedhaeftningIndholdData>  </Vedhaeftning>  </VedhaeftningSamling>  </ns2:Afsendelse> |

#### VedhaeftningSamling (resource)

VedhaeftningSamling contains any additional attachments to a DP2 Afsendelse. It contains a list of Vedhaeftning, which are described in the table below.

|  |  |
| --- | --- |
| **Field** | **Description** |
| VedhaeftningNavn | Name of the attachment (without file extension) |
| VedhaeftningIndholdData | Data of the attachment (in Base64) |
| FilformatNavn | File extension (pdf, docx, html, txt, etc.) |
| BilagIdentifikator | Not used by DP |
| IndholdStoerrelseMaal | Not used by DP |
| VedhaeftningIndholdURLreference | Not used by DP |

### Legacy services: Sending DP/DP2 messages via SFTP

This page describes the available SFTP legacy service for sending DP/DP2 messages. The first section describes the integration, and the following sections describe the two bulk message formats that can be sent via SFTP.

#### Authentication

When setting up the sender system via Administrative Access, an SSH-key should be provided by the user for SFTP sender systems. Once the system has been created, DP will provide an SSH-username.

This username-key pair should be used to authenticate when logging into the SFTP-server.

#### Folders

Each sender system has its own folders on the SFTP server. The table describing each of those folders can be found in the section “Bulk Memo SFTP service”.

#### Uploading

In order to successfully upload an XML BULK or EBCDIC file:

1. Upload the content file to the corresponding folder. The name of the file should not include “.KLAR”
2. Upload ImportReadyFile file which indicates that the file from the first point has been uploaded successfully with the name of the content file with the addition of “KLAR” as described below:
   1. If the content file name starts with “EBOKS.DATA” prefix, the “KLAR” marker needs to be added right after that prefix, e.g. ‘EBOKS.DATA.Oyyyyy.Lxxxx’ → ‘EBOKS.DATA.**KLAR.**Oyyyyy.Lxxxx’
   2. If the content file name does not start with “EBOKS.DATA”, the “KLAR” marker should be added at the beginning of the file name, e.g. ‘Oyyyyy. Lxxxx’ → ‘**KLAR.**Oyyyyy.Lxxxx’.

#### XML Bulk

Bulk messages can be sent via SFTP in the MasseforsendelseAfsendelseSamling XML format. The following table describes the fields in the bulk XML format MasseforsendelseAfsendelseSamling.

|  |  |
| --- | --- |
| **Field** | **Description** |
| SystemIdentifikator | Not used by DP |
| KundeIdentifikator | Not used by DP |
| MasseforsendelseAfsendelseSamlingDannetDatoTid | Not used by DP |
| Afsendelse | Details of this format can be found in “*Sending DP/DP2 messages via REST*” page |

This section describes the available SFTP legacy services for sending DP/DP2 messages. These can be used by any sender system that has been set up for DP.

#### DP1 EBCDIC

Bulk messages can be sent via SFTP in an EBCDIC record file format. Each DP1 EBCDIC file has the following structure:

1 headerrecord.  
For each message:  
1 parameterrecord  
1 datastartrecord  
1 or more data records  
1 dataslutrecord  
None or several attachments, each consists of  
1 vedhæftningstartrecord  
1 or more vedhæftningrecord  
1 vedhæftningslutrecord  
1 trailerrecord

Each of the records is preceded with a two-byte field containing the recordlength: for example, Headerrecord is always preceded by 97.

Each of the records, except data records (Datarecord and Vedhæftningrecord) are encoded in EBCDIC (cp277) and data records are encoded in ASCII.

If the documents sent in the message have more than 30000 bytes they are divided into multiple parts. Each part except the last one must have exactly 30000 bytes, so a 100000 bytes file is split into 30000, 30000, 30000 and 10000 bytes parts. Each of the parts is in Datarecord for the main attachment and in Vedhæftningrecord for additional attachments.

#### Headerrecord - length 97

|  |  |  |  |
| --- | --- | --- | --- |
| **Field name** | **Type** | **Length** | **Explanation** |
| Record-type | String | 8 | Always 'EBOKS004' |
| Struktur-version | String | 3 | Record version Currently ’005’ |
| Data-type | String | 30 | Here 'Data for e-Boks' |
| Kunde-nr | String | 15 | Not used by DP |
| Dannelsestidspunkt | String | 26 | Time stamp for example ‘2000-12-24-12.30.45.123456’ |
| Afsendersystem | String | 15 | Not used by DP |

#### Parameterrecord - length 783

|  |  |  |  |
| --- | --- | --- | --- |
| **ValørdatoField name** | **Type** | **Length** | **Explanation** |
| Record-type | String | 8 | Always 'EBOKS005' |
| Struktur-version | String | 3 | Record version Currently ’006’ |
| MaterialeId | String | 15 | Specifies the content type. Number, which clearly defines the type of dispatch. The content must be numerical, specified with right-alignment and prefixed by 0 |
| Bruger-type | String | 20 | P for CPR-number, V for CVR-number Left-aligned followed by blank spaces |
| Bruger | String | 50 | Identification of recipient (CPR or CVR number) Left-aligned followed by blank spaces |
| Filler | String | 2 | Left blank |
| Valørdato | String | 10 | Specifies when the message should be sent. The format is: YYYY-MM-DD. Must be no more than 5 days in the future. If it is not specified, the message will be sent to the recipient immediately |
| Valørtid | String | 8 | Not used by DP |
| Højre del af emne | String | 50 | Optional for the customer. The text is composed with the fixed text from materialeId. In the case of NemSMSs, this text is not used |
| Filformat | String | 10 | The file format used in the message. In the case of NemSMS messages, this field must be 'txt' |
| MeddelelsesId | String | 30 | Identifier of the message |
| Number of appendices | String | 2 | Numerical. Must be 0 for NemSMS messages |
| Bilags-id 1-10 | String | 80 | Not used by DP |
| Filler | String | 15 | Should be left blank |
| FESD sag Identifikator | String | 36 | Identifies the case. Optional and not used for NemSMS messages |
| FESD sagsklassifikationIdentifikator | String | 36 | Classification of the case. Optional and not used for NemSMS messages |
| FESD Aktoer-Identifikatior | String | 36 | Identifies the participant. Optional and not used for NemSMS messages |
| FESD Dokument-Identifikatior | String | 36 | Identifies a document. Optional and not used for NemSMS messages |
| Dialogtråd | String | 26 | Not used by DP |
| Svartype | String | 1 | 'D' can be responded to, use the default mailbox for the content type. 'A' - uses the mailbox specified in the response mailbox. Otherwise: cannot be used. Not used for NemSMS messages |
| Svarpostkasse | String | 15 | Marks (together with Svaremne) which contact point a reply to the message should be sent to |
| Svaremne | String | 15 | Marks (together with Svarpostkasse) which contact point a reply to the message should be sent to |
| Antal vedhæftninger | String | 15 | Numerical. Must be 0 for NemSMS messages |
| Tidsfrist | String | 10 | A date which specifies a deadline associated with the message. Specification of a deadline is optional. The format is: YYYY-MM-DD. Not used for NemSMS messages |
| Note | String | 254 | Not used by DP |

#### Datastartrecord - length 26

|  |  |  |  |
| --- | --- | --- | --- |
| **Field name** | **Type** | **Length** | **Explanation** |
| Record-type | String | 8 | Always 'EBOKS014' |
| Struktur-version | String | 3 | Record version Currently ’005’ |
| Antal bytes | String | 15 | Number of bytes in the message data record |

#### Datarecord - length variable

|  |  |  |  |
| --- | --- | --- | --- |
| **Field name** | **Type** | **Length** | **Explanation** |
| Datarecord | String | 30000 | Variable max. length. 30,000 bytes |

#### Dataslutrecord - length 11

|  |  |  |  |
| --- | --- | --- | --- |
| **Field name** | **Type** | **Length** | **Explanation** |
| Record-type | String | 8 | Always 'EBOKS015' |
| Struktur-version | String | 3 | Record version Currently ’006’ |

#### Vedhæftningstartrecord - length 290

|  |  |  |  |
| --- | --- | --- | --- |
| **Field name** | **Type** | **Length** | **Explanation** |
| Record-type | String | 8 | Always 'EBOKS025' |
| Struktur-version | String | 3 | Record version Currently ’001’ |
| Antal bytes | String | 15 | Number of bytes contained in the attachment |
| Filformat | String | 10 | File format of the attachment |
| Name | String | 254 | File name of the attached document |

#### Vedhæftningrecord - length variable

|  |  |  |  |
| --- | --- | --- | --- |
| **Field name** | **Type** | **Length** | **Explanation** |
| Vedhæftningrecord | String | 30000 | Variable max. length. 30,000 bytes |

#### Vedhæftningslutrecord - length 11

|  |  |  |  |
| --- | --- | --- | --- |
| **Field name** | **Type** | **Length** | **Explanation** |
| Record-type | String | 8 | Always 'EBOKS026' |
| Struktur-version | String | 3 | Record version Currently ’001’ |

#### Trailerrecord - length

|  |  |  |  |
| --- | --- | --- | --- |
| **Field name** | **Type** | **Length** | **Explanation** |
| Record-type | String | 8 | Always 'EBOKS007' |
| Struktur-version | String | 3 | Record version Currently ’003’ |
| Antal parameterrecord | String | 15 | Number of documents/messages |
| Antal records | String | 15 | Total number of records in the file |

#### DP2 EBCDIC

Bulk messages can also be sent via SFTP in a DP2 EBCDIC record file format. The DP2 EBCDIC format is for the most part identical to DP1, but it has two additional non-mandatory records Extra user record and AdvisteksterRecord.

Each DP2 EBCDIC file has the following structure

1 headerrecord.  
For each message:  
1 parameterrecord  
0-9 extra user record  
0-1 advistekstrecord  
1 datastartrecord  
1 or more data records  
1 dataslutrecord  
None or several attachments, each consists of  
1 vedhæftningstartrecord  
1 or more vedhæftningrecord  
1 vedhæftningslutrecord  
1 trailerrecord

#### Extra user record – length 83

If one message should be sent to multiple users, the extra users should be specified in the Extra user record.

|  |  |  |  |
| --- | --- | --- | --- |
| **Field name** | **Type** | **Length** | **Explanation** |
| Record-type | String | 8 | Always 'EBOKS013' |
| Struktur-version | String | 3 | Record version Currently ’004’ |
| Bruger-type | String | 15 | P for CPR-number, V for CVR-number Left-justified followed by blank spaces |
| Bruger | String | 15 | Identification of recipient (CPR or CVR number) Left-justified followed by blank spaces |
| Filler | String | 2 | Always completed with 'DK' |

#### AdvisteksterRecord - length 2059

Custom notification messages can be specified in AdvisteksterRecord.

|  |  |  |  |
| --- | --- | --- | --- |
| **Field name** | **Type** | **Length** | **Explanation** |
| Record-type | String | 8 | Always 'EBOKS051' |
| Struktur-version | String | 3 | Record version Currently ’001’ |
| AdvisTxtMail | String | 1024 | Specifies a text for use in individual notifications |
| AdvisTxtSms | String | 1024 | Not used in DP |

### Legacy services: Sending DP/DP2 messages via SMTP

It is possible for an authority to send and reply messages to an end user’s digital mailbox. This section describes the service for sending DP/DP2 messages via SMTP.

The model is automated meaning that the messages can be sent via an automated sender system. The following criteria should be uphold to send messages.

#### Sending DP messages

1. The sender system should be configured in Administrative Access to sent via SMTP
2. The mail must be signed with the OCES certificate registered on the sender system via Administrative Access
3. The sender system use the OCES certificate for encryption. The certificate can be uploaded to Administrative Access
4. To specify meta data, sender system must attach a metadata file with the name 'dkalmetadata.xml'  
     
   Example of dkalmetadata.xml for DP1

|  |
| --- |
| <?xml version="1.0" encoding="UTF-8" standalone="yes"?>  <EpostAfsendelseMetadata xmlns="urn:oio:dkal:1.0.0" xmlns:ns2="urn:oio:adir:dagpenge:2009.07.01">  <SystemIdentifikator>444</SystemIdentifikator>  <MeddelelseIdentifikator>123</MeddelelseIdentifikator>  <AfsendelseModtager>  <ns2:CPRnummerIdentifikator>1234</ns2:CPRnummerIdentifikator>  </AfsendelseModtager>  <MeddelelseIndholdstypeIdentifikator>1</MeddelelseIndholdstypeIdentifikator>  <MeddelelseSvarTypeNavn>angivet</MeddelelseSvarTypeNavn>  <MeddelelseSvarPostkasseIdentifikator>321</MeddelelseSvarPostkasseIdentifikator>  <MeddelelseSvarEmneIdentifikator>123</MeddelelseSvarEmneIdentifikator>  <MeddelelseTraadIdentifikator>asdasd</MeddelelseTraadIdentifikator>  <MeddelelseFESDmetadata>  <FESDdokumentIdentifikator>b2770188-e83f-4334-b575-e6a15a5b5918</FESDdokumentIdentifikator>  <FESDaktoerIdentifikator>870d5800-2963-4638-a9ae-6cdb7a2c3ca6</FESDaktoerIdentifikator>  <FESDsagIdentifikator>e900aa10-2781-45ea-b9a1-99cc7762dca0</FESDsagIdentifikator>  <FESDsagsklassifikationIdentifikator>846efeab-1bd6-429d-bfc8-ead1d2dd5575</FESDsagsklassifikationIdentifikator>  </MeddelelseFESDmetadata>  </EpostAfsendelseMetadata> |

1. The sender needs to set an identifier of the sender system in the MIME message. This is done by setting the custom header X-API-KEY to the API-key of the sender system that is provided when creating the sender system
2. The actual content of the message must be added to the body, and any attachment can be added. Note that the subject field will not necessarily be encrypted. In front of the subject field, an indicator text for the content type is added
3. All mails should be signed with the senders certificate (uploaded in Administrative Access when the sender system was created), encrypted with DP’s certificate, and sent to [dp@test.digitalpost.dk](mailto:dp@test.digitalpost.dk). A designated production environment email address will be added

##### Afsendelse metadata

The format of the meta data that is used for sending messages is described below. The meta data is an attached XML file named dkalmetadata.xml.

|  |  |
| --- | --- |
| **Field** | **Description** |
| SystemIdentifikator | Not used by DP |
| AfsendelseModtager | The recipients of the message identified by a CPR or CVR number. |
| MeddelelseIndholdstypeIdentifikator | ID used to set standard values for similar messages. The following values can be set through this field:   * A title / title prefix * The message can be marked Mandatory * MeddelelseSvarTypeNavn * MeddelelseSvarPostkasseIdentifikator * MeddelelseSvarEmneIdentifikator * MeddelelseTypeNavn   If this value is not set, the required data has to be set in the Afsendelse XML |
| MeddelelseIdentifikator | The identifier of the message |
| MeddelelseSvarTypeNavn | Standard: The value from MeddelelseIndholdstypeIdentifikator is used  Angivet: The recipient can reply to the message  IkkeMuligt: The recipient can not reply to the message |
| MeddelelseSvarPostkasseIdentifikator | Marks (together with MeddelelseSvarEmneIdentifikator) which contact point a reply to the message should be sent to |
| MeddelelseSvarEmneIdentifikator | Marks (together with MeddelelseSvarPostkasseIdentifikator) which contact point a reply to the message should be sent to |
| MeddelelseTraadIdentifikator | Not used by DP |
| MeddelelseTidsfristDato | Marks a deadline for replying to the message |
| MeddelelseTidsfristTekst | Not used by DP |
| MeddelelseFESDmetadata | The values in these fields will be returned in any reply to the message |
| BilagSamling | Not used by DP |

###### Receipts

After sending the message (either if it was sent successfully or rejected) the sender will be provided with a receipt email. The email will be signed with DP’s certificate and encrypted with the sender system’s certificate.

For successfully sent emails, attached to the receipt are an XML file with relevant information named dkalkvittering.xml.

Example of dkalkvittering.xml with schema EpostAfsendelseKvittering.XSD

|  |
| --- |
| <EpostAfsendelseKvittering xmlns="urn:oio:dkal:1.0.0">  <MeddelelseIdentifikator>1234-5678-3456</MeddelelseIdentifikator>  <AfsendelseModtager>  <CPRnummerIdentifikator>12345678</CPRnummerIdentifikator>  </AfsendelseModtager>  <AfsendelseDatoTid>2009-12-24T12:34:56</AfsendelseDatoTid>  <SystemIdentifikator>43</SystemIdentifikator>  </EpostAfsendelseKvittering> |

The recipient of the receipt can be configured in Administrative Access.

When an email cannot be delivered to the end user, an error receipt is sent back to the sender address. This always happens and requires no setup. Error receipts can be identified with the subject field starting with the text 'Fejlkvittering:'. The error itself is described via an XML attachment. The attachment is described via the form 'Fejl.XSD'

Example of a error message with schema Fejl.XSD

|  |
| --- |
| <?xml version="1.0" encoding="UTF-8" standalone="yes"?>  <Fejl xmlns="urn:oio:dkal:1.0.0">  <FejlKode>2001</FejlKode>  <FejlTekst>NgDP error codes: dp.invalid, NgDP error texts: cvc-enumeration-valid: Value 'planlagtDp2' is not facet-valid with respect to enumeration '[afventer, fremsendt, planlagt]'. It must be a value from the enumeration.</FejlTekst>  </Fejl> |

##### Metadata for receipts

The format of the meta data that is used for sending receipts is described below. The meta data is an attached XML file named dkalkvittering.xml.

|  |  |
| --- | --- |
| **Field** | **Description** |
| SystemIdentifikator | Not used by DP |
| AfsendelseModtager | The recipient of the message identified by a CPR or CVR number |
| MeddelelseTraadIdentifikator | Not used by DP |
| MeddelelseIdentifikator | The identifier of the message |
| AfsendelseDatoTid | The time where the initial sender message was sent |

#### Sending DP2 messages

1. The sender system should be configured in Administrative Access to sent via SMTP
2. The mail must be signed with the OCES certificate registered on the sender system via Administrative Access
3. The sender system use the OCES certificate for encryption. The certificate can be uploaded to Administrative Access
4. To specify meta data, sender system must attach a meta data file with the name 'dkalmetadata.xml'  
     
   Example of dkalmetadata.xml for DP2

|  |
| --- |
| <?xml version="1.0" encoding="UTF-8" standalone="yes"?>  <ns2:EpostAfsendelseMetadata xmlns="urn:oio:dkal:1.0.0" xmlns:ns2="urn:oio:dkal:2.0.0">  <SystemIdentifikator>444</SystemIdentifikator>  <ns2:MeddelelseIdentifikator>123</ns2:MeddelelseIdentifikator>  <AfsendelseModtagerSamling>  <ns2:AfsendelseModtager>  <CPRnummerIdentifikator>1234</CPRnummerIdentifikator>  </ns2:AfsendelseModtager>  <ns2:AfsendelseModtager>  <CVRnummerIdentifikator>12345</CVRnummerIdentifikator>  </ns2:AfsendelseModtager>  </AfsendelseModtagerSamling>  <MeddelelseIndholdstypeIdentifikator>1</MeddelelseIndholdstypeIdentifikator>  <MeddelelseSvarTypeNavn>angivet</MeddelelseSvarTypeNavn>  <MeddelelseSvarPostkasseIdentifikator>321</MeddelelseSvarPostkasseIdentifikator>  <MeddelelseSvarEmneIdentifikator>123</MeddelelseSvarEmneIdentifikator>  <MeddelelseTraadIdentifikator>asdasd</MeddelelseTraadIdentifikator>  <MeddelelseTidsfristDato>+999999999-12-31T23:59:59.999999999</MeddelelseTidsfristDato>  <MeddelelseTidsfristTekst>tidsfristtekst</MeddelelseTidsfristTekst>  <MeddelelseFESDmetadata>  <FESDdokumentIdentifikator>b2770188-e83f-4334-b575-e6a15a5b5918</FESDdokumentIdentifikator>  <FESDaktoerIdentifikator>870d5800-2963-4638-a9ae-6cdb7a2c3ca6</FESDaktoerIdentifikator>  <FESDsagIdentifikator>e900aa10-2781-45ea-b9a1-99cc7762dca0</FESDsagIdentifikator>  <FESDsagsklassifikationIdentifikator>846efeab-1bd6-429d-bfc8-ead1d2dd5575</FESDsagsklassifikationIdentifikator>  </MeddelelseFESDmetadata>  <AfsendelseAdviseringMailTekst>Notfication text</AfsendelseAdviseringMailTekst>  </ns2:EpostAfsendelseMetadata> |

1. The sender needs to set an identifier of the sender system in the MIME message. This is done by setting the custom header X-API-KEY to the UUID of the sender system
   1. The value of the X-API-KEY will be replaced by an API key that will be provided when creating the sender system
2. The actual content of the message must be added to the body, and any attachments can be added. Note that the subject field will not necessarily be encrypted. In front of the subject field, an indicator text for the content type is added
3. All mails should be sent signed and encrypted with DP’s certificate to [dp@test.digitalpost.dk](mailto:dp@test.digitalpost.dk). A designated production email address will be added

##### Afsendelse metadata

The format of the meta data that is used for sending messages is described below. The meta data is an attached XML file named dkalmetadata.xml.

|  |  |
| --- | --- |
| **Field** | **Description** |
| SystemIdentifikator | Not used by DP |
| AfsendelseModtagerSamling | The recipients of the message identified by a CPR or CVR number. The afsendelse is treated individually for each recipient, and the sender system will receive a receipt for each |
| MeddelelseIndholdstypeIdentifikator | ID used to set standard values for similar messages. The following values can be set through this field:   * A title / title prefix * The message can be marked Mandatory * MeddelelseSvarTypeNavn * MeddelelseSvarPostkasseIdentifikator * MeddelelseSvarEmneIdentifikator * MeddelelseTypeNavn   If this value is not set, the required data has to be set in the Afsendelse XML. |
| MeddelelseIdentifikator | The identifier of the initial sender message |
| MeddelelseSvarTypeNavn | Standard: The value from MeddelelseIndholdstypeIdentifikator is used  Angivet: The recipient can reply to the message  IkkeMuligt: The recipient can not reply to the message |
| MeddelelseSvarPostkasseIdentifikator | Marks (together with MeddelelseSvarEmneIdentifikator) which contact point a reply to the message should be sent to |
| MeddelelseSvarEmneIdentifikator | Marks (together with MeddelelseSvarPostkasseIdentifikator) which contact point a reply to the message should be sent to |
| MeddelelseTraadIdentifikator | Not used by DP |
| MeddelelseTidsfristDato | Marks a deadline for replying to the message |
| MeddelelseTidsfristTekst | Not used by DP |
| MeddelelseFESDmetadata | The values in these fields will be returned in any reply to the message |
| BilagSamling | Not used by DP |
| MeddelelseKvitteringsTypeNavn | Not used by DP |
| MeddelelseKvitteringPostkasseIdentifikator | Not used by DP |
| AfsendelseAdviseringMailTekst | Specifies individual notification text for email notifications |

##### Receipts

After sending the message (either if it was sent successfully or rejected) the sender will be provided with a receipt email. The email will be signed with DP’s certificate and encrypted with the sender system’s certificate.

For successfully sent emails, attached to the receipt is an XML file with relevant information named dkalkvittering.xml.

Example of dkalkvittering.xml with schema EpostAfsendelseKvttering.XSD

|  |
| --- |
| <?xml version="1.0" encoding="utf-8"?>  <dkal2:EpostAfsendelseKvittering xmlns:dkal1="urn:oio:dkal:1.0.0" xmlns:dkal2="urn:oio:dkal:2.0.0">  <dkal1:SystemIdentifikator>931</dkal1:SystemIdentifikator>  <dkal2:MeddelelseIdentifikator>000931171272</dkal2:MeddelelseIdentifikator>  <dkal1:AfsendelseModtagerSamling>  <dkal2:AfsendelseModtager>  <dkal1:CPRnummerIdentifikator>0703740001</dkal1:CPRnummerIdentifikator>  </dkal2:AfsendelseModtager>  </dkal1:AfsendelseModtagerSamling>  <dkal1:MeddelelseTraadIdentifikator>2016A10A14A10B05B41B713953</dkal1:MeddelelseTraadIdentifikator>  <dkal1:AfsendelseDatoTid>2016-11-24T01:02:03+04:05</dkal1:AfsendelseDatoTid>  </dkal2:EpostAfsendelseKvittering> |

The recipient of the receipt can be configured in Administrative Access.

When an email cannot be delivered to the end user, an error receipt is sent back to the sender address. This always happens and requires no setup. Error receipts can be identified with the subject field starting with the text 'Fejlkvittering:'. The error itself is described via an XML attachment. The attachment is described via the form 'Fejl.XSD'.

Example of error receipt with schema Fejl.XSD in DP2:

|  |
| --- |
| <?xml version="1.0" encoding="UTF-8" standalone="yes"?>  <ns2:Fejl xmlns="urn:oio:dkal:1.0.0" xmlns:ns2="urn:oio:dkal:2.0.0">  <FejlKode>2001</FejlKode>  <FejlTekst>NgDP error codes: dp.invalid, NgDP error texts: cvc-enumeration-valid: Value 'planlagtDp2' is not facet-valid with respect to enumeration '[afventer, fremsendt, planlagt]'. It must be a value from the enumeration.</FejlTekst>  <ns2:EpostAfsendelseKvittering>  <SystemIdentifikator>0</SystemIdentifikator>  <ns2:MeddelelseIdentifikator>1234</ns2:MeddelelseIdentifikator>  <AfsendelseModtagerSamling>  <ns2:AfsendelseModtager>  <CPRnummerIdentifikator>0000000000</CPRnummerIdentifikator>  <CVRnummerIdentifikator>00000000</CVRnummerIdentifikator>  </ns2:AfsendelseModtager>  <ns2:AfsendelseModtager/>  </AfsendelseModtagerSamling>  <AfsendelseDatoTid>+999999999-12-31T23:59:59.999999999</AfsendelseDatoTid>  </ns2:EpostAfsendelseKvittering>  </ns2:Fejl> |

##### Metadata for receipts

The format of the meta data that is used for sending receipts is described below. The meta data is an attached XML file named dkalkvittering.xml.

|  |  |
| --- | --- |
| **Field** | **Description** |
| SystemIdentifikator | Not used by DP |
| AfsendelseModtagerSamling | The recipients of the message identified by a CPR or CVR number. The afsendelse is treated individually for each recipient, and the sender system will receive a receipt for each |
| MeddelelseTraadIdentifikator | Not used by DP |
| MeddelelseIdentifikator | The identifier of the initial sender message |
| AfsendelseDatoTid | The time where the initial sender message was sent |

### Legacy error codes

This chapter describes the error codes DP will return in legacy receipt formats.

#### XML/CSV receipt error codes

Wherever possible, DP error codes will be mapped to legacy error codes. If no mapping is available, the error code will be 2999. The error text is always ‘DP error codes: {error codes}, DP error texts: {error messages}’.  
The error codes are mapped according to the following table:

|  |
| --- |
| **do**.not.deliver.until.date.too.early=2001  **do**.not.deliver.until.date.too.late=6004  recipient.is.closed=6003  recipient.is.exempt=4090  recipient.nem.sms.is.not.allowed=6003  recipient.not.found=4007  recipient.nem.sms.subscription.not.found=6003  recipient.nem.sms.subscription.mobile.number.not.verified=6003  recipient.mailbox.not.found=4007  recipient.mailbox.and.**default**.recipient.system.not.found=4007  sender.mandatory.message.not.allowed=3002  memo.file.size.too.large=2002  dp.invalid=2001  dp.neither.cpr.nor.cvr.given=4018  dp.both.cpr.and.cvr.given=4019  dp.cpr.invalid=4042  dp.cvr.invalid=4043  dp.indholdstype.not.found=4012  dp.meddelelse.tidsfrist.data.is.empty=4063  dp.vedhaeftning.navn.too.**long**=4069  dp.meddelelse.titel.tekst.too.**long**=4071  dp.afsendelse.advisering.mail.tekst.too.**long**=4120  dp.not.allowed=3002  dp.vedhaeftning.indhold.data.required=4052  dp.meddelelse.indhold.data.required=4052  dp.postkasse.id.not.found=4016  dp.postkasse.emne.id.not.found=4017  dp.contact.point.id.not.found=4020  dp.materialeId.mapping.not.found=4059  dp.materialeId.mapping.not.found.and.defaultMaterialeId.not.set=4005  message.neither.encrypted.nor.signed=9001  wrong.certificate=3004  invalid.certificate=3004 |

#### Record receipt error codes

Wherever possible, DP error codes will be mapped to legacy error codes. If no mapping is available, the error code will be 99. The error text is always ‘{error codes}’.  
The error codes are mapped according to the following table:

|  |
| --- |
| **do**.not.deliver.until.date.too.late -> 17  recipient.is.closed -> 11  recipient.is.exempt -> 11  recipient.nem.sms.is.not.allowed -> 7  recipient.not.found -> 11  recipient.nem.sms.subscription.not.found -> 7  recipient.nem.sms.subscription.mobile.number.not.verified -> 7  recipient.mailbox.not.found -> 11  recipient.mailbox.and.**default**.recipient.system.not.found -> 11  memo.file.size.too.large -> 22  dp.neither.cpr.nor.cvr.given -> 13  dp.meddelelse.tidsfrist.data.is.empty -> 63  dp.afsendelse.advisering.mail.tekst.too.**long** -> 72  dp.vedhaeftning.indhold.data.required -> 21  dp.meddelelse.indhold.data.required -> 21  dp.contactpointid.mapping.not.found -> 65  dp.materialeId.mapping.not.found -> 9 |

### Fetching registration status for contact

This section describes the available REST legacy services for fetching the registration status of a contact in the contact-registry of Digital Post. These can be used by any authority sender system that has been set up for DP.

|  |  |
| --- | --- |
| **/afsendersystem/{systemid}/tilmeldinger/{indholdstypeId}** | |
| description | Returns a boolean if the contact is able to receive a specific indholdstype. For NemSMS the end-user needs a verified mobil number, for the service to return true. |
| content-type | application/xml |
| accept | application/xml |
| encoding | UTF-8 |
| request-type | GET |
| Responses | 200, 400, 401, 403, 404 |
| Return Data | Version 1 Fejl xml for 400 BAD REQUEST and 404 NOT FOUND, otherwise Boolean |
| Input parameters | |
| systemId | UUID of the sender system in the system-registry |
| indholdstypeId | ID of the indholdstype that is requested. The field have to be specified |
| Input search parameters | |
| CVR | Specifies the end-user of type company and authority. CPR or CVR should be specified. |
| CPR | Specifies the end-user of type citizen. CPR is without hyphen. CPR or CVR should be specified. |
| Content | |
| bool | Boolean in xml |
| Example | |
| GET /afsendersystem/a8616175-d285-484d-a2e6-b4a870cedee7/25?cpr=0703740001  <Boolean>false</Boolean> | |

### Fetching registration status list

This section describes the available REST legacy services for fetching the registration status list.

The return type defaults to JSON, so the accept-header is needed if XML is wanted.

|  |  |
| --- | --- |
| **/afsendersystem/{systemid}/tilmeldingsliste** | |
| description | Returns a list of id’s (0, 1, 2…) that can be used to fetch each part list |
| content-type | application/xml |
| accept | application/xml, application/xml |
| encoding | UTF-8 |
| request-type | GET |
| Responses | 200, 400, 401, 403, 404 |
| Return Data | Version 1 Fejl xml for 400 BAD REQUEST and 404 NOT FOUND, otherwise TilmeldingSamlingReferenceSamling |
| Input parameters | |
| systemId | UUID of the sender system in the system-registry |
| Example | |
| GET /afsendersystem/a8616175-d285-484d-a2e6-b4a870cedee7/tilmeldingsliste/  "tilmeldingSamlingIdentifikator" : 11, "tilmeldingDelSamlingIdentifikator" : [ 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13 ], "tilmeldingSamlingURLreference" : "0", "tilmeldingDelSamlingURLreference" : [ ], "tilmeldingSamlingDannetDatoTid" : "2021-10-12T11:03:12.782Z", "systemIdentifikator" : 0, "tilmeldingSamlingKompletIndikator" : true | |

### Fetching contact registration status part list

This section describes the available REST legacy services for fetching the registration status part list. Together all the part lists contains all contacts registered for digital post or nemsms in DP. In the returned data a contact will have one row for the digital post registration and another for the nemsms registration. These can be used by any authority sender system that has been set up for Digital Post.

|  |  |
| --- | --- |
| **/afsendersystem/{systemid}/tilmeldingsliste/{tilmeldingslisteId}/{dellisteId}** | |
| description | Returns a Tilmeldingsdata in a csv format with registration status of a part of the contacts in the DP contact-registry. |
| content-type | application/xml, application/json |
| encoding | UTF-8 |
| request-type | GET |
| Responses | 200, 401, 403, 404 |
| Return Data | Version 1 Fejl xml for 404 NOT FOUND, otherwise Tilmeldingsdata. Data is returned in a semicolon separated format (CSV). |
| Input parameters | |
| systemId | UUID of the sender system in the system-registry |
| tilmeldingslisteId | ID of the registration status list (tilmeldingsliste) that is requested. The id is an integer and have to be specified. |
| dellisteId | ID of the registration status part list (delliste) that is requested. The id is an integer and have to be specified. |
| Return data | |
| Tilmeldingsdata | Tilmeldingsdata in csv format. 1. row: Header that describes the columns. This will only be on the first list. 2..n row: Contains the following fields (Modtager, ModtagerType, Indholdstype, Tilmeldt) separated by semicolons.  Modtager: CPR / CVR number of the contact, formatted without special characters.  ModtagerType: Indicates whether the contact is a citizen (P )or a company (V).  Indholdstype: Indicates whether the contact is registered for digital post (D) or nemsms (S).  Tilmeldt: Indicates whether the contact is registered. Will always be 1. |
| Example | |
| GET /afsendersystem/a8616175-d285-484d-a2e6-b4a870cedee7/tilmeldingsliste/0/0  Modtager;ModtagerType;Indholdstype;Tilmeldt 99889988;V;D;1 0101009999;P;D;1 0202009998;P;D;1 0303009997;P;D;1 0303009997;P;S;1  … 99889977;V;D;1 | |

### Bulk receipt list (massekvitteringsliste)

This section describes the available REST legacy services for fetching a list of receipt send with the asynchronous endpoint. A list of receipts can be fetched by a sender system and the list will contains all receipts for the messages send by that sender system. Once a list have been fetched, the same list will be returned until the list is deleted. The list can be deleted by the sender system, and it have to be deleted before a new list with newer receipts can be fetched. Receipts will also be deleted from the list if they have been delete by the clean-up scheduler.

Only XML return values are supported for this endpoint.

### Fetch bulk receipt list

|  |  |
| --- | --- |
| **/afsendersystem/{systemid}/masseafsendelser/kvitteringsliste/** | |
| description | Returns a bulk receipt list. |
| content-type | application/xml |
| accept | application/xml |
| encoding | UTF-8 |
| request-type | GET |
| Responses | 200, 401, 403 |
| Return Data | MasseforsendelseKvitteringSamling |
| Input parameters | |
| systemId | UUID of the sender system in the system-registry |
| Return data | |
| MasseforsendelseKvitteringSamling | MasseforsendelseKvitteringSamling with the fields :  MasseforsendelseKvitteringSamlingIdentifikator, MasseforsendelseKvitteringSamlingURLreference, MasseforsendelseKvitteringSamlingDannetDatoTid, SystemIdentifikator and MasseforsendelseKvitteringData |
| Fields in MasseforsendelseKvitteringSamling | |
| MasseforsendelseKvitteringSamlingIdentifikator | Id of the bulk receipt list |
| MasseforsendelseKvitteringSamlingURLreference |  |
| MasseforsendelseKvitteringSamlingDannetDatoTid | Timestamp of when the list was created |
| SystemIdentifikator | Always 0 in DP |
| MasseforsendelseKvitteringData | Contain a line for every receipt and have fields: AfsendelseModtager, AfsendelseModtagerType, MeddelelseIndholdstype Identifikator, MeddelelseIdentifikator, IndholdStoerrelseMaal, FejlKode and KvitteringTekst. The fields are separated by semicolons. |
| Fields in MasseforsendelseKvitteringData | |
| AfsendelseModtager | CPR / CVR number of the contact, formatted without special characters. |
| AfsendelseModtagerType | Indicates whether the contact is a citizen (P )or a company (V). |
| MeddelelseIndholdstype Identifikator | Specifies the indholdstype from the message |
| MeddelelseIdentifikator | Specifies the MeddelelseIdentifikator |
| IndholdStoerrelseMaal | Specifies the size in bytes of the message |
| FejlKode | Error code |
| KvitteringTekst | Error message |
| Example | |
| GET /afsendersystem/a8616175-d285-484d-a2e6-b4a870cedee7/masseafsendelser/kvitteringsliste/   |  | | --- | | <MasseforsendelseKvitteringSamlingType>  <masseforsendelseKvitteringSamlingIdentifikator>3322230</masseforsendelseKvitteringSamlingIdentifikator>  <systemIdentifikator>0</systemIdentifikator>  <masseforsendelseKvitteringSamlingDannetDatoTid>2020-05-07T11:53:24.064Z</masseforsendelseKvitteringSamlingDannetDatoTid>  <masseforsendelseKvitteringData>  2307921515;P;3;;82032;0;  2707921414;P;300;testId;4007;NgDP error codes: recipient.not.found, NgDP error texts: Contact with CPR 2307921514 does not exist  88778877;V;85;testId2;52042;0;  2412001010;P;998;testId3;27977;0;  </masseforsendelseKvitteringData>  </MasseforsendelseKvitteringSamlingType> | | |

### Delete bulk receipt list

|  |  |
| --- | --- |
| **/afsendersystem/{systemid}/masseafsendelser/kvitteringsliste/{kvitteringslisteId}** | |
| description | Delete the last fetched bulk receipt list. |
| content-type | application/xml |
| encoding | UTF-8 |
| request-type | DELETE |
| Responses | 204, 404 |
| Return Data | 204 No content for delete, otherwise 404 not found |
| Input parameters | |
| systemId | UUID of the sender system in the system-registry |
| kvitteringslisteId | ID of the bulk receipt list |
| Example | |
| DELETE /afsendersystem/a8616175-d285-484d-a2e6-b4a870cedee7/778855 | |

### Handling of reply destination in Digital Post contact structure when sending DP/ DP2

Sending message in DP/DP2 format which needs to be replied to, should contain references to *postkasser* and *postkasseemner*, either stated in the message itself or in the Material indicated with *MeddelelseIndholdstypeIdentifikator.* When the message is transformed into the MeMo format, these will be translated into a contact point in the DP contact structure.

**Migration of Material from Eboks to Digital Post**

Note that the Materiale currently available for testing has been migrated from the Eboks PRODUCTION environment on the 8th of July 2021 to the DP TEST- and PRODUCTION environment.

At the DP go live ultimo november newly created Materiale created on the Eboks PRODUCTION environment between the 8/7-2021 to 21/11-2021 will be migrated to the DP TEST- and PRODUCTION environment.

#### The mapping of postkasse and postkasseemne to DP contact structure

The mapping of *postkasseId* and *postkasseemneId* from E-boks solution to the DP contact structure, can be handled manually. In order to alter the current contact structure, *postkasse* and *postkasseemner* can be added or removed from contact groups and contact points. This can be handled through Administrative Access.

#### Default Material

In the DP sender systems can have a default material added. When the sender system sends a DP/DP2 message without *MeddelelseIndholdstypeIdentifikator,* then the default material from the sender system will be used for the transformation of the message from DP/DP2 to MeMo. If material or its elements are not specified in the DP/DP2 message or on the sender system, then the DP/DP2 will be rejected by DP. A material is a value that says something about the title of the message, the message type, or which mailbox (contact point) replies should be sent to.  
The default material can be specified on sender systems through Administrative Access. Therefore, the default value is specified only on specific systems, and not on standard systems.

#### Use cases for using material and postkasseId and postkasseEmneId when sending DP/DP2

In the following, three use cases for setting a reply desitination is shown. The use cases are:

* *postkasseId* and *postkasseemneId* are specified in the message
* *postkasseId* and *postkasseemneId* are specified in the material the *MeddelelseIndholdstypeIdentifikator* points to
* *postkasseId* and *postkasseemneId* are specified in a default material for the sender system

**Use case 1 - Specified in the message**

PostkasseId and postkasseemneId can be stated in the message by the usage of the fields:  
*MeddelelseSvarPostkasseIdentifikator* (line: 13) and *MeddelelseSvarEmneIdentifikator (line: 14)* where the field *MeddelelseSvarTypeNavnType* is set to *angivet (line: 12) .*

|  |
| --- |
| <?xml version="1.0" encoding="UTF-8"?>  <Afsendelse xmlns="urn:oio:dkal:1.0.0" xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">  <MeddelelseIdentifikator>Test</MeddelelseIdentifikator>  <AfsendelseModtager>  <CPRnummerIdentifikator xmlns="urn:oio:adir:dagpenge:2009.07.01">2307921515</CPRnummerIdentifikator>  </AfsendelseModtager>  <MeddelelseTypeNavn>meddelelse</MeddelelseTypeNavn>  <MeddelelseIndholdstypeIdentifikator>3</MeddelelseIndholdstypeIdentifikator>  <MeddelelseTitelTekst>test test test</MeddelelseTitelTekst>  <MeddelelseIndholdData>T0JTISBzZSB2ZWRow6ZmdG5pbmcgZm9yIGRhdGE=</MeddelelseIndholdData>  <FilformatNavn>pdf</FilformatNavn>  <MeddelelseSvarTypeNavn>angivet</MeddelelseSvarTypeNavn>  <MeddelelseSvarPostkasseIdentifikator>99999</MeddelelseSvarPostkasseIdentifikator>  <MeddelelseSvarEmneIdentifikator>888888</MeddelelseSvarEmneIdentifikator>  </Afsendelse> |

**Use case 2 - Specified in the material**

If the *postkasseId* and *postkasseemneId* from a given material should be used, then it is needed that your organisation hassuch a material, with the values specified. Furthermore, do the *postkasseId* and *postkasseemneId* on the material must be related to a contact point. In this use case the field *MeddelelseSvarTypeNavnType* is set to *standard (line: 12) .*

|  |
| --- |
| <?xml version="1.0" encoding="UTF-8"?>  <Afsendelse xmlns="urn:oio:dkal:1.0.0" xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">  <MeddelelseIdentifikator>Test</MeddelelseIdentifikator>  <AfsendelseModtager>  <CPRnummerIdentifikator xmlns="urn:oio:adir:dagpenge:2009.07.01">2307921515</CPRnummerIdentifikator>  </AfsendelseModtager>  <MeddelelseTypeNavn>meddelelse</MeddelelseTypeNavn>  <MeddelelseIndholdstypeIdentifikator>3</MeddelelseIndholdstypeIdentifikator>  <MeddelelseTitelTekst>test test test</MeddelelseTitelTekst>  <MeddelelseIndholdData>T0JTISBzZSB2ZWRow6ZmdG5pbmcgZm9yIGRhdGE=</MeddelelseIndholdData>  <FilformatNavn>pdf</FilformatNavn>  <MeddelelseSvarTypeNavn>standard</MeddelelseSvarTypeNavn>  </Afsendelse> |

**Use case 3 - Specified in a default material**

In this use case is a *MeddelelseIndholdstypeIdentifikator* not specified, however the MeddelelseSvarTypeNavnType is still set to *standard*. DP will use the default material if the sender system has it, otherwise it will reject the massage.

|  |
| --- |
| <?xml version="1.0" encoding="UTF-8"?>  <Afsendelse xmlns="urn:oio:dkal:1.0.0" xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">  <MeddelelseIdentifikator>Test</MeddelelseIdentifikator>  <AfsendelseModtager>  <CPRnummerIdentifikator xmlns="urn:oio:adir:dagpenge:2009.07.01">2307921515</CPRnummerIdentifikator>  </AfsendelseModtager>  <MeddelelseTypeNavn>meddelelse</MeddelelseTypeNavn>  <MeddelelseTitelTekst>test test test</MeddelelseTitelTekst>  <MeddelelseIndholdData>T0JTISBzZSB2ZWRow6ZmdG5pbmcgZm9yIGRhdGE=</MeddelelseIndholdData>  <FilformatNavn>pdf</FilformatNavn>  <MeddelelseSvarTypeNavn>standard</MeddelelseSvarTypeNavn>  </Afsendelse> |

**NB**. The *postkasseId* and *postkasseemneId* in DP is not environment specific, thus the prod values will be used on both the test environment and on production environment. Moreover, the materials in the DP solution is migrated from the E-boks solution. Both the materials and the postkasse data is static, and the specific values are not meant to be changed or altered.

### Contact-registry in Record format

Every night, at 02.00 AM, a copy of the contact registry is fetched and mapped over to a Record format.

The Record file is uploaded to the SFTP server that can be accessed with SSH username & private part of the SSH key for a sender system. The file is found under the /contacts/ folder.

The file is called 'TILMELD.K0000000.DYYMMDD, where YYMMDD is the date of the upload, eg. 'TILMELD.K0000000.D210313'.

The following tables descibe the fields in each record.

#### Header record

|  |  |  |
| --- | --- | --- |
| Field name | Length | Description |
| Record-type | 8 | Always set to 'EBOKS001' |
| Struktur-version | 3 | Always set to '005' |
| Data-type | 30 | Always set to 'Tilmeldingsliste' Left-aligned with following blank spaces. |
| Kunde-Id | 15 | Filled out with 0s |
| Dannelsestidspunkt | 26 | Timestamp e.g ‘2000-12-24-12.30.45.123456’ |
| Filler | 18 | Filled out with blank spaces |
| System-Id | 15 | Filled out with 0s |
| KompletListe | 1 | Always set to 'J' |

#### Parameter record

|  |  |  |
| --- | --- | --- |
| Field name | Length | Description |
| Record-type | 8 | Always set to 'EBOKS002' |
| Struktur-version | 3 | Always set to '006' |
| Tilmeldingsgryppe | 15 | Set to **D** if the contact is registered to Digital Post and **S** if the contact is registered to NemSMS. There is one parameterrecord for each registration. Right-aligned with 0s infront. |
| Bruger-Type | 20 | **P** for CPR number and **V** for CVR number. Left-aligned with following blank spaces. |
| Bruger | 50 | Identification of the contact in the form of either CPR or CVR number. Left-aligned with following blank spaces. |
| Filler | 4 | Filled out with blank spaces. |
| Tilmeldt | 1 | Always set to 'J' |

#### Trailer record

|  |  |  |
| --- | --- | --- |
| Field name | Length | Description |
| Record-Type | 8 | Always set to 'EBOKS003' |
| Struktur-version | 3 | Always set to '003' |
| Antal parameterrecords | 15 | The amount of parameterrecords in the file. Right-aligned with 0s infront. |
| Filler | 74 | Filled out with blank spaces. |

**Eksempelfil:**

|  |
| --- |
| EBOKS001005Tilmeldingsliste 0000000000000002021-10-08-11.03.46.813095  000000000000000J  EBOKS00200600000000000000DV 43768500  J  EBOKS00200600000000000000DP 1004749983  J  EBOKS00200600000000000000DP 2412001010  J  EBOKS00200600000000000000DV 65307316  J  EBOKS00200600000000000000SP 0610740002  J  EBOKS00200600000000000000DP 0610740001  J  EBOKS00200600000000000000SP 1212042538  J  EBOKS00200600000000000000DV 82828282  J  EBOKS00200600000000000000DV 99881101  J  EBOKS003003000000000000009 |

### Contact in legacy csv

#### Purpose of the sender sftp

To support legacy systems, solution should export all the contacts persisted in DP to .csv file format and upload it to shared folder on sftp server under contacts directory. Each record is separated with ';' delimiter. Where it might be downloaded by external sender systems. Described above procedure is to be repeated each night.

#### Contact Legacy csv file format

**Header**

Header contains metadata of job.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Attribute** | **Description** | **Required** | **Value** | **Header** |
| createdDateTime | job end date and time | Yes | Date time | DannetDatoTid |
| systemIdentifier | Producer system id | Yes | 0 - integer, only one system generates csv file. | SystemIdentifikator |
| complete | The end job status | Yes | 1 - completed, otherwise job is not finished and no csv file uploaded | KompletIndikator |

**Entry**

Model is created by mapping correct values from contact model to csv entry.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Attribute** | **Description** | **Required** | **Value** | **Header** |
| recipient | cprNumber or cvrNumber | Yes | Numeric value | Modtager |
| recipientType | Denotes what type of recipient it is | Yes | V - company type or P - citizen | ModtagerType |
| messageType | Denotes type of subscription | Yes | D - digital post or S - nemSms | Indholdstype |
| signedUp | If recipient is subscribed to given messageType, this field is set to 1. Records with status 0 are cut out from file. | Yes | 1 - subscription type exist, 0 otherwise. | Tilmeldt |

**e.g.**

DannetDatoTid;SystemIdentifikator;KompletIndikator  
2021-11-09 00:18:21;0;1  
Modtager;ModtagerType;Indholdstype;Tilmeldt  
1111112323;P;D;1  
12211223;V;D;1

Entry - 3

…

Entry - n

### Distribution REST services

Note, that services sending and receiving MeMo’s and DP/DP2 messages only support XML (not JSON).

Generally all non-memo responses are JSON, but XML is supported by using the Accept header.

## Inbound services

From the distribution component, the following services are exposed:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Service** | **URL** | **Data returned** | **Usage** | **Required roles** | **Consumer** |
| Send MeMo messages | POST  /memos/ | Technical receipt with transmissionId | Sending .tar.lzma files containing MeMo’s, or single xml memo files, to the solution | Sender system | Sender system |
| Send DP messages (asynchronously) | PUT  /dp/afsendersystem/{systemid}/masseafsendelser/{meddelelsesid} |  | Sending DP messages to the solution | Sender system | Sender system |
| Send DP2 messages (asynchronously) | PUT  /dp2/afsendersystem/{systemid}/masseafsendelser/{meddelelsesid} |  | Sending DP2 messages to the solution | Sender system | Sender system |
| Send DP messages (synchronously) | PUT  /dp/afsendersystem/{systemid}/afsendelser/{meddelelsesid} |  | Sending DP messages to the solution | Sender system | Sender system |
| Send DP2 messages (synchronously) | PUT  /dp2/afsendersystem/{systemid}/afsendelser/{meddelelsesid} |  | Sending DP2 messages to the solution | Sender system | Sender system |
| Recall delayed MeMo message | DELETE  /memos/{memo-id} |  | Recalling MeMos before due date | Sender system  Public authority administrator | Sender system  Public authority administrator |
| Fetch MeMo | GET  /memos/{memo-id} | MeMo | Fetching memo’s from DP (publish-subscribe) | Recipient System | Recipient System |
| Fetch list of available MeMos | GET  /memos/ | List of UUID’s | Fetching a list of all available MeMos for the recipient system (publish subscribe) | Recipient System | Recipient System |
| Send business receipt | POST  /memos/{memo-id}/receipt/ |  | Sending a business receipt to DP | Recipient System | Recipient System |
| Fetch business receipt | GET  /receipts/{receiptId} | Business receipt | Fetching business receipts from DP (REST\_PULL) | Sender System | Sender System |
| Fetch list of available business receipts | GET  /receipts/ | List of UUID’s | Fetching a list of available business receipts for the sender system (REST\_PULL) | Sender System | Sender System |

### Send MeMo messages

In the ‘Send MeMo messages’ request, the MeMos should be sent as a requests body with a accompanying Content-Type header:

* A tar archive compressed with LZMA, the type also used when archiving using MeMo-lib, with the [Content-Type](https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/Content-Type) header set to application/x-lzma

or

* An xml file, which requires the [Content-Type](https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/Content-Type) header to be set to application/xml.

The tar-lzma file can be named as wanted, but the individually files in the archive must be named the same as the messageUUID from the MeMo.

The name of the xml file when sending a single MeMo, should be messageUUID from MeMo, without the .xml file extension.

|  |  |  |
| --- | --- | --- |
| **file type** | **Content-Type header** | **filename** |
| tar.lzma | application/x-lzma | any filename |
| xml | application/xml | messageUUID |

#### Should I send single messages or bulks?

|  |
| --- |
| It is important as a sender that you choose the correct way of sending messages as using the wrong method can negatively impact other senders. |

Digital Post includes two distinct methods for sending messages via REST, each designed to cater to different communication needs. Messages can either be sent as a single message, or as bulks of multiple messages (a tar.lzma file with only one message is seen as a single message).

Single messages are meant for individual, one-on-one communication. It is intended for when an employee needs to write and send a message to a single recipient, and the message should be sent immediately.

On the other hand, bulk messages are designed for mass communication or mass sending messages. This interface allows for the distribution of messages to a large group of recipients. It's especially useful when sending out general notifications to all members of a particular group, for instance, all citizens in a municipality.  
It can also be used for sending single messages, if e.g. the sender system is designed to store all messages that are sent within the day, and then sending all those messages at the same time.

Choosing the appropriate method based on your communication needs ensures efficient messaging and optimal communication, as sending bulks via as single messages can negatively impact other senders. Always remember to take into account the number of messages you are sending when deciding which interface to use - If it is more than one, they should be sent as bulks.

## Outbound services

From the distribution component, there are the following outbound services. The URLs which are used in the outbound services are the either the endpoint or the receiptEndpoint which the registered on the relevant sender or receiver system.

|  |  |  |  |
| --- | --- | --- | --- |
| **Service** | **Usage** | **Data sent** | **Consumer** |
| Send receipt via REST | Sending business receipts to sender systems | Business receipt with transmissionId | Sender system |
| Send message via REST | Sending MeMo messages to recipient systems | MeMo | Recipient System |
| Send sms notification | Sending SMS notifications to citizens | Notification message, mobile number, sender name | SMS-gateway |
| Send memo notification | Sending a notification about an available memo (publish subscribe) | URL of available MeMo | Recipient System |

### Sender system respond to received Business Receipt

The sender system is expected to respond with status code **200 OK, 201 CREATED** or **202 ACCEPTED**andwithout a response body upon successfully receiving a Business Receipt.

#### Outbound MeMo REST Push request

This page describes how a MeMo is sent by Digital Post via REST Push.

A MeMo is pushed to a REST PUSH recipient system with mutual SSL using HTTP POST to the recipient system’s endpoint. The MeMo’s messageUUID is appended to the URL, using request parameter name memo-message-uuid. This allows the recipient system to identify the MeMo if the body of the message cannot be interpreted, for some reason.

Example:

|  |
| --- |
| https://dp.skat.dk/modtagersystem/kontaktpunkt-1?memo-message-uuid=a66fcd7b-3392-4c69-ae2e-48f5c2e5ad98 |

Or if the URL already had parameters:

|  |
| --- |
| https://dp.skat.dk/modtagersystem?kontaktpunkt=1&memo-message-uuid=a66fcd7b-3392-4c69-ae2e-48f5c2e5ad98 |

The Content-Type header is: application/xml

Example:

|  |
| --- |
| POST /modtagersystem?kontaktpunkt=1&memo-message-uuid=b6fcb074-2843-4f66-8481-682715232ac9 HTTP/1.1  Host: dp.skat.dk  Content-Type: application/xml  Content-Length: 18634    <**memo**:Message xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:memo="https://DigitalPost.dk/MeMo-1" memoVersion="1.0" memoSchVersion="1.0.0" xmlns:xsd="http://www.w3.org/2001/XMLSchema#">  <**memo**:MessageHeader>  <**memo**:messageType>DIGITALPOST</**memo**:messageType>  <**memo**:messageUUID>b6fcb074-2843-4f66-8481-682715232ac9</**memo**:messageUUID>  <**memo**:label>Til forvaltningen</**memo**:label>  <**memo**:reply>true</**memo**:reply>  <**memo**:mandatory>false</**memo**:mandatory>  <**memo**:legalNotification>false</**memo**:legalNotification>  <**memo**:Sender>  <**memo**:senderID>0610328534</**memo**:senderID>  <**memo**:idType>CPR</**memo**:idType>  <**memo**:label>Lone Hansen</**memo**:label>  </**memo**:Sender>  <**memo**:Recipient>  <**memo**:recipientID>63636363</**memo**:recipientID>  <**memo**:idType>CVR</**memo**:idType>  </**memo**:Recipient>  </**memo**:MessageHeader>  <**memo**:MessageBody>  <**memo**:createdDateTime>2020-06-29T12:00:00Z</**memo**:createdDateTime>  <**memo**:MainDocument>  <**memo**:File>  <**memo**:encodingFormat>text/html</**memo**:encodingFormat>  <**memo**:filename>Hoveddokument</**memo**:filename>  <**memo**:language>da</**memo**:language>  <**memo**:content>JVBER....</**memo**:content>  </**memo**:File>  </**memo**:MainDocument>  </**memo**:MessageBody>  </**memo**:Message> |

#### Inbound MeMo REST Push request

This page contains examples of how to send MeMo messages to DP via REST Push as a sender system.

MeMos must be POST'ed using certificate from active sender system or authentication will fail with a 401.

#### Sending single MeMo

HTTP POST with Content-Type application/xml to Digital Post /memos/ endpoint. messageUUID of the MeMo must be added as a request parameter named 'memo-message-uuid' and Content-Length header **must** be correct:

|  |
| --- |
| POST /apis/v1/memos/?memo-message-uuid=b6fcb074-2843-4f66-8481-682715232ac9 HTTP/1.1  Host: digitalpost.dk  Content-Type: application/xml  Content-Length: 18634    <**memo**:Message xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:memo="https://DigitalPost.dk/MeMo-1" memoVersion="1.0" memoSchVersion="1.0.0" xmlns:xsd="http://www.w3.org/2001/XMLSchema#">  <**memo**:MessageHeader>  <**memo**:messageType>DIGITALPOST</**memo**:messageType>  <**memo**:messageUUID>b6fcb074-2843-4f66-8481-682715232ac9</**memo**:messageUUID>  <**memo**:label>Til forvaltningen</**memo**:label>  <**memo**:reply>true</**memo**:reply>  <**memo**:mandatory>false</**memo**:mandatory>  <**memo**:legalNotification>false</**memo**:legalNotification>  <**memo**:Sender>  <**memo**:senderID>0610328534</**memo**:senderID>  <**memo**:idType>CPR</**memo**:idType>  <**memo**:label>Lone Hansen</**memo**:label>  </**memo**:Sender>  <**memo**:Recipient>  <**memo**:recipientID>63636363</**memo**:recipientID>  <**memo**:idType>CVR</**memo**:idType>  </**memo**:Recipient>  </**memo**:MessageHeader>  <**memo**:MessageBody>  <**memo**:createdDateTime>2020-06-29T12:00:00Z</**memo**:createdDateTime>  <**memo**:MainDocument>  <**memo**:File>  <**memo**:encodingFormat>text/html</**memo**:encodingFormat>  <**memo**:filename>Hoveddokument</**memo**:filename>  <**memo**:language>da</**memo**:language>  <**memo**:content>JVBER....</**memo**:content>  </**memo**:File>  </**memo**:MainDocument>  </**memo**:MessageBody>  </**memo**:Message> |

#### Sending multiple MeMos

**Raw body**

|  |
| --- |
| POST /apis/v1/memos/ HTTP/1.1  Host: digitalpost.dk  Content-Type: application/x-lzma  Content-Length: 2223513431    "<contents here>" |

Using curl:

|  |
| --- |
| curl --location --request POST 'https://digitalpost.dk/apis/v1/memos/' \  --header 'Content-Type: application/x-lzma' \  --data-binary '@/home/user/memos/memo\_archive.tar.lzma' |

**Multipart**

Form element name must be file. Form element content type must be application/x-lzma.

|  |
| --- |
| curl --location --request POST 'https://digitalpost.dk/apis/v1/memos/' \  --header 'Content-Type: multipart/form-data'  --header 'Accept: application/json' \  --form 'file=@"/home/user/memos/memo\_archive.tar.lzma"' |

#### REST\_PULL service protocol

The service protocol REST\_PULL can be used for both recipient and sender rest systems.  
  
For recipient system the service protocols REST\_PULL and REST\_PUBLISH\_SUBSCRIBE are quite similar with the only difference being whether a notification will be sent to the systems endpoint, when a MeMo is ready to be fetched, or not. The purpose of the service protocol REST\_PULL is for recipient systems to explicitly choose not to have notifications every time a MeMo is ready to be fetched. REST\_PULL systems cannot have endpoints.  
Sender systems with REST\_PULL service protocol do not get business receipts send to a receipt-endpoint. It is the sender systems responsibility to fetch business receipt associated with that system.

Systems with REST\_PULL can at any time fetch all available MeMos and/or receipts. See more of how this fecthing is done in [Fetching MeMos](https://digidp.atlassian.net/wiki/spaces/DP/pages/182517998/Fetching+MeMos) or [Fetching receipts](https://digidp.atlassian.net/wiki/spaces/~404685048/pages/800489772/Fetching+business+receipts).

#### Distribution SMTP services

## Inbound services

From the distribution component the following mail services are exposed.

### Send message to Digital Post

MeMo messages can be sent to the DP at memo@test.digitalpost.dk. The message should have a MIME format, be signed with the senders certificate (uploaded via Administrative Access when creating the system) and encrypted with DP’s certificate.

As well as having the following properties:

* The sender needs to set an identifier of the sender system in the MIME message. This is done by setting the custom header X-API-KEY to the UUID of the sender system
* It should contain one attached MeMo message xml named {messageUUID}. (If multiple are sent, only the first is handled)
* The certificate serial number used to sign the message must match the certificate serial number for the respective sender system

## Outbound services

From the distribution component there are the following outbound SMTP services.

### Send MeMos to SMTP Recipient Systems

MeMos which are sent to SMTP Recipient systems are encrypted with sender’s certificate (public key) and signed using DP certificate (private key). MeMos can be decrypted with sender's certificate (private key) and signature needs to be validated using DP certificate (public key).

Both notifications and technical receipts are signed and encrypted. They should be decrypted with sender's certificate (private key) and signature needs to be validated using DP certificate (public key).

#### Send notification to users with EmailSubscriptions

When a new message appears in a user's mailbox, mailbox-indexer generates an event that is consumed by sender-smtp.  
This event prompts the sending of an email notifying of this new message to said user through SMTP (if respective user has 1 or more (up to 5) emails attached as EmailSubscriptions)

* The notification text provided in the email is fetched from MeMo fields (if available), and prioritized as such:
  + AdditionalNotification (MeMo field)
  + Notification (MeMo field)
  + A default notification text

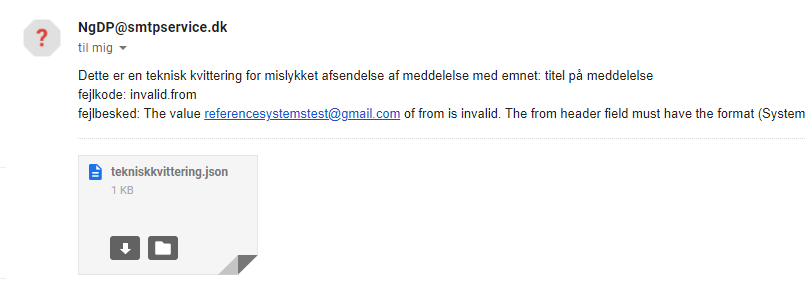
### Send technical receipts to senders

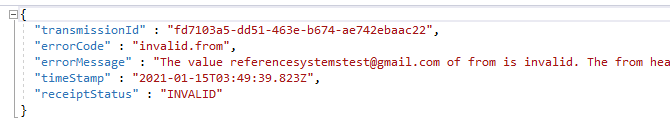
When messages in MIME format has been fetched by fetcher-imap, an event will be produced to sender-smtp.

There are two primary types of technical receipt scenarios:

* A valid technical receipt, if the fetcher-imap encountered no issues in the handling of the MIME message.
  + In this case, the service looks up the senderSystem ID from memoMetaData to grab the receiptEndpoint, and send a technical receipt to this endpoint.
* An invalid technical receipt, if the fetcher-imap did encounter issues in the handling of the MIME message.
  + In this case, the service will fetch the needed endpoint from the MIME\_FROM property in the event produced by fetcher-imap.

The technical receipt mail carries a JSON file with receipt information:





TransmissionId is generated by the solution and returned. This ID will also be present on the business receipt, and can help connect techinal and business receipts.

#### SMTP MeMo example

MeMo messages can be send to the DP mail server. MeMo mails to be fetched from the IMAP server must adhere to a few rules for successful processing.

Attached picture of example of valid mail:



Attached picture of valid mail in client (outlook in this case):



**Overall structure:**

The MIME message must be composed of **two** body parts:

* A multipart/mixed part which at minimum must contain the MeMo XML attached - name of MeMo file must be <messageUUID> or <messageUUID>.xml
* A application/pkcs7-signature or application/x-pkcs7-signature part

**Rules:**

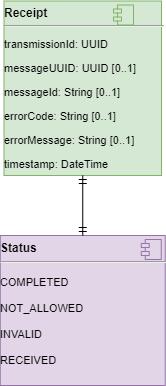
* In the MIME-sender property, the format must be <senderSystemId>@<domain> - where senderSystemId is the UUID of the sender system in DP
* It should contain one attached MeMo message xml named {messageUUID} or {messageUUID}.xml. (If multiple are sent, only the first is handled)
* The certificate serial number used to sign the message must match the certificate serial number for the respective sender system.

The table below describes the rules based on error codes that can be triggered through the processing when fetching mails from the IMAP server:

|  |  |
| --- | --- |
| **Error code** | **Message** |
| mime.message.invalid | Mime message must contain two body parts, a multipart/mixed part containing the message, and a application/pkcs7-signature or application/x-pkcs7-signature part |
| invalid.from | The value {0} of from is invalid. The from header field must have the format (senderSystemId)@(domain), where senderSystemId is the UUID of the sender system in the system registry |
| wrong.certificate | Serial number of certificate used to sign message with id {0} does not match the certificate serial number for sender system with id {1} |
| invalid.certificate | Could not read certificate of message with id {0} |
| no.message.attachment | No application/json or application/xml attachment found on message with id {0} |
| system.not.found | No system found with id {0} |

### Digital Post Receipt domain model

This page describes the model for Digital Post receipts. This page is only relevant for sender systems registered to use MeMo receipts.



The above diagrams shows the domain model for receipts. Fields of type String contains a maximum of 512 characters.

DP operates with two types of receipts, namely Technical Receipts, and Business Receipts. Technical receipts are sent immediately when the message(s) is received by Digital Post, and confirms that Digital Post has received the message. Business receipts are the final state for the sender - This can e.g. be that even though Digital Post was able to receive the message, the memo xml has failed validation, or that the message has been successfully validated by DP, and will be sent.

The meaning of the statuses are:

* COMPLETED: Digital Post has validated the messages.
* NOT\_ALLOWED: The sender is not allowed to send this message
* INVALID: There is something invalid in the request - E.g. the MeMo is not in the correct format
* RECEIVED: Digital Post has received the message, and will start to validate it. The message is still the responsibility of the sender

Some examples below refer to error codes and error messages in both technical and business receipts. Those are used to indicate issues, that the system encountered during message validation. There are various types, and one can look up the details of them under the section **“Back-end validation and error codes in distribution”**.

The following sections will go through technical and business receipts for each protocol.

### REST receipt procedure

#### Technical receipts

The technical receipt indicates if DP have received the message or not. If the message is received it will be further processed and validated. Please note that the results of the processing and validation will be in the business receipt and not in the technical receipt. The technical receipt for REST protocol is the REST response. This response will only contain a body if the call is succeeded, and HTTP status 201 is returned. The body will contain a JSON receipt. The field values are:

|  |
| --- |
| {  "transmissionId": "86f13750-8068-44c1-93cf-a915998831cf",  "timeStamp": "2020-12-15T08:23:32.583Z",  "receiptStatus": "RECEIVED"  } |

Code snippet above contains correct “Technical Receipt”. The status of request is ‘201 Created’. And below is example of incorrect request status 400.

|  |
| --- |
| {  "code": "ValidationException",  "message": "File type 'null' not allowed. Allowed file types: application/xml, application/x-lzma",  "fieldErrors": []  } |

Short description of fields in receipt:

* transmissionId: A UUID generated by Digital Post. This UUID will also be returned in the business receipt, and should be used to chain the two together
* timeStamp: Timestamp of when the message was received, which is always now (UTC timezone)
* receiptStatus: RECEIVED meaning that Digital Post have received the message

#### Business receipt

The business receipt for the REST protocol are generated when the message has been either rejected or passed the validation. The business receipt will contain the result of the validation, and a positiv result means that the message will be delivered to the recipient.

If a sender system it set up with the service protocol *REST\_PUSH* the receipt will be POST’ed to the receipt endpoint of the sender system. If the sender system is set up with the service protocol *REST\_PULL* the receipts can be found by the sender system by issuing a GET request to the endpoint */receipts/*. In both cases the body of the receipt will be in the JSON format.

Examples of REST business receipts are shown below:

|  |
| --- |
| {  "transmissionId": "86f13750-8068-44c1-93cf-a915998831cf",  "messageUUID": "182bd6d1-ab9f-48fb-84f6-4f243ace9780",  "messageId": "MSG-81220",  "errorCode": **null**,  "errorMessage": **null**,  "timeStamp": "2020-12-15T08:23:32.583Z",  "receiptStatus": "COMPLETED"  } |

Code snippet above shows ‘Business Receipt’ with status ‘COMPLETED’. And below with status 'INVALID'.

|  |
| --- |
| {  "transmissionId": "8e62eb2a-8ef7-4034-8bb4-4021ecd9c377",  "messageUUID": "182bd6d1-ab9f-48fb-84f6-4f243ace9780",  "messageId": **null**,  "errorCode": "message.uuid.not.unique",  "errorMessage": "The MessageUUID 182bd6d1-ab9f-48fb-84f6-4f243ace9780 is invalid. MessageUUID must be a unique UUID",  "timeStamp": "2020-12-15T08:23:32.583Z",  "receiptStatus": "INVALID"  } |

Short description of fields in receipt:

* transmissionId: A UUID generated by DP. This UUID will also be returned in the technical receipt, and should be used to chain the two together
* messageUuid: UUID of MeMo if available, otherwise empty (e.g. if tar.lzma extraction fails)
* messageId: Read from MeMo if defined, otherwise empty
* errorCode: Code of error, that appeared during validation process
* errorMessage: Description of error
* timestamp: Now (UTC)
* receiptStatus: Status of message in a system

### SMTP receipt procedure

#### Technical receipts

The technical receipt for SMTP protocol is a response to the S/MIME message that has been sent to DP. The subject of the message is Teknisk Kvittering fra Digital Post. This response will have an attached JSON, that either reports success (ReceiptStatus RECEIVED) or failure (ReceiptStatus INVALID). The field values are:

* TransmissionId: A UUID generated by DP. This UUID will also be returned in the business receipt, and should be used to chain the two together
* MessageUuid: Empty
* ErrorCode: Error code (examples found under section "Back-end validation and error codes indistribution") in case of INVALID status, otherwise *null*
* ErrorMessage: Error message (examples found under section "Back-end validation and error codes in distribution") in case of INVALID status, otherwise *null*
* Timestamp: Now (UTC)
* ReceiptStatus: RECEIVED/INVALID

#### Business receipts

The business receipts for SMTP protocol are sent when the message has been either rejected, or if the message is now the responsibility of the recipient. The subject of the message is Forretningskvittering fra Digital Post. The receipt will be sent to the receipt endpoint (email address), with a JSON receipt as the body. The field values are:

* TransmissionId: A UUID generated by DP. This UUID will also be returned in the technical receipt, and should be used to chain the two together
* MessageUuid: UUID of MeMo if available, otherwise empty
* MessageId: Read from MeMo if defined, otherwise empty
* ErrorCode: Error code (examples found under section "Back-end validation and error codes in distribution") in case of INVALID or NOT\_ALLOWED status, otherwise *null*
* ErrorMessage: Error message (examples found under section "Back-end validation and error codes in distribution") in case of INVALID or NOT\_ALLOWED status, otherwise *null*
* Timestamp: Now (UTC)
* ReceiptStatus: COMPLETED/NOT\_ALLOWED/INVALID

### SFTP receipt procedure

#### Technical receipts

Technical receipts for SFTP protocol are created on the SFTP server for each tar.lzma container that has been sent to DP via SFTP. The receipt is sent in XML format. The field values are:

* TransmissionId: The UUID part of the name of the uploaded .tar.lzma-file (SFTP .tar.lzma files should be named <UUID>.tar.lzma)
  + *03-01-2022 This feature is not currently implemented but is planned to be released before public go-live.*
* MessageUuid: Empty
* ErrorCode: *null* for RECEIVED status otherwise error code
* ErrorMessage: *null* for RECEIVED status otherwise error message
* Timestamp: Now (UTC)
* ReceiptStatus: RECEIVED/INVALID

#### Business receipts

A SFTP protocol business receipt is sent for each individual message when the message has been through a validation procedure. If the message fails validation a business receipt with the status NOT\_ALLOWED or INVALID is sent. The reason of a failed validation can be deducted from the ErrorCode and ErrorMessage of the returned receipt. If the message passes validation the receipt will have the status COMPLETED. A business receipt with status COMPLETED signifies that DP has accepted the message and now holds the responsibility to deliver the message to the recipient. The receipts are uploaded to the receipt folder of the sender system on the SFTP server with a body in the XML format. The field values are:

* TransmissionId: The UUID part of the name of the uploaded .tar.lzma file (SFTP .tar.lzma files should be named <UUID>.tar.lzma)
  + *03-01-2022 This feature is not currently implemented but is planned to be released before public go-live.*
* MessageUuid: UUID of MeMo if available, otherwise empty
* MessageId: Read from MeMo if defined, otherwise empty
* ErrorCode: *null* for COMPLETED status otherwise error code e.g. “recipient.is.exempt”
* ErrorMessage: *null* for COMPLETED status otherwise error message e.g. “Recipient with cvr <cvr\_number> is exempt”
* Timestamp: Now (UTC)
* ReceiptStatus: COMPLETED/NOT\_ALLOWED/INVALID

### JSON examples

#### Technical receipt examples

|  |
| --- |
| {  "transmissionId": "aa1c5cde-4f52-4ff4-b9c5-d737bf478544",  "timeStamp": "2020-07-02T08:23:07.026Z",  "receiptStatus": "RECEIVED"  } |

#### Business receipt examples

|  |
| --- |
| {  "transmissionId": "fb55c3ad-8e9c-4229-b781-3e603a4657b6",  "timeStamp": "2021-07-23T10:47:41.241Z",  "receiptStatus": "RECEIVED"  } |

|  |
| --- |
| {  "transmissionId": "54f1aa41-8be6-4510-9bb9-04c38085a384",  "messageUUID": "040dcd16-e9f3-4bed-a49f-2039187be467",  "messageId": "MSG-12340",  "errorCode": "recipient.is.exempt",  "errorMessage": "Recipient with cvr 11223344 is exempt",  "timeStamp": "2020-06-10T09:25:55.033Z",  "receiptStatus": "NOT\_ALLOWED"  } |

|  |
| --- |
| {  "transmissionId": "c18aca74-ac56-458a-96ae-4a1b63fe0391",  "messageUUID": "040dcd16-e9f3-4bed-a49f-2039187be467",  "messageId": "MSG-12341",  "errorCode": "recipient.not.found, sender.not.found",  "errorMessage": "Recipient with CVR 24586369 does not exist, Organisation with cvr 24545784 does not exist",  "timeStamp": "2020-06-10T09:07:01.012Z",  "receiptStatus": "INVALID"  } |

|  |
| --- |
| {  "transmissionId": "238179a2-b1fe-4504-b1b7-6be7856974d3",  "messageUUID": "9c2ea15d-61fb-4ba9-9366-42f8b194c262",  "messageId": **null**,  "errorCode": **null**,  "errorMessage": **null**,  "timeStamp": "2020-06-25T12:55:00.126Z",  "receiptStatus": "COMPLETED"  } |

|  |
| --- |
| {  "transmissionId": "65753850-a841-4288-9b74-46b7fb9ef2f6",  "errorCode": "archive.processing.failed",  "errorMessage": "An error occurred while processing the archive: Unable to detect compression format",  "timeStamp": "2021-07-23T10:45:56.640Z",  "receiptStatus": "INVALID"  } |

### Bulk MeMo SFTP Service

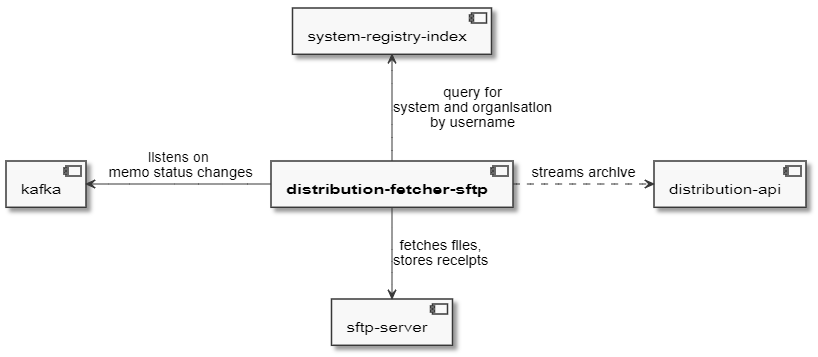
* [Component diagram](#scroll-bookmark-286)
* [Sequence diagram](#scroll-bookmark-287)
* [SFTP Server folder structure](#scroll-bookmark-288)
* [Receipt XML](#scroll-bookmark-289)

In order to support bulk memos upload, the usage of SFTP protocol has been proposed. In order to utilize it, a distribution-fetcher-sftp component has been implemented. It is responsible for connecting to the server, fetching the archives and streaming them into distribution-api, which is then responsible for its further processing.

It also uploads receipts to the SFTP server (technical and business ones).

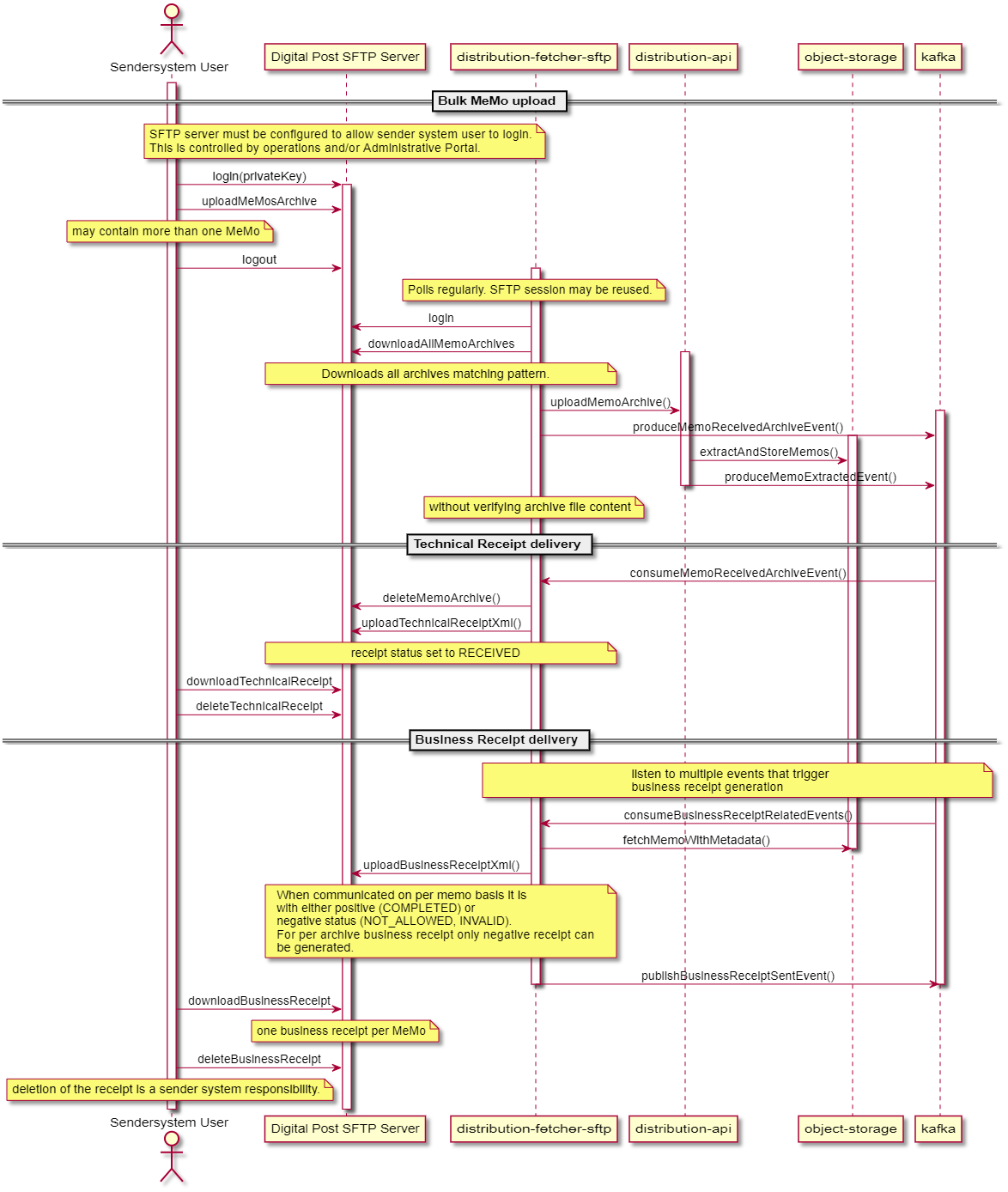
#### Component diagram

The below diagram shows the components that are a part of SFTP flow.



#### Sequence diagram

The below diagram shows the SFTP flow sequence.



#### SFTP Server folder structure

|  |  |  |  |
| --- | --- | --- | --- |
| **Root folder (per user)** | **Subfolders** | **User access rights** | **Description** |
| /{username}  Note: SFTP server uses chroot functionality to prevent access to filesystem thus real path is different (and is up to server admins) | . | **dir**: RX | Home folder. Contains subfolders defined below.  The username is the id of the systems identity id encoded in base62 to ensure that it is always under 32 characters. |
|  | memos/{bulkMemoTarLzmaUUID}.tar.lzma | **files:** RW  **dir:** WX | Contains bulk memo .tar.lzma files uploaded by sender system. File name must end with .tar.lzma and be unique identifier (UUID). Files that do not match this naming pattern are ignored.  Example file name:  91581881-23cb-40b0-9e50-9c4d500649e3.tar.lzma |
|  | memos/tmp/\*.\* | **files:** RW  **dir:** WX | Sender systems are required to use a pattern of **copy&move**, ie. files are to be uploaded to memos/tmp folder first and then moved to memos |
|  | receipts/ | **files:** R  **dir:** WX | Contains technical receipts and business receipts.  Technical receipt status is either RECEIVED (technical receipt) or negative (INVALID, NOT\_ALLOWED) - if tar.lzma file is corrupted or incorrect in any other way. The UUID of the technical receipt is taken from the uploaded tar.lzma file and is also the transmission ID of both the technical receipt and the business receipt.  Business receipts are generated by DP - one per MeMo |
|  | receipts/tmp/\*.\* | None | Temporary folder used by system user to write business receipts before they are moved to receipts folder |
|  | dp/receipt/ebcdic | **files:** R  **dir:** WX | Contains business receipts generated by DP - one per EBCDIC message. |
|  | dp/receipt/ebcdic/tmp/\*.\* | None | Temporary folder used by system user to write DP EBCDIC receipts before they are moved to dp/receipts/ebcdic folder |
|  | dp/receipts/xml/ | **files:** R  **dir:** WX | Contains business receipts generated by DP - one per DP/DP2 message |
|  | dp/receipts/xml/tmp/\*.\* | None | Temporary folder used by system user to write DP/DP2 receipts before they are moved to dp/receipts/xml folder |
|  | dp/xml/{EBOKS.DATA.\*.\*} | **files:** R  **dir:** WX | Contains BULK XML file and expects \*.KLAR\* file in same directory to signal that original file uploaded successfully |
|  | dp/xml/tmp/\*.\* | None | Temporary folder used by system user to write BULK XML before they are moved to dp/xml folder |
|  | dp/ebcdic/{EBOKS.DATA.\*.\*} | **files:** R  **dir:** WX | Contains EBCDIC file and expects \*.KLAR\* file in same directory to signal that original file uploaded successfully |
|  | dp/ebcdic/tmp/\*.\* | None | Temporary folder used by system user to write EBCDIC before they are moved to dp/ebcdic folder |

|  |
| --- |
| Users SHOULD not create additional folders themselves. |

#### Receipt XML

|  |
| --- |
| <?**xml** version='1.0' encoding='UTF-8'?>  <**receipt**>  <**transmissionId**>a5345a15-e5cc-4f6d-b2c7-97d0b036bfdf</**transmissionId**>  <**messageUUID**>e9f3bd3f-11d0-4af2-a72f-01327c5bcc96</**messageUUID**>  <**messageId**>MSG-12345</**messageId**>  <**errorCode**>(optional)123</**errorCode**>  <**errorMessage**>(optional) some error message</**errorMessage**>  <**timeStamp**>2020-06-05T12:00:00Z</**timeStamp**>  <**receiptStatus**>COMPLETED</**receiptStatus**>  </**receipt**> |

(Note: real xml will not be pretty printed, but rather a single line text file, in UTF-8 encoding)

### Distribution use case examples

#### Sending memo messages over REST PUSH

In DP a sender system can send a message (MeMo) using the Distribution API to CREATE/POST a MeMo to the DP system. The sender system is notified using technical and business receipts.

Sending a message from a sender system using REST PUSH:

POST /memos/

Status is CREATED, 201 http status. The response is shown below.

|  |
| --- |
| {  "transmissionId": "aa1c5cde-4f52-4ff4-b9c5-d737bf478544",  "timeStamp": "2020-07-02T08:23:07Z",  "receiptStatus": "RECEIVED"  } |

The response entails a technical receipt.

The receipt has a transmissionId to identify the transaction.

When the message have been validated by DP and is ready to either be delivered to a recipients mailbox or send to the recipients receiver system, a business receipt is generated by DP and send to the sender systems receipt endpoint.

An example of a business receipt is shown below:

|  |
| --- |
| {  "transmissionId": "238179a2-b1fe-4504-b1b7-6be7856974d3",  "messageUUID": "e60394cd-1ba9-4ff0-833b-9a05113b3df1",  "messageId": "MSG-12345",  "errorCode": **null**,  "errorMessage": **null**,  "timeStamp": "2020-06-25T12:55:00.262362",  "receiptStatus": "COMPLETED"  } |

The business receipt status is completed. In use cases where a negative receipt is sent, a message is shown to identify the problem. In both cases, DP expect the sender system to respond to the business receipt with either 200 OK or 202 ACCEPTED with an empty body.

#### Fetching a single memo over REST PULL

MeMos can both be fetched individually by the use of the ID for a specific message or a system can fetch all available MeMos for that given system.

In DP a REST\_PUBLISH\_SUBSCRIBE recipient system is notified when a MeMo message is available for it to fetch. However, a REST\_PULL recipient systems do not get this notification and are responsible themselves to fetch the currently available MeMos if any.

**Fetching a single MeMo for a REST\_PUBLISH\_SUBSCRIBE recipient system**

When a new MeMo is available, a MeMo notification is sent to a REST\_PUBLISH\_SUBSCRIBE recipient system endpoint, the expected response is 200 OK.

Recipient system service protocol:  
serviceProtocol: ServiceProtocolType.REST\_PUBLISH\_SUBSCRIBE

Recipient system endpoint example:  
https://host:8204/memos/

The MeMo notification contains an URL that points to the location of the new MeMo.

##### (url) path example:

https://digital\_post\_host:port/memos/e60394cd-1ba9-4ff0-833b-9a05113b3df1

The path includes the ID for that specific message.

The recipient system then makes a

GET /memos/e60394cd-1ba9-4ff0-833b-9a05113b3df1

which fetches the MeMo.

#### Fetching a list of MeMos for a REST\_PULL recipient system

A REST\_PULL recipient system is responsible itself to fetch the currently available MeMos. A list of all available MeMos will be returned.

The recipient system has to make a

GET /memos/

and a list is returned.

#### Deleting a MeMo for a REST\_PULL recipient system

A REST\_PULL recipient system can send a business receipt containing a MeMo ID of the message that it has fetched.

POST /memos/{memo id}/receipt

The MeMo with the given ID will be found in the list of available MeMos for the system. The MeMo will firstly be deleted off of the list containing available MeMos, and afterwards from object storage. The MeMo can no longer be fetched.

It is only possible to send a business receipt for a single MeMo at a time.

### Fetching business receipts

REST\_PULL sender systems are responsible for fetching available receipts themselves. A list with UUID's of available receipts for a given system can be fetched, after which the receipts can be fetched individually by the use of the UUID’s. The receipts will be deleted after fetched individually, unless otherwise is specified, or deleted by a schedule cleaner that will delete receipt older than 7 days.

### Fetching receipts for a REST\_PULL sender system

A REST\_PULL sender system is responsible for fetching the currently available receipts.

#### Fetching a list of available receipts

The sender system has to make a GET as such:

GET /receipts/

A list of all available receipts UUID will be returned.  
Example of result:

|  |
| --- |
| {  "content": [  "966925f3-569a-4d9a-b688-f49eac9e2c7b",  "888e528f-1ef0-44ad-ab57-910344cf2003",  "e93ab749-1b58-4f40-808c-77327ced20bf",  "c8e94729-b2b9-49bc-99f0-4e7f9863fde0",  "759e3921-36a7-4353-971e-b0edd93e35ab",  "c46d1f6a-8947-4758-bb5f-51e356d58a1d"  ],  "number": 0,  "size": 20,  "totalElements": 6,  "totalPages": 1  } |

And a list of UUID’s for the individually is returned, which can be used to fetch a single receipt.

Search parameter can be use in order to go through multiple pages. Moreover the size per page can also be change by the usage of search parameter. As standard 20 receipts are returned per page.

Example:

/receipts/?size=100

#### Fetching a single receipt with automatic deletion GET /receipts/{receiptId} Example of request:

|  |
| --- |
| GET /receipts/966925f3-569a-4d9a-b688-f49eac9e2c7b |

Result when using request above:

|  |
| --- |
| <Receipt>  <transmissionId>3fa532b9-2b94-4fc3-b979-5d6c3bbf2a3e</transmissionId>  <messageUUID>c2ec7c7a-f197-4ade-887e-6ce0d3a9267c</messageUUID>  <timeStamp>2021-07-15T12:08:38.715Z</timeStamp>  <receiptStatus>COMPLETED</receiptStatus>  </Receipt> |

Please notice that like in the ‘Fetching Memo’ flow, the result is in xml format and that the receipt will be deleted after it have be returned.

#### Fetching a single receipt without automatic deletion

If automatic deletion of a receipt after fetching is not desired, adding the “delete=” queryparam allows one to specify whether the receipt should be deleted after fetching:

#### GET /receipts/{receiptId}/?delete=false

True can also be specified, but then in effect it works the same was as the regular **GET** /receipts/{receiptId}

#### Deleting a single receipt without fetching

It is also possible to delete a single receipt without fetching it.

#### DELETE /receipts/{receiptId}

### Bulk-fetching receipts

It is also possible to fetch a list of receipts for the sender system by using **GET** /receipts-bulk/. The only supported query parameters are *size* and *page* for paging. After receipts are fetched, each receipt should be deleted individually.

#### Example request

*GET* /receipts-bulk/?size=2&page=3

gives a response like

|  |
| --- |
| {  "currentPage": 3,  "totalPages": 1004,  "elementsOnPage": 2,  "totalElements": 2007,  "receipts": [  {  "transmissionId": "0206bdc8-a254-4b33-8eac-b847afc6afa6",  "messageUUID": "46f39e7f-b5f4-4bf3-99ba-7651c4c7f5a1",  "errorCode": "message.uuid.not.unique",  "errorMessage": "The MessageUUID 46f39e7f-b5f4-4bf3-99ba-7651c4c7f5a1 is invalid. MessageUUID must be a unique UUID",  "timeStamp": "2022-12-06T07:59:13.554Z",  "receiptStatus": "INVALID"  },  {  "transmissionId": "fdb52007-f5e2-430c-ac92-3c9f95e25ff3",  "messageUUID": "46f39e7f-b5f4-4bf3-99ba-7651c4c7f5a1",  "errorCode": "message.uuid.not.unique",  "errorMessage": "The MessageUUID 46f39e7f-b5f4-4bf3-99ba-7651c4c7f5a1 is invalid. MessageUUID must be a unique UUID",  "timeStamp": "2022-12-06T07:59:14.377Z",  "receiptStatus": "INVALID"  }  ]  } |

### Sending business receipts as a recipient system

When DP sends a MeMo through REST PUSH to a recipient system a Business Receipt is expected. A Business Receipt confirms that the recipient system has successfully been able to handle the received MeMo and that the MeMo will not need to be resent. The format of the business receipt is described in the section **“DP Receipt domain model“**. If DP does not receive a successful Business Receipt when a MeMo has been sent to a recipient system, DP will resend the MeMo according to the flow described in the section **“Flow for resending messages”**.

When a recipient system responds to DP the value of the field *receiptStatus* for Business Receipts does not matter to DP. DP evaluates a Business receipt as successful when the field *errorMessage* is empty (and the field *errorCode* is not "virus.detected").

If the errorCode is virus.detected the message will not be resent. Any other negative receipts will be logged in the event-log, but will not otherwise impact the re-sending of messages.

## Flow for resending messages

The following section describes the flow for resending MeMo's to SMTP and REST recipient systems. Technical- and business receipts or lack thereof triggers the resending flow. These receipts are exchanged between Digital Post and recipient systems to ensure correct delivery and integrity of messages. The flow is terminated as soon as the recipient system returns a positive business receipt or after 7 days.

When the flow is terminated by 7 days passing without a positive business receipt, the memo is redirected, and a mail is sent to the email address contacts registered on the system notifying of the new destination. The redirections is as follows:

* If the failure was to a default recipient system, the memo is saved in the mailbox for the organisation, which can be accessed through <http://virk.dk>.
* If the failure was to a non-default recipient system, the memo is sent to the default recipient system. If it fails for two days to this system, the message is saved in the mailbox.

**Technical receipts**

(Only for REST\_PUSH systems): The following flow is triggered if there has not been received a technical receipt with "http 200/201/202":

1. The message is resent after **10 minutes**
2. The message is resent after **8 hours**
3. The message is resent after **16 hours**
4. The message is resent after **24 hours\***
5. The message is resent every **24 hours for up to 6 days\***

**Business receipts**

If a message is send from Digital Post to a recipient system and a positive technical receipt is returned, Digital Post will await a business receipt from that recipient system. If one is not returned without an error message, the following flow is triggered (excl. returned receipts with error code "virus.detected")

1. The message is resent after **8 hours**
2. The message is resent after **16 hours**
3. The message is resent after **24 hours\***
4. The message is resent every **24 hours for up to 6 days\***

When using **SMTP** technical receipts are not returned by the recipient system, thus the resending flow is only initiated by a lack of returned business receipt (virus.detected excluded) or if the message softbounces. If the message is hardbounced by the recipient system, the resending flow is not initiated.

1. The message is resent after **8 hours**
2. The message is resent after **16 hours**
3. The message is resent after **24 hours**
4. The message is resent every **24 hours for up to 6 days**

**Retrying to default recipient system**

After the full technical or business retry flow to a recipient system is completed without a positive business receipt, the message is attempted delivered to the default recipient system, then again after **10 minutes** and **25 hours**. If no delivery can be made to the default recipient system, the message is delivered to the default mailbox, which can be accessed through <http://virk.dk> or commercial view clients.

**System deactivation**

If messages to a system fail, the email contacts of the systems will be notified before the message is redirected at day 7. After a message starts failing (no positive business receipt is returned), a notification mail will be sent on day 2 and day 5 after the message was sent for the first time, assuming it is still failing.

The mail will notify that something is failing, which may lead to system deactivation, and will give some info on what system fails and what messages are failing.

To avoid spamming the contacts, the mails are sent based on the earliest failure - i.e. if a memo on day 0 and day 1 are failing, mails are sent on day 2 and 5 for the first message, but not on day 3 and day 6.

When a memo is redirected on day 7, the DP system will check if the recipient system has any message within the last 7 days where it has received both a positive technical receipt and a positive business receipt for any message. If this is not the case, the system will be deemed systematically failing, and will be deactivated so future messages do not get delayed for 7 days of retry before the recipient receives them.  
This means that if only a few messages of many fail (e.g. due to an edge case bug in the recipient system), the recipient system will not be deactivated, and only the failing messages are redirected.

**Handling of error code** virus.detected

If a negative receipt is returned with the error "virus.detected=Virus fundet I modtaget payload for MeMo med id {0}" the flow is not initiated and the message is handled manually by the central administration.

**Retries cleanup**

In order to clean up the database used by distribution-retry-store, a scheduled job is run daily, which is responsible for removing retries older than a configurable number of days (two weeks by default).

## Flow for resending business receipts - REST Push protocol

When sending MeMos from a sender system with REST\_PUSH protocol, Digital Post will retry sending the business receipt if the business receipts fails to deliver to the sender systems provided endpoint.

Digital Post will retry sending the business receipt every 6 hours for 5 days.

If Digital Post is ultimately unable to deliver the business receipt, it will not be lost. The validation status of the message can be found via the event-log. The sender system can do a lookup in the event log to find the validation status.

## HTML whitelist for document validation

### Introduction

Before messages are delivered, both from sender systems and replying through the mailbox. The HTML of the message is validated to ensure that it can be rendered and does not contain any malicious content. Therefore Digital Post implements two different policies for HTML validation:

* Strict
  + A policy for end users that are writing or attaching HTML to a Message from a view client
* Lenient
  + Used when validating incoming MeMos before distribution
  + The lenient white list is a super set of the strict one

### Strict

* No comments allowed at all
* Only inline styling on elements

|  |  |
| --- | --- |
| **Elements** | **Attributes**  Inline styling is allowed on all elements that support styling |
| **Global**  All elements |  |
| *Elements where attribute is relevant* | * role * title * aria-hidden * aria-label * aria-level * aria-orientation * aria-placeholder * aria-sort * aria-relevant * aria-activedescendant * aria-colcount * aria-colindex * aria-colspan * aria-describedby * aria-details * aria-labelledby * aria-posinset * aria-rowcount * aria-rowindex * aria-rowspan |
| **Main**  Main blocks such as <html>, <body>, etc. |  |
| html | * xmlns * lang |
| head |  |
| meta | * charset * content * name * http-equiv   + content-security-policy   + content-type |
| title |  |
| body | * lang |
| **Semantic** Html 5 semantic elements |  |
| address |  |
| article |  |
| aside |  |
| details |  |
| figcaption |  |
| figure |  |
| footer |  |
| header |  |
| main |  |
| mark |  |
| nav |  |
| section |  |
| summary |  |
| time |  |
| **Blocks**  Allow common block elements including <p>, <h1>, etc. |  |
| p |  |
| div |  |
| h1 |  |
| h2 |  |
| h3 |  |
| h4 |  |
| h5 |  |
| h6 |  |
| hr |  |
| ul |  |
| ol |  |
| li |  |
| blockquote |  |
| **Formatting**  Allows common formatting elements including <b>, <i>, etc. |  |
| b |  |
| i |  |
| font | color, face, size |
| s |  |
| u |  |
| o |  |
| sup |  |
| sub |  |
| ins |  |
| del |  |
| strong |  |
| strike |  |
| tt |  |
| code |  |
| big |  |
| small |  |
| br |  |
| span |  |
| em |  |
| **Tables**  Allow common table elements |  |
| table | * summary * align * valign |
| tr | * align * valign |
| td | * align * valign |
| th | * align * valign |
| colgroup | * align * valign |
| caption |  |
| col | * align * valign |
| thead | * align * valign |
| tbody | * align * valign |
| tfoot | * align * valign |
| **Links** |  |
| a | * href   + https, mailto * target   + \_blank |
| **Images**  Allow <img> elements from from embedded sources only |  |
| img | * alt * src   + data:image * border * height * width |
| **Styles**  Allow certain safe CSS properties in style="..." attributes. <style> element is not allowed | **Properties** |
| * -moz-border-radius * -moz-border-radius-bottomleft * -moz-border-radius-bottomright * -moz-border-radius-topleft * -moz-border-radius-topright * -moz-box-shadow * -moz-outline * -moz-outline-color * -moz-outline-style * -moz-outline-width * -o-text-overflow * -webkit-border-bottom-left-radius * -webkit-border-bottom-right-radius * -webkit-border-radius * -webkit-border-radius-bottom-left * -webkit-border-radius-bottom-right * -webkit-border-radius-top-left * -webkit-border-radius-top-right * -webkit-border-top-left-radius * -webkit-border-top-right-radius * -webkit-box-shadow * azimuth * background * background-attachment * background-color * background-image * background-position * background-repeat * border * border-bottom * border-bottom-color * border-bottom-left-radius * border-bottom-right-radius * border-bottom-style * border-bottom-width * border-collapse * border-color * border-left * border-left-color * border-left-style * border-left-width * border-radius * border-right * border-right-color * border-right-style * border-right-width * border-spacing * border-style * border-top * border-top-color * border-top-left-radius * border-top-right-radius * border-top-style * border-top-width * border-width * box-shadow * caption-side * color * cue * cue-after * cue-before * direction * elevation * empty-cells * font * font-family * font-size * font-stretch * font-style * font-variant * font-weight * height * image() * letter-spacing * line-height * linear-gradient() * list-style * list-style-image * list-style-position * list-style-type * margin * margin-bottom * margin-left * margin-right * margin-top * max-height * max-width * min-height * min-width * outline * outline-color * outline-style * outline-width * padding * padding-bottom * padding-left * padding-right * padding-top * pause * pause-after * pause-before * pitch * pitch-range * quotes * radial-gradient() * rect() * repeating-linear-gradient() * repeating-radial-gradient() * rgb() * rgba() * richness * speak * speak-header * speak-numeral * speak-punctuation * speech-rate * stress * table-layout * text-align * text-decoration * text-indent * text-overflow * text-shadow * text-transform * text-wrap * unicode-bidi * vertical-align * voice-family * volume * white-space * width * word-spacing * word-wrap | * url   + data uri only * -moz-inline-box, -moz-inline-stack, -moz-pre-wrap, -o-pre-wrap, -pre-wrap, 100, 200, 300, 400, 500, 600, 700, 800, 900, above, absolute, aliceblue, all-scroll, always, antiquewhite, aqua, aquamarine, armenian, at, auto, avoid, azure, baseline, behind, beige, below, bidi-override, bisque, black, blanchedalmond, blink, block, blue, blueviolet, bold, bolder, border-box, both, bottom, break-word, brown, burlywood, cadetblue, capitalize, caption, center, center-left, center-right, chartreuse, child, chocolate, circle, cjk-decimal, clip, closest-corner, closest-side, code, col-resize, collapse, condensed, contain, content-box, continuous, coral, cornflowerblue, cornsilk, cover, crimson, crosshair, cursive, cyan, darkblue, darkcyan, darkgoldenrod, darkgray, darkgreen, darkkhaki, darkmagenta, darkolivegreen, darkorange, darkorchid, darkred, darksalmon, darkseagreen, darkslateblue, darkslategray, darkturquoise, darkviolet, dashed, decimal, decimal-leading-zero, deeppink, deepskyblue, default, digits, dimgray, disc, disclosure-closed, disclosure-open, dodgerblue, dotted, double, e-resize, ellipse, ellipsis, embed, ethiopic-numeric, expanded, extra-condensed, extra-expanded, fantasy, far-left, far-right, farthest-corner, farthest-side, fast, faster, female, firebrick, fixed, floralwhite, forestgreen, fuchsia, gainsboro, georgian, ghostwhite, gold, goldenrod, gray, green, greenyellow, groove, hand, hebrew, help, hidden, hide, high, higher, hiragana, hiragana-iroha, honeydew, hotpink, icon, indianred, indigo, inherit, inline, inline-block, inline-table, inset, inside, invert, italic, ivory, japanese-formal, japanese-informal, justify, katakana, katakana-iroha, khaki, korean-hangul-formal, korean-hanja-formal, korean-hanja-informal, large, larger, lavender, lavenderblush, lawngreen, left, left-side, leftwards, lemonchiffon, level, lightblue, lightcoral, lightcyan, lighter, lightgoldenrodyellow, lightgreen, lightgrey, lightpink, lightsalmon, lightseagreen, lightskyblue, lightslategray, lightsteelblue, lightyellow, lime, limegreen, line-through, linen, list-item, local, loud, low, lower, lower-alpha, lower-greek, lower-latin, lower-roman, lowercase, ltr, magenta, male, maroon, medium, mediumaquamarine, mediumblue, mediumorchid, mediumpurple, mediumseagreen, mediumslateblue, mediumspringgreen, mediumturquoise, mediumvioletred, menu, message-box, middle, midnightblue, mintcream, mistyrose, mix, moccasin, monospace, move, n-resize, narrower, navajowhite, navy, ne-resize, no-content, no-display, no-drop, no-repeat, none, normal, not-allowed, nowrap, nw-resize, oblique, oldlace, olive, olivedrab, once, orange, orangered, orchid, outset, outside, overline, padding-box, palegoldenrod, palegreen, paleturquoise, palevioletred, papayawhip, peachpuff, peru, pink, plum, pointer, powderblue, pre, pre-line, pre-wrap, progress, purple, red, relative, repeat, repeat-x, repeat-y, ridge, right, right-side, rightwards, rosybrown, round, row-resize, royalblue, rtl, run-in, s-resize, saddlebrown, salmon, sandybrown, sans-serif, scroll, se-resize, seagreen, seashell, semi-condensed, semi-expanded, separate, serif, show, sienna, silent, silver, simp-chinese-formal, simp-chinese-informal, skyblue, slateblue, slategray, slow, slower, small, small-caps, small-caption, smaller, snow, soft, solid, space, spell-out, springgreen, square, static, status-bar, steelblue, sub, super, suppress, sw-resize, table, table-caption, table-cell, table-column, table-column-group, table-footer-group, table-header-group, table-row, table-row-group, tan, teal, text, text-bottom, text-top, thick, thin, thistle, to, tomato, top, trad-chinese-formal, trad-chinese-informal, transparent, turquoise, ultra-condensed, ultra-expanded, underline, unrestricted, upper-alpha, upper-latin, upper-roman, uppercase, vertical-text, violet, visible, w-resize, wait, wheat, white, whitesmoke, wider, x-fast, x-high, x-large, x-loud, x-low, x-slow, x-small, x-soft, xx-large, xx-small, yellow, yellowgreen |

### Lenient

All allowed in the Strict policy are allowed here, plus the following elements:

* Comments allowed
* Global styling element allowed with no restrictions other than URLs referencing http/https are blocked
* Element style attribute with no restrictions other than URLs referencing http/https are blocked
* Attributes id, class allowed on all supported elements

|  |  |
| --- | --- |
| **Elements** | **Attributes** |
| **Main**  Further options on elements such as <html>, <body>, etc. |  |
| html | * xmlns:v * xmlns:o * xmlns:w * xmlns:m |
| body | * link * vlink |
| **Blocks**  Further options on block elements |  |
| span | * lang |
| o:p |  |
| p | * align |
| div | * align |
| hr | * size * width * align |
| picture |  |
| source | * srcset   + data:image * src   + data:image * media * type |
| pre |  |
| cite |  |
| ol | * type * start |
| ul | * type |
| **Links** |  |
| a | * name |
| **Tables** |  |
| table | * border * cellspacing * cellpadding * width |
| td | * scope * headers * colspan * width * rowspan * nowrap * height |
| th | * scope * headers * colspan * width * rowspan * nowrap * height |
| colgroup | * width |
| col | * width * height * span |
| **Styles** |  |
| style | Global style tag allowed.  Local style attribute unrestricted.  Only URLs to external resources are blocked. |

### Testing

HTML content can be tested against the white list validator using endpoint:

* /validations/
  + Looks like this on the test environment: curl --location --request POST 'https://api.test.digitalpost.dk/apis/v1/validations/' --header 'Content-Type: text/html'

POST request with content-type: text/html and request body with HTML, will return 200 OK with a response body containing code html.validator.approved, if it finds no validation errors in the HTML:

|  |
| --- |
| {  "code": "html.validator.approved",  "message": "Approved: Html validation using NgDP whitelist - LENIENT policy found 0 errors.",  "fieldErrors": []  } |

If it finds validation errors, it returns 400 BAD REQUEST including a response body with code html.validator.rejected, and a list of errors:

|  |
| --- |
| {  "code": "html.validator.rejected",  "message": "Rejected: HTML validation using NgDP whitelist - LENIENT policy - found 1 errors.",  "fieldErrors": [  {  "resource": "errorMessage",  "code": "html.validator.rejected.element",  "message": "Filen test indeholder element \"link\", som enten ikke tilladt eller som indeholder data, der ikke er tilladt."  }  ]  } |

#### Policy

The default policy used is LENIENT. The policy can be switched using request parameter policy:

* /validations/?policy=STRICT
  + Looks like this on the test environment: curl --location --request POST 'https://api.test.digitalpost.dk/apis/v1/validations/?policy=STRICT' --header 'Content-Type: text/html'

# Access request registry

The following services are exposed from the access-request.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Service** | **URL** | **Data returned** | **Usage** | **Required roles** |
| Query access requests | GET /access-requests/ | Paged list of AccessRequest | Fetching all access-requests user has access to with optional paging, sorting and filtering. | * Employee * Citizen * System manager |
| Create access request | POST /access-requests/ | AccessRequest | Creating access request | * Employee * Citizen * System manager |
| Update access request | PUT /access-requests/{id} | AccessRequest | Updating access request | * Employee * Citizen * System manager |
| Delete access request | DELETE /access-requests/{id} | No content | Deleting access request | * Employee * Citizen * System manager |
| Fetch access request | GET /access-requests/{id} | AccessRequest | Fetching single access request | * Employee * Citizen * System manager |
| Update documentation content | PUT  /access-requests/{accessRequestId}/documentations/{id}/content | AccessRequest | Update the contents of a documentation | * Employee * Citizen * System manager |
| Fetch documentation content | GET  /access-requests/{accessRequestId}/documentations/{id}/content | Raw bytes | Fetching documentation content bytes | * Employee * Citizen * System manager |
| Activate access request target | POST /access-requests/target/activate | AccessRequest | Activating a target using activation code | * Employee |
| Fetch organisation lookup | GET /organisation-lookup/{cvrNumber} | OrganisationLookup | Fetching information about the organisation, such as name status active curator, liquidator and executor |  |
| Citizen lookup | GET /citizen-lookup/  Parameters:   * cprNumber * firstNameProvided * lastNameProvided | CitizenLookupResult | Fetching information about a citizen: identityId, full name,, and if citizen’s status is CLOSED |  |

## Access request registry - introduction

* [Purpose of the registry](#scroll-bookmark-318)
* [Privilege requests](#scroll-bookmark-319)
* [Delegation requests](#scroll-bookmark-320)
* [Appointed delegation requests](#scroll-bookmark-321)
* [Connection agreement requests](#scroll-bookmark-322)
* [Terms approval requests](#scroll-bookmark-323)
* [User administrator’s statement of truth privilege requests](#scroll-bookmark-324)
* [Lost user administrator privilege requests](#scroll-bookmark-325)
* [Special privilege requests](#scroll-bookmark-326)
* [Delegated support admin privilege request](#scroll-bookmark-327)
* [Concepts](#scroll-bookmark-328)

## Purpose of the registry

The access request registry is the request management store of DP. It stores access requests and exposes services to handle these. Eight different types of access requests are currently supported and they are all represented as one AccessRequest resource but marked as one of these types:

* Privilege requests
* Delegation requests
* Appointed delegation requests
* Connection agreement requests
* Terms approval requests
* User administrator’s statement of truth privilege requests
* Lost user administrator privilege requests
* Special privilege requests
* Legal owner of inactive or closed company privilege request
* Delegated support admin privilege request

## Privilege requests

A privilege request is filed by either a citizen or an organisation requesting a privilege to a mailbox or an organisation. In certain cases a request can also be triggered by an event elsewhere in DP. A privilege request consists of information about the target of the request and the requested privilege. To assist in the processing, documents may be attached to the request. Examples of scenarios:

* A new employee requesting a privilege (or membership of user group) to own organisation
* A citizen or organisation requesting a ‘executor of estate’ privilege to a deceased citizen's mailbox
* A citizen or organisation requesting party representative privileges to another person's or citizen's mailbox
* A system manager the request on behalf of a citizen or organisation

A privilege request will upon submission end up on a list of incoming requests presented to:

* The citizen owning the mailbox to which access is requested
* The user administrator of the organisation to which access is requested
* A system manager handling the special cases like for instance curators an liquidators etc.

The access can be granted or rejected and the requesting party will be notified of the decision. If access is granted the privileges will be created in the identity registry. If the target of the privilege is unknown to the system an email with activation code and instructions will be sent.

### Journalized: False

## Delegation requests

Delegation requests are filed by a citizen or an organisation granting one or more privileges to their own mailbox or organisation to another employee or citizen - delegating, so to speak, the privileges. A user group may be used instead of or on combination with privileges. Examples of scenarios:

* A citizen delegating legal representative privileges to another citizen, to an organisation or to a specific employee of that organisation
* A citizen “trusting“ another citizen so messages can be forwarded to that citizen
* A user administrator of an organisation delegating a privilege to a new employee of her own organisation
* A user administrator of an organisation delegating a read only access to another organisation or to a specific employee of that organisation

These requests are automatically approved and privileges assigned immediately if target is known, if not an activation code sent instead.

### Journalized: False

## Appointed delegation requests

The same principle as regular delegation requests, but here a user administrator can delegate privileges to the organisations employees, that give access to another organisation or a citizen, if those privileges have been appointed to the user administrator’s organisation.

### Journalized: False

## Connection agreement requests

Request filed by an authority to request access to DP. It requires a physical signature on a document, and manual approval. Upon approval the organisation changes type in the system registry (from COMPANY to AUTHORITY) and is then ready to approve the terms.

### Journalized: True

## Terms approval requests

Request filed by an authority to request access to DP. It is automatically approved if organisation is correctly registered and an activation code sent to the user administrator organisation by email.

### Journalized: False

## User administrator’s statement of truth privilege requests

Uses registration type code from system registry to categorize the requester. If within correct category, the request is automatically approved and NPTE privileges are granted to the requester.

### Journalized: True

## Lost user administrator privilege requests

Requested by Erhvervsstyrelsen on behalf of an organisation with no access to their user administrator. Automatically approved.

### Journalized: True

## Special privilege requests

Like a regular privilege request in it’s structure, but only regarding the special privileges:

* Curator
* Executor of estate
* Liquidator

The access can be granted or rejected, by system managers and access request registry administrators and the requesting party will be notified of the decision. If access is granted the privileges will be created in the identity registry.

### Journalized: True

## Delegated support admin privilege request

A delegated support admin privilege request is filed by a DSS support employee requesting privilege to a company or an authority which requests a support assistance. The request is auto-approved and results in creation of 2 privileges in the identity registry:

* An appointed privilege a grantee of which is the company Erhvervsstyrelsen (ERST)
* A privilege with DSS support employee as grantee and a privilege appointed to ERST as a parent privilege

A scope of the both privileges is a company or authority requesting support assistance and their expiration is set to 30 minutes. The privileges can be revoked earlier when the support employee revokes the privileges manually.

A support employee can have only one open delegated support admin privilege request, so if a support employee requests another one, the currently open one will be revoked.

### Journalized: False

## Concepts

A request is a request regarding either privileges or user groups. User groups are only allowed on delegation request. A request can consist of multiple privileges or groups. A request consists of 3 participants:

* requester
  + The participant initiating the request. When delegating, typically a user administrator or a citizen. When requesting privileges typically an employee or a citizen.
  + Although a system manager may create a request on behalf of an organisation, it is still the organisation or citizen that goes her, a they are the actual requester.
* accessTo
  + The scope of the request, which is an organisation or a citizen. If granted the target will get some sort of access to this participant.
* target
  + Who is the recipient of the privileges

In many cases two of these will be the exact same participant, but the registry insists on a complete registration of all three participants. This provides the consistency needed when dealing with a rather flexible API, such as this.

See [Access request registry - commom use case examples](https://digidp.atlassian.net/wiki/spaces/DP/pages/809074849/Access+request+registry+-+commom+use+case+examples) for examples.

## Access request registry - common use case examples

* [Creating and working with drafts](https://digidp.atlassian.net/wiki/spaces/DP/pages/819494917/Creating+and+working+with+drafts)
* [Attaching documents to request](https://digidp.atlassian.net/wiki/spaces/DP/pages/819462167/Attaching+documents+to+request)
* [Citizen granting (delegating) access to mailbox](https://digidp.atlassian.net/wiki/spaces/DP/pages/809533760/Citizen+granting+%28delegating%29+access+to+mailbox)
* [Citizen requesting access to other citizens mailbox](https://digidp.atlassian.net/wiki/spaces/DP/pages/882835838/Citizen+requesting+access+to+other+citizens+mailbox)
* [Lookup relevant information about citizens and companies](https://digidp.atlassian.net/wiki/spaces/DP/pages/1164312622/Lookup+relevant+information+about+citizens+and+companies)

## Introduction

All request can be created in state DRAFT. In DRAFT state the requests can be worked on and refined until the user deems the request ready for submission. A request will still be validated in DRAFT state and name resolving etc. will also occur.

|  |
| --- |
| DRAFT state can be skipped and a request can be submitted directly in state SUBMITTED, but DRAFT state is required if you need to attach documents to the requests as content cannot be added/changed after submission. |

A request in state DRAFT can be read, edited, and deleted following these rules:

|  |  |
| --- | --- |
| **Request created by** | **Users who can read, edit, delete the request** |
| Citizen | Only the exact same citizen |
| User administrator | User administrators of same organisation |
| Employee | User administrators of same organisation |
| System manager and “Special“ request | All system managers |

## User administrator delegates privilege to employee identified with an e-mail address

### Request

|  |
| --- |
| {  "requestType": "DELEGATION\_REQUEST",  "requestState": "DRAFT",  "requester": {  "externalId": "30714024",  "externalIdType": "CVR"  },  "accessTo": {  "externalId": "30714024",  "externalIdType": "CVR"  },  "privileges": [  "MESSAGE\_EMPLOYEE"  ],  "userGroups": [],  "target": {  "requestParticipant": {  "externalId": "30714024",  "externalIdType": "CVR",  "emailAddress": "clan@netcompany.com"  }  }  } |

### Response

201 CREATED

Notice how external ids have been accompanied by the identity ids, and the user performing the request have been inserted into employeeIdentityId.

|  |
| --- |
| {  "id": "1ba3a286-bb03-43ac-b874-e21a5bdee636",  "version": 0,  "transactionId": "F6DAVnsfzRF0HbqUJ67yJMsUAtV15MoG",  "createdDateTime": "2021-07-18T14:23:47.805Z",  "lastUpdated": "2021-07-18T14:23:47.805Z",  "requestType": "DELEGATION\_REQUEST",  "requestState": "DRAFT",  "requester": {  "id": "0ddd0ad1-cc43-45c3-bb9c-2dacd574a4e9",  "version": 0,  "createdDateTime": "2021-07-18T14:23:47.813Z",  "lastUpdated": "2021-07-18T14:23:47.813Z",  "externalId": "30714024",  "externalIdType": "CVR",  "organisationIdentityId": "6337c256-133a-46bb-b85c-d65eb751b37f",  "employeeIdentityId": "c810f086-5331-4336-bb4b-47dbdc27f3d7"  },  "accessTo": {  "id": "b45b6e83-5086-469f-87f0-dff6875de338",  "version": 0,  "createdDateTime": "2021-07-18T14:23:47.807Z",  "lastUpdated": "2021-07-18T14:23:47.807Z",  "externalId": "30714024",  "externalIdType": "CVR",  "organisationIdentityId": "6337c256-133a-46bb-b85c-d65eb751b37f"  },  "privileges": [  "MESSAGE\_EMPLOYEE"  ],  "userGroups": [],  "createdByIdentityId": "c810f086-5331-4336-bb4b-47dbdc27f3d7",  "createdOnBehalfOf": **false**,  "target": {  "id": "8330cbfe-158d-47f5-9250-d6f47eefbc59",  "version": 0,  "createdDateTime": "2021-07-18T14:23:47.813Z",  "lastUpdated": "2021-07-18T14:23:47.813Z",  "requestParticipant": {  "id": "919a73d9-b27d-4f4d-b309-ed40bf4f9b13",  "version": 0,  "createdDateTime": "2021-07-18T14:23:47.814Z",  "lastUpdated": "2021-07-18T14:23:47.814Z",  "externalId": "30714024",  "externalIdType": "CVR",  "organisationIdentityId": "6337c256-133a-46bb-b85c-d65eb751b37f",  "emailAddress": "clan@netcompany.com"  }  },  "documentations": [],  "processedAutomatically": **false**  } |

## Usage of generic identity id in access requests

A generic identity id is a part of the requestParticipant in the access request. By using the generic identity the same flow is followed as when one of the identity type specific IDs is used (citizenIdentityId, employeeIdentityId or OrganisationIdentityId), however the solution will determine the type of the identity id. Furthermore, it will provide af field called identityTypeResolved in the response. When the identity type is determine from the generic identity id one or several of the identity type specific IDs will in the response alongside with the generic identity id.

### Request

|  |
| --- |
| {  "requestType": "DELEGATION\_REQUEST",  "requestState": "DRAFT",  "requester": {  "identityId": "7f79954b-ded0-4af3-8a56-ca69791e4685"  },  "accessTo": {  "identityId": "462829eb-26c3-48a5-8020-10166a15c976"  },  "privileges": [  "MESSAGE\_EMPLOYEE"  ],  "userGroups": [],  "target": {  "requestParticipant": {  "externalId": "30714024",  "externalIdType": "CVR",  "emailAddress": "clan@netcompany.com"  }  }  } |

### Response

|  |
| --- |
| {  "id": "1ba3a286-bb03-43ac-b874-e21a5bdee636",  "version": 0,  "transactionId": "F6DAVnsfzRF0HbqUJ67yJMsUAtV15MoG",  "createdDateTime": "2021-07-18T14:23:47.805Z",  "lastUpdated": "2021-07-18T14:23:47.805Z",  "requestType": "DELEGATION\_REQUEST",  "requestState": "DRAFT",  "requester": {  "id": "0ddd0ad1-cc43-45c3-bb9c-2dacd574a4e9",  "version": 0,  "createdDateTime": "2021-07-18T14:23:47.813Z",  "lastUpdated": "2021-07-18T14:23:47.813Z",  "externalId": "30714024",  "externalIdType": "CVR",  "organisationIdentityId": "6337c256-133a-46bb-b85c-d65eb751b37f",  "employeeIdentityId": "7f79954b-ded0-4af3-8a56-ca69791e4685",  "identityId": "7f79954b-ded0-4af3-8a56-ca69791e4685",  "identityTypeResolved": "EMPLOYEE"  },  "accessTo": {  "id": "b45b6e83-5086-469f-87f0-dff6875de338",  "version": 0,  "createdDateTime": "2021-07-18T14:23:47.807Z",  "lastUpdated": "2021-07-18T14:23:47.807Z",  "externalId": "2412001010",  "externalIdType": "CPR",  "citizenIdentityId": "462829eb-26c3-48a5-8020-10166a15c976"  "identityId": "462829eb-26c3-48a5-8020-10166a15c976",  "identityTypeResolved": "CITIZEN"  },  "privileges": [  "MESSAGE\_EMPLOYEE"  ],  "userGroups": [],  "createdByIdentityId": "c810f086-5331-4336-bb4b-47dbdc27f3d7",  "createdOnBehalfOf": **false**,  "target": {  "id": "8330cbfe-158d-47f5-9250-d6f47eefbc59",  "version": 0,  "createdDateTime": "2021-07-18T14:23:47.813Z",  "lastUpdated": "2021-07-18T14:23:47.813Z",  "requestParticipant": {  "id": "919a73d9-b27d-4f4d-b309-ed40bf4f9b13",  "version": 0,  "createdDateTime": "2021-07-18T14:23:47.814Z",  "lastUpdated": "2021-07-18T14:23:47.814Z",  "externalId": "30714024",  "externalIdType": "CVR",  "organisationIdentityId": "6337c256-133a-46bb-b85c-d65eb751b37f",  "emailAddress": "clan@netcompany.com"  }  },  "documentations": [],  "processedAutomatically": **false**  } |

### Attaching documents to request

Documents can be attached to a DRAFT request and it follows a 2-step approach.

## 1. Add documentation element

First we need to tell the resource what documents will be attached. This goes into the "documentations" list of the resource. It can be present at creation (POST) or they can be added removed, switched around, or edited using PUT.

When adding a “documentation” element, the only required field is documentationType.We can choose from a list of predefined types, that can be seen in the OpenApi documentation:



Current version can be found here: <https://test.digitalpost.dk/api>

Let us add a BUSINESS\_CARD to the request we created on the previous [page](https://digidp.atlassian.net/wiki/spaces/DP/pages/819494917/Creating+and+working+with+drafts).

### Request

PUT /access-requests/1ba3a286-bb03-43ac-b874-e21a5bdee636

|  |
| --- |
| {  "requestType": "DELEGATION\_REQUEST",  "requestState": "DRAFT",  "requester": {  "id": "0ddd0ad1-cc43-45c3-bb9c-2dacd574a4e9",  "version": 0,  "createdDateTime": "2021-07-18T14:23:47.813Z",  "lastUpdated": "2021-07-18T14:23:47.813Z",  "externalId": "30714024",  "externalIdType": "CVR",  "organisationIdentityId": "6337c256-133a-46bb-b85c-d65eb751b37f",  "employeeIdentityId": "c810f086-5331-4336-bb4b-47dbdc27f3d7"  },  "accessTo": {  "id": "b45b6e83-5086-469f-87f0-dff6875de338",  "version": 0,  "createdDateTime": "2021-07-18T14:23:47.807Z",  "lastUpdated": "2021-07-18T14:23:47.807Z",  "externalId": "30714024",  "externalIdType": "CVR",  "organisationIdentityId": "6337c256-133a-46bb-b85c-d65eb751b37f"  },  "privileges": [  "MESSAGE\_EMPLOYEE"  ],  "target": {  "id": "8330cbfe-158d-47f5-9250-d6f47eefbc59",  "version": 0,  "createdDateTime": "2021-07-18T14:23:47.813Z",  "lastUpdated": "2021-07-18T14:23:47.813Z",  "requestParticipant": {  "id": "919a73d9-b27d-4f4d-b309-ed40bf4f9b13",  "version": 0,  "createdDateTime": "2021-07-18T14:23:47.814Z",  "lastUpdated": "2021-07-18T14:23:47.814Z",  "externalId": "30714024",  "externalIdType": "CVR",  "organisationIdentityId": "6337c256-133a-46bb-b85c-d65eb751b37f",  "emailAddress": "clan@netcompany.com"  }  },  "documentations": [  {  "documentationType": "BUSINESS\_CARD"  }  ]  } |

### Response

|  |
| --- |
| {  "id": "1ba3a286-bb03-43ac-b874-e21a5bdee636",  "version": 1,  "transactionId": "F6DDqxoMf52quFkwgdfTUsLJLurGqHGD",  "createdDateTime": "2021-07-18T14:23:47.805Z",  "lastUpdated": "2021-07-18T14:48:58.049Z",  "requestType": "DELEGATION\_REQUEST",  "requestState": "DRAFT",  "requester": {  "id": "0ddd0ad1-cc43-45c3-bb9c-2dacd574a4e9",  "version": 0,  "createdDateTime": "2021-07-18T14:23:47.813Z",  "lastUpdated": "2021-07-18T14:23:47.813Z",  "externalId": "30714024",  "externalIdType": "CVR",  "organisationIdentityId": "6337c256-133a-46bb-b85c-d65eb751b37f",  "employeeIdentityId": "c810f086-5331-4336-bb4b-47dbdc27f3d7"  },  "accessTo": {  "id": "b45b6e83-5086-469f-87f0-dff6875de338",  "version": 0,  "createdDateTime": "2021-07-18T14:23:47.807Z",  "lastUpdated": "2021-07-18T14:23:47.807Z",  "externalId": "30714024",  "externalIdType": "CVR",  "organisationIdentityId": "6337c256-133a-46bb-b85c-d65eb751b37f"  },  "privileges": [  "MESSAGE\_EMPLOYEE"  ],  "userGroups": [],  "createdByIdentityId": "c810f086-5331-4336-bb4b-47dbdc27f3d7",  "createdOnBehalfOf": **false**,  "target": {  "id": "8330cbfe-158d-47f5-9250-d6f47eefbc59",  "version": 0,  "createdDateTime": "2021-07-18T14:23:47.813Z",  "lastUpdated": "2021-07-18T14:23:47.813Z",  "requestParticipant": {  "id": "919a73d9-b27d-4f4d-b309-ed40bf4f9b13",  "version": 0,  "createdDateTime": "2021-07-18T14:23:47.814Z",  "lastUpdated": "2021-07-18T14:23:47.814Z",  "externalId": "30714024",  "externalIdType": "CVR",  "organisationIdentityId": "6337c256-133a-46bb-b85c-d65eb751b37f",  "emailAddress": "clan@netcompany.com"  }  },  "documentations": [  {  "id": "4a583058-84f4-4491-ba0a-bcc80643713c",  "version": 0,  "createdDateTime": "2021-07-18T14:48:58.047Z",  "lastUpdated": "2021-07-18T14:48:58.047Z",  "documentationType": "BUSINESS\_CARD"  }  ],  "processedAutomatically": **false**  } |

Notice the documentation element:

|  |
| --- |
| {  "id": "4a583058-84f4-4491-ba0a-bcc80643713c",  "version": 0,  "createdDateTime": "2021-07-18T14:48:58.047Z",  "lastUpdated": "2021-07-18T14:48:58.047Z",  "documentationType": "BUSINESS\_CARD"  } |

We will need the assigned id to add the byte content to the attachment, which is the second step.

## 2. Upload byte content to documentation

It is done using a PUT multipart request.

### Request

PUT /access-requests/1ba3a286-bb03-43ac-b874-e21a5bdee636/documentations/4a583058-84f4-4491-ba0a-bcc80643713c/content

Notice the documentation id in the URL.  The If-Match header must be set to the version of the documentation element.

The name of the multipart form element must be ‘file’. Here is a Curl example:

|  |
| --- |
| curl --location --request PUT 'https://dev.digdp.nchosting.dk/apis/v1/access-requests/1ba3a286-bb03-43ac-b874-e21a5bdee636/documentations/4a583058-84f4-4491-ba0a-bcc80643713c/content' \  --header 'If-Match: 0' \  --header 'Authorization: Bearer eyJh...' \  --form 'file=@"../business-card.pdf"' |

The client must NOT base64 encode the file content. The maximum size of the file is 10 MB.

Allowed file types are: application/pdf, image/png, image/jpeg.

### Response

|  |
| --- |
| {  "id": "1ba3a286-bb03-43ac-b874-e21a5bdee636",  "version": 2,  "transactionId": "F6DGzMEQshbb6c3RXtB7J9DM7zZcWkSd",  "createdDateTime": "2021-07-18T14:23:47.805Z",  "lastUpdated": "2021-07-18T15:12:35.222Z",  "requestType": "DELEGATION\_REQUEST",  "requestState": "DRAFT",  "requester": {  "id": "0ddd0ad1-cc43-45c3-bb9c-2dacd574a4e9",  "version": 0,  "createdDateTime": "2021-07-18T14:23:47.813Z",  "lastUpdated": "2021-07-18T14:23:47.813Z",  "externalId": "30714024",  "externalIdType": "CVR",  "organisationIdentityId": "6337c256-133a-46bb-b85c-d65eb751b37f",  "employeeIdentityId": "c810f086-5331-4336-bb4b-47dbdc27f3d7"  },  "accessTo": {  "id": "b45b6e83-5086-469f-87f0-dff6875de338",  "version": 0,  "createdDateTime": "2021-07-18T14:23:47.807Z",  "lastUpdated": "2021-07-18T14:23:47.807Z",  "externalId": "30714024",  "externalIdType": "CVR",  "organisationIdentityId": "6337c256-133a-46bb-b85c-d65eb751b37f"  },  "privileges": [  "MESSAGE\_EMPLOYEE"  ],  "userGroups": [],  "createdByIdentityId": "c810f086-5331-4336-bb4b-47dbdc27f3d7",  "createdOnBehalfOf": **false**,  "target": {  "id": "8330cbfe-158d-47f5-9250-d6f47eefbc59",  "version": 0,  "createdDateTime": "2021-07-18T14:23:47.813Z",  "lastUpdated": "2021-07-18T14:23:47.813Z",  "requestParticipant": {  "id": "919a73d9-b27d-4f4d-b309-ed40bf4f9b13",  "version": 0,  "createdDateTime": "2021-07-18T14:23:47.814Z",  "lastUpdated": "2021-07-18T14:23:47.814Z",  "externalId": "30714024",  "externalIdType": "CVR",  "organisationIdentityId": "6337c256-133a-46bb-b85c-d65eb751b37f",  "emailAddress": "clan@netcompany.com"  }  },  "documentations": [  {  "id": "4a583058-84f4-4491-ba0a-bcc80643713c",  "version": 1,  "createdDateTime": "2021-07-18T14:48:58.047Z",  "lastUpdated": "2021-07-18T15:12:35.223Z",  "mediaType": "application/pdf",  "filename": "business-card.pdf",  "documentationType": "BUSINESS\_CARD",  "size": 71929  }  ],  "processedAutomatically": **false**  } |

### Citizen requesting access to other citizens mailbox

A citizen A requesting LEGAL\_REPRESENTATIVE to another citizen B. Requester and Target is the citizen A requesting the privilege. AccessTo is the citizen B that must approve and grant the request.

## 1. Citizen A submits request

### Request

POST /access-requests/

|  |
| --- |
| {  "requestType": "PRIVILEGE\_REQUEST",  "requestState": "SUBMITTED",  "requester": {  "citizenIdentityId": "dab86db0-98c0-4062-bb70-39bdb3bd9cab"  },  "accessTo": {  "externalId": "0610749832",  "externalIdType": "CPR",  "firstNameProvided": "Lone",  "lastNameProvided": "Boaikvopg"  },  "privileges": [  "LEGAL\_REPRESENTATIVE"  ],  "target": {  "requestParticipant": {  "citizenIdentityId": "dab86db0-98c0-4062-bb70-39bdb3bd9cab"  }  }  } |

### Response

|  |
| --- |
| {  "id": "02a307ca-227f-445b-8176-46ad04eb802e",  "version": 1,  "transactionId": "F7MjMnGST1vbcsqxf6AZnU9CbWoyEoRh",  "createdDateTime": "2021-08-10T19:27:17.290Z",  "lastUpdated": "2021-08-10T19:27:19.538Z",  "requestType": "PRIVILEGE\_REQUEST",  "requestState": "SUBMITTED",  "requester": {  "id": "5ec74ef8-d22b-40e4-94ec-ef1262dd3d3c",  "version": 0,  "createdDateTime": "2021-08-10T19:27:17.291Z",  "lastUpdated": "2021-08-10T19:27:17.291Z",  "externalId": "0610742121",  "externalIdType": "CPR",  "citizenIdentityId": "dab86db0-98c0-4062-bb70-39bdb3bd9cab"  },  "accessTo": {  "id": "eeccfad4-e150-4a87-bd77-373eaf5dfc09",  "version": 0,  "createdDateTime": "2021-08-10T19:27:17.290Z",  "lastUpdated": "2021-08-10T19:27:17.290Z",  "externalId": "0610749832",  "externalIdType": "CPR",  "citizenIdentityId": "44ba8c4a-cc01-4f7e-bf4c-8188f9203f28",  "firstNameProvided": "Lone",  "lastNameProvided": "Boaikvopg",  "firstNameResolved": "Lone Boaikvopg"  },  "privileges": [  "LEGAL\_REPRESENTATIVE"  ],  "userGroups": [],  "createdByIdentityId": "dab86db0-98c0-4062-bb70-39bdb3bd9cab",  "createdOnBehalfOf": **false**,  "target": {  "id": "2bf85948-26c1-40f4-a0fc-f0d22a7e8a90",  "version": 0,  "createdDateTime": "2021-08-10T19:27:17.291Z",  "lastUpdated": "2021-08-10T19:27:17.291Z",  "requestParticipant": {  "id": "a6a3291d-ff6d-4b08-8929-61adc0b05c91",  "version": 0,  "createdDateTime": "2021-08-10T19:27:17.291Z",  "lastUpdated": "2021-08-10T19:27:17.291Z",  "externalId": "0610742121",  "externalIdType": "CPR",  "citizenIdentityId": "dab86db0-98c0-4062-bb70-39bdb3bd9cab"  }  },  "documentations": [  {  "id": "b6b27ed1-158b-4a43-9949-b916c4d0fa3f",  "version": 0,  "createdDateTime": "2021-08-10T19:27:19.537Z",  "lastUpdated": "2021-08-10T19:27:19.537Z",  "mediaType": "application/xml",  "filename": "generic",  "documentationType": "ACCESS\_REQUEST\_MESSAGE\_MEMO",  "size": 2347,  "scanState": "NOT\_INFECTED"  },  {  "id": "1b1717f8-f707-456d-9fec-7c083d82bf18",  "version": 0,  "createdDateTime": "2021-08-10T19:27:19.538Z",  "lastUpdated": "2021-08-10T19:27:19.538Z",  "mediaType": "application/xml",  "filename": "generic",  "documentationType": "ACCESS\_REQUEST\_MESSAGE\_MEMO",  "size": 2347,  "scanState": "NOT\_INFECTED"  }  ],  "processedAutomatically": **false**,  "submissionDateTime": "2021-08-10T19:27:17.303Z",  "submittedByIdentityId": "dab86db0-98c0-4062-bb70-39bdb3bd9cab"  } |

## 2. Citizen B queries to see incoming requests

Once submitted it will turn up when citizen B queries.

### Request

GET /access-requests/

### Response

|  |
| --- |
| {  "currentPage": 0,  "totalPages": 1,  "elementsOnPage": 1,  "totalElements": 1,  "accessRequests": [  {  "id": "02a307ca-227f-445b-8176-46ad04eb802e",  "version": 1,  "transactionId": "F7MjMnGST1vbcsqxf6AZnU9CbWoyEoRh",  "createdDateTime": "2021-08-10T19:27:17.290Z",  "lastUpdated": "2021-08-10T19:27:19.538Z",  "requestType": "PRIVILEGE\_REQUEST",  "requestState": "SUBMITTED",  "requester": {  "id": "5ec74ef8-d22b-40e4-94ec-ef1262dd3d3c",  "version": 0,  "createdDateTime": "2021-08-10T19:27:17.291Z",  "lastUpdated": "2021-08-10T19:27:17.291Z",  "externalId": "0610742121",  "externalIdType": "CPR",  "citizenIdentityId": "dab86db0-98c0-4062-bb70-39bdb3bd9cab"  },  "accessTo": {  "id": "eeccfad4-e150-4a87-bd77-373eaf5dfc09",  "version": 0,  "createdDateTime": "2021-08-10T19:27:17.290Z",  "lastUpdated": "2021-08-10T19:27:17.290Z",  "externalId": "0610749832",  "externalIdType": "CPR",  "citizenIdentityId": "44ba8c4a-cc01-4f7e-bf4c-8188f9203f28",  "firstNameProvided": "Lone",  "lastNameProvided": "Boaikvopg",  "firstNameResolved": "Lone Boaikvopg"  },  "privileges": [  "LEGAL\_REPRESENTATIVE"  ],  "userGroups": [],  "createdByIdentityId": "dab86db0-98c0-4062-bb70-39bdb3bd9cab",  "createdOnBehalfOf": **false**,  "target": {  "id": "2bf85948-26c1-40f4-a0fc-f0d22a7e8a90",  "version": 0,  "createdDateTime": "2021-08-10T19:27:17.291Z",  "lastUpdated": "2021-08-10T19:27:17.291Z",  "requestParticipant": {  "id": "a6a3291d-ff6d-4b08-8929-61adc0b05c91",  "version": 0,  "createdDateTime": "2021-08-10T19:27:17.291Z",  "lastUpdated": "2021-08-10T19:27:17.291Z",  "externalId": "0610742121",  "externalIdType": "CPR",  "citizenIdentityId": "dab86db0-98c0-4062-bb70-39bdb3bd9cab"  }  },  "documentations": [],  "processedAutomatically": **false**,  "submissionDateTime": "2021-08-10T19:27:17.303Z",  "submittedByIdentityId": "dab86db0-98c0-4062-bb70-39bdb3bd9cab"  }  ]  } |

## 3. Citizen B approves request

### Request

PUT /access-requests/02a307ca-227f-445b-8176-46ad04eb802e

|  |
| --- |
| {  "requestType": "PRIVILEGE\_REQUEST",  "requestState": "APPROVED",  "requester": {  "id": "5ec74ef8-d22b-40e4-94ec-ef1262dd3d3c",  "version": 0,  "createdDateTime": "2021-08-10T19:27:17.291Z",  "lastUpdated": "2021-08-10T19:27:17.291Z",  "externalId": "0610742121",  "externalIdType": "CPR",  "citizenIdentityId": "dab86db0-98c0-4062-bb70-39bdb3bd9cab"  },  "accessTo": {  "id": "eeccfad4-e150-4a87-bd77-373eaf5dfc09",  "version": 0,  "createdDateTime": "2021-08-10T19:27:17.290Z",  "lastUpdated": "2021-08-10T19:27:17.290Z",  "externalId": "0610749832",  "externalIdType": "CPR",  "citizenIdentityId": "44ba8c4a-cc01-4f7e-bf4c-8188f9203f28",  "firstNameProvided": "Lone",  "lastNameProvided": "Boaikvopg",  "firstNameResolved": "Lone Boaikvopg"  },  "privileges": [  "LEGAL\_REPRESENTATIVE"  ],  "userGroups": [],  "createdByIdentityId": "dab86db0-98c0-4062-bb70-39bdb3bd9cab",  "createdOnBehalfOf": **false**,  "target": {  "id": "2bf85948-26c1-40f4-a0fc-f0d22a7e8a90",  "version": 0,  "createdDateTime": "2021-08-10T19:27:17.291Z",  "lastUpdated": "2021-08-10T19:27:17.291Z",  "requestParticipant": {  "id": "a6a3291d-ff6d-4b08-8929-61adc0b05c91",  "version": 0,  "createdDateTime": "2021-08-10T19:27:17.291Z",  "lastUpdated": "2021-08-10T19:27:17.291Z",  "externalId": "0610742121",  "externalIdType": "CPR",  "citizenIdentityId": "dab86db0-98c0-4062-bb70-39bdb3bd9cab"  }  },  "documentations": [],  "processedAutomatically": **false**,  "submissionDateTime": "2021-08-10T19:27:17.303Z",  "submittedByIdentityId": "dab86db0-98c0-4062-bb70-39bdb3bd9cab"  } |

### Response

|  |
| --- |
| {  "id": "02a307ca-227f-445b-8176-46ad04eb802e",  "version": 2,  "transactionId": "F7MlCBdrICUttJJkdYOhv3strTN6vDO8",  "createdDateTime": "2021-08-10T19:27:17.290Z",  "lastUpdated": "2021-08-10T19:41:03.957Z",  "requestType": "PRIVILEGE\_REQUEST",  "requestState": "APPROVED",  "requester": {  "id": "5ec74ef8-d22b-40e4-94ec-ef1262dd3d3c",  "version": 0,  "createdDateTime": "2021-08-10T19:27:17.291Z",  "lastUpdated": "2021-08-10T19:27:17.291Z",  "externalId": "0610742121",  "externalIdType": "CPR",  "citizenIdentityId": "dab86db0-98c0-4062-bb70-39bdb3bd9cab"  },  "accessTo": {  "id": "eeccfad4-e150-4a87-bd77-373eaf5dfc09",  "version": 0,  "createdDateTime": "2021-08-10T19:27:17.290Z",  "lastUpdated": "2021-08-10T19:27:17.290Z",  "externalId": "0610749832",  "externalIdType": "CPR",  "citizenIdentityId": "44ba8c4a-cc01-4f7e-bf4c-8188f9203f28",  "firstNameProvided": "Lone",  "lastNameProvided": "Boaikvopg",  "firstNameResolved": "Lone Boaikvopg"  },  "privileges": [  "LEGAL\_REPRESENTATIVE"  ],  "userGroups": [],  "createdByIdentityId": "dab86db0-98c0-4062-bb70-39bdb3bd9cab",  "createdOnBehalfOf": **false**,  "target": {  "id": "2bf85948-26c1-40f4-a0fc-f0d22a7e8a90",  "version": 1,  "createdDateTime": "2021-08-10T19:27:17.291Z",  "lastUpdated": "2021-08-10T19:41:03.958Z",  "requestParticipant": {  "id": "a6a3291d-ff6d-4b08-8929-61adc0b05c91",  "version": 0,  "createdDateTime": "2021-08-10T19:27:17.291Z",  "lastUpdated": "2021-08-10T19:27:17.291Z",  "externalId": "0610742121",  "externalIdType": "CPR",  "citizenIdentityId": "dab86db0-98c0-4062-bb70-39bdb3bd9cab"  },  "targetState": "GRANTED"  },  "documentations": [],  "processedAutomatically": **false**,  "submissionDateTime": "2021-08-10T19:27:17.303Z",  "submittedByIdentityId": "dab86db0-98c0-4062-bb70-39bdb3bd9cab",  "approvalDateTime": "2021-08-10T19:40:54.672Z",  "approvedByIdentityId": "44ba8c4a-cc01-4f7e-bf4c-8188f9203f28"  } |

# Sender-/Receiver Systems

## Rate-limiting

To ensure that Digital Post is operational the API is protected by a rate limiter. The rate limiter ensures that a caller can only perform a set number of requests within a certain period. And if the caller exceeds this limit the call will be rejected by the API and instead get a HTTP 429 - Too Many Requests.

To ensure that all incoming requests can be rate-limited, the rate-limiting is placed in the centralized [gateway](https://digidp.atlassian.net/wiki/spaces/DP/pages/831521704/Gateway) component which is responsible for routing external traffic into specific underlying microservices. Therefore the implementation is based around the Spring Cloud Gateways implementation and thus conveniently exposes a set of headers to the caller, which they can utilize to ensure they are not rate-limited.

The headers which are added to make external requests are the following:

* X-RateLimit-Remaining This header exposes the number of remaining tokens in the bucket that the caller has available. When this reaches 0 the caller is rejected
* X-RateLimit-Requested-Tokens The number of tokens that the request removed from the bucket when performing the request. This is used to differentiate between ‘cheap’ and ‘expensive’ requests. How a request is determined to be either cheap or expensive is internally determined and can therefore change without notice as the system is optimized.
* X-RateLimit-Burst-Capacity the bust capacity is the number of tokens which the bucket initially contained
* X-RateLimit-Replenish-Rate The rate of how fast tokens are re-added to the bucket each second.

The way Digital Post determines which bucket to assign callers is internally handled by the Digital Post and can therefore be tweaked without notice. However, callers can expect that the implementation follows the identity of the caller. Meaning that if a sender system is calling, Digital Post assigns a bucket to that system. In the case of standard systems, the limiting is based on who the standard system is acting on behalf of. Meaning that if a standard system is serving two different authorities, these will be rate-limited independently.

If Digital Post is unable to determine the identity of the caller the rate limit will fall back to the IP of the caller.

The current configuration in both the test environment and production is set so callers have a bucket of 10 tokens, which effectively means that callers can have at most 10 calls per second to the API.

## Patterns for integration to Digital Post

This section contains high level architecture of how Sender and Receiver Systems should be structured using the MeMo-lib. Inherently this also applies the reference implementations which shows a practical example of how the architecture is applied. To reduce the complexity the following section only address operations related to handling MeMo messages.

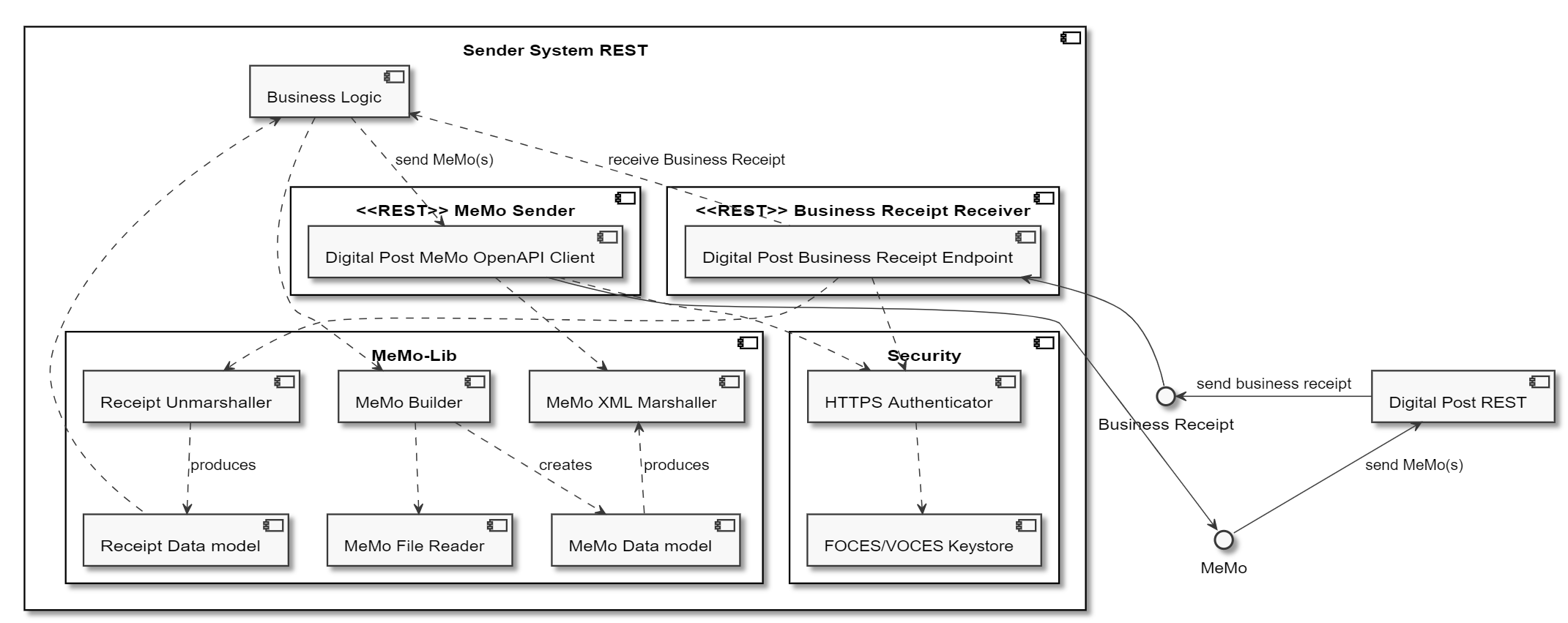
Since Digital Post does not constrain the business logic of when a Sender systems trigger messages and how receiver systems handles received messages all the diagrams contain a “Business Logic” component, which is a representation of where the business related logics could be placed.

## Sender system

The following section gives an overview of how sender systems should be structured.

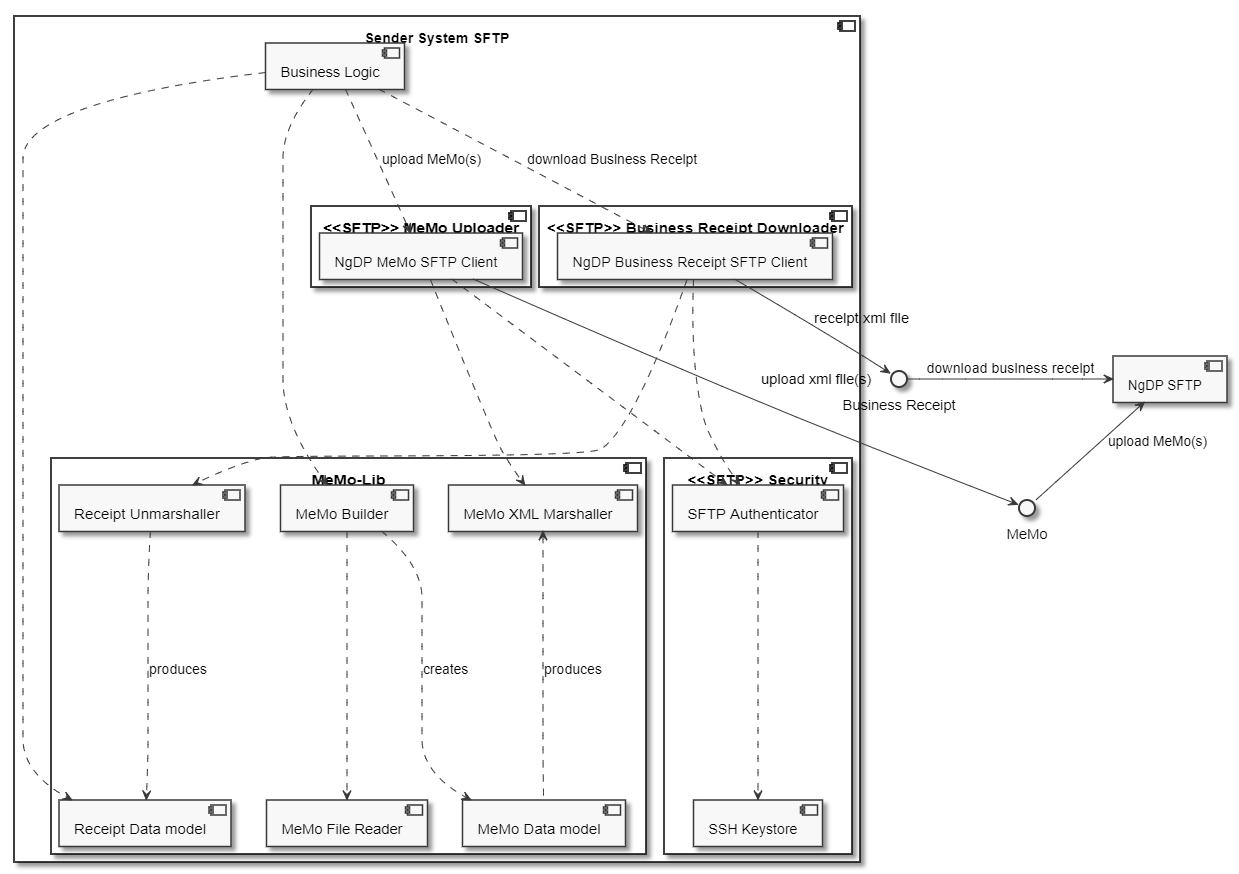
### REST Protocol

A Sender System using the REST service protocol is expected to communicate with Digital Post via the HTTP based interface and authenticating using mutual SSL. The REST protocol is ideal to send messages in synchronously to Digital Post and having them distributed immediately.



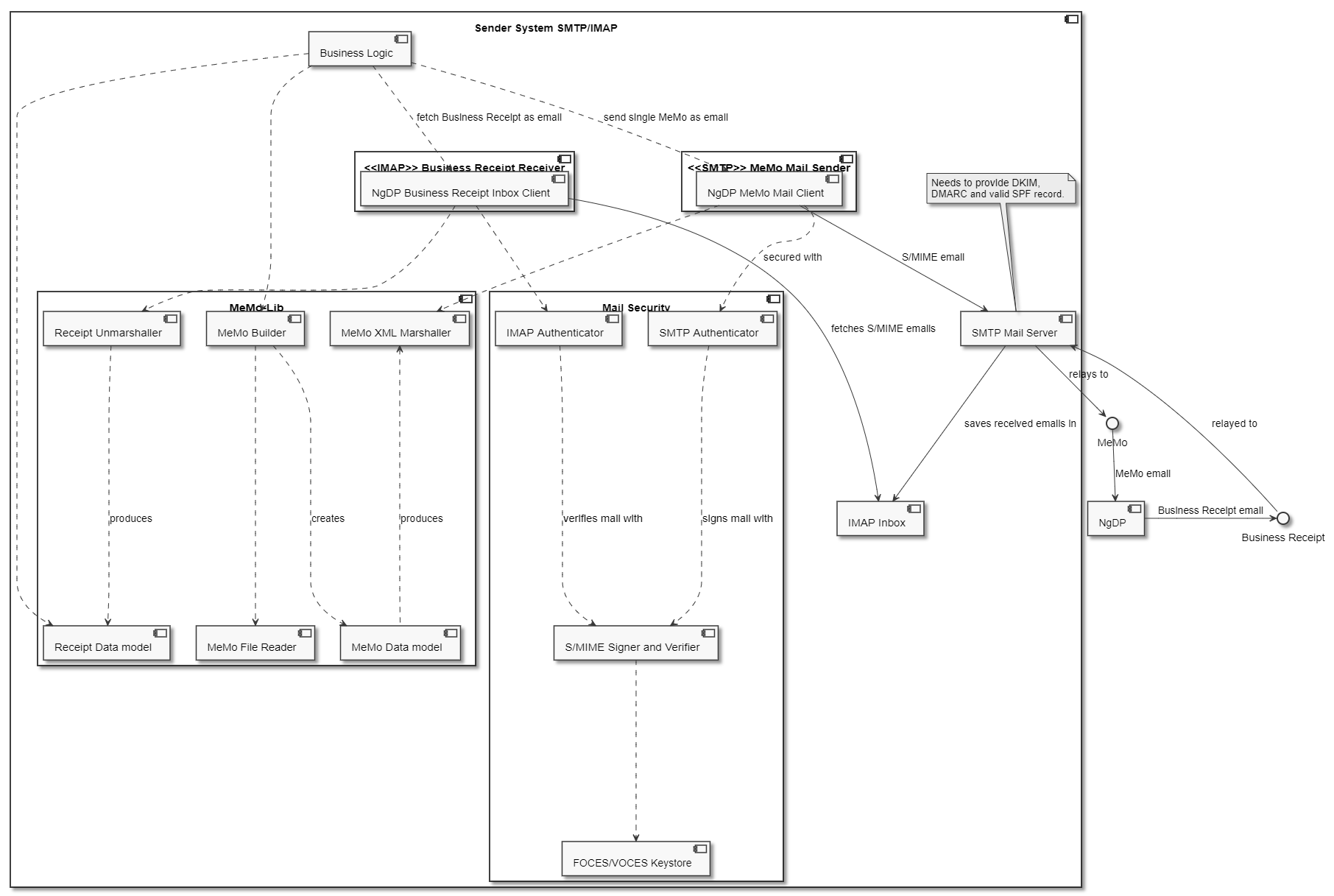
### SFTP Protocol

The SFTP service protocol is ideal when sending large quantities of messages. Which can be delivered asynchronously.



### SMTP/IMAP Protocol

Using the SMTP / IMAP protocol is ideal when you want to integrate Digital Post into an existing message platform.

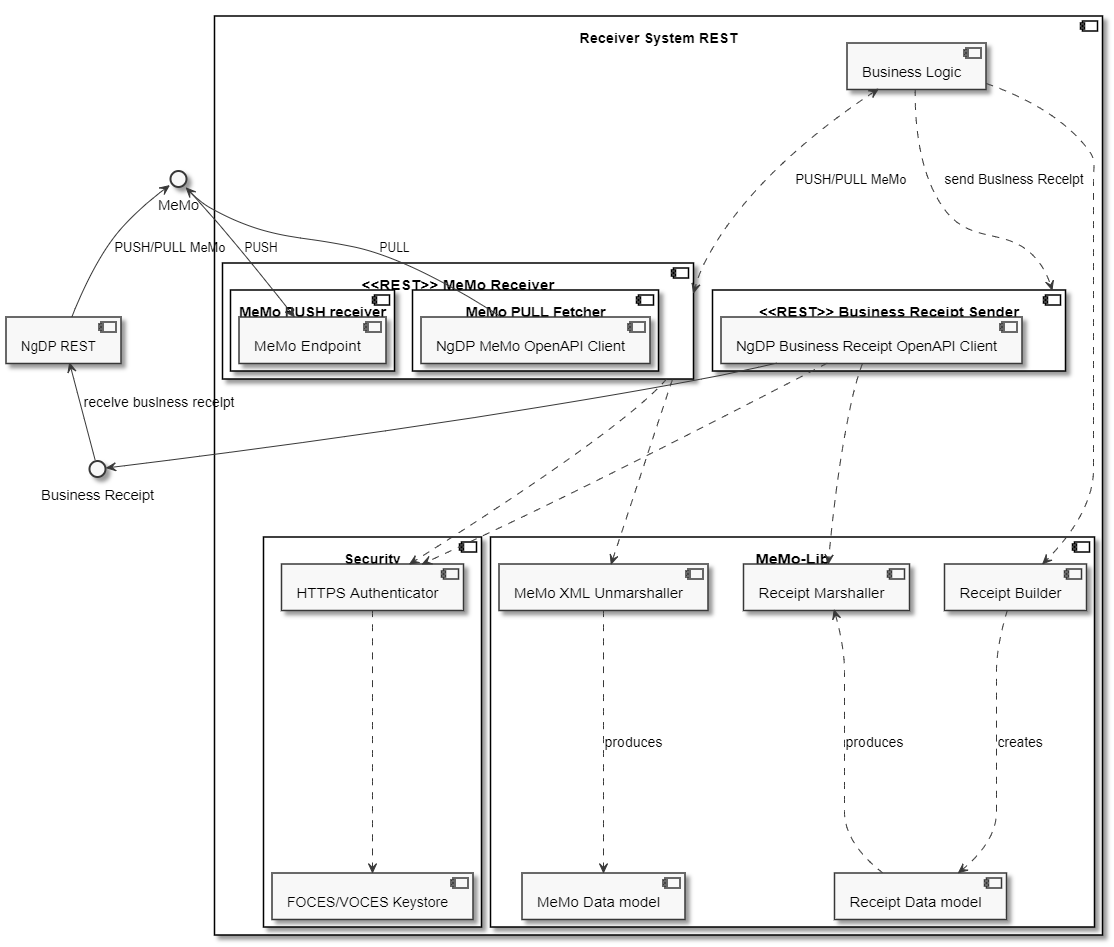


## Receiver system

The following section gives an overview of how receiver systems should be structured.

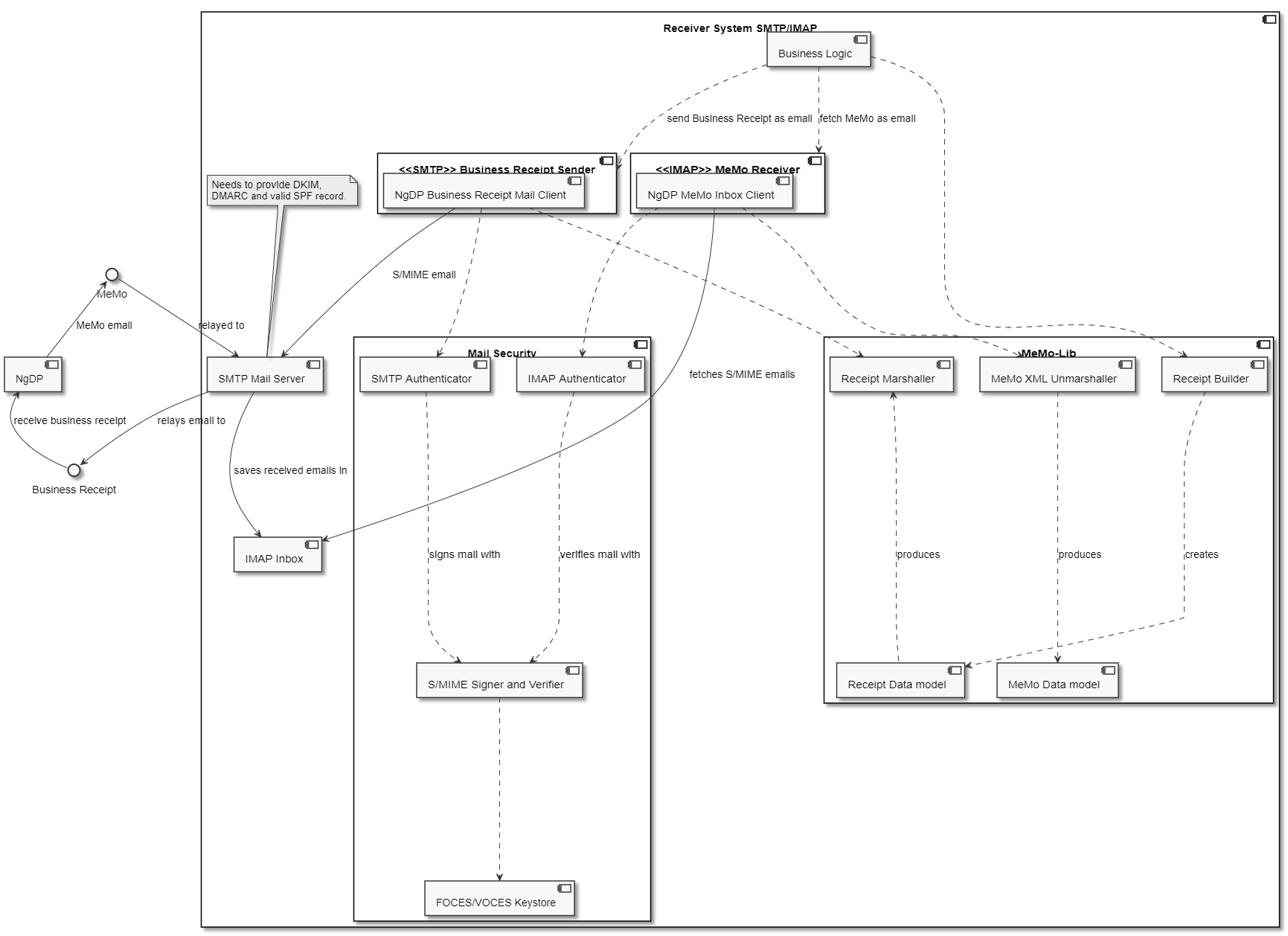
### REST Protocol

The HTTP based REST service protocol can be used both to receive messages as they arrive using the PUSH variant where Digital Post will “push” messages as they are distributed directly to the receiver system. Or as the receiver systems pleases using the pull variant.



### SMTP/IMAP Protocol

Using the SMTP / IMAP protocol is ideal when you want to integrate Digital Post into an existing message platform.



### Sending Memo

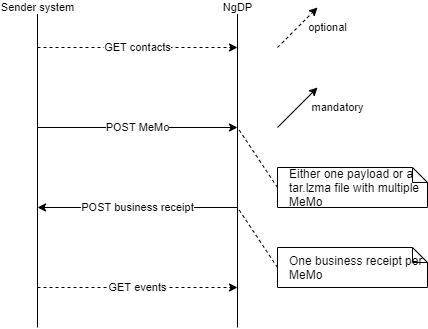
A sender system can deliver MeMos to Digital Post using three different protocols: HTTP, SMTP, and SFTP.

The term REST is often used instead of HTTP since DP is a rest-full API built on HTTP.

Sending a single MeMo is done just using the XML of the MeMo, but multiple MeMos can be delivered in the same request using tar.lzma packaging and compression.

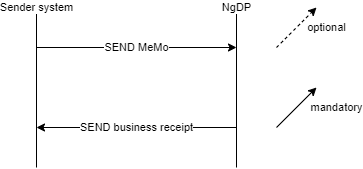
#### Sender system sending one or more MeMo over HTTP (REST)

1. Optional: Sender system lookup if one or more recipients are exempted from Digital Post and/or subscribes to NemSMS
2. Sender system sends either
   1. one MeMo over REST, which must be a <messageUUID> or <messageUUID>.xml
   2. Sending one or more MeMos in a tar.lzma file (name of MeMo file inside tar must be <messageUUID>.xml or <messageUUID>)  
      DP responds with technical receipt in the form of HTTP status code
3. DP sends one business receipt per MeMo. Sender system responds with HTTP status code
4. Optional: Sender system can lookup events for the sent MeMo, e.g. to see if messages are waiting for delivery date (valørdato)



#### Sender system sending one MeMo over SMTP

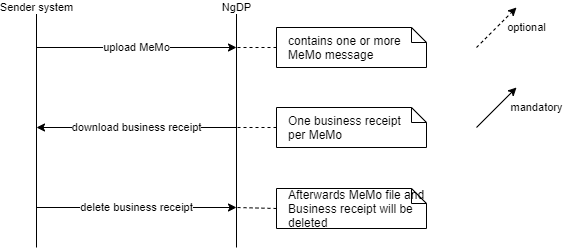
1. Sender system sends an email with MeMo file as an attachment (name of MeMo file in email must be <messageUUID>.xml or <messageUUID>)
2. DP sends an email with Business Receipt xml file as an attachment



#### Sender system sending file with multiple MeMos over SFTP

DP SFTP support standard SFTP functionality, e.g. compression and transferring/modifying multiple files at-once, either with wildcards or list of files to be addressed.

1. Sender system uploads tar.lzma file with one or more MeMo files (name of MeMo files inside tar must be <messageUUID>.xml or <messageUUID>)
   1. uploading file to temp subfolder
   2. move the file to main folder
2. DP uploads one Business Receipt per MeMo
   1. uploading file to temp subfolder
   2. move the file to main folder
3. Sender system downloads the Business receipt files
4. Sender system deletes the Business receipt files



### Receiving Memo

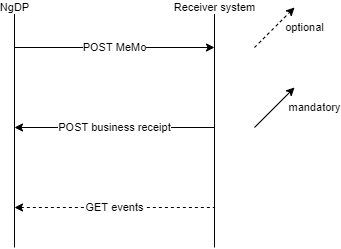
A recipient system can receive MeMos from Dp using two different protocols: HTTP and SMTP.

The term REST is often used instead of HTTP since DP is a rest-full API built on HTTP.

Using REST/HTTP is done in one of three ways: REST PUSH, REST PUBLISH SUBSCRIBE, or REST PULL, which is configured when attaching the recipient system.

#### Delivering one MeMo over REST PUSH

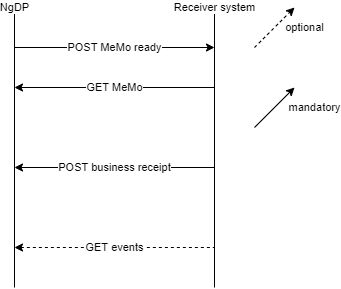
1. DP sends a single MeMo xml, receiver system responds with HTTP status code
2. Recipient system sends a Business Receipt, DP responds with HTTP status code
3. Optional: Recipient system fetches events regarding MeMo



#### Delivering one MeMo over REST PUBLISH SUBSCRIBE

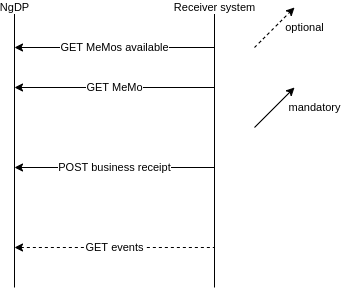
Note: Default recipient systems must use this pattern

1. DP publish new MeMo is ready (one publish call per MeMo), including MeMo id, Receiver system responds with HTTP status code
2. Recipient system fetches MeMo, DP responds with HTTP status code
3. Recipient system sends Business receipt, DP responds with HTTP status code
4. Optional: Recipient system fetches events about the MeMo



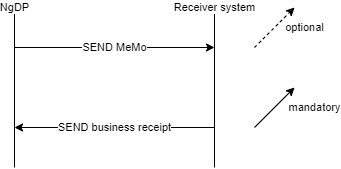
#### Delivering one MeMo over REST PULL

1. Recipient system fetches list of available MeMos, DP responds with HTTP status code
2. Recipient system fetches MeMo, DP responds with HTTP status code
3. Recipient system sends Business receipt, DP responds with HTTP status code
4. Optional: Recipient system fetches events about the MeMo



#### Delivering one MeMo over SMTP

1. DP sends an email with MeMo xml file as an attachment (name of MeMo file in email must be <messageUUID>.xml or <messageUUID>)
2. Recipient system sends an email with Business Receipt xml file as an attachment



# Encoding formats, Environments and Error codes

## Encoding format whitelist for files of documents

List of allowed values for the encodingFormat field of the File ressource.

### Main document

* application/pdf
* text/html
* text/plain

### Additional documents

* image/bmp
* text/csv
* application/vnd.fujixerox.ddd
* application/msword
* application/vnd.openxmlformats-officedocument.wordprocessingml.document
* application/x-stata-dta
* image/gif
* text/html
* text/calendar
* image/jpeg
* video/quicktime
* audio/mpeg
* video/mp4
* application/vnd.oasis.opendocument.spreadsheet
* application/vnd.oasis.opendocument.text
* application/pdf
* image/png
* application/rtf
* application/x-spss-sav
* image/tiff
* text/plain
* audio/wav
* application/vnd.ms-excel
* application/vnd.openxmlformats-officedocument.spreadsheetml.sheet
* application/xml
* text/xml

### Technical documents

* application/xml
* text/xml
* application/json

## Access to environments

All REST services for an Digital Post environment are exposed from or called from a single IP.

### Making requests - important!

When requesting, make sure you include “apis”-path as well as version.

Example: <https://api.test.digitalpost.dk/apis/v1/contacts/>

### TEST environment

|  |  |  |
| --- | --- | --- |
| **Protocol** | **Inbound/outbound, from Digital Post perspective** | **Digital Post IP** |
| REST | Inbound | 80.198.95.44 |
| SMTP | Inbound | 80.198.95.41 |
| SFTP | Inbound | 80.198.95.42 |
| REST | Outbound | 80.198.95.62 |

|  |  |  |  |
| --- | --- | --- | --- |
| **Component** | **URL** | **Port** | **Protocol** |
| Mailserver | [http://smtp.test.digitalpost.dk](http://smtp.test.digitalpost.dk/) | 25 | SMTP |
| SFTP-server | [http://sftp.test.digitalpost.dk](http://sftp.test.digitalpost.dk/) | 22 | SFTP |
| Distribution | [https://api.test.digitalpost.dk](https://api.test.digitalpost.dk/) | 443 | HTTPS/REST |
| Kontaktregister | [https://api.test.digitalpost.dk](https://api.test.digitalpost.dk/) | 443 | HTTPS/REST |
| Systemregister | [https://api.test.digitalpost.dk](https://api.test.digitalpost.dk/) | 443 | HTTPS/REST |
| Opbevaring | [https://api.test.digitalpost.dk](https://api.test.digitalpost.dk/) | 443 | HTTPS/REST |
| Hændelseslog | [https://api.test.digitalpost.dk](https://api.test.digitalpost.dk/) | 443 | HTTPS/REST |

### PROD Environment

|  |  |  |
| --- | --- | --- |
| **Protocol** | **Inbound/Outbound from Digital Post perspective** | **Digital Post IP** |
| REST | Inbound | 80.198.95.23 |
| SMTP | Inbound | 80.198.95.21 |
| SFTP | Inbound | 80.198.95.22 |
| REST | Outbound | 80.198.95.62 |

|  |  |  |  |
| --- | --- | --- | --- |
| **Component** | **URL** | **Port** | **Protocol** |
| Mailserver | [smtp.digitalpost.dk](http://smtp.digitalpost.dk/) | 25 | SMTP |
| SFTP-server | [sftp.digitalpost.dk](http://sftp.digitalpost.dk/) | 22 | SFTP |
| Distribution | [api.digitalpost.dk](http://api.digitalpost.dk/) | 443 | HTTPS/REST |
| Kontaktregister | [api.digitalpost.dk](http://api.digitalpost.dk/) | 443 | HTTPS/REST |
| Systemregister | [api.digitalpost.dk](http://api.digitalpost.dk/) | 443 | HTTPS/REST |
| Opbevaring | [api.digitalpost.dk](http://api.digitalpost.dk/) | 443 | HTTPS/REST |
| Hændelseslog | [api.digitalpost.dk](http://api.digitalpost.dk/) | 443 | HTTPS/REST |

#### E-mails from Digital Post

* Unrepliable e-mails (notifications, receipts and rights notifications) are sent from [noreply@digitalpost.dk](mailto:noreply@digitalpost.dk)

### Guidelines for downtime

Due to maintenance on the test environment, downtime can sometimes be experienced.

To reduce the inconvenience of downtime, the following guidelines have been established.

#### Guidelines

1. Regular operation time for the test-environment is weekdays 08:00 - 17:00 as well as the entire weekend. In this time period external users can expect stable operation of the test environment.
   1. Excluded from this are Thursdays from 08:00 - 14:00, which are reserved for maintenance of the DP test-environment, during which instability in the environment can be expected.
2. Hotfixes to the Digital Post test-environment will occur as part of Thursday’s service window/maintenance or outside of normal business hours.
3. Release of a new version of Digital Post. Usually when a new release is installed on one of the above environments there are no downtime, sometimes in-flight requests can be dropped and you can experience longer response times. make sure that you are familiar with the release calendar
   1. [Digitaliser.dk - Releasekalender](https://digitaliser.dk/digital-post/releasekalender)
   2. Under special circumstances there can be downtime related to the installation of a new version. This will always be announced beforehand on [digitaliser.dk](http://digitaliser.dk).

## Error codes

This section describes all the errors codes returned by DP.

* Front-end validation and error codes in the view client
* Back-end validation and error codes in distribution
* Recipient system error codes

### Front-end validation and error codes in the Viewclient

This section describes the front-end errorcodes from DP, i.e. the error codes that may be returned synchronously from the backend via REST CRUD calls.

#### Access services

|  |
| --- |
| # Access  access.accessType.notNull=Typen skal udfyldes.  access.mailboxId.notNull=Postkassens id skal udfyldes.  access.maximum.number.of.emails.exceeded=Der kan maksimalt oprettes {0} antal e-mails til notifikation **for** denne type adgang.  access.maximum.number.of.sms.exceeded=Der kan maksimalt oprettes {0} antal sms til notifikation **for** denne type adgang.  access.emailNotificationSubscriptions.email.required=E-mailadressen skal udfyldes.  access.emailNotificationSubscriptions.email.invalid=E-mailadressen er ikke gyldig.  access.emailNotificationSubscriptions.email.alreadyUsed=E-mailadressen {0} er allerede angivet én gang.  access.emailNotificationSubscriptions.confirmationTime.verificationRequired=Verifikation skal gennemføres før e-mailadressen kan bekræftes.  access.smsNotificationSubscription.mobileNumber.required=Mobiltelefonnummeret skal udfyldes.  access.smsNotificationSubscription.mobileNumber.invalid=Mobiltelefonnummeret {0} er ikke et gyldigt dansk nummer.  access.smsNotificationSubscription.confirmationTime.verificationRequired=Verifikation skal gennemføres før mobiltelefonnummeret kan bekræftes.  access.pushNotificationSubscriptions.deviceId.required=Device id skal udfyldes.  access.pushNotificationSubscriptions.deviceId.alreadyUsed=Device id {0} er allerede angivet én gang.  access.optedOutOfNotificationsDateTime.isInAFuture=datoen må ikke være i fremtiden. |

#### Access request services

|  |
| --- |
| accessRequest.requestType.required=Type (requestType) skal udfyldes.  accessRequest.requestType.invalid=Ugyldig værdi angivet.  accessRequest.privilegeEndDate.invalid=En anmodning må ikke oprettes eller godkendes med udløbet slutdato.  accessRequest.privilegeEndDateTime.invalid=En anmodning må ikke oprettes eller godkendes med udløbet slutdato.  accessRequest.privilegeEndDateTime.invalid.after=Anmodningens sluttidspunkt må ikke være efter {0}.  accessRequest.privilegeEndDate.withPrivilegeEndDateTime=En anmodning kan ikke indeholde både privilegeEndDate og privilegeEndDateTime.  accessRequest.requestType.modified=Det er ikke tilladt at skifte type.  accessRequest.requestState.required=Tilstand (requestState) skal udfyldes.  accessRequest.requestState.invalid=Ugyldig værdi angivet.  accessRequest.create.requestState.invalid=En anmodning kan kun oprettes med requestState DRAFT.  accessRequest.requestState.invalidTransition=En anmodning kan ikke skifte fra {0} til {1}.  accessRequest.requester.required=Anmoderen (requester) skal udfyldes.  accessRequest.accessTo.required=Rettighedsejer (accessTo) skal udfyldes.  accessRequest.target.required=Rettighedsmodtager (target) skal udfyldes.  accessRequest.documentations.mediaType.invalid=Ugyldig type {0} - tilladte typer {1}.  accessRequest.documentations.tooLarge=Ugyldig filstørrelse - maks. størrelse {0} bytes.  accessRequest.documentations.size=Antallet af vedhæftninger må ikke overstiger {0}.  accessRequest.documentations.documentationType.required=Dokumenttype skal udfyldes.  accessRequest.documentations.documentationType.specific.required=En dokument af typen {0} skal vedhæftes.  accessRequest.documentations.content.infected=Virus detekteret. Filen kan ikke tilføjes.  accessRequest.requestParticipant.contact.not.allowed=Det er ikke muligt at lave et forespørgsel til denne kontakt  accessRequest.requestParticipant.identity.notFound=Deltageren {0} kunne ikke identificeres.  accessRequest.requestParticipant.identity.identityType=Typen på deltageren {0} kunne ikke identificeres.  accessRequest.requestParticipant.identity.required=Deltagerens identitet kunne ikke afgøres. Enten skal externalId og type udfyldes eller også identitets id udfyldes.  accessRequest.requestParticipant.identity.mismatch=Deltagerens angivne interne identifikator matcher ikke med den eksterne.  accessRequest.requestParticipant.identityType.mismatch=Deltagerens identitet er en blanding af borger og virksomhed/medarbejder og dermed ugyldig.  accessRequest.requestParticipant.externalId.pattern=externalId skal indeholde gyldigt CPR-nummer (10 tegn) eller CVR-nummer (8 tegn).  accessRequest.requestParticipant.externalId.cprLookup.timeout.small=Der er foretaget **for** mange fejlende CPR opslag. Yderligere forsøg er blokeret i {0} minutter.  accessRequest.requestParticipant.externalId.cprLookup.timeout.large=Der er foretaget **for** mange fejlende CPR opslag. Yderligere forsøg er blokeret i {0} minutter.  accessRequest.requestParticipant.externalIdType.required=Det skal angives om den eksterne nøgle er CPR-nummer eller CVR-nummer.  accessRequest.requestParticipant.externalIdType.invalid=Det skal angives om den eksterne nøgle er CPR-nummer eller CVR-nummer.  accessRequest.requestParticipant.firstNameProvided.required=Fornavn skal udfyldes.  accessRequest.requestParticipant.lastNameProvided.required=Efternavn skal udfyldes.  accessRequest.requestParticipant.firstNameProvided.size=Længden af fornavnet må ikke overstige 256 tegn.  accessRequest.requestParticipant.lastNameProvided.size=Længden af efternavnet må ikke overstige 256 tegn.  accessRequest.requestParticipant.emailAddress.required=E-mailadresse skal udfyldes  accessRequest.requestParticipant.emailAddress.invalid=Ugyldig e-mailadresse.  accessRequest.requestParticipant.emailAddress.size=Længden af e-mailadressen må ikke overstige 256 tegn.  accessRequest.requestParticipant.alias.size=Længden af alias må ikke overstige 256 tegn.  accessRequest.requestParticipant.position.size=Længden af stillingsbetegnelse må ikke overstige 256 tegn.  accessRequest.requestParticipant.mustBeCitizen=Typen af deltager være en borger i denne rolle  accessRequest.requestParticipant.mustBeOrganisation=Typen af deltager være en organisation i denne rolle  accessRequest.target.requestParticipant.required=requestParticipant skal udfyldes på target.  accessRequest.target.requestParticipant.mustMatchRequester=target skal matche requester.  accessRequest.accessTo.mustMatchRequester=accessTo skal matche requester.  accessRequest.privileges.npte.required=Ved USER\_ADMIN\_STATEMENT\_OF\_TRUTH\_PRIVILEGE\_REQUEST skal privilegierne være foruddefinerede NPTE privilegier: {0}  accessRequest.accessTo.mustMatchAppointments=Ved APPOINTED\_DELEGATION\_REQUEST skal accessTo matche allerede tildelte privilegiers scope.  accessRequest.illegalModification=Ugyldig redigering af anmodning foretaget.  accessRequest.rejectionReason.required=Begrundelse (rejectionReason) skal udfyldes.  accessRequest.revocationReason.required=Begrundelse (revocationReason) skal udfyldes.  accessRequest.userGroups.notAllowed=Brugergrupper kan kun benyttes på direkte tildelinger indenfor samme organisdation.  accessRequest.privilegesOrUserGroups.required=Privilegier og/eller brugergrupper skal udfyldes.  accessRequest.privileges.userAdmin.required=Privilegiet skal være ORGANISATION\_USER\_ADMINISTRATOR (og kun det).  accessRequest.privileges.special.required=Ved specialanmodninger (CURATOR, EXECUTOR\_OF\_ESTATE, LIQUIDATOR) skal privilegiet være én og kun én af de 3.  accessRequest.privileges.special.notAllowed=Special-privilegier (CURATOR, EXECUTOR\_OF\_ESTATE, LIQUIDATOR) er kun tilladt på specialanmodninger (SPECIAL\_PRIVILEGIE\_REQUEST) eller videre delegeringer (APPOINTED\_DELEGATION\_REQUEST).  accessRequest.privileges.required=Privilegier skal udfyldes.  accessRequest.privileges.legalOwnerOfInactiveOrClosedCompany.required=Ved LEGAL\_OWNER\_OF\_INACTIVE\_OR\_CLOSED\_COMPANY\_PRIVILEGE\_REQUEST skal privilegierne være alle foruddefinerede privilegier: {0}  accessRequest.accessTo.legalOwnerOfInactiveOrClosedCompany.hasCurator=Organisationen har kurator/bobestyrer tilknyttet.  accessRequest.privileges.delegatedSupportAdminPrivilegeRequest.required=Ved DELEGATED\_SUPPORT\_ADMIN\_PRIVILEGE\_REQUEST skal privilegierne være alle foruddefinerede privilegier: {0}  accessRequest.target.activation.code.required=Aktiveringskode skal udfyldes.  accessRequest.withdrawal.illegal=Anmodningen kan ikke trækkes tilbage når privilegierne/grupperne er tildelt.  accessRequest.termsApproval.notAuthorityAndNotCorrectIndustryCode=Organisationen skal være en myndighed eller have branche code 841100.  accessRequest.privileges.privilegeType.ORGANISATION\_USER\_ADMINISTRATOR=Rettighedsadministrator  accessRequest.privileges.privilegeType.LEGAL\_REPRESENTATIVE=Ekstern læseadgang  accessRequest.privileges.privilegeType.CURATOR=Kurator  accessRequest.privileges.privilegeType.LIQUIDATOR=Likvidator  accessRequest.privileges.privilegeType.EXECUTOR\_OF\_ESTATE=Bobestyrer  accessRequest.privileges.privilegeType.DANISH\_BUSINESS\_AUTHORITY\_SERVICE\_EMPLOYEE=Erhvervsservicemedarbejder  accessRequest.privileges.privilegeType.MESSAGE\_WRITE=Skriveadgang  accessRequest.privileges.privilegeType.MESSAGE\_EMPLOYEE=Avanceret adgang  accessRequest.privileges.privilegeType.MESSAGE\_BASIC=Basisadgang  accessRequest.privileges.privilegeType.ORGANISATION\_ADMINISTRATOR=Systemopsætning  accessRequest.privileges.privilegeType.ACCESS\_REQUEST\_ADMINISTRATOR=Anmodningsadministrator  accessRequest.privileges.privilegeType.ACTION\_LOG\_ADMINISTRATOR=Log **for** medarbejderhandlinger  accessRequest.privileges.privilegeType.SEARCH\_LOG\_ADMINISTRATOR=Log **for** søgninger  accessRequest.privileges.privilegeType.MESSAGE\_LOG\_ADMINISTRATOR=Log **for** meddelelser  accessRequest.privileges.privilegeType.STATISTICS\_ADMINISTRATOR=Statistikadministrator  accessRequest.privileges.privilegeType.SYSTEM\_MANAGER=Systemforvalter  accessRequest.privileges.privilegeType.CITIZEN\_SERVICE\_EMPLOYEE=Borgerservicemedarbejder  accessRequest.privileges.privilegeType.SUPPORT=Support adgang  accessRequest.privileges.privilegeType.TRUSTED\_RECIPIENT=Digital post - modtager  accessRequest.privileges.privilegeType.CONTACT\_ADMINISTRATOR=Kontaktstrukturadministrator  accessRequest.privileges.privilegeType.LEGAL\_OWNER=Ejer  accessRequest.privileges.privilegeType.COURTS\_OF\_DENMARK=Domstolsstyrelsen  accessRequest.privileges.privilegeType.LEGAL\_OWNER\_OF\_INACTIVE\_OR\_CLOSED\_COMPANY=Legal ejer  accessRequest.requestType.UNKNOWN=Ukendt  accessRequest.requestType.PRIVILEGE\_REQUEST=Privilegie anmodning  accessRequest.requestType.DELEGATION\_REQUEST=Delegations anmodning  accessRequest.requestType.APPOINTED\_DELEGATION\_REQUEST=Tildelt delegations anmodning  accessRequest.requestType.CONNECTION\_AGREEMENT\_REQUEST=Ny offentlig afsender  accessRequest.requestType.TERMS\_APPROVAL\_REQUEST=Underskrivelse af vilkår  accessRequest.requestType.USER\_ADMIN\_STATEMENT\_OF\_TRUTH\_PRIVILEGE\_REQUEST=Ny virksomheds rettighedsadministrator  accessRequest.requestType.USER\_ADMIN\_LOST\_PRIVILEGE\_REQUEST=Mistet rettighedsadministrator  accessRequest.requestType.SPECIAL\_PRIVILEGE\_REQUEST=Udpeget anmodning  accessRequest.requestType.LEGAL\_OWNER\_OF\_INACTIVE\_OR\_CLOSED\_COMPANY\_PRIVILEGE\_REQUEST=Legal ejer privilegie anmodning  accessRequest.requestStateType.UNKNOWN=Ukendt  accessRequest.requestStateType.DRAFT=Udkast  accessRequest.requestStateType.SUBMITTED=Indsendt  accessRequest.requestStateType.APPROVED=Godkendt  accessRequest.requestStateType.REJECTED=Afvist  accessRequest.requestStateType.REVOKED=Tilbagekaldt  citizenLookup.identity.notFound=Deltageren {0} kunne ikke identificeres.  citizenLookup.firstNameProvided.required=Fornavn skal udfyldes.  citizenLookup.lastNameProvided.required=Efternavn skal udfyldes.  organisationLookup.organisationId.notFound =Organisationen {0} kunne ikke identificeres.  citizenLookup.cprLookup.timeout.small=Der er foretaget **for** mange fejlende CPR opslag. Yderligere forsøg er blokeret i {0} minutter.  citizenLookup.cprLookup.timeout.large=Der er foretaget **for** mange fejlende CPR opslag. Yderligere forsøg er blokeret i {0} minutter. |

#### Folder Services

|  |
| --- |
| # Folder  folder.folderType.illegal=Standardmapper kan ikke oprettes.  folder.folderType.notNull=Mappens type skal udfyldes.  folder.name.notBlank=Navn på mappen skal udfyldes.  folder.create.too.many.folders=Der kan maksimalt oprettes {0} antal mapper.  folder.create.standard.folder.exists=En standardmappe af typen {0} eksisterer allerede.  folder.create.standard.folder.with.parent.not.allowed=En standardmappe må ikke være undermappe.  folder.createOrUpdate.name.exists.at.level=Der findes allerede en mappe der hedder {0} på dette niveau.  folder.createOrUpdate.too.many.levels=Antallet af mappe-niveauer må ikke over stige {0}.  folder.createOrUpdate.non.existing.parent=Overmappen {0} eksistererer ikke i postkassen.  folder.createOrUpdate.standard.parent.not.allowed=En standardmappe kan ikke have undermapper  folder.createOrUpdate.name.size=Antal tegn i feltet name må ikke overstige {0} tegn.  folder.createOrUpdate.name.illegal=Mappens navn må ikke indeholde en af følgende tegn: {0}  folder.update.standard.folder.may.no.be.updated=En standardmappe må ikke opdateres.  folder.delete.messages.exists=Mappen kan ikke slettes, da den indeholder meddelelser.  folder.delete.subfolders.exists=Mappen kan ikke slettes, da den indeholder undermapper.  folder.delete.standard.folders.may.not.be.deleted=Mappen kan ikke slettes, da det er en standardmappe. |

#### Message services

|  |
| --- |
| # Message  message.draft.sendercontactPoint.not.allowed=Afsenders kontakpunkt {0} må ikke have nogen værdi  message.reply.label=Sv: {0}  message.forward.label=Vs: {0}  message.create.reply.notAllowed=Denne meddelelse kan ikke besvares.  message.update.reply.sender.notNull=Feltet sender må ikke slettes **for** besvarelser.  message.update.reply.recipient.notNull=Feltet recipient må ikke slettes **for** besvarelser.  message.update.reply.sender.id.not.allowed=Feltet senderId må ikke opdateres **for** besvarelser.  message.update.reply.sender.id.type.not.allowed=Feltet senderIdType må ikke opdateres **for** besvarelser.  message.update.reply.recipient.not.allowed=Feltet recipient må ikke opdateres **for** besvarelser.  message.update.reply.label.not.allowed=Feltet label må ikke opdateres **for** besvarelser.  message.update.folderId.notNull=Mappe skal angives.  message.update.folderId.notFound=Den angivne mappe eksisterer ikke i postkassen.  message.update.legallyNotified.not.allowed=Kun forkyndelser kan markeres forkyndt.  message.update.folderType.not.allowed=En meddelelse i tilstand {0} kan ikke flyttes til mappe af typen {1}  message.create.draft.folderType.not.allowed=Meddelelse skal placeres i DRAFTS-mappen; ikke i {0}.  message.create.draft.messageType.not.allowed=Meddelelsestypen {0} er ikke tilladt her. Det skal være {1}.  message.create.draft.folderId.notFound=Den angivne mappe eksisterer ikke i postkassen.  message.create.forward.email.invalid=Den angivne email {0} er ikke valid.  message.create.forward.notAllowed=Denne meddelelse kan ikke videresendes.  message.create.forward.comment.size=Antal tegn i feltet {0} må ikke overstige {1} tegn.  message.create.forward.recipientId.notBlank=Modtagers id skal angives  message.create.forward.recipientId.email.invalid=E-mailadressen {0} er ugyldig.  message.create.forward.recipientId.cpr.invalid=CPR-nummeret {0} er ugyldigt  message.create.forward.recipientId.cvr.invalid=CVR-nummeret {0} er ugyldigt  message.create.forward.recipientIdType.notNull=Modtagertypen skal angives  message.create.forward.recipientIdType.invalid=Ugyldig modtagertype {0}. CPR/CVR/EMAIL er gyldige værdier.  message.create.forward.senderLabel.size=Antal tegn i feltet {0} må ikke overstige {1} tegn.  message.create.forward.recipientLabel.size=Antal tegn i feltet {0} må ikke overstige {1} tegn.  message.create.forward.size.exceeds.allowed.size=Total filstørrelse {0} overskrider tilladt størrelse til email-videresendelse: {1}.  message.maximum.number.of.additionalContentData.exceeded=Der kan maksimalt oprettes {0} antal additionalContentData.  message.document.actions.status.of.additionalActionStatusData.exceeded=Der kan maksimalt oprettes {0} antal additionalActionStatusData.  message.document.main.file.name=hoveddokument.html  message.document.main.file.encodingFormat=text/html  message.document.main.file.language=da  message.document.main.label=Hoveddokument  message.document.draftMessageRequired=Meddelelsen skal være en kladdemeddelelse **for** at kunne tilføje dokument.  message.document.number.higher.than.allowed=Grænsen **for** antal dokumenter der kan tilføjes beskeden er oversteget: {0}  message.file.encodingFormat.required=Format skal udfyldes - f.eks. 'text/html'.  message.file.encodingFormat.size=Feltet {0} må ikke overstige 256 tegn.  message.file.filename.required=Filnavn skal udfyldes.  message.file.filename.size=Feltet {0} må ikke overstige 256 tegn.  message.file.content.size=Filen er **for** stor.  message.file.content.infected=Virus detekteret. Filen kan ikke tilføjes.  message.file.language.required=Sprog skal udfyldes - f.eks. 'da'.  message.file.encoding.format.invalid=Det angivne format {0} er ikke tilladt **for** dokumenter af typen {1}. Tilladte formater: {2}  message.file.encoding.format.invalid.**switch**.to.html=encodingFormat kan ikke skiftes til text/html da det eksisterende indhold ikke er validt html.  message.file.name.invalid=Det angivne filtypenavn i {0} er ikke tilladt **for** filer med format {1}. Tilladte filtypenavne: {2}  message.file.draftMessageRequired=Meddelelsen skal være en kladdemeddelelse **for** at kunne tilføje fil.  message.file.number.higher.than.allowed=Grænsen **for** antal filer der kan tilføjes beskeden er oversteget: {0}  message.total.file.size.exceeds.allowed.size=Total filstørrelse overskrider tilladt størrelse: {0}  message.recipient.recipientIdType.invalid=Ugyldig modtagertype {0}. CPR/CVR er gyldige værdier.  message.recipient.recipientId.cpr.invalid=CPR-nummeret {0} er ugyldigt.  message.recipient.recipientId.cvr.invalid=CVR-nummeret {0} er ugyldigt.  message.recipient.recipientId.email.invalid=E-mailen {0} er ugyldig.  message.recipient.contactPoint.contactInfo.exceeds.max=Der kan højst angives 2 kontaktinformationer.  message.recipient.contactPoint.contactInfo.label.notBlank=Informationsfeltets navn skal angives.  message.recipient.contactPoint.contactInfo.value.notBlank=Informationsfeltets værdi skal angives.  message.send.label.required=Feltet label skal udfyldes ved afsendelse.  message.send.sender.label.required=Feltet sender.label skal udfyldes ved afsendelse.  message.send.invalid.folder=En meddelelse kan kun sendes fra 'Kladder'-mappen.  message.send.invalid.state=Kun kladder kan sendes. Denne meddelelse er en i tilstanden {0}.  message.send.documents.required=Der skal tilføjes et hoveddokument.  message.send.documents.main.exceeds.max=Der kan kun tilføjes et hoveddokument.  message.send.documents.files.required=Der skal tilføjes minimum en fil til et dokument.  message.send.documents.files.content.required=Der skal tilføjes indhold til filen {0}. |

#### System Fetch services

|  |
| --- |
| # SystemFetch  systemFetch.organisationId.notNull=Organisationens id skal udfyldes  systemFetch.contactPointId.notNull=Kontaktpunkt id skal udfyldes  systemFetch.systemFetchStatusType.invalid=Status typen må kun sættes til STOPPED  systemFetch.currently-running=Systemafhentning er i gang **for** postkassen  systemFetch.organisationId.notFound=Organisationen kan ikke findes.  systemFetch.organisationId.invalid=Organisation matcher ikke postkassen. Postkassse CVR:{0}. Organisation CVR: {1}".  systemFetch.organisationId.changed=Organisationen kan ikke skiftes.  systemFetch.systemId.notFound=Systemet kan ikke findes.  systemFetch.systemId.inactive=Systemet er ikke aktivt.  systemFetch.systemId.invalid=Systemet er ikke et modtagersystem.  systemFetch.contactPointId.notFound=Kontaktpunktet kan ikke findes.  systemFetch.contactPointId.inactive=Kontaktpunktet er ikke aktivt.  systemFetch.contactPointId.changed=Kontaktpunktet kan ikke skiftes på kørende Systemafhentning. Stop og start ny.  systemFetch.systemFetchStatusType.cannotStop=En FINISHED systemafhentning kan ikke stoppes.  systemFetch.queryString.maxLength=Antal tegn i søgefeltet må ikke overstige {0} tegn. |

#### Asynchronous distribution messages

If a message cannot be distributed, an error message will be generated in senders mailbox. It can contain the following errors:

|  |
| --- |
| # From distribution-validator  memo.infected=Virus detekteret i et bilag  recipient.not.found=Ukendt modtager  memo.invalid=Intern fejl  recipient.is.closed=Modtageren kan ikke modtage digital post  recipient.is.exempt=Modtageren er fritaget  recipient.contact.point.not.allowed.**for**.id.type=Kontaktpunkt kan ikke angives **for** denne modtager  recipient.mailbox.and.**default**.recipient.system.not.found=Modtageren kan ikke findex.  sender.not.found=Afsenderen (dig) kan ikke findes i systemet. |

#### File and document services

Initial validation of file attachments occur while creating and editing a draft through the view client. Files are HTML validated against a whitelist. This entails the following error codes in case of rejection:

|  |
| --- |
| # Html validator  html.validator.rejected=Filen {0} kunne ikke genkendes som et gyldigt html-dokument  html.validator.rejected.comments=Filen {0} indeholder kommentarer. Kommentarer er ikke tilladt.  html.validator.rejected.element=Filen {0} indeholder element \"{1}\", som enten ikke tilladt eller som indeholder data, der ikke er tilladt.  html.validator.rejected.element.attributes=Filen {0} indeholder element \"{1}\" med attribut \"{2}\", der enten ikke er tilladt attribut, eller som indeholder data, der ikke er tilladt.  html.validator.rejected.unknown-element=Filen {0} indeholder url i en ikke godkendt placering. Det er sandsynligvis i en style attribut. Kun data url'er er tilladt. |

#### Contact services

|  |
| --- |
| danish.mobile.number.invalid="{0}" er ugyldigt dansk mobilnummer  danish.mobile.number.needed=Dansk mobilnummer er påkrævet  exemption.end.update.not.allowed=Opdatering af fritagelses slutdato er ikke tilladt  exemption.start.invalid=Ugyldig startdato **for** fritagelse  exemption.after.voluntary.registration.not.allowed=Det er ikke tilladt at fritage sig selv efter frivillig tilmelding  invalid.exemption.start.or.end=Fritagelses start- og sluttidspunkt må ikke sættes  id.not.exist=ID findes ikke  registration.status.needed=Registreringsstatus er påkrævet  type.wrong="{0}" er den forkerte type  type.update.not.allowed=Typeopdatering er ikke tilladt  changed.date.update.not.allowed=Opdatering at status.changedDate er ikke tilladt  access.denied=Adgang nægtet  terms.id.missing=Ingen vilkår accepteret  terms.id.invalid=Ugyldige vilkår accepteret  removal.of.confirmedDateTime.not.allowed=Det er ikke tilladt at fjerne en nemSMS bekræftelse  **new**.confirmedDateTime.must.be.after.old.confirmedDateTime=En ny bekræftelse: {0} skal være nyere end den foregående: {1}  confirmedDateTime.can.not.be.in.the.future=Bekræftelse ugyldig:{0} en bekræftelse kan ikke være i fremtiden  cannot.confirm.nem.sms.**for**.nonexistent.verification=NemSMS nummeret skal verificeres før det kan bekræftes  terms.type.missing=Type **for** vilkår mangler at bliver angivet  terms.version.missing=Vilkårs version skal angives  voluntary.registration.status.closed=Frivillig registrering ikke tilladt **for** lukkede kontakter  voluntary.registration.age=Frivillig registrering ikke tilladt **for** borgere under 15  voluntary.registration.eligible=Frivillig registrering kun tilladt **for** borgere som er berettiget til frivilig registrering  voluntary.registration.active=Frivillig registrering kan ikke udføres **for** allerede registreret borger |

#### Identity services

|  |
| --- |
| clientdetails.clientSecretRequired.mandatory=Hemmelighed er påkrævet **for** klienter!  clientdetails.clientsecret.invalid=Den givne hemmelighed er ugyldig  clientdetails.clientsecret.mangled=Den angivne hemmelighed matcher delvist den eksisterende hemmelighed! Dette skyldes enten den nye hemmelighed ligner den gamle **for** meget, eller input er beskadiget pga. manglende håndtering af specialtegn.    grantee.identity.group.immutable=Gruppe reference er uforanderlig  grantee.identity.group.invalid=Invalid gruppe reference  grantee.identity.group.required=Gruppe reference er påkrævet  grantee.identity.group.**default**.immutable=DEFAULT-gruppe er uforanderlig  grantee.identity.invalid=Invalid reference til identitet  grantee.identity.required=Reference til identitet er påkrævet  grantee.issuer.invalid=Udstedende identitet er ugyldig  grantee.issuer.required=Udstedende identitet er påkrævet    identifier.empty=Minimum én identifikator er påkrævet  identifier.type.empty=Type er påkrævet  identifier.type.invalid=Type er ugyldig  identifier.type.unambiguous.constraint.violation=Utvetydige type(r): {0} overtræder maximum antal: 1 **for** én identitet  identifier.value.empty=Værdien er påkrævet  identifier.value.invalid=Værdien er ugyldig    identity.citizenName.invalid=Person navn er ugyldig  identity.invalid=Invalid identitet  identity.parent.id.invalid=Forældrereference er ugyldig  identity.parent.type.invalid=Forældretype er ugyldig  identity.parent.employee.invalid=Forældre CVR og medarbejder RID kombination er ugyldig  identity.type.empty=Type er påkrævet  identity.type.invalid=Type er ugyldig  identity.email.invalid=E-mailen er ugyldig  identity.email.empty=Identiteten har ikke registeret en e-mail som kan verificeres    identityGroup.issuer.invalid=Udstedende identitet er ugyldig  identityGroup.issuer.required=Udstedende identitet er påkrævet  identityGroup.name.required=Navn er påkrævet  identityGroup.owner.invalid=Ejer er ugyldig  identityGroup.owner.required=Ejer er påkrævet  identityGroup.type.invalid=Type er ugyldig    identityPrivilege.delegated.type.update.not.allowed=Delegeret privilegie understøtter ikke manuel redigering  identityPrivilege.group.immutable=Gruppe reference er uforanderlig  identityPrivilege.group.invalid=Gruppe er ugyldig  identityPrivilege.group.required=Gruppe er påkrævet  grantee.identity.group.parent.invalid=Gruppe skal have fælles parent ejer med parent-privilegiets gruppe  identityPrivilege.issuer.invalid=Udstedende identitet er ugyldig  identityPrivilege.issuer.required=Udstedende identitet er påkrævet  identityPrivilege.parent.id.invalid=Forældrereference er ugyldig  identityPrivilege.parent.scope.invalid=Forældreafgrænsning afviger fra afgrænsning  identityPrivilege.parent.source.invalid=Forældrekilde afviger fra kilde  identityPrivilege.parent.type.invalid=Forældretype afviger fra type  identityPrivilege.scope.invalid=Identitetsreference er ugyldig **for** privilegiets afgrænsning  identityPrivilege.scope.required=Identitetsreference er påkrævet **for** privilegiets afgrænsning  identityPrivilege.source.appointed.invalid=Kilde APPOINTED må ikke kombineres med parentPrivilegeId  identityPrivilege.source.invalid=Kilde er ugyldig  identityPrivilege.source.required=Kilde er påkrævet  identityPrivilege.type.invalid=Type er ugyldig  identityPrivilege.type.required=Type er påkrævet  identityPrivilege.type.invalid.scope.power-of-attorney=Modtager af ægte fuldmagt kan både være borger og virksomhed  identityPrivilege.type.invalid.grantee.full-power-of-attorney=Adgangshaver af fuld adgang kan kun være en borger eller virksomhed  identityPrivilege.type.invalid.type=Adgangsgiver kan højst have 10 læse- eller fulde adgange  identity.subscription.confirmationTime.removal.not.allowed=Bekræftelsestidspunkt må ikke fjernes  identity.subscription.confirmationTime.must.be.more.recent=Bekræftelsestidspunkt skal være nyere end det eksisterende  identity.subscription.confirmationTime.cannot.confirm.unverified.email=Verifikation skal gennemføres før e-mailadressen kan bekræftes igen    directPrivilege.grantee.scope.invalid=Modtager (grantee) skal være forskellig fra afgrænsning (scope)  directPrivilege.grantee.invalid=Modtager (grantee) er ugyldig  directPrivilege.grantee.required=Modtager (grantee) er påkrævet  directPrivilege.grantee.parent.invalid=Modtager (grantee) skal have fælles parent med parent-privilegiets grantee |

#### Verification services

|  |
| --- |
| verification.notification.email.subject=Pinkode - bekræft e-mailadresse i Digital Post  verification.notification.email.text=Du har opdateret din email adresse. Brug venligst denne PIN {0} til at bekræfte din email adresse  verification.notification.sms.text=Pinkode: {0} \n\nBekræft mobilnummer med pinkoden på borger.dk, Virk.dk, e-Boks.dk eller mit.dk. \n\nNår du har bekræftet, kan du modtage servicebeskeder (NemSMS) \n\nMed venlig hilsen \nNemSMS  verification.notification.sms.text.mailbox=Pinkode: {0} \n\nBekræft mobilnummer med pinkoden på borger.dk, Virk.dk, e-Boks.dk eller mit.dk. \n\nNår du har bekræftet, får du SMS om ny post. \n\nMed venlig hilsen \nDigital Post  verification.notification.sms.link.text={0}\n\nTryk på linket **for** at bekræfte din tilmelding til NemSMS\n\nMed venlig hilsen\nNemSMS    verification.max.attempts=Det maksimale antal forsøg er overskredet.  verification.validity.exceeded=PIN er udløbet.  verification.pin.invalid=Ugyldig PIN  verification.pin.is.present.**for**.non.verifying.state=Verifikations pinkode må kun være til stede **for** verifying tilstand  verification.not.in.verifying.state=Nuværende verifikation er ikke i verificerings-tilstand  verification.state.invalid=Tilstanden på verifikationen er ugyldig  verification.channel.**null**.or.blank=Værdien skal være udfyldt  verification.channelType.**null**.or.unknown=Værdien skal være udfyldt  verification.identityId.**null**=Værdien skal være udfyldt  verification.already.exists=Verifikation findes allerede  verification.is.not.verified=Verifikation er ikke i verificerings-tilstand  verification.linkToken.can.be.only.present.**for**.mobile=Link verifikationsflow kan kun startes **for** mobil tilmelding  verification.linkToken.is.present.**for**.non.verifying.state=Link token må kun være til stede **for** "verifying" tilstand  verification.invalid.linkToken=Token må ikke være tom  verification.invalid.flowType=Den angivne type af flow er udyldig  verification.link.token.cannot.be.present.**for**.pin.flow=Verifikations link ikke tilladt i pin verificerings flow  verification.pin.cannot.be.present.**for**.link.flow=Verifikations pin ikke tilladt i link verificerings flow |

#### Contact subscription services

|  |
| --- |
| contact.id.duplicated=Contact'en findes allerede i abonnement  contact.id.invalid=Kontakt id må ikke være blankt    url.invalid=Notifikationsadressen er ikke en gyldig URL |

#### System subscription services

|  |
| --- |
| cvr.duplicated=Et eller flere CVR nummer optræder flere gange  cvr.invalid=CVR nummeret er ugyldigt  url.invalid=Notifikationsadressen er ikke en gyldig URL |

#### System registry services

Error codes which can occour when creating or updating ContactPoints, ContactGroups, Systems and Organisations.

|  |
| --- |
| activefrom.needed.when.activeto.stated="aktiv fra" er påkrævet når "aktiv til" er angivet  activefrom.invalid="Aktiv fra skal være før {0}"  activefrom.required="Aktiv fra skal angives"  activeto.not.allowed="Et standard modtagersystem kan ikke inaktiveres, og derfor kan aktiv til ikke angives."  activeto.invalid={0} er ikke før {1}  active.invalid="Kontakt punktet kan ikke være aktivt da dets associerede modtager system ikke er aktivt."  cvr.number.not.exist="CVR {0} does not exist"  active.period.invalid={0, choice, 1#Det følgende systems aktive periode sammenfalder |1< De følgende systemers aktive periode sammenfalder}:{1} kun {2, choice, 1#1 aktivt modtager system|1<{1, number, integer}aktive modtager systemer} er tilladt  authority.type.not.allowed=Kun myndigheder må have en myndigheds type  authority.type.invalid="{0}" er en ugyldig myndigheds type  contact.groups.circular.dependencies=Kontakt grupper må ikke have cirkulær afhængighed  cvr.number.invalid="{0}" er et ugyldigt CVR nummer  delegated.cvr.does.not.exist=Fuldmagt kan ikke gives til et cvr-nummer der ikke findes  danish.phone.number.needed=Dansk mobilnummer er påkrævet  delegated.cvr.cannot.equal.owner.organization.cvr=Et system kan kun deleges til et andet CVR nummer end den organisation som systemet tilhører  danish.phone.number.invalid="{0}" er et ugyldigt dansk mobilnummer  email.address.needed=Email er påkrævet  email.address.invalid="{0}" er ugyldig email  externalLink.invalid="{0}" er et ugyldig URL  externalLinkText.needed=Eksternt link tekst påkrævet  endpoint.is.invalid="{0}" er et ugyldigt endepunkt  endpointCertificateType.is.invalid="{0}" er en ugyldigt certifikattype  field.value.is.needed=Værdi påkrævet  field.ip.range.not.singular=Når en IP addresseinterval angives må kun ét element angives  cannot.parse.ip=Kan ikke læse IP addresse  cannot.parse.ip.range=Kan ikke læse IP addresseinterval  ip.list.too.big=For mange IP addresser angivet (over "{0}")  ip.list.not.unique=En eller flere IP addresser er ikke unikke  invalid.ip.range.size=Mængden af IP addresser i addresseintervallet er **for** stor (over "{0}")  invalid.parent.group=Gruppen du forsøger at ligge "{0}" under er en ugyldig gruppe  invalid.system.type.delegated.cvr = "Only a sender / receiver system can be able to be delegated to another CVR number"  oces.**public**.certificate.validation.failed=Det uploadede certifikat fejlede valideringen med kode: {0}  oces.**public**.certificate.not.foces.or.voces=Det uploadede certifikat med subject serial number {0} er ikke et funktions- eller virksomhedscertifikat  oces.**public**.certificate.not.issued.by.nets=Det uploadede certifikat er ikke signeret med NETS rodcertifikat  oces.**public**.certificate.could.not.be.parsed=Det uploadede certifikat kunne ikke læses, er det et gyldigt certifikat?  nets.root.cert.not.found=Kunne ikke finde NETS rod certifikat, kan ikke validere uploadede certifikat  ssh.**public**.key.too.large=Den uploadede fil er **for** stor  ssh.**public**.key.missing=Den uploadede fil er tom  organisation.type.invalid="{0}" er en ugyldig organisations type  receiptEndpoint.is.invalid="{0}" er et ugyldigt kvitterings-endepunkt  sender.system.endpoint.is.not.**null**=Afsendersystemer kan ikke have et endepunkt  recipient.system.receipt.endpoint.not.**null**=Modtagersystemer kan ikke have et kvitterings-endepunkt  service.protocol.invalid.**for**.attaching.certificates=Et system med service protokol {0} kan ikke have et tilknyttet certifikat  service.protocol.invalid.**for**.attaching.ssh-keys=Et system med service protokol {0} kan ikke have en tilknyttet ssh-key  service.protocol.invalid.**for**.system.type=Et system med typen "{0}" kan ikke have service protokollen "{1}"  service.protocol.not.supported="{0}" er ikke en understøttet service protokol  technical.contact.required.**for**.standard.system.template=Teknisk kontaktperson skal udfyldes  service.protocol.sft.missing.ip=IP er påkrævet ved brug af service protokollen SFTP  targets.type.needed=Målgruppe er påkrævet  logo.organisationType.invalid=Logo er kun tilladt **for** myndigheder  logo.contentType.invalid="{0}" er ikke en understøttet filtype. Tilladt type er "image/png". Sørg **for** at filen ender på .png.  logo.fileSize.invalid=Filstørrelsen på {0} bytes skal være mellem {1} og {2} bytes  logo.dimensions.invalid=Filens dimensioner skal være kvadratiske og minimum {0}px, {1}px  logo.file.invalid=Filen kan ikke læses. Fejl: {0}  receiptEndpoint.not.empty=Modtagersystemer med service protokol REST\_PULL kan ikke have et kvitterings-endepunkt  endpoint.not.**null**=Modtagersystemer med service protokol REST\_PULL kan ikke have et endepunkt  code.version.is.needed=Klassifikations version skal angives **for** kontakt punkt med kode type: {0}  contact.point.code.type.exist=Kode typen: {0}, eksisterer allerede **for** dette kontakt punkt  contact.point.code.type.custom.limit.exceeded=Det er ikke tilladt at have {0} af kontakt punkt kode typen CUSTOM  system.api.token.renew.not.allowed=Systemer med et certifikat må ikke opdatere deres API Token  contact.group.delete.subgroup.exists=Kontaktgruppen kan ikke slettes, da den har undergrupper  contact.group.delete.related.contact.point.exists=Kontaktgruppen kan ikke slettes, da den har et eller flere kontaktpunker hørende til gruppen  post.kasse.id.invalid=Postkasse id: {0}, tilhører ikke organisationen med id: {1}  post.kasse.emne.id.invalid=PostkasseEmne id: {0}, tilhører ikke postkasse id: {1}  pair.postkasseid.postkasseemneid.invalid= Pair of postkasseId: {0} and postkasseEmneId: {1}, is not unique.  post.kasse.id.already.used=Postkasse id: {0}, er tilnyttet på kontakt gruppe med id: {1}  system.has.contact.point=System id: {0} har kontaktpunkt tilknyttet: {1}  notification.email.system.is.**default**.recipient.and.sender=Afsender- og modtagersystem (Primærsystem)  notification.email.system.is.recipient.and.sender=Afsender- og modtagersystem  notification.email.system.is.**default**.recipient=Modtagersystem (Primærsystem)  notification.email.system.is.recipient=Modtagersystem  notification.email.system.is.sender=Afsendersystem  notification.email.system.is.unknown=Ukendt  notification.email.standard.system.inheritance.subject=Ny tilslutning til standardsystem  notification.email.standard.system.inheritance.property.is.**null**=Ikke valgt  notification.email.standard.system.closed.subject=Orientering om inaktiv systemleverandør  notification.email.standard.system.closed.content.system.type.is.**default**.recipient.and.sender=Det betyder, at jeres aktive primærsystem ''{0}'' ikke vil blive vedligeholdt. I kan dermed risikere, at systemet på et tidspunkt ikke længere kan sende og modtage jeres post korrekt.  notification.email.standard.system.closed.content.system.type.is.recipient.and.sender=Det betyder, at jeres aktive system ''{0}'' ikke vil blive vedligeholdt. I kan dermed risikere, at systemet på et tidspunkt ikke længere kan sende og modtage jeres post korrekt  notification.email.standard.system.closed.content.system.type.is.**default**.recipient=Det betyder, at jeres aktive primærsystem ''{0}'' ikke vil blive vedligeholdt. I kan dermed risikere, at systemet på et tidspunkt ikke længere kan modtage jeres post korrekt.  notification.email.standard.system.closed.content.system.type.is.recipient=Det betyder, at jeres aktive system ''{0}'' ikke vil blive vedligeholdt. I kan dermed risikere, at systemet på et tidspunkt ikke længere kan modtage jeres post korrekt.  notification.email.standard.system.closed.content.system.type.is.sender=Det betyder, at jeres aktive system ''{0}'' ikke vil blive vedligeholdt. I kan dermed risikere, at systemet på et tidspunkt ikke længere kan sende jeres post korrekt. |

#### Push notification settings error codes

These error codes can be encountered when creating/updating Settings in the push-notification-settings-store (see *”Push notification integrations”*). They’re mostly for a few values for Firebase Cloud Messaging settings, where Google expects values in a certain format / value range. We refer to the official documentation to ensure creating well-formed settings: <https://firebase.google.com/docs/reference/fcm/rest/v1/projects.messages>

|  |
| --- |
| access.denied=Adgang nægtet  field.update.not.allowed=Dette felt må ikke opdateres  identity.does.not.exist=Den angivne identitet findes ikke  value.is.not.a.rrggbb.value=Den angivne værdi er ikke på formatet "#rrggbb", eksempelvis #0fab88  value.is.not.a.rgba.value=Den angivne værdi er ikke ordentligt json objekt med nøglerne 'red', 'green', 'blue', 'alpha', og værdi mellem 0 og 1  value.is.between.zero.and.one=Den angive værdi er ikke mellem 0 og 1 (inklusiv) |

### Back-end validation and error codes in distribution

This section describes the back-end error codes, that are returned when a message fails to be distributed. If sending with the MeMo format these error codes are returned asynchronously, either as Receipts to sender systems, or as error messages in the mailbox.  
If nothing is mentioned, the error codes are returned in business receipts for messages received via Digital Post via all protocols.

### Business receipt error codes

#### Validation

|  |
| --- |
| memo.infected=Antivirus scan found threats  memo.invalid={0}  **do**.not.deliver.until.date.too.early='Do not deliver until date' can not be in the past  **do**.not.deliver.until.date.too.late='Do not deliver until date' is too late. Maximum number of days allowed is {0}  message.uuid.does.not.match.file.name=The MessageUUID {0} does not match the UUID in the filename {1}  message.uuid.not.unique=The MessageUUID {0} is invalid. MessageUUID must be a unique UUID  file.name.invalid=Filename {0} is invalid. The format of the filename should be '{'UUID'}' or '{'UUID'}'.xml  contact.point.id.format.not.allowed={0} contactPointID {1} invalid. Expected format UUID  notification.length.over.limit=Notification can not be longer than {0}  empty.notification.not.allowed=Empty notification is not allowed **for** MeMo of type NEMSMS    # Recipient  recipient.is.closed=Recipient with {0} {1} is {2}  recipient.is.exempt=Recipient with {0} {1} is exempt  recipient.nem.sms.is.not.allowed=Recipient of type {0} can not receive nem sms messages  recipient.contact.point.id.required=Contact point must contain a contact point id  recipient.contact.point.not.allowed.**for**.id.type=Contact point not allowed **for** recipient with id type {0}  recipient.cpr.invalid=The format of the cpr number: {0} is incorrect  recipient.cvr.invalid=The format of the cvr number: {0} is incorrect  recipient.not.found={0} with {1} {2} does not exist  recipient.nem.sms.subscription.not.found=Recipient with {0} {1} does not have a nem sms subscription  recipient.nem.sms.subscription.mobile.number.not.verified=Recipient with {0} {1} has not verified the mobile number {2}  recipient.mailbox.not.found=Recipient with {0} {1} does not have a mailbox  recipient.mailbox.and.**default**.recipient.system.not.found=Recipient with {0} {1} does not have a mailbox or **default** recipient system  recipient.type.cannot.receive.legal.notifications=Recipient is of type {0} and therefore cannot receive legal notifications.    # Sender  sender.not.found={0} with {1} {2} does not exist  sender.organisation.id.does.not.match=The sender organisation in the message does not match {0} which was resolved when the message was received  sender.cpr.invalid=The format of the cpr number: {0} is incorrect  sender.cvr.invalid=The format of the cvr number: {0} is incorrect  sender.system.not.found=The sender system {0} which was resolved when the message was received does not exist on the organisation {1}  sender.system.is.not.activated=The sender system {0} has not been activated yet. The activation date of the system is {1}  sender.system.is.deactivated=The sender system {0} was deactivated at {1}  sender.mandatory.message.not.allowed=Sender is not allowed to send mandatory messages  sender.legal.notification.not.allowed=Sender is not allowed to send legal notifications  sender.type.not.allowed=Only authorities can send messages to recipients of type {0}  sender.**do**.not.deliver.until.date.not.allowed=Senders of type {0} are not allowed to send messages with a 'do not deliver until date'  sender.system.forward.not.allowed=Sender systems may not forward messages through Digital Post  id.type.invalid=Invalid {0} id type {1}    # Files  file.format.not.allowed=File encodingFormat(s) {0} **for** one or more files in {1} document not allowed. Only the following are allowed **for** **this** type of document: {2}  file.extension.not.allowed=One or more invalid file exentions in one or more files is not allowed: {0}  memo.file.size.too.large=File size of memo is too large. Allowed file size is {0} bytes.  file.empty.not.allowed=One or more of the attachments in the message are empty    # Html validator  html.validator.rejected=Filen {0} kunne ikke genkendes som et gyldigt html-dokument  html.validator.rejected.comments=Filen {0} indeholder kommentarer. Kommentarer er ikke tilladt.  html.validator.rejected.element=Filen {0} indeholder element \"{1}\", som enten ikke tilladt eller som indeholder data, der ikke er tilladt.  html.validator.rejected.element.attributes=Filen {0} indeholder element \"{1}\" med attribut \"{2}\", der enten ikke er tilladt attribut, eller som indeholder data, der ikke er tilladt.  html.validator.rejected.unknown-element=Filen {0} indeholder url i en ikke godkendt placering. Det er sandsynligvis i en style attribut. Kun data url'er er tilladt.    # File and Document number validator  message.document.number.higher.than.allowed=The limit **for** the number of documents that can be added to the message has been exceeded: {0}. Limit is {1}.  message.file.number.higher.than.allowed=The limit **for** the number of files that can be added to the document \"{0}\" has been exceeded: {1}. Limit is {2}. |

#### Transformation

|  |
| --- |
| dp.invalid={0}  attention.invalid={0}  dp.vedhaeftning.indhold.data.required=Feltet vedhaeftningIndholdData skal være udfyldt **for** vedhaeftninger  dp.vedhaeftning.fil.format.required=Feltet filFormat skal være udfyldt **for** vedhaeftninger  dp.meddelelse.indhold.data.required=Feltet meddelelseIndholdData skal være udfyldt  dp.meddelelse.fil.format.navn.required=Feltet filFormat skal være udfyldt  dp.service.besked.meddelelse.indhold.data.must.be.plain.text=Feltet filFormat skal være txt **for** service besked  dp.filformat.unknown=Ukendt filFormat {0}  dp.materialeId.mapping.not.found=Der blev ikke fundet nogen værdi **for** felt {0} **for** materialeId {1}  dp.materialeId.mapping.not.found.and.defaultMaterialeId.not.set=Default indholdstype ikke opsat på system  transformed.memo.invalid=Der kunne ikke genereres en valid MeMo ud fra DP/DP2 beskeden. Fejlbesked: {0}  dp.not.allowed= Manglende rettighed til kald af operation  dp.postkasse.id.not.found=PostkasseId må ikke være tom  dp.materialeId.not.found=Materiale {0} eksisterer ikke  system.id.invalid=Det angivne systemId matcher ikke systemId'et fra API nøglen  dp.receipt.list.not.found=Listen af kvitteringer kan ikke findes  meddelelse.id.not.equal=Det angivne meddelelsesId i beskedens data matcher ikke meddelelsesId'et i den angivne url-request |

#### Dp/Dp1 transform error codes

|  |
| --- |
| **do**.not.deliver.until.date.too.early=2001  **do**.not.deliver.until.date.too.late=6004  recipient.is.closed=6003  recipient.is.exempt=4090  recipient.nem.sms.is.not.allowed=6003  recipient.not.found=4007  recipient.nem.sms.subscription.not.found=6003  recipient.nem.sms.subscription.mobile.number.not.verified=6003  recipient.mailbox.not.found=4007  recipient.mailbox.and.**default**.recipient.system.not.found=4007  sender.mandatory.message.not.allowed=3002  memo.file.size.too.large=2002  dp.invalid=2001  dp.neither.cpr.nor.cvr.given=4018  dp.both.cpr.and.cvr.given=4019  dp.cpr.invalid=4042  dp.cvr.invalid=4043  dp.indholdstype.not.found=4012  dp.meddelelse.tidsfrist.data.is.empty=4063  dp.vedhaeftning.navn.too.**long**=4069  dp.meddelelse.titel.tekst.too.**long**=4071  dp.afsendelse.advisering.mail.tekst.too.**long**=4120  dp.not.allowed=3002  dp.vedhaeftning.indhold.data.required=4052  dp.meddelelse.indhold.data.required=4052  dp.postkasse.id.not.found=4016  dp.contact.point.id.not.found=4020  dp.materialeId.not.found=4059  dp.materialeId.mapping.not.found=4059  dp.materialeId.mapping.not.found.and.defaultMaterialeId.not.set=4005  message.neither.encrypted.nor.signed=9001  wrong.certificate=3004  invalid.certificate=3004  dp.tilmeldingslisteid.not.found=4032  dp.dellisteid.not.found=4033  dp.dellist.invalid=4034  dp.receipt.list.not.found=2030  system.id.invalid=3001 |

#### Extracting memos from archive

|  |
| --- |
| archive.processing.failed=An error occurred **while** processing the archive: {0}  no.archive.entry=No archive entry could be found in the file  file.name.uuid.is.not.valid=The file name {0} does not contain a valid UUID |

### Technical receipt error codes

#### SMTP messages

The following error codes can be returned in technical receipts for messages sent via SMTP

|  |
| --- |
| wrong.certificate=Serial number of certificate used to sign message with subject {0} does not match the certificate serial number **for** sender system with id {1}  system.not.found.id=No system found with id {0}  system.not.found.authenticationToken=No system found with authentication token {0}  mail.decryption.failed=Failed to decrypt mail  mime.message.not.signed=Mime message with subject {0} was not signed.  mime.message.not.encrypted=Mime message with subject {0} was not encrypted.  mime.invalid.api.key=Mime header with name {0} has an invalid value: {1}  mime.missing.api.key=Missing required mime header {0}  no.certificate.found=Could not find certificate on system with systemId {0}.  invalid.signature.certificate=Certificate used to sign message with subject {0} from organisation with organisationId {1} is invalid. The mail should be encrypted with NgDP certificate and signed with sender certificate.  invalid.encryption.certificate=Certificate used to encrypt message with subject {0} is invalid. The mail should be encrypted with NgDP certificate.  message.neither.encrypted.nor.signed=Mime message with subject {0} was neither encrypted nor signed.    no.message.attachment=No application/xml attachment found on message with subject {0}  no.receipt.attachment=No application/json attachment found on message with subject {0}  file.name.uuid.is.not.valid=The file name {0} does not contain a valid UUID  system.wrong.type=Wrong protocol type {0} or receipt format {1} of a system |

#### Recipient-system error codes

This section describes the errorCodes that Recipient-systems may set in the Business Receipt they send to DP upon receiving (REST\_PUSH) or fetching (REST\_PULL/REST\_PUBLISH\_SUBSCRIBE) MeMos from the solution.

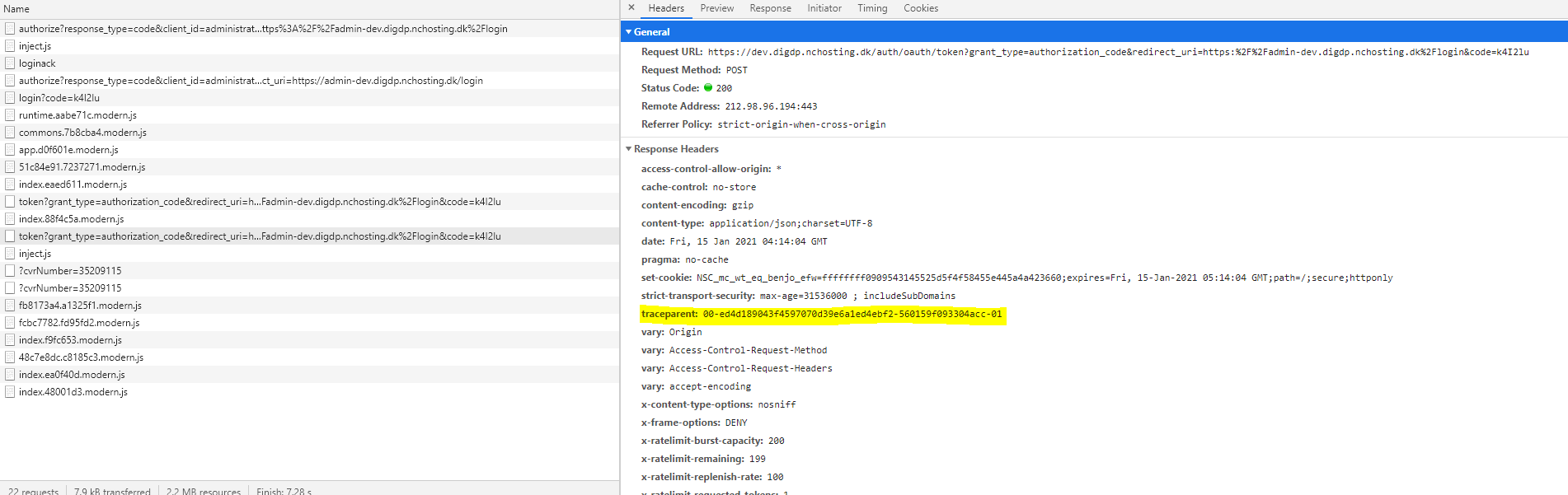
##### Business receipt errorcodes

|  |
| --- |
| virus.detected=Virus fundet i modtaget payload **for** MeMo med id {0} |

### Tracing requests using W3C headers

Digital Post provides support for tracing HTTP requests via [W3C Trace Context specification](https://www.w3.org/TR/trace-context/#overview) for external parties. In summary, the following HTTP headers are available from the responses returned from request:

* traceparent : This HTTP header field identifies the incoming request in a tracing system. See the [RFC](https://www.w3.org/TR/trace-context/#traceparent-header) for details. **This is the header that will be used for external request tracing**. See screenshot for example:



* tracestate : This HTTP header is to provide additional vendor-specific trace identification information across different distributed tracing systems and is a companion header for the traceparent field. See [RFC](https://www.w3.org/TR/trace-context/#tracestate-header) for details.

|  |
| --- |
| The tracestate header is currently missing from the current implementation but it is NOT used in tracing requests, therefore, it is safe to ignore this header. |

To request support, external parties can provide the traceparent header from the response to their requests. The value of this header is in the format similar to this example: 00-0af7651916cd43dd8448eb211c80319c-b7ad6b7169203331-01

# Java/.Net Core, Security perspective, MeMo-lib and Test

## Reference Systems for Java and .Net Core

### Overall description and purpose

The purpose of the Reference Systems built for Java and .Net Core is to provide *code by example* by providing reference implementations of sender- and recipient-systems for Digital Post (DP). Authorities and developers can use the examples in the Reference Systems to gain an overall understanding of how these implementations can be programmed, and what systems must be able to handle in the interaction with DP.

The reference implementation contains examples of sending- and/or receiving MeMos as well as receipts across these protocols:

* REST\_PUSH and REST\_PUBLISH\_SUBSCRIBE (publish/subscribe using long-polling)
* SFTP (sender-system only)
* SMTP

### Supported platforms

The Reference Systems have been built for both Java and .Net Core, and made available as public Bitbucket repositories. Naturally, there are some differences between the Java and .Net Core versions, but there has been an emphasis on code-reuse and streamlining the implementation as much as possible to reduce the scope of differences. The available versions can be found here:

* Java version (<https://bitbucket.org/nc-dp/reference-systems-for-java/src/master/> )
* .Net Core version (<https://bitbucket.org/nc-dp/reference-systems-for-dotnet/src/master/> )

### Application architecture

The application architecture for both versions of the Reference Systems involves separating responsibilities and protocols into individual sub-modules.

Each sub-module can act as an independent runnable application to trigger specific flows. (The .Net pendant to sub-module is a “Project” within a “Solution”).

#### Sub-modules (Java name / .Net Core name)

##### ssl-client / SSLClient

Is used for authentication and used for handling mutual SSL handshake between the Reference Systems and DP. The client allows one to use a certificate in combination with an API-key to ensure validation.

##### utility-library / UtilityLibrary

Provides model and service resources for MeMos and receipts, handling the creation, parsing and logging of MeMos as well as the creation, sending and logging of positive or negative receipts.

##### system-rest-push / RestPush

Reference implementation of REST\_PUSH protocol sender- and recipient-system. Showcases how REST requests to DP can be implemented, utilizing the RestClient provided by ssl-client - and how the recipient-systems can provide an endpoint for receiving MeMos from DP - as well as the processing of these received MeMos and the creation and sending of Business Receipts back to DP.

##### system-rest-publish-subscribe / RestPublishSubscribe

Reference implementation of REST\_PUBLISH\_SUBSCRIBE protocol recipient-system. Showcases how REST calls can be made to DP to fetch information about MeMos currently available for fetching, as well as fetching these one by one from DP. Also contains example code to create and sends back Business Receipts to DP.

##### system-smtp / Smtp

Reference implementation of SMTP protocol sender- and recipient-system. Showcases how a scheduler can be configured to poll a mail server for new entries, and trigger the MemoFetcherService which fetches and processes mails containing MeMos. Also contains example code which creates, maps and sends back Business Receipts to DP.

As MeMos and receipts must be mapped to the MimeMessage format for SMTP flows, this sub-module also contains SMTP specific mapping logic in MeMoToMimeMapper and ReceiptToMimeMapper.

##### system-sftp / Sftp

Reference implementation of SFTP protocol sender-system. Showcases how created MeMos can be bundled to a TAR.LZMA file and uploaded to DP’s SFTP server - as well as downloading available receipts from the SFTP server.

### Resources and webinars

More information as well as webinars covering some of the examples in the Reference Systems can be found here: <https://digst.dk/it-loesninger/naeste-generation-digital-post/for-myndigheder-og-it-leverandoerer/for-it-leverandoerer/referenceimplementeringer-og-memo-lib/>

### REST protocol examples

#### Overall description

The Reference Systems REST examples are configured to represent an organisation integrated to Digital Post (DP) with a REST\_PUSH protocol sender- and recipient-system (system-rest-push sub-module) and a REST\_PUBLISH\_SUBSCRIBE recipient-system (system-rest-publish-subscribe sub-module).

For the Reference Systems REST protocol examples, the two endpoints of sender- and recipient-systems are exemplified by exposed RestController endpoints that provide examples of how MeMos and receipts can be received and processed:

* Handling of receipts being sent to the *receiptEndpoint* of a system is exemplified by an endpoint exposed by the Reference Systems application, which can process the receipt and print relevant information to the console.
* Handling of a validated and processed MeMo being sent to the *endpoint* of a system is likewise exemplified by an endpoint exposed by the Reference Systems application, which can process the MeMo and print relevant information to the console.

#### Purpose

It is the aim of the Reference Systems REST protocol examples to provide insight into the interaction between sender- and recipient-systems and DP, with extensive console logging and in-code commentary utilized to describe each process. This is achieved through example code showcasing how MeMos and receips can be created and sent as well as received and processed.

The examples for REST\_PUSH can be found within the Reference Systems repositories here:

* Java: <https://bitbucket.org/nc-dp/reference-systems-for-java/src/master/system-rest-push/>
* .Net Core: <https://bitbucket.org/nc-dp/reference-systems-for-dotnet/src/master/RestPush/>

And for REST\_PUBLISH\_SUBSCRIBE:

* Java: <https://bitbucket.org/nc-dp/reference-systems-for-java/src/master/system-rest-publish-subscribe/>
* .Net Core: <https://bitbucket.org/nc-dp/reference-systems-for-dotnet/src/master/RestPull/>

##### REST flow

The Reference Systems contains example implementations of the 4 primary REST steps, described below.

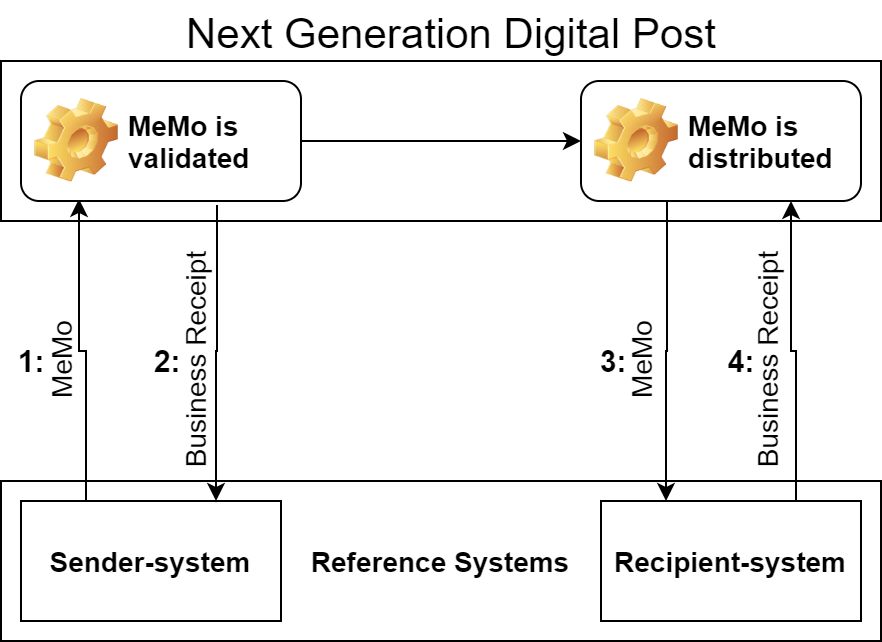
1. The sender-system sends a MeMo, e.g to a recipient-system. DP receives the MeMo and returns a Technical receipt in the response body.
2. DP validates and processes the MeMo and sends back a business receipt for the sender-system (for REST\_PUSH), to notify whether this step was successful. For REST\_PULL the business receipt is made available.
3. If successful, DP distributes the MeMo to the (in this example) recipient-system (REST\_PUSH), or the recipient-system fetches the MeMo from DP (REST\_PUBLISH\_SUBSCRIBE or REST\_PULL).
4. Recipient-system creates and sends back a Business Receipt to DP to notify whether it successfully received the MeMo (and DP uses this information to delete the MeMo from internal storage).

In the implementation of the Reference Systems, observing the steps of the REST protocol flows can be initiated by running either the system-rest-push or system-rest-publish-subscribe applications.

Running the system-rest-push application will create a single MeMo as well as a tar.lzma of 3 MeMos, and send these to DP, with the recipient being configurable. This mimics the process of sending MeMos as a sender-system, as DP will handle everything from there onwards (given that the MeMo adheres to the agreed format and that the recipient is reachable).

Running the system-rest-publish-subscribe application will start the process of the REST\_PUBLISH\_SUBSCRIBE recipient-system fetching a list of available MeMos and attempting to fetch each of these from DP.

A simplified representation of the Reference Systems application and how it showcases the interactions that sender- and recipient-systems will have with DP is presented below. It exemplifies the usage of receipts to communicate whether actions where successful or not.



#### Reference REST sender-system

The following section will describe the two primary REST protocol interactions between sender-systems and DP. They are:

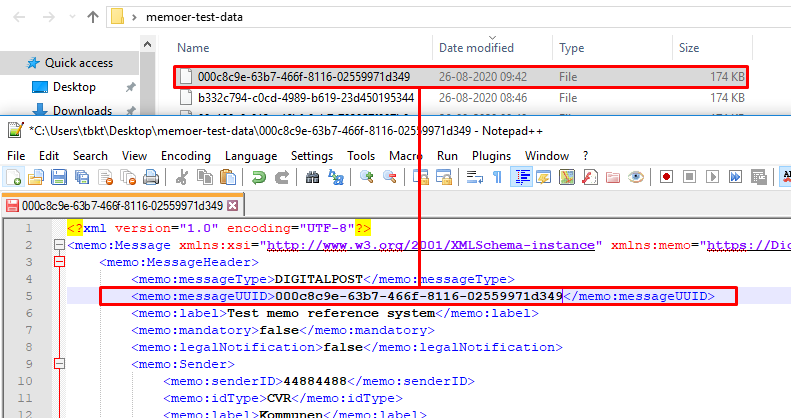
1. Sending MeMo
2. Receiving Receipt

##### Sending MeMo

The Reference sender-system can send a MeMo intended to the DP test environment. The DP endpoint is {environmentUrl}/memos/, exposed at the following path on e.g. test01: <https://api.test.digitalpost.dk/apis/v1/memos/>. The endpoint consumes a Resource - in the format of either:

* A tar.lzma file with content-type ‘application/x-lzma’ containing XML files
* An XML file with content-type ‘application/xml’

Additionally, the name of the XML files should be the messageUUID from the MeMo. The .xml extension is optional. See picture below:



In the case of the Reference Systems application, the service is able to create MeMo messages that adhere to these rules, which besides actively being sent by the Reference sender-system, will also give insight into the structure and contents of a MeMo.

For an in-depth overview of inbound REST services, see chapter *Inbound services* in *Digital Post - Technical Integration* for more information.

###### Initiating flow

Sending a MeMo is doable by running the appropriate application within the Reference System application. Running the system-rest-push application will create a single MeMo separately as an XML as well as a tar.lzma containing 3 MeMos, and proceed to send these. The system-rest-push sub-module utilizes a service in the utility-library sub-module to create MeMos programmatically by utilizing memo-lib. Running this application also exposes the REST\_PUSH recipient-system endpoint which provides an example to how MeMos can be received.

##### Receiving receipts from DP

Immediately upon receiving the MeMo, DP will send back a Technical Receipt. If this step was successful, DP will validate and process the MeMo and send a Business Receipt back to the sender-system at the *receiptEndpoint* URL. The sender-system responds with a HTTP Status code to notify DP of successful or unsuccessful delivery of the receipt. The Reference Systems contain example code showcasing this interaction. This Business Receipt contains information on the status of DP’s processing of the MeMo. (see chapter 4.9.3 *Receipt domain model* and chapter 4.9.4 *REST receipts* in *Digital Post - Technical Integration* for more information).

Specifically for the Reference sender-system example, the receipts from DP will be logged to the console for an overview of its content and status. A positive Business Receipt to the sender-system notifies that the MeMo is now considered the responsibility of DP, and this stage marks the end of interaction between the sender-system and DP for the respective MeMo.

#### Reference REST recipient-system

##### Receiving MeMo

There are two ways in which a REST protocol recipient-system can receive MeMos.

###### REST\_PUSH

If the recipient of a MeMo is a REST\_PUSH recipient-system, DP will send MeMos to the recipient-system endpoint as soon as these have been validated by DP. An example of this is exposed by a RestController in the system-rest-push sub-module, which will initiate the processing of any received MeMo from DP.

###### REST\_PUBLISH\_SUBSCRIBE

If the recipient of a MeMo is a REST\_PUBLISH\_SUBSCRIBErecipient-system, DP will not send MeMos immediately upon validation, but instead send a notification to the recipient-system, allowing the recipient-system to fetch them when needed. An example of this functionality is implemented in the system-rest-publish-subscribe sub-module, where the recipient-system can call two DP endpoints: One for fetching a list of available MeMos - and one for fetching each available MeMo in this list.

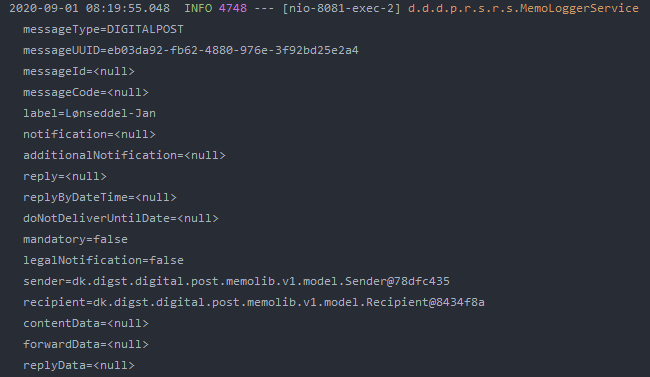
##### Processing MeMo

When the MeMo has been received through either REST\_PUSH or REST\_PUBLISH\_SUBSCRIBE, the following process of parsing it and sending back Business Receipts is identical.

The Reference System examples implements functionality of the publicly available MeMo-lib, by utilizing a parser to parse the MessageHeader from the received MeMo.

Relevant information from the MeMo MessageHeader will then be logged to the console for an overview of its content.

Draft example shown below of MeMo MessageHeader parsed and logged to console:

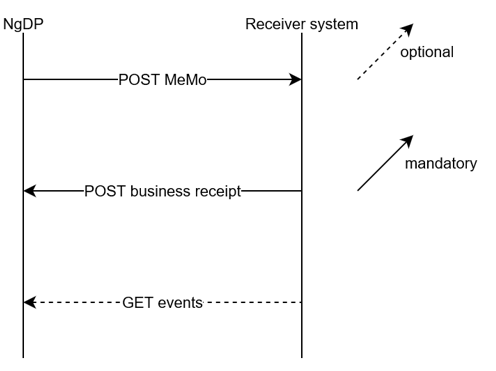


##### Sending Business Receipt to DP

When the Reference recipient-system has parsed the received MeMo, it will send a Business Receipt back to DP. The memoId is a PathVariable, which must be the UUID of the MeMo. This informs DP which specific MeMo the Business Receipt is a response to.

The Reference System will automatically create a Business Receipt if it receives a MeMo, and send it to the correct endpoint. If the Reference System application encounters an error in the parsing of the MeMo MessageHeader, this error will populate the ErrorMessage field of the Business Receipt.

Thus, the Business Receipt as built by the Reference System application can be either positive or negative, dependent on (in the context of the Reference System application) whether the MeMo sent from DP was passable.



Upon receiving a positive Business Receipt from a recipient-system, DP will delete the MeMo from internal storage.

### SMTP protocol examples

#### Overall description

The Reference Systems SMTP examples are configured to represent an organisation integrated to Digital Post (DP) with a SMTP protocol sender- and recipient-system.

For the Reference Systems SMTP examples, these two endpoints of sender- and recipient-systems are exemplified by e-mail addresses point to a test email where MeMos and receipts can be received and processed:

* Handling of receipts being sent to the *receiptEndpoint* of a system is exemplified by a test email client, which the Reference Systems can fetch from and process the receipt and print relevant information to the console.
* Handling of a validated and processed MeMo being sent to the *endpoint* of a system is likewise exemplified by a test email client, which the Reference Systems can fetch from and process the MeMo and print relevant information to the console.

#### Purpose

It is the aim of the Reference Systems SMTP protocol examples to provide insight into the interaction between sender- and recipient-systems and DP, with extensive console logging and in-code commentary utilized to describe each process. This is achieved through example code showcasing how MeMos and receipts can be created and sent as well as received and processed.

The examples for SMTP can be found within the Reference Systems repositories here:

* Java: <https://bitbucket.org/nc-dp/reference-systems-for-java/src/master/system-smtp/>
* .Net Core: <https://bitbucket.org/nc-dp/reference-systems-for-dotnet/src/master/Smtp/>

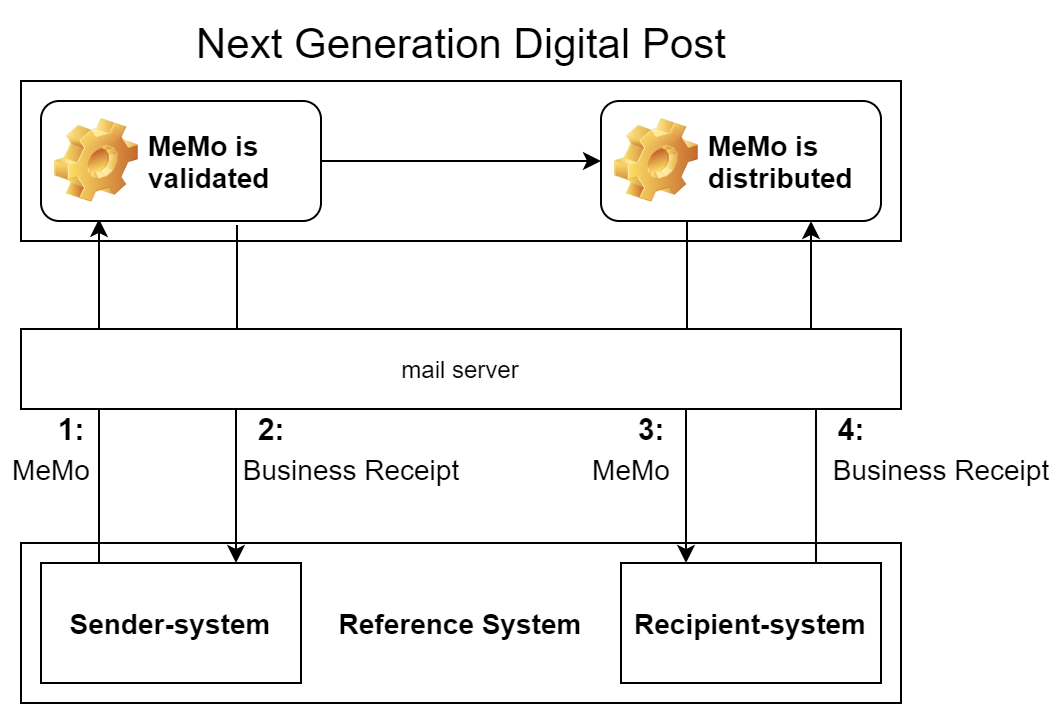
##### SMTP flow

The Reference Systems contains example implementations of the 4 primary SMTP steps, described below.

1. The sender-system sends a MeMo in the MimeMessage format, e.g to a recipient-system. This MeMo is signed with the sender-system certificate and the sender-system Api-key is added as a header - it is also encrypted with DP public certificate.
2. DP validates and processes the MeMo and sends back a receipt for the sender-system, to notify whether this step was successful.
3. If successful, DP distributes the MeMo to the recipient (in this example) recipient-system.
4. Recipient-system creates and sends back a Business Receipt to DP to notify whether it successfully received the MeMo (and DP uses this information to delete the MeMo from internal storage).

In the implementation of the Reference Systems, observing the steps of the SMTP protocol flow can be initiated by running the system-smtp application. The Reference sender-system will create a MeMo, map it to MimeMessage format, sign, encrypt and send it. This mimics the process of sending MeMos as a sender-system, as DP should handle everything from there onwards (given that the MeMo adheres to the agreed format and that the recipient is reachable).

A simplified representation of the Reference Systems application and how it showcases the interactions that SMTP sender- and recipient-systems will have with DP is presented below. It exemplifies the usage of receipts to communicate whether actions where successful or not.



#### Reference SMTP sender-system

The following section will describe the two primary SMTP protocol interactions between sender-systems and DP. They are:

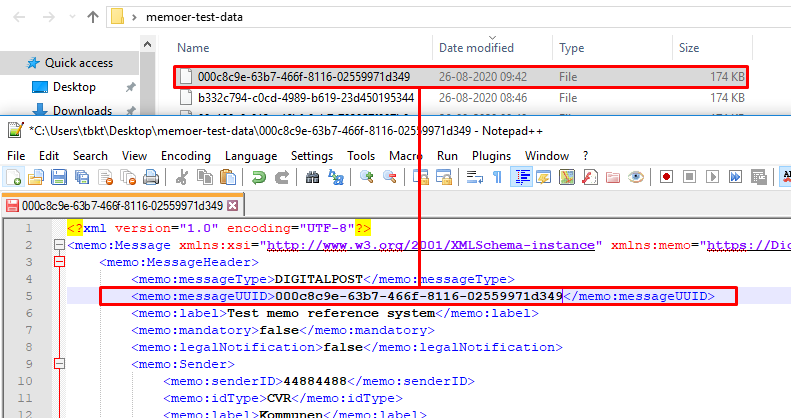
1. Sending MeMo
2. Receiving Receipt

##### Sending MeMo

The Reference sender-system can send a MeMo to the DP test environment mail server. The DP mail endpoint is [memo@test.digitalpost.dk](mailto:memo@test.digitalpost.dk). The message should have a MIME format and follow this structure:

* A multipart/mixed part which at minimum must contain the MeMo XML attached - name of MeMo file must be {messageUUID} ({messageUUID}.xml also allowed) (see picture below)
* An application/pkcs7-signature or application/x-pkcs7-signature part
* The sender-system Api-key must be added as a header to the MimeMessage.

Name of MeMo file:



Attached picture of valid mail in client (outlook in this case):



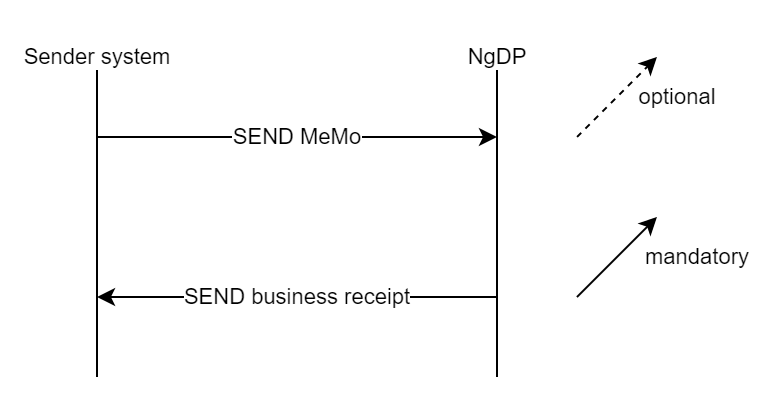
In the case of the Reference Systems application, the service is able to create MeMo messages that adhere to these rules, which besides being sent in by the Reference sender-system, will give insight into the structure and contents of a MeMo. Signing of the MeMo before sending to DP mail server is also done, using in-repository stored certificate stored in a PKCS12 key store.

###### Initiating flow

Sending a MeMo as the Reference sender-system will be doable by running the appropriate application within the system-smtp sub-module. The application will then create and send a valid MeMo intended for the Reference recipient-system to DP.

##### Receiving receipts from DP

Immediately upon receiving the MeMo, DP will send back a Technical Receipt. If this step was successful, DP will validate and process the MeMo and send a Business Receipt back to the sender-system at the *receiptEndpoint* email address. In case of the Reference Systems, The sender-system downloads this Business Receipt and prints it to console.



This Business Receipt contains information on the status of DP’s processing of the MeMo.

Specifically for the Reference sender-system example, the Business Receipt from DP will be logged to the console for an overview of its content and status. A positive Business Receipt to the sender-system notifies that the MeMo is now considered the responsibility of DP, and this stage marks the end of interaction between the sender-system and DP for the respective MeMo.

#### Reference SMTP recipient-system

##### Receiving MeMo

The Reference Systems SMTP example showcases how a scheduler can be implemented to continuously poll a mail server for new entries. In the Reference implementation, the SMTP recipient-system will try to:

1. Fetch IMAPMessage from mail server.
2. Analyze the content to validate whether the IMAPMessage adheres to the correct format.
3. Decrypt and extract the attached MeMo as a BodyPart from the IMAPMessage.
4. Attempt to validate the certificate used to sign the BodyPart.

The Reference recipient-system fetches the validated and processed MeMo by DP from the mail client configured as the *endpoint* field. The Reference System implements functionality of the publicly available Memo-lib, by utilizing a parser to parse the MessageHeader from the received MeMo.

Relevant information from the MeMo MessageHeader will then be logged to the console for an overview of its content.

##### Sending Business Receipt to DP

When the Reference recipient-system has parsed the received MeMo, it will send a Business Receipt back to DP mail server. The endpoint for this Business Receipt is exposed by DP, at the following endpoint on test01: receipt@test.digitalpost.dk

The memoId is a PathVariable, which must be the UUID of the MeMo. This informs DP which specific MeMo the Business Receipt is a response to.

The Reference System will automatically create a Business Receipt when it has received a MeMo, and send it to this endpoint. If the Reference System application encountered an error in the parsing of the MeMo MessageHeader, this error will populate the ErrorMessage field of the Business Receipt.

Thus, the Business Receipt as built by the Reference System application can be either positive or negative, dependent on (in the context of the Reference System application) whether the MeMo sent from DP was parsable.

Upon receiving a positive Business Receipt from the Reference recipient-system, DP will delete the MeMo from internal storage.

#### Configuring mutual SSL for REST flow

The examples showcased are from the Java implementation, however the .Net implementation is similar.

There are two main rules for a correct setup with a valid certificate:

* The CVR of the certificate must be the same as the Organisation’s CVR that the sender-system is a part of - or alternatively the sender-system must have a “Systemfuldmagt” which points to that CVR.
* The API-key must match the API-key of the sender-system that is to be used for sending.

The certificate and API-key is easily configured in the Reference Systems. In the below example from the Java version, egress-rest-client.properties (located in \reference-systems-for-java\ssl-client\src\main\resources\configI) is setup to allow for plug-n-play of valid certificates and API-keys.

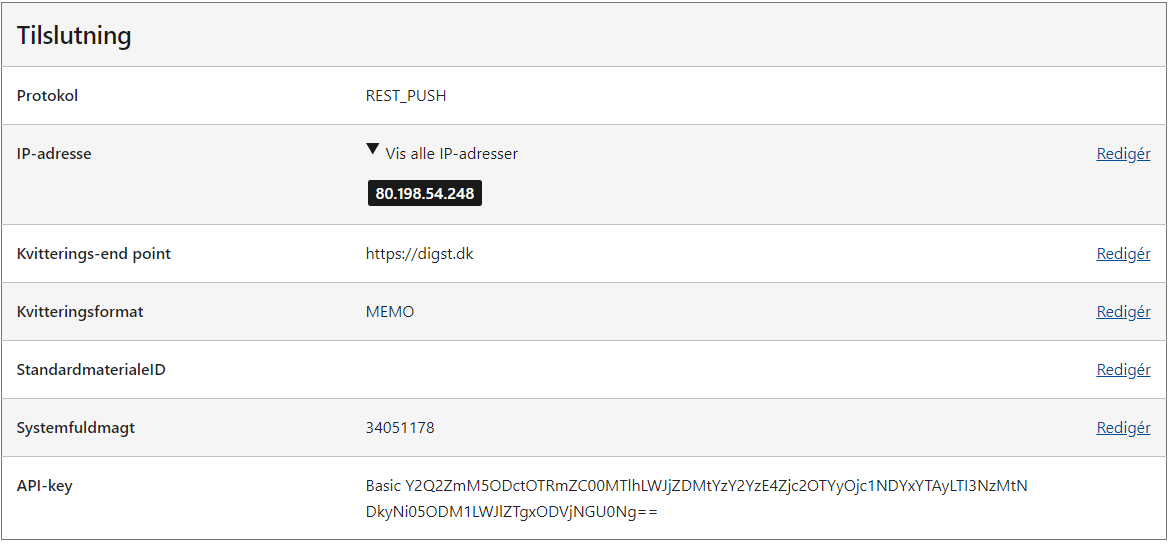
For example, let’s say we have an Organisation with CVR 64942212, with a sender-system we want to send with:

|  |
| --- |
| # keystore and truststore locations should be common to all components  dk.digst.digital.post.egress-rest.keyStoreLocation=classpath:/sender-system-keystore/VOCES\_gyldig\_2022.p12  dk.digst.digital.post.egress-rest.keyStorePassword=Test1234  dk.digst.digital.post.egress-rest.type=PKCS12  dk.digst.digital.post.egress-rest.alias=VOCES\_gyldig\_2022.p12  dk.digst.digital.post.egress-rest.trustselfsigned=false  dk.digst.digital.post.egress-rest.senderSystemApiToken=OTc5MzUxMDEtNGY1Mi00MzE2LTk4YzktYTdjZWQ5NzI5YzdiOmM0MmU5ZTBhLWZhY2ItNGE0ZS1hYWViLTNlNTE0NzVkOGY5MQ==  dk.digst.digital.post.egress-rest.readtimeout=30000  dk.digst.digital.post.egress-rest.connecttimeout=60000    # proxy is optional  dk.digst.digital.post.egress-rest.proxy= |

* We must point the SSL client to a valid certificate with the CVR 64942212:
  + **keyStoreLocation**: Location of the certificate.
  + **keyStorePassword**: Password of the certificate.
  + **type**: Type of the certificate.
  + **alias**: Name of the certificate.
* The API-key must match the API-key of the sender-system we want to send as (can be found in Administrative Access).
  + **senderSystemApiToken**: The API-key of the sender-system (without the “Basic “ prefix)

Thus, in the above example, the “VOCES\_gyldig\_2022.p12” certificate must have the CVR 64942212, and the senderSystemApiToken (OTc5MzUxMDEtNGY1Mi00MzE2LTk4YzktYTdjZWQ5NzI5YzdiOmM0MmU5ZTBhLWZhY2ItNGE0ZS1hYWViLTNlNTE0NzVkOGY5MQ==) must be present on a sender-system on the Organisation.

An exception to the first rule is when “Systemfuldmagt” is used. This allows a sender-system to send on behalf of another CVR, example:



The above pictured sender-system would allow us to send with a certificate that has the CVR 34051178 - even though the Organisation this sender-system is a part of does not have this CVR.

### SFTP protocol examples

#### Overall description

The Reference Systems SFTP application are configured to represent an organisation integrated to Digital Post (DP) with a SFTP protocol sender-system, as recipient-systems can not be SFTP type.

#### Purpose

It is the aim of the Reference Systems SFTP protocol examples to provide insight into the interaction between sender-systems and DP, with extensive console logging and in-code commentary utilized to describe each process.

The examples for SFTP can be found within the Reference Systems repositories here:

* Java: <https://bitbucket.org/nc-dp/reference-systems-for-java/src/master/system-sftp/>
* .Net Core: <https://bitbucket.org/nc-dp/reference-systems-for-dotnet/src/master/Sftp/>

##### SFTP flow

The Reference Systems contains example implementations of the 3 primary SFTP steps, described below:

1. A sender-system uploads a MeMo to DP’s SFTP server.
2. DP fetches the MeMo, and returns a receipt to the SFTP server.
3. The sender-system fetches the receipt from the SFTP server.

In the implementation of the Reference Systems, observing the steps of the SFTP protocol flow can be initiated by running the system-sftp application.

When this application is run, utility-library will be utilized to create 3 MeMos and adding these to a tar.lzma file. This tar.lzma file will automatically be added as the payload to a request to the SFTP server, and the request will be made and the tar.lzma sent.

The application is configured to use a poller which polls the receipt folder on the SFTP server every second for new entries. Whenever a receipt is sent to the SFTP server by DP, the application will shortly thereafter fetch it, log it to console and delete it.

## Security Perspective

### Authentication

The sections below outlines how Digital Post authenticates and authorizes users.

Authentication consist of unambiguously verifying the user identity ~ are they who they claim to be.

Authorization consist of unambiguously verifying the user access ~ are they allowed to perform a given action, or view a resource etc.

* [Authentication regarding external systems](https://digidp.atlassian.net/wiki/spaces/DP/pages/6030641/Authentication+regarding+external+systems)
* [Authentication for internal requests](https://digidp.atlassian.net/wiki/spaces/DP/pages/6651905/Authentication+for+internal+requests)
* [Authentication for external users](https://digidp.atlassian.net/wiki/spaces/DP/pages/853380479/Authentication+for+external+users)
* [Authentication for internal systems](https://digidp.atlassian.net/wiki/spaces/DP/pages/853380591/Authentication+for+internal+systems)
* [Authorization](https://digidp.atlassian.net/wiki/spaces/DP/pages/856721219/Authorization)

#### Authentication regarding external systems

In Digital Post there are multiple patterns for authentication depends on the integrations method and service consumer. In the below table the different authentication schemes are listed for the systems integrating to DP.

|  |  |  |  |
| --- | --- | --- | --- |
| **Protocol** | **Type** | **Authentication** | **Document** |
| SMTP | Sender system | S/Mime signing |  |
| SFTP | Sender system | SSH asynchronous keys |  |
| REST | Datafordeler | Ingress: PKI | [Access request for CPR](https://digidp.atlassian.net/wiki/spaces/DP/pages/10584065/Access+request+for+CPR) |
| Egress: PKI | [Access request for CPR](https://digidp.atlassian.net/wiki/spaces/DP/pages/10584065/Access+request+for+CPR) |
| REST | Sender system | Mutual SSL using API Token | [Authentication of REST Sender/Receiver systems](https://digidp.atlassian.net/wiki/spaces/DP/pages/856719705) |
| REST | Receiver system |
| REST | SMS gateway | Egress: DP’s outgoing REST IP’s are whitelisted in TDC’s gateway and the gateway authenticates DP’s calls based on this whitelist and basic username/password combo. |  |
| REST | PrintService | Basic authentication | [Printservice](https://digidp.atlassian.net/wiki/spaces/DP/pages/2656719/Printservice) |
| REST | View clients | External user’s NemLog-in SAML token and OpenID Connect | [Authentication for external users](https://digidp.atlassian.net/wiki/spaces/DP/pages/853380479/Authentication+for+external+users) |

### Allowed certificate cipher suites

Not all cipher suites are allowed when accessing Digital Post, as many are outdated or insecure. These are the allowed cipher suites:

|  |
| --- |
| **Suites** |
| TLS1.3-AES256-GCM-SHA384 |
| TLS1.3-AES128-GCM-SHA256 |
| TLS1.2-ECDHE-RSA-AES256-GCM-SHA384 |
| TLS1.2-ECDHE-RSA-AES128-GCM-SHA256 |

## MeMo-lib

### Purpose

The library should provide a simple API to construct and serialize messages in the MeMo format. By utilizing the builder pattern it should be possible to construct a representation of the message and let the library handle the serialization to a zip archive. The file must contain a manifest file and one or more MeMo message file(s) in either xml or json format.

The library should also provide a simple API to access the zip archive. This includes verifying the correctness of the content including the actual MeMo message(s) and the manifest file. The API should also include methods to deserialize the MeMo message.

### Repository

MeMo-lib is currently available in Java and .NET. It is accessible at the following repositories:

<https://bitbucket.org/nc-dp/memo-lib-java/src/>

<https://bitbucket.org/nc-dp/memo-lib-dot-net/src>

For information and usage, read the readme files in the repositories.

# Access to Test environments

## Test

### Prerequisites

* None of the sender systems have access to sending services (forkyndelser)
* Only public authorities may send mandatory (obligatoriske) mails
* Requires a **NemLog-in VOCES/FOCES TEST-certificate** for the test environment.
* Modern **TLS-versions (+1.2) and cipher suites (See 'Allowed certificate cipher suites')**
* Only one **receiver system per CVR**

For more information on how to connect to the test environment please see “Connect to Digital Post”.

### Acquisition of an OCES test certificate

To get a NemLog-in OCES TEST-certificate, you must acquire one via Nemlog-in. Follow this guide to create a certificate <https://www.nemlog-in.dk/vejledningertiltestmiljo/>.

Be aware that the test organisations mentioned in this guide cannot be used in Digital Post’s test environment. Neither can login with MitId Erhverv. Only the certificates issued in DevTest4 can be used in Digital Post if it is connected to you real CVR number.

## Access to the administration portal on the test environment

### Step 1 - Create a MitID simulator identity

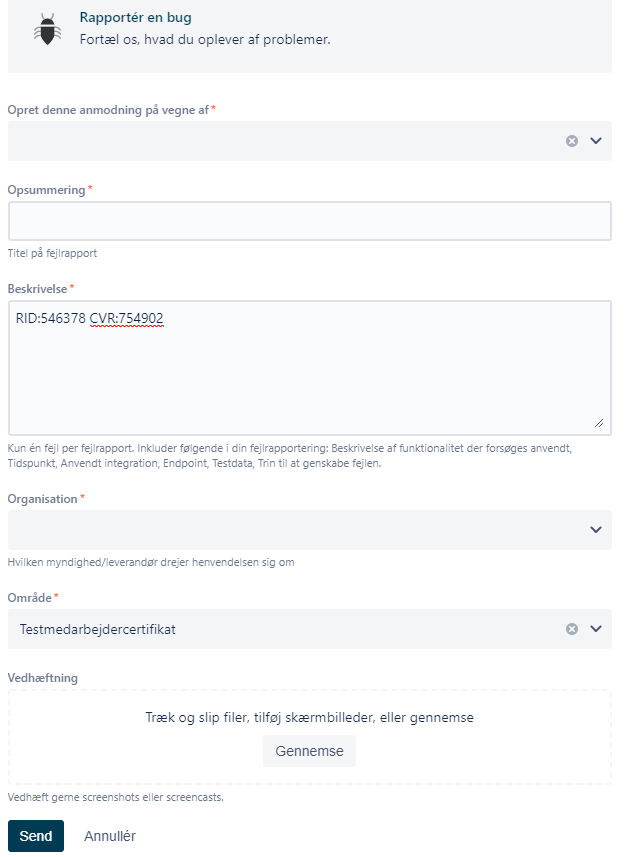
To gain access to the Digital Post Test portal and create test users, privileges etc., a user needs to create a test user identity via the MitID-simulator <https://mitidsimulator.test-nemlog-in.dk/Home/Create>

### Step 2 - Get the new identity enrolled in DP test environment

To make your new test identity work, you need to create a service request in DP’s Servicedesk: <https://digidp.atlassian.net/servicedesk/customer/portal/>

You create a bug, by:

1. Access DP’s Servicedesk.
2. Create a bug, by selecting “Rapporter én bug”.
3. Next to “Beskrivelse” enter the **RID** and **CVR** number as well as the **name** of the test user.
4. Next to “Område” select “Testmedarbejdercertifikat”.
5. Press “send”.



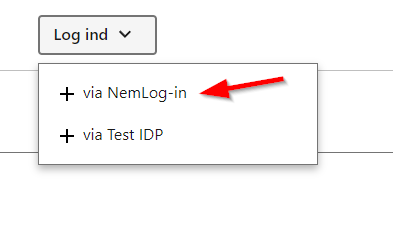
After pressing ”send” Netcompany will process your request and send relevant data to NemLog-in’s integration test environment.

Otherwise, the changes will not apply, because Nemlog-in’s integration test environment will not see them and therefore it will not be able to send the correct information to your log on web page.

### Step 3 - Test login for a test user

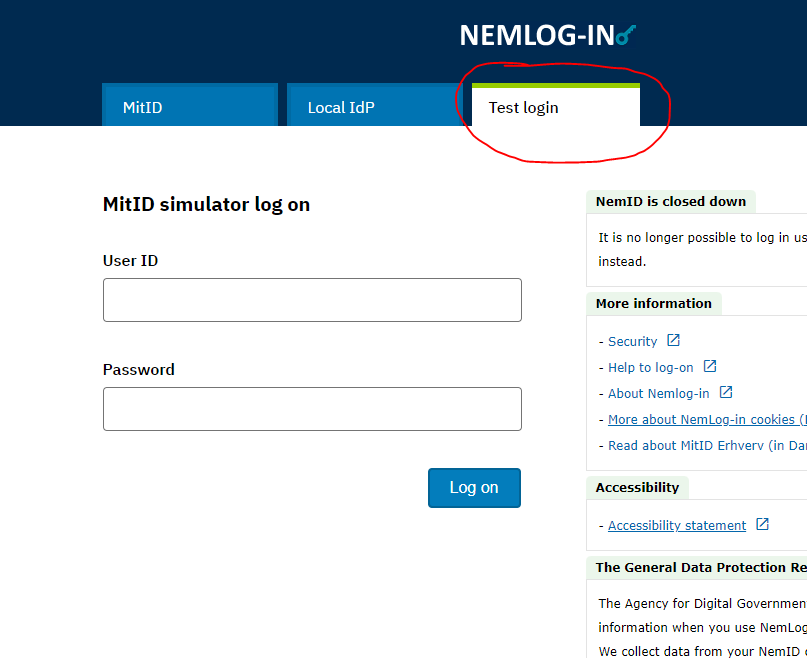
Once step 2 is completed you can login to the Test Portal using NemLog-in -> Test login. You should consider this first login as you bootstrapping login were you obtain access to the Test Portal where you are able to create new users with access to both the mailbox and or Administrative Access. Additionally, you are also able to create test users to similar external parties. E.g. you can create a citizen login which can receive messages send by you and that you can view in the demo client of Borger.dk.

The Test Portal can be access using this link: [Testportal (digitalpost.dk)](https://testportal.test.digitalpost.dk/login)



You will be asked to login using NemLog-in. You can login by selecting “Test login”

Enter your username from the MitID simulator identity and password.



### Links

* NemLog-in Testmiljøer Portal [Nemlog-in miljøer](https://test-nemlog-in.dk/testportal/)
* Nemlog-in guides [NemLog-in Vejledningsliste](https://www.nemlog-in.dk/vejledningertiltestmiljo/)
* digital Post Test Portal [Testportal digital Post](https://testportal.test.digitalpost.dk/login)

### Points of attention for suppliers

Suppliers will be granted the same privileges as authorities, hereby it is possible for suppliers to test both sending and receiving digital post.

* If you do not want to be granted the same privileges as an authority, this must be explicitly mentioned when contacting the Netcompany service desk.

## Access to the Test Portal on the test environment

The Test Portal makes it possible to create test data in the “Digital Post” test environment.

Test data consists of test-contacts and test-authorities, with accompanying mailboxes. You create the test-contacts and -authorities, by entering a CPR- or VAT/CVR-number.

The purpose of the Test Portal is to make it easier for authorities and companies as well as their suppliers to test the “Digital Post” in relation to their own sender- and receiver systems. The authorities and companies are given the possibility to create test data themselves via the Test Portal and therefore they are not limited by what Netcompany and Agency for Digitisation (DIGST) have defined in the test environments.

The guide for the Test Portal can be found through this link <https://digitaliser.dk/digital-post/vejledninger/testportalen> .

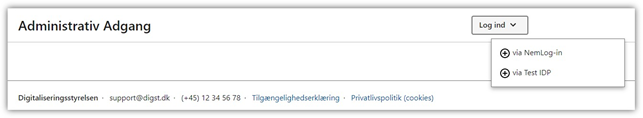
## How to login to Administrative Access with test-identities

When you access <https://admin.test.digitalpost.dk/login> you should have two login options, when you press “Log ind”.

If you select “Via NemLog-in”, you will have to login using your MitID simulator identity.

If you select “Via Test IDP”, you will have the option to login using one of your test-identities.

Press “via Test IDP“



When you login using a test-identity, you must enter the username next to “Brugernavn” and the password next to “Kodeord”.

Press “Login”.



After a successful authentication, the user should have access to the overview page as shown below.



# Troubleshooting, SFTP server, SDLC, OpenID Connect, Connect

## Troubleshooting

These are the typical problems encountered during testing:

* Wrong IP-address.
  + Is it the correct IP address you have added in Administrativ Adgang you are calling from?
* Nemlogin Production certificate used **instead of Nemlogin test certificate**.
  + Has the certificate expired?
* Need to call a specific end point - otherwise you are stuck in the firewall.
* Old ciphers (see 'Prerequisites')
* Non-existent CPR-/CVR number in the dataset.

### Certificate policies

Sender and receiver systems must expose *their* endpoints with mutual SSL. Similar to when a sender or receiver system is calling Digital Post, where Digital Post is exposing a “web certificate” as oppose to an OCES certificate. When Digital Post is calling you, Digital Post is using the OCES certificate and you are expected to identify using a “web certificate” as defined in this section.

#### Format

Web certificates are required to be provided in the X.509 standard, <https://en.wikipedia.org/wiki/X.509>. As well as the entire trust chain.

#### Validity

Every certificate has a validity period. A certificate may be either:

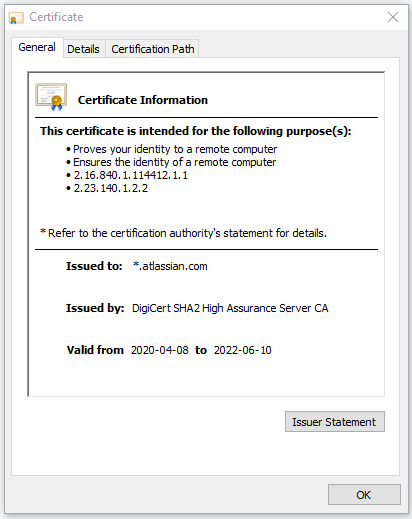
* not yet valid
* **valid**
* expired

Please mind the dates and replace certificates when they are getting close to expiration date.

|  |
| --- |
| It is advisable to have a **defined** **process of certificate replacement**. A new certificate can have overlapping validity period with the old one, thus certificates can be replaced in real-time without any consequences. |

#### Hostname verification

In order for Web certificate to be verified positively “issued to” field containing domains allowed for the certificate must **match the domain in URL**. An example: a page is hosted on a URL starting with https://test.atlassian.net/… and it’s **valid** certificate is issued for all subdomains of atlassian.net (so called star certificate).



#### Accepted issuer CAs

When verifying certificates it is crucial to have them issued by CA (Certificate Authority) that is included in the Oracle Java Root Certificate Program since they are included in the distribution Oracle’s Java Runtime Environment (JRE). Almost all typical certificate issuers are already included, however, mind that your issuer might is not supported. **It is a client's responsibility to use keys acceptable by Java.**See <https://www.oracle.com/java/technologies/javase/carootcertsprogram.html> for details.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **issuer name** | **fingerprint (SHA-1)** | **valid from** | **valid to** |
| 1 | AAA Certificate Services | D1:EB:23:A4:6D:17:D6:8F:D9:25:64:C2:F1:F1:60:17:64:D8:E3:49 | 2004-01-01T00:00Z | 2028-12-31T23:59Z |
| 2 | AC RAIZ FNMT-RCM | EC:50:35:07:B2:15:C4:95:62:19:E2:A8:9A:5B:42:99:2C:4C:2C:20 | 2008-10-29T15:59Z | 2030-01-01T00:00Z |
| 3 | AC RAIZ FNMT-RCM SERVIDORES SEGUROS | 62:FF:D9:9E:C0:65:0D:03:CE:75:93:D2:ED:3F:2D:32:C9:E3:E5:4A | 2018-12-20T09:37Z | 2043-12-20T09:37Z |
| 4 | Actalis Authentication Root CA | F3:73:B3:87:06:5A:28:84:8A:F2:F3:4A:CE:19:2B:DD:C7:8E:9C:AC | 2011-09-22T11:22Z | 2030-09-22T11:22Z |
| 5 | AddTrust External CA Root | 02:FA:F3:E2:91:43:54:68:60:78:57:69:4D:F5:E4:5B:68:85:18:68 | 2000-05-30T10:48Z | 2020-05-30T10:48Z |
| 6 | AddTrust Qualified CA Root | 4D:23:78:EC:91:95:39:B5:00:7F:75:8F:03:3B:21:1E:C5:4D:8B:CF | 2000-05-30T10:44Z | 2020-05-30T10:44Z |
| 7 | AffirmTrust Commercial | F9:B5:B6:32:45:5F:9C:BE:EC:57:5F:80:DC:E9:6E:2C:C7:B2:78:B7 | 2010-01-29T14:06Z | 2030-12-31T14:06Z |
| 8 | AffirmTrust Networking | 29:36:21:02:8B:20:ED:02:F5:66:C5:32:D1:D6:ED:90:9F:45:00:2F | 2010-01-29T14:08Z | 2030-12-31T14:08Z |
| 9 | AffirmTrust Premium | D8:A6:33:2C:E0:03:6F:B1:85:F6:63:4F:7D:6A:06:65:26:32:28:27 | 2010-01-29T14:10Z | 2040-12-31T14:10Z |
| 10 | AffirmTrust Premium ECC | B8:23:6B:00:2F:1D:16:86:53:01:55:6C:11:A4:37:CA:EB:FF:C3:BB | 2010-01-29T14:20Z | 2040-12-31T14:20Z |
| 11 | Amazon Root CA 1 | 8D:A7:F9:65:EC:5E:FC:37:91:0F:1C:6E:59:FD:C1:CC:6A:6E:DE:16 | 2015-05-26T00:00Z | 2038-01-17T00:00Z |
| 12 | Amazon Root CA 2 | 5A:8C:EF:45:D7:A6:98:59:76:7A:8C:8B:44:96:B5:78:CF:47:4B:1A | 2015-05-26T00:00Z | 2040-05-26T00:00Z |
| 13 | Amazon Root CA 3 | 0D:44:DD:8C:3C:8C:1A:1A:58:75:64:81:E9:0F:2E:2A:FF:B3:D2:6E | 2015-05-26T00:00Z | 2040-05-26T00:00Z |
| 14 | Amazon Root CA 4 | F6:10:84:07:D6:F8:BB:67:98:0C:C2:E2:44:C2:EB:AE:1C:EF:63:BE | 2015-05-26T00:00Z | 2040-05-26T00:00Z |
| 15 | ANF Secure Server Root CA | 5B:6E:68:D0:CC:15:B6:A0:5F:1E:C1:5F:AE:02:FC:6B:2F:5D:6F:74 | 2019-09-04T10:00Z | 2039-08-30T10:00Z |
| 16 | Autoridad de Certificacion Firmaprofesional CIF A62634068 | AE:C5:FB:3F:C8:E1:BF:C4:E5:4F:03:07:5A:9A:E8:00:B7:F7:B6:FA | 2009-05-20T08:38Z | 2030-12-31T08:38Z |
| 17 | Baltimore CyberTrust Root | D4:DE:20:D0:5E:66:FC:53:FE:1A:50:88:2C:78:DB:28:52:CA:E4:74 | 2000-05-12T18:46Z | 2025-05-12T23:59Z |
| 18 | Buypass Class 2 Root CA | 49:0A:75:74:DE:87:0A:47:FE:58:EE:F6:C7:6B:EB:C6:0B:12:40:99 | 2010-10-26T08:38Z | 2040-10-26T08:38Z |
| 19 | Buypass Class 3 Root CA | DA:FA:F7:FA:66:84:EC:06:8F:14:50:BD:C7:C2:81:A5:BC:A9:64:57 | 2010-10-26T08:28Z | 2040-10-26T08:28Z |
| 20 | CA Disig Root R2 | B5:61:EB:EA:A4:DE:E4:25:4B:69:1A:98:A5:57:47:C2:34:C7:D9:71 | 2012-07-19T09:15Z | 2042-07-19T09:15Z |
| 21 | Certigna | B1:2E:13:63:45:86:A4:6F:1A:B2:60:68:37:58:2D:C4:AC:FD:94:97 | 2007-06-29T15:13Z | 2027-06-29T15:13Z |
| 22 | Certigna Root CA | 2D:0D:52:14:FF:9E:AD:99:24:01:74:20:47:6E:6C:85:27:27:F5:43 | 2013-10-01T08:32Z | 2033-10-01T08:32Z |
| 23 | certSIGN ROOT CA | FA:B7:EE:36:97:26:62:FB:2D:B0:2A:F6:BF:03:FD:E8:7C:4B:2F:9B | 2006-07-04T17:20Z | 2031-07-04T17:20Z |
| 24 | certSIGN ROOT CA G2 | 26:F9:93:B4:ED:3D:28:27:B0:B9:4B:A7:E9:15:1D:A3:8D:92:E5:32 | 2017-02-06T09:27Z | 2042-02-06T09:27Z |
| 25 | Certum CA | 62:52:DC:40:F7:11:43:A2:2F:DE:9E:F7:34:8E:06:42:51:B1:81:18 | 2002-06-11T10:46Z | 2027-06-11T10:46Z |
| 26 | Certum EC-384 CA | F3:3E:78:3C:AC:DF:F4:A2:CC:AC:67:55:69:56:D7:E5:16:3C:E1:ED | 2018-03-26T07:24Z | 2043-03-26T07:24Z |
| 27 | Certum Trusted Network CA | 07:E0:32:E0:20:B7:2C:3F:19:2F:06:28:A2:59:3A:19:A7:0F:06:9E | 2008-10-22T12:07Z | 2029-12-31T12:07Z |
| 28 | Certum Trusted Network CA 2 | D3:DD:48:3E:2B:BF:4C:05:E8:AF:10:F5:FA:76:26:CF:D3:DC:30:92 | 2011-10-06T08:39Z | 2046-10-06T08:39Z |
| 29 | Certum Trusted Root CA | C8:83:44:C0:18:AE:9F:CC:F1:87:B7:8F:22:D1:C5:D7:45:84:BA:E5 | 2018-03-16T12:10Z | 2043-03-16T12:10Z |
| 30 | CFCA EV ROOT | E2:B8:29:4B:55:84:AB:6B:58:C2:90:46:6C:AC:3F:B8:39:8F:84:83 | 2012-08-08T03:07Z | 2029-12-31T03:07Z |
| 31 | Chambers of Commerce Root | 6E:3A:55:A4:19:0C:19:5C:93:84:3C:C0:DB:72:2E:31:30:61:F0:B1 | 2003-09-30T16:13Z | 2037-09-30T16:13Z |
| 32 | Chambers of Commerce Root - 2008 | 78:6A:74:AC:76:AB:14:7F:9C:6A:30:50:BA:9E:A8:7E:FE:9A:CE:3C | 2008-08-01T12:29Z | 2038-07-31T12:29Z |
| 33 | COMODO Certification Authority | 66:31:BF:9E:F7:4F:9E:B6:C9:D5:A6:0C:BA:6A:BE:D1:F7:BD:EF:7B | 2006-12-01T00:00Z | 2029-12-31T23:59Z |
| 34 | COMODO ECC Certification Authority | 9F:74:4E:9F:2B:4D:BA:EC:0F:31:2C:50:B6:56:3B:8E:2D:93:C3:11 | 2008-03-06T00:00Z | 2038-01-18T23:59Z |
| 35 | COMODO RSA Certification Authority | AF:E5:D2:44:A8:D1:19:42:30:FF:47:9F:E2:F8:97:BB:CD:7A:8C:B4 | 2010-01-19T00:00Z | 2038-01-18T23:59Z |
| 36 | DigiCert Assured ID Root CA | 05:63:B8:63:0D:62:D7:5A:BB:C8:AB:1E:4B:DF:B5:A8:99:B2:4D:43 | 2006-11-10T00:00Z | 2031-11-10T00:00Z |
| 37 | DigiCert Assured ID Root G2 | A1:4B:48:D9:43:EE:0A:0E:40:90:4F:3C:E0:A4:C0:91:93:51:5D:3F | 2013-08-01T12:00Z | 2038-01-15T12:00Z |
| 38 | DigiCert Assured ID Root G3 | F5:17:A2:4F:9A:48:C6:C9:F8:A2:00:26:9F:DC:0F:48:2C:AB:30:89 | 2013-08-01T12:00Z | 2038-01-15T12:00Z |
| 39 | DigiCert Global Root CA | A8:98:5D:3A:65:E5:E5:C4:B2:D7:D6:6D:40:C6:DD:2F:B1:9C:54:36 | 2006-11-10T00:00Z | 2031-11-10T00:00Z |
| 40 | DigiCert Global Root G2 | DF:3C:24:F9:BF:D6:66:76:1B:26:80:73:FE:06:D1:CC:8D:4F:82:A4 | 2013-08-01T12:00Z | 2038-01-15T12:00Z |
| 41 | DigiCert Global Root G3 | 7E:04:DE:89:6A:3E:66:6D:00:E6:87:D3:3F:FA:D9:3B:E8:3D:34:9E | 2013-08-01T12:00Z | 2038-01-15T12:00Z |
| 42 | DigiCert High Assurance EV Root CA | 5F:B7:EE:06:33:E2:59:DB:AD:0C:4C:9A:E6:D3:8F:1A:61:C7:DC:25 | 2006-11-10T00:00Z | 2031-11-10T00:00Z |
| 43 | DigiCert Trusted Root G4 | DD:FB:16:CD:49:31:C9:73:A2:03:7D:3F:C8:3A:4D:7D:77:5D:05:E4 | 2013-08-01T12:00Z | 2038-01-15T12:00Z |
| 44 | D-TRUST Root Class 3 CA 2 2009 | 58:E8:AB:B0:36:15:33:FB:80:F7:9B:1B:6D:29:D3:FF:8D:5F:00:F0 | 2009-11-05T08:35Z | 2029-11-05T08:35Z |
| 45 | D-TRUST Root Class 3 CA 2 EV 2009 | 96:C9:1B:0B:95:B4:10:98:42:FA:D0:D8:22:79:FE:60:FA:B9:16:83 | 2009-11-05T08:50Z | 2029-11-05T08:50Z |
| 46 | emSign ECC Root CA - C3 | B6:AF:43:C2:9B:81:53:7D:F6:EF:6B:C3:1F:1F:60:15:0C:EE:48:66 | 2018-02-18T18:30Z | 2043-02-18T18:30Z |
| 47 | emSign ECC Root CA - G3 | 30:43:FA:4F:F2:57:DC:A0:C3:80:EE:2E:58:EA:78:B2:3F:E6:BB:C1 | 2018-02-18T18:30Z | 2043-02-18T18:30Z |
| 48 | emSign Root CA - C1 | E7:2E:F1:DF:FC:B2:09:28:CF:5D:D4:D5:67:37:B1:51:CB:86:4F:01 | 2018-02-18T18:30Z | 2043-02-18T18:30Z |
| 49 | emSign Root CA - G1 | 8A:C7:AD:8F:73:AC:4E:C1:B5:75:4D:A5:40:F4:FC:CF:7C:B5:8E:8C | 2018-02-18T18:30Z | 2043-02-18T18:30Z |
| 50 | Entrust Root Certification Authority | B3:1E:B1:B7:40:E3:6C:84:02:DA:DC:37:D4:4D:F5:D4:67:49:52:F9 | 2006-11-27T20:23Z | 2026-11-27T20:53Z |
| 51 | Entrust Root Certification Authority - EC1 | 20:D8:06:40:DF:9B:25:F5:12:25:3A:11:EA:F7:59:8A:EB:14:B5:47 | 2012-12-18T15:25Z | 2037-12-18T15:55Z |
| 52 | Entrust Root Certification Authority - G2 | 8C:F4:27:FD:79:0C:3A:D1:66:06:8D:E8:1E:57:EF:BB:93:22:72:D4 | 2009-07-07T17:25Z | 2030-12-07T17:55Z |
| 53 | Entrust Root Certification Authority - G4 | 14:88:4E:86:26:37:B0:26:AF:59:62:5C:40:77:EC:35:29:BA:96:01 | 2015-05-27T11:11Z | 2037-12-27T11:41Z |
| 54 | [Entrust.net](http://Entrust.net) Certification Authority (2048) | 50:30:06:09:1D:97:D4:F5:AE:39:F7:CB:E7:92:7D:7D:65:2D:34:31 | 1999-12-24T17:50Z | 2029-07-24T14:15Z |
| 55 | ePKI Root Certification Authority | 67:65:0D:F1:7E:8E:7E:5B:82:40:A4:F4:56:4B:CF:E2:3D:69:C6:F0 | 2004-12-20T02:31Z | 2034-12-20T02:31Z |
| 56 | e-Szigno Root CA 2017 | 89:D4:83:03:4F:9E:9A:48:80:5F:72:37:D4:A9:A6:EF:CB:7C:1F:D1 | 2017-08-22T12:07Z | 2042-08-22T12:07Z |
| 57 | E-Tugra Certification Authority | 51:C6:E7:08:49:06:6E:F3:92:D4:5C:A0:0D:6D:A3:62:8F:C3:52:39 | 2013-03-05T12:09Z | 2023-03-03T12:09Z |
| 58 | GDCA TrustAUTH R5 ROOT | 0F:36:38:5B:81:1A:25:C3:9B:31:4E:83:CA:E9:34:66:70:CC:74:B4 | 2014-11-26T05:13Z | 2040-12-31T15:59Z |
| 59 | GeoTrust Global CA | DE:28:F4:A4:FF:E5:B9:2F:A3:C5:03:D1:A3:49:A7:F9:96:2A:82:12 | 2002-05-21T04:00Z | 2022-05-21T04:00Z |
| 60 | GeoTrust Primary Certification Authority | 32:3C:11:8E:1B:F7:B8:B6:52:54:E2:E2:10:0D:D6:02:90:37:F0:96 | 2006-11-27T00:00Z | 2036-07-16T23:59Z |
| 61 | GeoTrust Primary Certification Authority - G2 | 8D:17:84:D5:37:F3:03:7D:EC:70:FE:57:8B:51:9A:99:E6:10:D7:B0 | 2007-11-05T00:00Z | 2038-01-18T23:59Z |
| 62 | GeoTrust Primary Certification Authority - G3 | 03:9E:ED:B8:0B:E7:A0:3C:69:53:89:3B:20:D2:D9:32:3A:4C:2A:FD | 2008-04-02T00:00Z | 2037-12-01T23:59Z |
| 63 | GeoTrust Universal CA | E6:21:F3:35:43:79:05:9A:4B:68:30:9D:8A:2F:74:22:15:87:EC:79 | 2004-03-04T05:00Z | 2029-03-04T05:00Z |
| 64 | Global Chambersign Root - 2008 | 4A:BD:EE:EC:95:0D:35:9C:89:AE:C7:52:A1:2C:5B:29:F6:D6:AA:0C | 2008-08-01T12:31Z | 2038-07-31T12:31Z |
| 65 | GlobalSign | D6:9B:56:11:48:F0:1C:77:C5:45:78:C1:09:26:DF:5B:85:69:76:AD | 2009-03-18T10:00Z | 2029-03-18T10:00Z |
| 66 | GlobalSign | 1F:24:C6:30:CD:A4:18:EF:20:69:FF:AD:4F:DD:5F:46:3A:1B:69:AA | 2012-11-13T00:00Z | 2038-01-19T03:14Z |
| 67 | GlobalSign | 80:94:64:0E:B5:A7:A1:CA:11:9C:1F:DD:D5:9F:81:02:63:A7:FB:D1 | 2014-12-10T00:00Z | 2034-12-10T00:00Z |
| 68 | GlobalSign | 69:69:56:2E:40:80:F4:24:A1:E7:19:9F:14:BA:F3:EE:58:AB:6A:BB | 2012-11-13T00:00Z | 2038-01-19T03:14Z |
| 69 | GlobalSign Root CA | B1:BC:96:8B:D4:F4:9D:62:2A:A8:9A:81:F2:15:01:52:A4:1D:82:9C | 1998-09-01T12:00Z | 2028-01-28T12:00Z |
| 70 | GlobalSign Root E46 | 39:B4:6C:D5:FE:80:06:EB:E2:2F:4A:BB:08:33:A0:AF:DB:B9:DD:84 | 2019-03-20T00:00Z | 2046-03-20T00:00Z |
| 71 | GlobalSign Root R46 | 53:A2:B0:4B:CA:6B:D6:45:E6:39:8A:8E:C4:0D:D2:BF:77:C3:A2:90 | 2019-03-20T00:00Z | 2046-03-20T00:00Z |
| 72 | GLOBALTRUST 2020 | D0:67:C1:13:51:01:0C:AA:D0:C7:6A:65:37:31:16:26:4F:53:71:A2 | 2020-02-10T00:00Z | 2040-06-10T00:00Z |
| 73 | Go Daddy Class 2 Certification Authority | 27:96:BA:E6:3F:18:01:E2:77:26:1B:A0:D7:77:70:02:8F:20:EE:E4 | 2004-06-29T17:06Z | 2034-06-29T17:06Z |
| 74 | Go Daddy Root Certificate Authority - G2 | 47:BE:AB:C9:22:EA:E8:0E:78:78:34:62:A7:9F:45:C2:54:FD:E6:8B | 2009-09-01T00:00Z | 2037-12-31T23:59Z |
| 75 | GTS Root R1 | E5:8C:1C:C4:91:3B:38:63:4B:E9:10:6E:E3:AD:8E:6B:9D:D9:81:4A | 2016-06-22T00:00Z | 2036-06-22T00:00Z |
| 76 | GTS Root R2 | 9A:44:49:76:32:DB:DE:FA:D0:BC:FB:5A:7B:17:BD:9E:56:09:24:94 | 2016-06-22T00:00Z | 2036-06-22T00:00Z |
| 77 | GTS Root R3 | ED:E5:71:80:2B:C8:92:B9:5B:83:3C:D2:32:68:3F:09:CD:A0:1E:46 | 2016-06-22T00:00Z | 2036-06-22T00:00Z |
| 78 | GTS Root R4 | 77:D3:03:67:B5:E0:0C:15:F6:0C:38:61:DF:7C:E1:3B:92:46:4D:47 | 2016-06-22T00:00Z | 2036-06-22T00:00Z |
| 79 | Hellenic Academic and Research Institutions ECC RootCA 2015 | 9F:F1:71:8D:92:D5:9A:F3:7D:74:97:B4:BC:6F:84:68:0B:BA:B6:66 | 2015-07-07T10:37Z | 2040-06-30T10:37Z |
| 80 | Hellenic Academic and Research Institutions RootCA 2015 | 01:0C:06:95:A6:98:19:14:FF:BF:5F:C6:B0:B6:95:EA:29:E9:12:A6 | 2015-07-07T10:11Z | 2040-06-30T10:11Z |
| 81 | Hongkong Post Root CA 1 | D6:DA:A8:20:8D:09:D2:15:4D:24:B5:2F:CB:34:6E:B2:58:B2:8A:58 | 2003-05-15T05:13Z | 2023-05-15T04:52Z |
| 82 | Hongkong Post Root CA 3 | 58:A2:D0:EC:20:52:81:5B:C1:F3:F8:64:02:24:4E:C2:8E:02:4B:02 | 2017-06-03T02:29Z | 2042-06-03T02:29Z |
| 83 | IdenTrust Commercial Root CA 1 | DF:71:7E:AA:4A:D9:4E:C9:55:84:99:60:2D:48:DE:5F:BC:F0:3A:25 | 2014-01-16T18:12Z | 2034-01-16T18:12Z |
| 84 | IdenTrust Public Sector Root CA 1 | BA:29:41:60:77:98:3F:F4:F3:EF:F2:31:05:3B:2E:EA:6D:4D:45:FD | 2014-01-16T17:53Z | 2034-01-16T17:53Z |
| 85 | ISRG Root X1 | CA:BD:2A:79:A1:07:6A:31:F2:1D:25:36:35:CB:03:9D:43:29:A5:E8 | 2015-06-04T11:04Z | 2035-06-04T11:04Z |
| 86 | [Izenpe.com](http://Izenpe.com) | 2F:78:3D:25:52:18:A7:4A:65:39:71:B5:2C:A2:9C:45:15:6F:E9:19 | 2007-12-13T13:08Z | 2037-12-13T08:27Z |
| 87 | LuxTrust Global Root | C9:3C:34:EA:90:D9:13:0C:0F:03:00:4B:98:BD:8B:35:70:91:56:11 | 2011-03-17T09:51Z | 2021-03-17T09:51Z |
| 88 | LuxTrust Global Root 2 | 1E:0E:56:19:0A:D1:8B:25:98:B2:04:44:FF:66:8A:04:17:99:5F:3F | 2015-03-05T13:21Z | 2035-03-05T13:21Z |
| 89 | Microsoft ECC Root Certificate Authority 2017 | 99:9A:64:C3:7F:F4:7D:9F:AB:95:F1:47:69:89:14:60:EE:C4:C3:C5 | 2019-12-18T23:06Z | 2042-07-18T23:16Z |
| 90 | Microsoft RSA Root Certificate Authority 2017 | 73:A5:E6:4A:3B:FF:83:16:FF:0E:DC:CC:61:8A:90:6E:4E:AE:4D:74 | 2019-12-18T22:51Z | 2042-07-18T23:00Z |
| 91 | NAVER Global Root Certification Authority | 8F:6B:F2:A9:27:4A:DA:14:A0:C4:F4:8E:61:27:F9:C0:1E:78:5D:D1 | 2017-08-18T08:58Z | 2037-08-18T23:59Z |
| 92 | NetLock Arany (Class Gold) F?tanúsítvány | 06:08:3F:59:3F:15:A1:04:A0:69:A4:6B:A9:03:D0:06:B7:97:09:91 | 2008-12-11T15:08Z | 2028-12-06T15:08Z |
| 93 | OISTE WISeKey Global Root GB CA | 0F:F9:40:76:18:D3:D7:6A:4B:98:F0:A8:35:9E:0C:FD:27:AC:CC:ED | 2014-12-01T15:00Z | 2039-12-01T15:10Z |
| 94 | OISTE WISeKey Global Root GC CA | E0:11:84:5E:34:DE:BE:88:81:B9:9C:F6:16:26:D1:96:1F:C3:B9:31 | 2017-05-09T09:48Z | 2042-05-09T09:58Z |
| 95 | QuoVadis Root CA 1 G3 | 1B:8E:EA:57:96:29:1A:C9:39:EA:B8:0A:81:1A:73:73:C0:93:79:67 | 2012-01-12T17:27Z | 2042-01-12T17:27Z |
| 96 | QuoVadis Root CA 2 | CA:3A:FB:CF:12:40:36:4B:44:B2:16:20:88:80:48:39:19:93:7C:F7 | 2006-11-24T18:27Z | 2031-11-24T18:23Z |
| 97 | QuoVadis Root CA 2 G3 | 09:3C:61:F3:8B:8B:DC:7D:55:DF:75:38:02:05:00:E1:25:F5:C8:36 | 2012-01-12T18:59Z | 2042-01-12T18:59Z |
| 98 | QuoVadis Root CA 3 | 1F:49:14:F7:D8:74:95:1D:DD:AE:02:C0:BE:FD:3A:2D:82:75:51:85 | 2006-11-24T19:11Z | 2031-11-24T19:06Z |
| 99 | QuoVadis Root CA 3 G3 | 48:12:BD:92:3C:A8:C4:39:06:E7:30:6D:27:96:E6:A4:CF:22:2E:7D | 2012-01-12T20:26Z | 2042-01-12T20:26Z |
| 100 | QuoVadis Root Certification Authority | DE:3F:40:BD:50:93:D3:9B:6C:60:F6:DA:BC:07:62:01:00:89:76:C9 | 2001-03-19T18:33Z | 2021-03-17T18:33Z |
| 101 | Secure Global CA | 3A:44:73:5A:E5:81:90:1F:24:86:61:46:1E:3B:9C:C4:5F:F5:3A:1B | 2006-11-07T19:42Z | 2029-12-31T19:52Z |
| 102 | SecureSign RootCA11 | 3B:C4:9F:48:F8:F3:73:A0:9C:1E:BD:F8:5B:B1:C3:65:C7:D8:11:B3 | 2009-04-08T04:56Z | 2029-04-08T04:56Z |
| 103 | SecureTrust CA | 87:82:C6:C3:04:35:3B:CF:D2:96:92:D2:59:3E:7D:44:D9:34:FF:11 | 2006-11-07T19:31Z | 2029-12-31T19:40Z |
| 104 | Security Communication RootCA1 | 36:B1:2B:49:F9:81:9E:D7:4C:9E:BC:38:0F:C6:56:8F:5D:AC:B2:F7 | 2003-09-30T04:20Z | 2023-09-30T04:20Z |
| 105 | Security Communication RootCA2 | 5F:3B:8C:F2:F8:10:B3:7D:78:B4:CE:EC:19:19:C3:73:34:B9:C7:74 | 2009-05-29T05:00Z | 2029-05-29T05:00Z |
| 106 | [SSL.com](http://SSL.com) EV Root Certification Authority ECC | 4C:DD:51:A3:D1:F5:20:32:14:B0:C6:C5:32:23:03:91:C7:46:42:6D | 2016-02-12T18:15Z | 2041-02-12T18:15Z |
| 107 | [SSL.com](http://SSL.com) EV Root Certification Authority RSA R2 | 74:3A:F0:52:9B:D0:32:A0:F4:4A:83:CD:D4:BA:A9:7B:7C:2E:C4:9A | 2017-05-31T18:14Z | 2042-05-30T18:14Z |
| 108 | [SSL.com](http://SSL.com) Root Certification Authority ECC | C3:19:7C:39:24:E6:54:AF:1B:C4:AB:20:95:7A:E2:C3:0E:13:02:6A | 2016-02-12T18:14Z | 2041-02-12T18:14Z |
| 109 | [SSL.com](http://SSL.com) Root Certification Authority RSA | B7:AB:33:08:D1:EA:44:77:BA:14:80:12:5A:6F:BD:A9:36:49:0C:BB | 2016-02-12T17:39Z | 2041-02-12T17:39Z |
| 110 | Starfield Class 2 Certification Authority | AD:7E:1C:28:B0:64:EF:8F:60:03:40:20:14:C3:D0:E3:37:0E:B5:8A | 2004-06-29T17:39Z | 2034-06-29T17:39Z |
| 111 | Starfield Root Certificate Authority - G2 | B5:1C:06:7C:EE:2B:0C:3D:F8:55:AB:2D:92:F4:FE:39:D4:E7:0F:0E | 2009-09-01T00:00Z | 2037-12-31T23:59Z |
| 112 | Starfield Services Root Certificate Authority - G2 | 92:5A:8F:8D:2C:6D:04:E0:66:5F:59:6A:FF:22:D8:63:E8:25:6F:3F | 2009-09-01T00:00Z | 2037-12-31T23:59Z |
| 113 | SwissSign Gold CA - G2 | D8:C5:38:8A:B7:30:1B:1B:6E:D4:7A:E6:45:25:3A:6F:9F:1A:27:61 | 2006-10-25T08:30Z | 2036-10-25T08:30Z |
| 114 | SwissSign Platinum CA - G2 | 56:E0:FA:C0:3B:8F:18:23:55:18:E5:D3:11:CA:E8:C2:43:31:AB:66 | 2006-10-25T08:36Z | 2036-10-25T08:36Z |
| 115 | SwissSign Silver CA - G2 | 9B:AA:E5:9F:56:EE:21:CB:43:5A:BE:25:93:DF:A7:F0:40:D1:1D:CB | 2006-10-25T08:32Z | 2036-10-25T08:32Z |
| 116 | SZAFIR ROOT CA2 | E2:52:FA:95:3F:ED:DB:24:60:BD:6E:28:F3:9C:CC:CF:5E:B3:3F:DE | 2015-10-19T07:43Z | 2035-10-19T07:43Z |
| 117 | TeliaSonera Root CA v1 | 43:13:BB:96:F1:D5:86:9B:C1:4E:6A:92:F6:CF:F6:34:69:87:82:37 | 2007-10-18T12:00Z | 2032-10-18T12:00Z |
| 118 | thawte Primary Root CA | 91:C6:D6:EE:3E:8A:C8:63:84:E5:48:C2:99:29:5C:75:6C:81:7B:81 | 2006-11-17T00:00Z | 2036-07-16T23:59Z |
| 119 | thawte Primary Root CA - G2 | AA:DB:BC:22:23:8F:C4:01:A1:27:BB:38:DD:F4:1D:DB:08:9E:F0:12 | 2007-11-05T00:00Z | 2038-01-18T23:59Z |
| 120 | thawte Primary Root CA - G3 | F1:8B:53:8D:1B:E9:03:B6:A6:F0:56:43:5B:17:15:89:CA:F3:6B:F2 | 2008-04-02T00:00Z | 2037-12-01T23:59Z |
| 121 | Trustwave Global Certification Authority | 2F:8F:36:4F:E1:58:97:44:21:59:87:A5:2A:9A:D0:69:95:26:7F:B5 | 2017-08-23T19:34Z | 2042-08-23T19:34Z |
| 122 | Trustwave Global ECC P256 Certification Authority | B4:90:82:DD:45:0C:BE:8B:5B:B1:66:D3:E2:A4:08:26:CD:ED:42:CF | 2017-08-23T19:35Z | 2042-08-23T19:35Z |
| 123 | Trustwave Global ECC P384 Certification Authority | E7:F3:A3:C8:CF:6F:C3:04:2E:6D:0E:67:32:C5:9E:68:95:0D:5E:D2 | 2017-08-23T19:36Z | 2042-08-23T19:36Z |
| 124 | T-TeleSec GlobalRoot Class 2 | 59:0D:2D:7D:88:4F:40:2E:61:7E:A5:62:32:17:65:CF:17:D8:94:E9 | 2008-10-01T10:40Z | 2033-10-01T23:59Z |
| 125 | T-TeleSec GlobalRoot Class 3 | 55:A6:72:3E:CB:F2:EC:CD:C3:23:74:70:19:9D:2A:BE:11:E3:81:D1 | 2008-10-01T10:29Z | 2033-10-01T23:59Z |
| 126 | TUBITAK Kamu SM SSL Kok Sertifikasi - Surum 1 | 31:43:64:9B:EC:CE:27:EC:ED:3A:3F:0B:8F:0D:E4:E8:91:DD:EE:CA | 2013-11-25T08:25Z | 2043-10-25T08:25Z |
| 127 | TWCA Global Root CA | 9C:BB:48:53:F6:A4:F6:D3:52:A4:E8:32:52:55:60:13:F5:AD:AF:65 | 2012-06-27T06:28Z | 2030-12-31T15:59Z |
| 128 | TWCA Root Certification Authority | CF:9E:87:6D:D3:EB:FC:42:26:97:A3:B5:A3:7A:A0:76:A9:06:23:48 | 2008-08-28T07:24Z | 2030-12-31T15:59Z |
| 129 | UCA Extended Validation Root | A3:A1:B0:6F:24:61:23:4A:E3:36:A5:C2:37:FC:A6:FF:DD:F0:D7:3A | 2015-03-13T00:00Z | 2038-12-31T00:00Z |
| 130 | UCA Global G2 Root | 28:F9:78:16:19:7A:FF:18:25:18:AA:44:FE:C1:A0:CE:5C:B6:4C:8A | 2016-03-11T00:00Z | 2040-12-31T00:00Z |
| 131 | USERTrust ECC Certification Authority | D1:CB:CA:5D:B2:D5:2A:7F:69:3B:67:4D:E5:F0:5A:1D:0C:95:7D:F0 | 2010-02-01T00:00Z | 2038-01-18T23:59Z |
| 132 | USERTrust RSA Certification Authority | 2B:8F:1B:57:33:0D:BB:A2:D0:7A:6C:51:F7:0E:E9:0D:DA:B9:AD:8E | 2010-02-01T00:00Z | 2038-01-18T23:59Z |
| 133 | UTN-USERFirst-Object | E1:2D:FB:4B:41:D7:D9:C3:2B:30:51:4B:AC:1D:81:D8:38:5E:2D:46 | 1999-07-09T18:31Z | 2019-07-09T18:40Z |
| 134 | VeriSign Class 3 Public Primary Certification Authority - G3 | 13:2D:0D:45:53:4B:69:97:CD:B2:D5:C3:39:E2:55:76:60:9B:5C:C6 | 1999-10-01T00:00Z | 2036-07-16T23:59Z |
| 135 | VeriSign Class 3 Public Primary Certification Authority - G4 | 22:D5:D8:DF:8F:02:31:D1:8D:F7:9D:B7:CF:8A:2D:64:C9:3F:6C:3A | 2007-11-05T00:00Z | 2038-01-18T23:59Z |
| 136 | VeriSign Class 3 Public Primary Certification Authority - G5 | 4E:B6:D5:78:49:9B:1C:CF:5F:58:1E:AD:56:BE:3D:9B:67:44:A5:E5 | 2006-11-08T00:00Z | 2036-07-16T23:59Z |
| 137 | VeriSign Universal Root Certification Authority | 36:79:CA:35:66:87:72:30:4D:30:A5:FB:87:3B:0F:A7:7B:B7:0D:54 | 2008-04-02T00:00Z | 2037-12-01T23:59Z |
| 138 | XRamp Global Certification Authority | B8:01:86:D1:EB:9C:86:A5:41:04:CF:30:54:F3:4C:52:B7:E5:58:C6 | 2004-11-01T17:14Z | 2035-01-01T05:37Z |

|  |
| --- |
| A list changes in time, depending on Java version used |

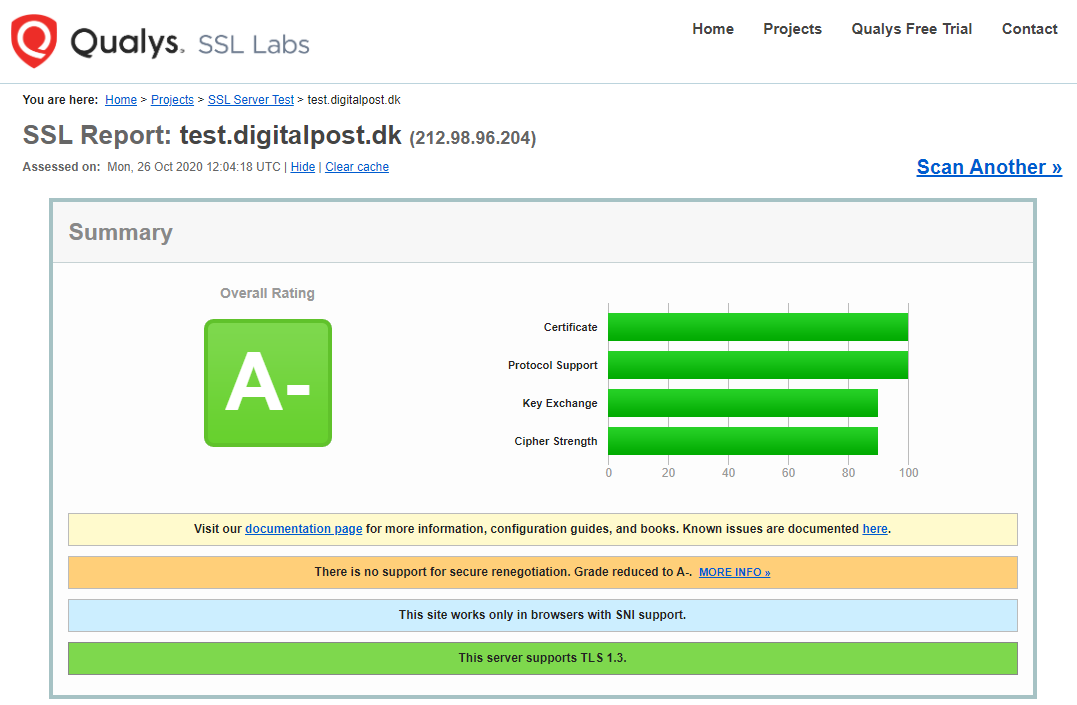
### Allowed ciphers

The following Ciphers are allowed when trying to connect to application endpoints, the list of allowed ciphers will change over time when new secure ciphers are supported, and older ciphers are getting insecure:

|  |
| --- |
| TLS1.3-AES256-GCM-SHA384 TLS1.2-ECDHE-RSA-AES256-GCM-SHA384 TLS1.2-ECDHE-RSA-AES128-GCM-SHA256 TLS1.3-AES128-GCM-SHA256 |

### Rating

There are many factors that affect certificate quality. To simplify this assessment process one can use so called rating tools. We are using SSL Labs rating, for example: <https://www.ssllabs.com/ssltest/analyze.html?d=test.digitalpost.dk>:



Rating is based on multiple factors (see <https://github.com/ssllabs/research/wiki/SSL-Server-Rating-Guide>), and as a rule of thumb:

|  |
| --- |
| **A+**, **A, A-** ratings are mandatory |

### Troubleshooting

Full certificate chain

We have observed that one of the functionalities that is frequently incorrectly configured is a **full certificate chain** being reported by server - this drops the rating to below A.

### Ensuring the authenticity of Digital Posts through the OCES certificate

#### HTTP calls with mutual SSL

When systems are calling Digital Post through mutual SSL, Digital Post ensures the validity and identity of the caller by performing a series of checks as described in [Mutual SSL authentication using API key](https://digidp.atlassian.net/wiki/spaces/DP/pages/742522912/Mutual+SSL+authentication+using+API+key) .

Similarly, receiver systems are expected to verify the identity of Digital Post, to ensure that the services are not misused by bad actors. One way of doing this is by pinning the certificate of Digital Post. Meaning that your receiver system *only* allows calls where the caller authenticate using Digital Post’s OCES certificate.

#### Systems using the SMTP protocol

When sender and receiver systems utilize the SMTP protocol all messages are either signed or encrypted using the public part of the Digital Posts OCES certificate. This means that a sender system using the SMTP protocol are expected to encrypt the messages sent to Digital Post with Digital Post’s certificate, to ensure that no other can read the message. As well as when Digital Post is sending message via SMTP to receiver systems they are signed using Digital Post’s certificate so that the receiver can ensure the authenticity of the messages.

#### Finding the certificate

To ensure that integrators can do certificate pinning and encryption/signature validation, Digital Post provides the public part of its OCES certificate.

The certificate can be found on <https://digitaliser.dk/digital-post/vejledninger/oces-certifikater> . Please note that different certificates are used in test and production.

### Point of attention regarding certificates

In relation to the initial testing of the DP (Digital Post) test environment, there has been issues regarding the test users Trust Chain in their main certificates.  
The following documentation and description are intended to indicate points of attention in relation to obtaining an intact Trust Chain and thereby a functional test certificate and ultimately access to DP’s administration portal.

#### Two types of certificates

It is important that you **always use a test certificate (MOCES) when accessing the test environment** - and a production certificate (MOCES) when accessing the production environment.

The Trust Chain for test- and production certificates are not identical and therefore using a production certificate may cause errors in the Trust Chain, when using it on the test environment.

Furthermore, production certificates can contain confidential data. By default, Netcompany does not have access to the test users certificate chains, so it is only in cases of issues that it may become necessary.

#### Certificates must be traceable

If an error in the Trust Chain occurs, it is the test user’s responsibility to be able to trace the certificate and thereby the associated certificate chain (part of the Trust Chain). When uploading a test certificate, the test environments frontend (Administrativ Access) does not check if the certificate chain is correct, therefore it is very important that the whole chain of the certificate can be traced back as, if an error occurs.

Furthermore, it is also very important, to be able to access the entire chain and not just the end of the certificate.

#### Check validity of a OCES certificate for test and production

##### In general

The description of OCES certificates can be found in this document (in Danish) <https://digitaliseringskataloget.dk/files/integration-files/020920201531/Kom%20godt%20i%20gang%20-%20certifikater.pdf> (Have a look at page 12)

##### Test certificates

For a test certificate, the OCES certificate must contain the primary (certificate #1) and secondary issuer (certificate #2)

The first certificate must contain the Issuer (CN) e.g:  
Issuer: CN=**TRUST2408 Systemtest VII Primary CA**, O=TRUST2408, C=DK

The second certificate must contain the Issuer (CN) e.g:  
Issuer: CN=**TRUST2408 Systemtest XXII CA**, O=TRUST2408, C=DK

##### Production Certificate

The first certificate must contain the Issuer (CN) e.g:  
**TRUST2408 OCES Primary CA**

##### Tool for checking validity of certificate

In order to check if a certificate (.pem, p12, .cer, .crt) is for production or test, different tools can be used. This example is a Windows tool:

|  |
| --- |
| certutil -dump <path to cert> |

<https://superuser.com/questions/580697/how-do-i-view-the-contents-of-a-pfx-file-on-windows>

This will dump the content of the certificate and you need to check that the information from the dump is correct.

A dump from a valid .p12 (pkcs12) test certificate:

|  |
| --- |
| ================ Certificate 0 ================  ================ Begin Nesting Level 1 ================  Element 0:  Serial Number: 4bea6e94  Issuer: CN=TRUST2408 Systemtest VII Primary CA, O=TRUST2408, C=DK  NotBefore: 12-05-2010 09:32  NotAfter: 12-01-2037 10:02  Subject: CN=TRUST2408 Systemtest VII Primary CA, O=TRUST2408, C=DK  Signature matches Public Key  Root Certificate: Subject matches Issuer  Cert Hash(sha1): d6b1f3e9319f68d36f1c71c48e47468130543bce  ---------------- End Nesting Level 1 ----------------  No key provider information  Cannot find the certificate and **private** key **for** decryption.    ================ Certificate 1 ================  ================ Begin Nesting Level 1 ================  Element 1:  Serial Number: 58187e74  Issuer: CN=TRUST2408 Systemtest VII Primary CA, O=TRUST2408, C=DK  NotBefore: 04-07-2017 07:18  NotAfter: 04-07-2032 07:48  Subject: CN=TRUST2408 Systemtest XXII CA, O=TRUST2408, C=DK  Non-root Certificate  Cert Hash(sha1): 896f3cdbdc384eba6b13105b2ca1654bc1b97437  ---------------- End Nesting Level 1 ----------------  No key provider information  Cannot find the certificate and **private** key **for** decryption.    ================ Certificate 2 ================  ================ Begin Nesting Level 1 ================  Element 2:  Serial Number: 5bad3b1a  Issuer: CN=TRUST2408 Systemtest XXII CA, O=TRUST2408, C=DK  NotBefore: 16-12-2019 15:31  NotAfter: 16-12-2022 15:31  Subject: SERIALNUMBER=CVR:30808460-FID:94731315 + CN=TU GENEREL FOCES gyldig (funktionscertifikat), O=NETS DANID A/S // CVR:30808460, C=DK  Non-root Certificate  Cert Hash(sha1): 21ad7d2d4280765bfe113b7dd5d62736c34e37bd  ---------------- End Nesting Level 1 ----------------  Provider = Microsoft Enhanced RSA and AES Cryptographic Provider  Encryption test passed  CertUtil: -dump command completed successfully. |

#### Certificates must be unique and not reused

It is very important that you do not reuse the test certificates, because the test environment cannot distinguish reused test certificates from each other, and this may cause a system error.  
Therefore, it is important that you use a unique test certificate, when creating a new system in Administrative Access.

#### Certificate expiration date

When creating or receiving a test certificate it is important that you keep track of when the certificate expires. Before it expires it must be renewed and uploaded again.

If it is possible for your organization, then create a test certificate that do not expire before the end of 2023, then you do not have to worry about renewing it.

### DP Service Desk

If you are in the process of integrating to Digital Post and are experiencing issues with the Solution - or lacking information regarding the interfaces which is not available in this document - it is possible to create a ticket via the Servicedesk. Use the following link and provide as much information as possible:

<https://digidp.atlassian.net/servicedesk/customer/portal/>

## SFTP server

### Generate an SSH-key to DP

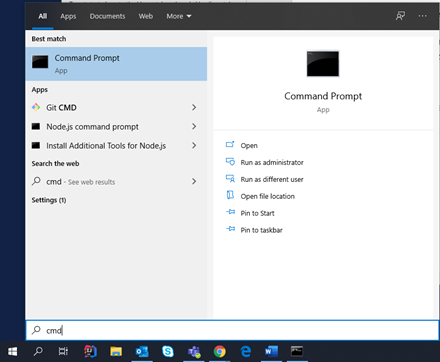
In the DP test environment SSH-key pairs are used to integrate your sender system, if you use the SFTP protocol.

In the following there are two examples of how you can make an SSH-key. Both examples will generate both a Private and a Public SSH-key. However, you should only upload the Public SSH-key id\_rsa.pub and not the Private SSH-key, that are generated.

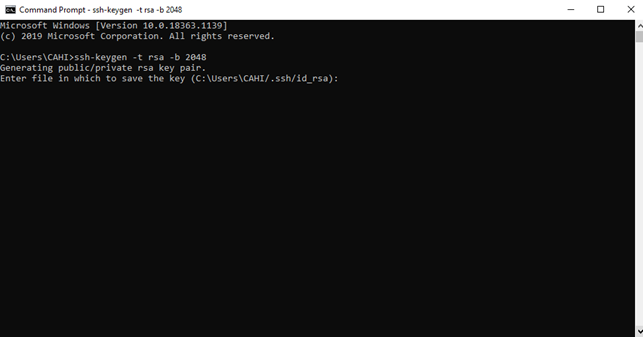
The first example is generated via Command Prompt/The Terminal and the second example is generated via program called PuTTY. Command Prompt works on Windows, Linux and MacOS and PuTTY is a Windows program.

#### Command Prompt:

To open Command Prompt, you must search “CMD” in the Windows search box and click on Command Prompt.



When Command Prompt is open, you must type “SSH-keygen -t rsa -b 2048” to generate an SSH-key. -t defines what type of SSH-key it should be and -b defines how many bits it is. In the DP test environment we use rsa which support 2048 bits. Once you have written it in the Command Prompt, it will ask where you want to save the file nd what it should be called, which is shown in the image below.



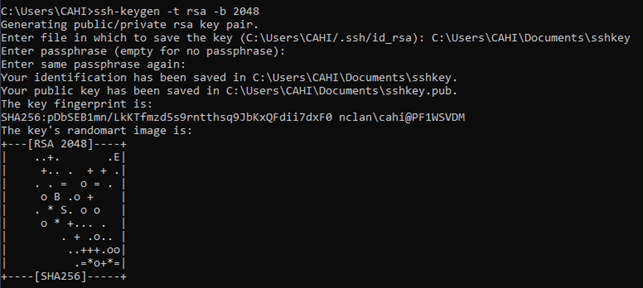
As you can see in the image above, Command Prompt suggest a place to save it an the name of the file. You can separate were it saves the file from the filename on backslash (\) and slash (/). Backslash specifies the file path (C:\Users\CAHI) and is both a new directory (/.ssh) and the filename (/id\_rsa). Here you must press “enter”, then the file will be saved in the user’s directory (C:\Users\CAHI), in a directory it creates (/.ssh/).

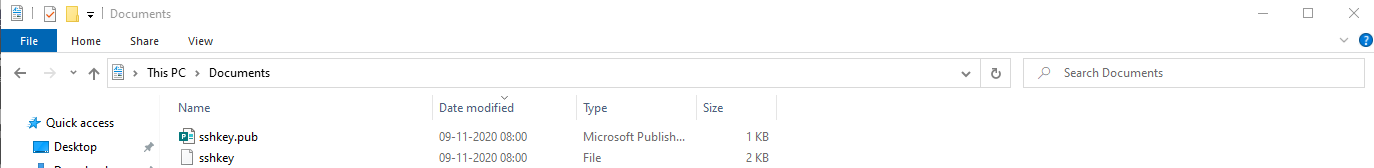
However, it can be specifies where the file is to be saved. An example of this could be “C:\Users\SAHI\Documents\sshkey”. In this example it will be saved in the Documents directory and the file will be named sshkey.

After pressing “enter” or specifying another location to save the file, the Command Prompt will ask for a password for the files. It is possible not to specify a password, however, it is recommended to enter a password. Please note that you must enter the password twice, which is shown in the image below:



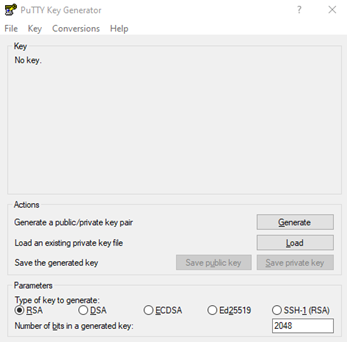
Once you have entered the password, the SSH-key will be generated and shown in the file directory, the SSH-key should look like the image below.





#### PuTTY:

Another way to generate an SSH-key is by using PuTTY. PuTTY can be downloaded from: <https://www.putty.org/> and should look like the image below, when you open it.

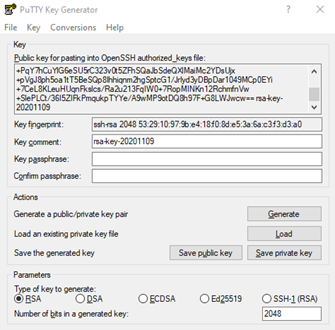


At the bottom of the PuTTY program, we can define different parameters. You must choose “rsa” and make sure the number of “bits” are 2048, which is shown in the image below.

To generate a SSH-key with these parameters, you must press “Generate”. After you have pressed “Generate” you must move the mouse over the empty field in the program. It will then use the mouse input to generate the SSH-key, which is shown in the following image:



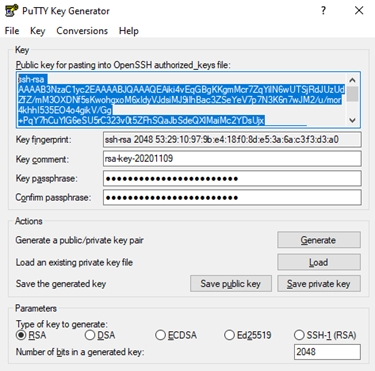
Once you have generate the SSH-key, the “save pblic key” and “save private key” will no longer be disabled and you will have the option to save the files. In addition it is possible to enter a password for the files, which is recommended.



When you have saved the files, both the Public and Private, you must open the Public SSH-key in Notepad or and other text application. Here you must change the text that is displayed in the Public SSH-key, to the SSH-key that is displayed at the top of PuTTY.

You can copy the text by highlighting it and then copy (Ctrl + C) + paste (Ctrl + V) it into the text program. This key should then replace the text in the file, so only the SSH-key from PuTTY is in the file. Remember to copy to copy all the text from PuTTY (Ctrl + A), as you can scroll through the window in PuTTY, which contains the whole key.

In the image below you can se, what it should look like, however it should just contain the SSH-key, that your program has generated.





Please note: The examples above are just two examples out of several, of how to make an SSH-key. If you are familiar with other programs, these can easily be used.

### Access the SFTP server

To access the SFTP server the following is needed:

1. A sender system with service protocol SFTP needs to be set up through Administrative Access (AA)
   1. A valid SSH key-pair: a public key and a private key. How to generate this can be found [here](https://digidp.atlassian.net/l/c/sExdoYrf).
      1. **Note:** the easiest method is to use the Command Prompt, as the PuTTy option may generate the private key in a format that is too new to work.
      2. **Note:** when creating the sender system in Administrative Access the value of IP field does not matter for non-prod environments when you are connected to the Netcompany VPN as it is whitelisted.

When the sender system has been set up the SFTP server can be accessed. You will need:

1. The SSH username from the sender system. It is called “SSH brugernavn” in AA.
2. The private key asociated with the public key you uploaded for the sender system in AA.

The easiest way to access the SFTP server is through the “links” folder in the devops/tools/kitty

1. Run WinSCP.exe
2. Create a new site with:
   1. File protocol = SFTP
   2. Host name = [sftp.test.digitalpost.dk](http://sftp.test.digitalpost.dk) (for non-prod environments)
   3. Port number = 22
   4. User name = the “SSH brugernavn” from your sender system
   5. Password = Blank
      1. Press “Advanced” and go to SSH → Authentication
      2. Upload the SSH private key matching the public key of your sender system. Press OK
   6. Press login

## Software Development Life Cycle (SDLC) for the API

This page describes the design for the management of Life Cycle for the API in Digital Post.

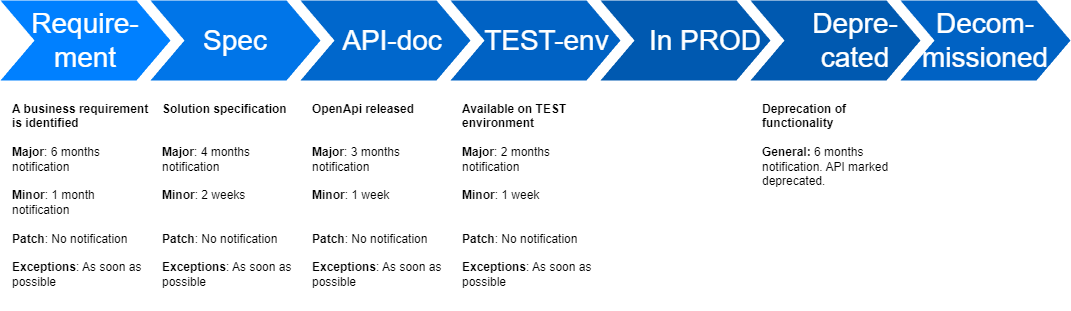
### Overall strategy

Breaking changes introduces complexity and reduces maintainability, so first rule in this overall strategy is: don't do breaking changes. Avoid them if possible and thereby avoid the versioning of APIs. Do enhancements in a backwards compatible way. This can be done by adding new properties or entities that can be safely ignored by older clients. Introduction of versioning should only be attempted if all other attempts fails.

### Versioning semantics (change category)

* Patch (x.x.**x**)
  + A patch that corrects small errors or adjusts the schema to allow for a more correct use of the standard.
  + A patch will always be backwards compatible.
  + Patches are not preceded by a notification.
  + No hearing is needed
* Minor (x.**x**.x)
  + A minor change allows for smaller (minor) changes to the specification. Changes must still be backwards compatible. A MeMo of previous version can be validated using the new schema, but a new version MeMo can not necessarily be validated against the old schema.
  + Examples
    - changing a required field to be non-required
    - adding a new non-required field
  + Changes should be preceded by a notification a least 1 month in advance.
  + No hearing is needed
  + A minimum of 4 months must pass before such a minor changed schema becomes obligatory to use.
  + Updated OpenApi specification will be published at least one week before TEST environment is updated.
  + Only one minor version is available at a time.
* Major (**x**.x.x)
  + A major update will change the schema in a non-backwards compatible way.
  + It could be adding new required fields to live up to some new regulatory rules.
  + Changes should be notified a least 6 months in advance.
  + Specification must be published at least 4 months in advance.
  + A minimum of 6 months must pass before such a major changed schema becomes obligatory to use.
  + A minimum of 3 months must pass after schema becomes obligatory to previous version is no longer supported.
* Exceptions
  + Exceptions to the above may occur in cases of emergency, such as security breaches.

The process shown below describes how a version of a service is controlled through its life cycle. The periods in the figure are tentative and are to be final decided.



## OIO OpenID Connect to Digital Post

|  |
| --- |
| This section not relevant for sender- and recipient systems as they are expected to use mutual SSL when integration with the Digital Post REST API |

In-order to integrate with Digital Post, front-end clients must utilize [OpenID Connect](https://openid.net/specs/openid-connect-core-1_0.html) (OIDC) when accessing the Digital Post REST API. Digital Post as a OpenID Connect provider (OP) adheres to the [OIO OpenID Connect profile](https://digst.dk/it-loesninger/standarder/openid-connect-profiler/) for OP, specifically a subset of the OIO OIDC profile as outlined below.

### OIO OIDC profile - Digital Post subset

**Background**

The OIO OIDC profile contains the outline for full-fledged OIDC multi-tenant support for NemLog-in, with support slated for sometime in the future. This is intended to support all OIDC needs across the public business-domain for all types of clients.

Implementing 1-1 OIO OIDC support into Digital Post is considered out-of-scope, instead Digital Post supports a subset of the OIO OIDC profile, simplified and tailored around the Digital Post use-cases and clients. Unless noted below, Digital Post OIDC adheres to the OIO OIDC profile.

**Client types**

Digital Post supports OIO "Native app" and "Web/JS app with a Backend" client types, as these clients are expected to be able to protect issued refresh tokens.

"Web/JS app without a Backend" (aka. refresh-token-rotation) is **unsupported**, as the security compromises to achieve a satisfactory user experience was deemed unacceptable by the Digital Post client forum.

**"Service token"**

The opaque "access token" and "service token" exchange outlined in the OIO OIDC profile is unsupported by Digital Post.  
Digital Post provides the OIDC access token (JWT), id token (JWT) and refresh token (opaque), in order to support common OIDC frameworks utilized by clients.

**Scopes**

Digital Post supports one auto-approved scope: openid, this scope encompasses access to Digital Post via JWT.

**OIO JWT profile**

The Digital Post JWT doesn't include the Assurance Level claims aal and ial.

When the Assurance Level is provided by a NSIS-compliant identity-provider, a service-provider is not allowed to repackage that information (source: NemLog-in).

The priv claim is omitted as privileges are administrated exclusively in Digital Post (separately from NemLog-in), and the verbose structure generates unnecessarily large values. Instead, the privileges specific to the Digital Post domain are present in the dpprivclaim. For users granted 55 or more privileges the dpprivref supports requesting the full set of privileges granted a user.

Besides the OIO JWT profile claims, Digital Post includes claims specific to the Digital Post domain.

* The authorities claim contains the unscoped authorities (aka roles) granted an identity (dpiid), specific to the Digital Post domain. The authorities also contain the privilege types, in order to support simplified access evaluation for the domain.
* The client\_id claim contains the OIO OIDC client ID.
* The dpiid claim contains the Digital Post identity ID, an opaque reference specific to the Digital Post domain principal.
* The dpprivref claim contains a URI that represents a reference to the Digital Post privileges granted an identity (dpiid). If this claim is present, the user is granted more claims than is supported by the header limits imposed by various user agents & backend interfaces. To avoid the header limit and excessive I/O the dpprivref claim is triggered for users granted 55 or more privileges. The dppriv claim can then be requested using the URI value ~ utilizing the /userinfo endpoint.
  + Example:

|  |
| --- |
| { .. "dpprivref" : "https://test.digitalpost.dk/auth/oauth/userinfo" .. } |

* The dppriv claim contains Digital Post privileges granted an identity (dpiid) in a compact format. For users granted <= 54 privileges the claim is included in the JWT, otherwise see dpprivref.
* The scopeclaim contains the scopes granted the client. Note that currently Digital Post only supports the scope openid.
* The sub claim is reserved for future use by NemLog-in3, currently the sub claim mirrors the dpiid claim to adhere to OAuth 2.0 JWT profile.

### OIDC client enrollment in Digital Post

In-order to register your client with Digital Post, OP requires the following information before a client is authorized with OP.

|  |  |
| --- | --- |
| **Information** | **Description** |
| Client name | Human readable name for OIDC client |
| Type | OIDC client type to register. Must be native app or Web/JS app with a Backend |
| Client ID | Unique ID for the view client - can be human readable or opaque |
| Client secret | Minimum 32 characters |
| Redirect URL | Redirect URIs can be any valid URI, i.e. both custom URL schemes (myapp://) as well as HTTP/S schemes are allowed. Note that the full redirect URI must be supplied.  A test-client is allowed to register multiple URLs, in-order to facilitate development & testing. Test clients are also permitted the use of HTTP-only schemes. |
| Post logout redirect URL (Digital Post) | Post logout redirect URLs used for logging a user out of Digital Post. Redirect URIs for logging out of Digital Post can be any valid URI, i.e. both custom URL schemes (myapp://) as well as HTTP/S schemes are allowed. Note that the full redirect URI must be supplied.  A test-client is allowed to register multiple URLs, in-order to facilitate development & testing. Test clients are also permitted the use of HTTP-only schemes. |
| Post logout redirect URL (NemLog-in) | Post logout redirect URLs for ending a user session in NemLog-in. Redirect URIs for logging out of NemLog-in are limited to HTTP/S-only schemes. Note that the full redirect URI must be supplied.  A test-client is allowed to register multiple URLs, in-order to facilitate development & testing. Test clients are also permitted the use of HTTP-only schemes. |
| AppSwitch return URI (only relevant native app clients) | The return URI the view client wishes to use for the MitID AppSwitch functionality. The same URI can be used across platforms. |

For details about logout, please see the section below regarding “RP-initiated Logout”.

An openid-configuration is available in each environment under the .well-known endpoint, e.g. <https://test.digitalpost.dk/auth/oauth/.well-known/openid-configuration>.

### RP-initiated Logout

Digital Post supports [OpenID Connect RP-Initiated Logout 1.0](https://openid.net/specs/openid-connect-rpinitiated-1_0.html), unless an exception is specified here.

#### Session

As the RP-initiated specification doesn’t specify how the OpenID Connect provider is expected to recognize a given session, Digital Post has chosen the access-token to represent an authorized session.

In-order to logout using the end\_session\_endpoint the request MUST include an access-token representing the Digital Post session. An expired access-token is also accepted. The access-token MUST be included as a Bearer authorization header.

#### Logout

A login in Digital Post often generates two “sessions” - one in NemLog-in and one in Digital Post. To ensure that all sessions are terminated when a user logs out of a view client it is important that a logout to both providers are initiated.

**Logout in NemLog-in**

View clients of the type “Web/JS app with a Backend” MUST terminate the users NemLog-in session when they log out of the view client.

To logout of NemLog-in requires calling the /auth/s9/multi-realm/logout endpoint exposed by Digital Post. Each view client have their own realm they MUST use when logging in and out of NemLog-in through Digital Post provided via the idp query parameter, see the section “NemLog-in Realms” below for details.

**Logout in Digital Post**

View clients of the type “Native app” MUST revoke the current refresh token if a new user is enrolling their device for Digital Post.

To logout of Digital Post, i.e. revoke the refresh token and any access tokens issued using the refresh token, the end\_session\_endpoint MUST be called. The request MUST include an id\_token\_hint to ensure the issued refresh token can be identified and revoked. Furthermore, the request MAY include a post\_logout\_redirect\_uri for redirecting the user after a successful logout of Digital Post as well as a state parameter if the view client wishes to maintain state as outlined in the RP-initiated specification.

**Logout notes**

Please note that while the OIDC specification outlines that a logout can be initiated by redirecting to the OP’s logout endpoint, Digital Post also supports GET/POST directly to the logout endpoint as a redirect request may exceed URI length limitation imposed by common browsers.

Note also that Digital Post delegates the responsibility for prompting the user to the client, in-line with NemLog-in’s current behavior and as such ignores any ui\_locales parameters.

### NemLog-in Realms

Each view client have their own realm they MUST use when authenticating users with NemLog-in as indicated below. The realms MUST be used during login and logout of NemLog-in. Login is handled through the authorization code grant flow and logout is done through the /auth/s9/multi-realm/logout endpoint.

|  |  |
| --- | --- |
| **View client** | **Realm (idp)** |
| Public view clients (borgerdk, virk) | nemlogin |
| mit.dk | mit-dk-nemlogin |
| e-Boks | e-boks-nemlogin |

The realm is indicated by supplying the idp query parameter during the authorization code grant flow or logout. It is not necessary to supply the idp query parameter when requesting an access token after login or during the refresh token flow.

#### Examples

**Authorization code grant flow**

For a public view client to log a user into Digital Post the authorization request would look as follows

GET https://test.digitalpost.dk/auth/oauth/authorize?idp=nemlogin&client\_id=...

**Termination of NemLog-in session (logout)**

For a public view client to log a user out of NemLog-in the endpoint /auth/s9/multi-realm/logout must be used together with the idp query parameter

GET https://test.digitapost.dk/auth/s9/multi-realm/logout?idp=nemlogin&f=...

In this instance the f query parameter is the post logout redirect URI (NemLog-in) provided during the client enrollment and is restricted to HTTP/S scheme only as outlined above.

### MitID AppSwitch support

Digital Post has support for MitID AppSwitch with the rollout of release 64. To make use of the AppSwitch functionality two query parameters must be supplied to Digital Post during the authorization code grant flow as follows

|  |  |
| --- | --- |
| **Query parameter** | **Description** |
| nemloginAppswitchReturnURI | The full return URI used by the MitID app to return the user back to the app which initiated the login. |
| nemloginAppswitchPlatform | An enum indication which platform the user is on. Must be either iOS or Android. |

### eIDAS eID Gateway

Similar to NemLog-in, Digital Post exposes a realm for authentication with the eIDAS eID Gateway. The eID realm can be used by supplying the idp query parameter with the value eid.

## Connect to Digital Post: Test and Prod

Whether you are testing or your systems have been built and/or adapted to the new interfaces they need to be connected to Digital Post on the test or prod environment respectively.

Be aware of your MOCES/VOCES/FOCES certificates: They are not the same and it depends on whether you are trying to connect to the test- or prod environment. Test certificates for test environment and prod certificates for prod environment.

Note: Connecting and configuring your sender- and receiver systems for Digital Post is done via the new administration portal ‘Administrativ adgang’. The following steps are only from a technical perspective for preparing your systems. For more information on how to set up your systems in the administration portal look under “References”.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Step** | **System** | | **Protocol** | | | **Where?** | **DP assistance** |
| Sender system | Receiver system | REST | SMTP | SFTP |
| Get NemLog-In VOCES/FOCES certificate | X | X | X | X |  | NemLog-in: <https://www.nemid.nu/> | NA |
| Generate SSH keys | X |  |  |  | X | Own servers | NA |
| Get employee login for NemLog-in |  |  |  |  |  | NemLog-in: <https://erhverv.pp.certifikat.dk/produkter/nemid_medarbejdersignatur/> | For test environment: See the section “Access to the administration portal on the test environment” |
| Integration to Contact registry and System registry | X |  | X |  |  | Own system | OpenAPI |
| Prepare mail server with DMARC, DKIM and SPF | X | X |  | X |  | Own system | NA |
| Prepare integration to DP SFTP | X |  |  |  | X | Own system | Reference-Sender-system |
| Integration to Distribution | X | X | X | X | X | Own system | Reference-Sender-/receiver-system |
| Administrate systems and technical contact person | X | X | X | X | X | Test:  <https://admin.test.digitalpost.dk/login>  Prod:  [https://admin.digitalpost.dk/login](https://admin.test.digitalpost.dk/login) | Manuals (see ”Reference”). |
| Administrate access to prod |  |  |  |  |  | Rights management portal (Rettighedsportalen)  <https://rettighedsportal.digitalpost.dk/home> | Manuals (see “Reference”). |
| Establish contact structure |  | X | X | X | X | Administration Portal: ’Administrativ Adgang’ | Manuals (see ”Reference”). |

### Roles in the administration portal “Administrativ Adgang”.

Roles and priviliges are defined in Rights Management Portal. See “References” for manual.

## Additional configuration support

It is possible to request additional configuration support for technical issues regarding your sender- and receiver systems.

You can create a case in our service desk <https://digidp.atlassian.net/servicedesk/customer/portals>