



QUICK-SIGN PORTAL INSTALLATION

The specifications and information in this document are subject to change without notice. Companies, names, and data used in examples herein are fictitious unless otherwise noted. This document may not be copied or distributed by any means, in whole or in part, for any reason, without the express written permission of RCDevs Security.

WebADM and OpenOTP are trademarks of RCDevs. All further trademarks are the property of their respective owners.

No guarantee is given for the correctness of the information contained in this document. Please send any comments or corrections to info@rcdevs.com.

Quick-Sign Portal Installation

[Quick-Sign Portal](#) [Install](#) [signature](#) [seal](#)

1. Introduction

The Quick-Sign Portal is very easy to install and configure; it can be integrated in an existing environment. It is a web PHP application which interacts with a mail server to permit users to send and sign documents. We recommend to install it on a dedicated machine and not on your WebADM/OpenOTP servers. This Quick-Sign portal is using the quicksign-milter, you must first configure the [Quick-Sign Milter](#) before deploying the Quick-Sign portal. WebADM/OpenOTP infrastructure must be already deployed and integrated with your LDAP backend. Your OpenOTP license must also support **Sign** option.

2. Installation

As indicated in the introduction, the Quick-Sign Portal needs several components to run. They are mandatory.

- › Web server with PHP activated
- › IMAP and SMTP server associated to a [Quick-Sign server](#)
- › LDAP server
 - › for authentication
 - › to retrieve information to fill the signers lists, the user's personal data, etc.
- › Redis server
 - › to store information about unsigned transactions such as rejected, cancelled, error (Portal Redis instance)
 - › to retrieve information from Quick-Sign server (Quick-Sign server Redis instance)
- › WebADM server for the junction to the OpenOTP process

Note

Note that it is possible to use only one Redis instance : the sessions from Quick-Sign server and from Portal are different to prevent data loss. Also, the Portal can run completely as standalone application : one server with its own mail server (independent of Company's), own LDAP (e.g. clone of the Company's).

2.1 RHEL/CentOS through RCDevs Repository

The installation of the Quick-Sign Milter as a service is done with the following command once RCDevs repository is installed:

```
yum install quicksign-milter
```

OR

```
dnf install quicksign-milter
```

2.2 Debian/Ubuntu through RCDevs Repository

The installation of the Quick-Sign Milter as a service is done with the following command once RCDevs repository is installed:

```
apt install quicksign-milter
```

3. Portal tree structure

The portal is an MVC application with the below structure. Only public folder has to be reachable by users; the system administrator should configure web server according to this recommendation. The core of the application is stored in the **app** folder, the public area stays in the **public** folder. The only file which has to be modified is the config.ini; it is located in **app/config** folder.

```
/qsPortal
├── app
│   ├── config
│   ├── controllers
│   ├── core
│   ├── locales
│   ├── media
│   ├── models
│   ├── utility
│   └── views
├── public
│   ├── css
│   ├── font
│   ├── img
│   └── js
```

The users documents are stored in another folder, different from the application. It permits an easier maintenance (backup, purge, etc.). The following tree example shows each user (IT Manager, HR, Rosy) who have 4 sub folders.

- › approved : contains all the document sent by the user and signed (so approved) by someone else.
- › mydocuments : we found here all the user's documents which can be sent for signature or sealing
- › sealed : all the documents the user sealed (signed for his own purpose)
- › signed : all the documents the user signed

All other documents (cancelled signatures, rejected signature, errors) are stored in Redis and are temporary. As you can see, each signed/sealed document is composed of

- › its original name concatenated with a base64 string
- › the same pattern for an index file which contains relevant information about the signature

All these stored files can be downloaded by the user.

```
documents/
├── IT Manager
│   ├── approved
│   │   ├── IT charter by Rebecca.pdf_<base64String>
│   │   ├── IT charter by Rebecca.pdf_<base64String>.idx
│   │   ├── Computer workstation security agreement.pdf_<base64String>
│   │   └── Computer workstation security agreement.pdf_<base64String>.idx
│   ├── mydocuments
│   │   ├── Computer workstation security agreement.pdf
│   │   ├── IT charter.pdf
│   │   ├── IT infrastructure project.pdf
│   │   ├── IT infrastructure project (validated).pdf
│   │   └── Network architecture.pdf
│   ├── sealed
│   │   ├── Computer workstation security agreement.pdf_<base64String>
│   │   ├── Computer workstation security agreement.pdf_<base64String>.idx
│   │   ├── IT infrastructure project (validated).pdf_<base64String>
│   │   └── IT infrastructure project (validated).pdf_<base64String>.idx
│   └── signed
│       ├── Alpha server purchase Q1-2023.pdf_<base64String>
│       ├── Alpha server purchase Q1-2023.pdf_<base64String>.idx
│       ├── CEO-IT service reorganization Q3-2022.pdf_<base64String>
│       ├── CEO-IT service reorganization Q3-2022.pdf_<base64String>.idx
│       ├── Subcontractor, confidentiality agreement Q4-2022.pdf_<base64String>
│       └── Subcontractor, confidentiality agreement Q4-2022.pdf_<base64String>.idx
├── HR
│   ├── approved
│   │   ├── Appointment 2022-11-11 by Pike.pdf_<base64String>
│   │   ├── Appointment 2022-11-11 by Pike.pdf_<base64String>.idx
│   │   ├── Long term contract by Rosy.pdf_<base64String>
│   │   └── Long term contract by Rosy.pdf_<base64String>.idx
│   ├── mydocuments
│   │   ├── Appointment 2022-11-11.pdf
│   │   ├── Long term contract.txt
│   │   ├── Long term contract template.pdf
│   │   └── Reorg project.pdf
│   ├── sealed
│   │   ├── Long term contract template.pdf_<base64String>
│   │   └── Long term contract template.pdf_<base64String>.idx
│   └── signed
├── Rosy
│   ├── ...
│   └── ...
```

4. Configuration

Please find below, the standard Portal configuration file for domain “*server.com*”.

4.1 Standard configuration file *config.ini*

```
[Portal]
applicationFullName    = "QuickSign Portal"
applicationFlag         = "QuickSignPortal"
portalId               = "a9a9a9a9-a9a9-a9a9-a9a9-a9a9a9a9a9a9"
; Do NOT forget the ending SLASH
url_base               = "/quicksign/"
refreshArray           = false
refreshDelay           = 5
[Logging]
logFileName            = "/var/log/quicksign/quicksign-portal.log"
level = WARNING
; DEBUG
; INFO
; WARNING
; ERROR
; CRITICAL
[Documents]
; Do NOT forget the ending DS
documentsFolder        = "/var/www/quicksign/documents/"
; System credentials for documents
documentsCredentials   = "0750"
[Postfix]
name                   = "server.com"
realDomain             = "server.com"
port                   = 143
type                   = ""
security               = ""
folder                 = "INBOX"
SMTPDebug              = false
signDomain             = "sign.server.com"
sealEmail              = "seal@sign.server.com"
manager               = "Postman"
password               = "mngPassword"
[Redis]
; Milter Redis
redisMilterHost        = "server.com"
redisMilterPort        = 6379
; Portal Redis (used for temporary status like "Error", "Rejected", etc.)
redisedFoldersHost     = "server.com"
redisedFoldersPort     = 6379
; Keep the intel inside the Portal Redis (in hours) ; this is just for Rejected, Cancelled & Error requests
keepAlive              = 120
[LDAP]
host                   = "webadm.server.com"
port                   = 389
; encryption can be none/ssl/tls
encrvption             = none
```

```

timeout                = 10
username               = "cn=admin,o=Root"
password              = "admPassword"
baseDn                = "o=Root"
filter                = "(objectclass=person)"
filterUser            = "(uid=%s)"
filterMail            = "(mail=%s)"
columnFirstname       = "givenname"
columnLastname        = "sn"
columnLogin           = "uid"
columnMail            = "mail"
[Media]
iconType              = "png"
iconExtension         = ".png"
extensionSeparator    = "_"
missing               = "_missing"

```

4.2 Parameters explanations

Variables	Meaning
[Portal]	
applicationFullname	Name to display especially in the top banner
applicationFlag	Short name used <ul style="list-style-type: none"> • to send Portal messages to milter • to read logs
portalId	Portal ID which permits to link the milter and the Portal emails. This ID has to be also written in the Milter configuration file to perform the link.
url_base	url used in the web browser, root of the Portal web site
refreshArray	Permits to refresh automatically the files lists
refreshDelay	Delay between two refreshes (in seconds)
[Logging]	
level = WARNING	Define log level ; can be DEBUG, INFO, WARNING, ERROR, CRITICAL
[Documents]	
documentsFolder	Root folder for documents storage. For each user's folder, there are 4 subfolders :

- *approved* for documents approved by user's colleagues
- *mydocuments* which contains user's documents
- *sealed* for user's sealed documents
- *signed* which contains documents signed by user

Note : the other Portal tabs are temporary virtual folders stored in Redis

documentsCredentials	Define credential for directory creation (disable on MS Windows)
[Postfix]	More information on PHP imap-open function
name	Mail Server name used to send signs and seals requests
realDomain	Real domain mail server as opposed to sign domain (below)
port	Mail Server port (cf. standard Postfix configuration)
type	pop or imap protocol or empty value
security	ssl, tls, notls or empty value
folder	INBOX folder
SMTPDebug	Debug protocol negotiations as boolean (default <i>false</i>)
signDomain	Signature domain, generally concatenation of <i>sign</i> and <i>real domain</i> value
sealEmail	Seal email ; it has to be seal@sign.server.com
manager	Manager of the mails from and to Portal. This special user will receive all emails from the milter to prevent standard users mailbox to be flooded
password	Manager password
[Redis]	Two Redis servers are needed. First, the Redis used by the milter with which the Portal can interact. Second, specific Portal Redis to manage cancellations, errors, rejections. These Redis servers can be merged into one thanks to the Portal prefix sessions
redisMilterHost	Milter Redis hostname
redisMilterPort	Milter Redis port
redisedFoldersHost	Portal Redis hostname
redisedFoldersPort	Portal Redis port
keepAlive	Time during which the information is kept in the Portal Redis (in hours)

refreshInterval	Time during which the information is kept in the portal cache (in hours)
[LDAP]	
	LDAP is used to authentication, to fill the users lists, the user personal information, etc.
host	LDAP server hostname
port	LDAP server port
encryption	Used to define encryption to LDAP (none/ssl/tls)
timeout	Standard LDAP timeout
username	Special user ID who is allowed to read the LDAP
password	Special user password
baseDn	LDAP base DN
filter	Used to search information about LDAP users (depends on your LDAP configuration). e.g. search in a group of <i>person</i> .
filterUser	Used to search information about LDAP users (depends on your LDAP configuration). e.g. search information according to the login name (here <i>uid</i>).
filterMail	Used to search information about LDAP users (depends on your LDAP configuration). e.g. search information according to the user's mail (here <i>mail</i>).
columnFirstname	Users' firstname (displayed in users lists)
columnLastname	Users' lastname (displayed in users lists)
columnLogin	LDAP User login ID
columnMail	LDAP User email
[Media]	
iconType	Type of pictures used in the portal
iconExtension	Extension of pictures used in the portal
extensionSeparator	Separator used to display pictures
missing	Display this picture is one is missing

especially for translation into foreign languages. No part of this manual may be reproduced in any way (photocopies, microfilm or other methods) or transformed into machine-readable language without the prior written permission of RCDevs Security S.A. The latter especially applies for data processing systems. RCDevs Security S.A. also reserves all communication rights (lectures, radio and television). The hardware and software names mentioned in this manual are most often the registered trademarks of the respective manufacturers and as such are subject to the statutory regulations. Product and brand names are the property of RCDevs Security. © 2024 RCDevs Security S.A., All Rights Reserved