

LSE LinOTP

Module Development Manual

© LSE Leading Security Experts GmbH

version 2.0.1
Apr 13, 2010



LSE Leading Security Experts GmbH
Robert-Koch-Str. 9
64331 Weiterstadt
Germany
Tel: +49 (0) 6151 / 9067 - 0
www.lsexperts.de

Contents

1 Introduction.....	3
1.1 System Overview.....	3
1.1.1 Components.....	3
1.2 License.....	5
1.3 Support.....	5
2 Authentication modules.....	6
2.1 Interface specification.....	6
2.1.1 check.....	6
2.1.2 simplecheck.....	7

1 Introduction

This manual describes the interfaces you need to know, if you plan to develop further modules for LinOTP 2.

1.1 System Overview

LSE LinOTP 2 is a framework that provides most flexible authentication with One Time Passwords (OTP). The dimension of flexibility is shown in this section.

1.1.1 Components

LSE LinOTP consists of several component types.

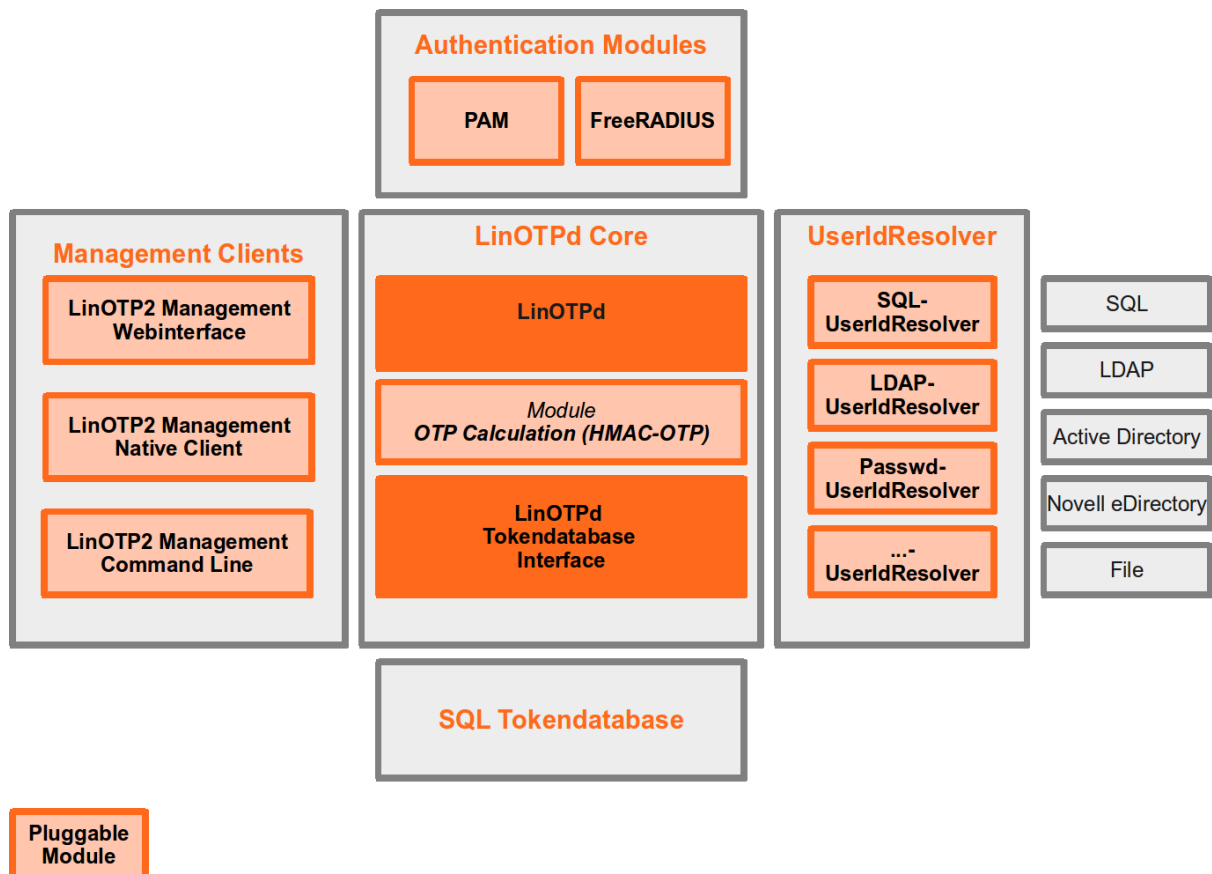


Figure 1: Components of LinOTP

The components are shown in Figure 1 'Components of LinOTP'.

The components are like pluggable modules that may or may not be used with the LinOTPd core. As the interfaces for component types are well defined, it is straightforward to implement new components without any impact on the LinOTPd core. Such new components can be easily plugged in during runtime.

LinOTPd core

This is the central server part, the LinOTPd core. LinOTPd is implemented in python and well tested with python 2.6.4 and will also run with python 2.5. It uses pylons for the communication of the other components with the core. Thus the other components like management clients and authentication modules will issue http requests to communicate with the LinOTPd core.

LinOTP stores all token information in an SQL database. MySQL, PostgreSQL and SQLite were tested successfully.

OTP Calculation

The LinOTPd core is capable of using different OTP algorithms for calculating the OTP values. Each OTP token is stored with its token type, that identifies how the OTP value is calculated.

At the moment the event based HMAC-OTP¹ and time based mOTP² is supported.

New modules for new tokens can be plugged in easily.

UserIdResolver

LinOTP uses external userstores to identify users to whom tokens get assigned. LinOTP does not modify the userstore. A UserIdResolver has a well defined interface. So a new useridresolver can be plugged into LinOTP so that users from another userstore can be used. LinOTP can use several userirdresolvers at the same time. This LinOTP Enterprise Edition provides a PasswdUserId-Resolver to access users from /etc/passwd, an LDAPIdResolver to use users from LDAP directories like openLDAP, Active Directory or Novell eDirectory and SQLUserIdResolver to access users in SQL databases.

So your users need to be stored either in an /etc/passwd file, in some kind of SQL database or in some kind of LDAP directory (OpenLDAP, AD, eDirectory).

Authentication Modules

LinOTP does not bind you to any authentication method. Although RADIUS (Remote Authentication Dial In User Service) is an often used protocol, it might not fit all the needs or might be sometimes to costly to set up. So LinOTP also provides an interface for authenticating users. At the moment LinOTP Enterprise Edition provides an authentication module for FreeRADIUS but also for the Unix PAM stack (Pluggable Authentication Module). Again as the LinOTP authentication interface is very lean, other authentication modules can be implemented easily.

Management Clients (adminclient)

The Enterprise Edition provides a GUI for Windows and Linux. This can be used for all administrative and token management tasks.

Using a management client the LinOTPd server and its UserIdResolvers can be configured. Tokens can be imported, enrolled, assigned or disabled. The management client also provides a view to the available users in the configured userstore. Of course - as LinOTP only has read access to the userstore - the users will not be managed within LinOTP.

1 RFC 4226

2 motp.sourceforge.net

1.2 License

The LinOTPd Server, the PasswdIdResolver, linotpadm.py, pam_linotp and rlm_linotp2 are under GNU General Public License (GPL) Version 2.

All other UserIdResolvers (LDAPIdResolver, SQLIdResolver) and the Management Client glinotpadm.py, the documentation and scripts are to be licensed from LSE Leading Security Experts GmbH.

1.3 Support

With licensing the LSE LinOTP 2 Enterprise Edition from LSE Leading Security Experts GmbH you are entitle to get support.

Support contact for LinOTP: linotp@lsexperts.de

Support for other issues: support@lsexperts.de

2 Authentication modules

Authentication is done by talking to LinOTP via a well defined web service interface. Either an authentication module or an application itself can handle this interface.

At the moment LinOTP comes with two authentication modules:

- LinOTP rlm_linotp2 for FreeRADIUS and
- LinOTP pam_linotp2 for the PAM system.

2.1 Interface specification

There are two interfaces, a full blown and a simple one:

- <https://linotpserver/validate/check>
- <https://linotpserver/validate/simplecheck>

Both interfaces take these three parameters:

- **user**: the username of the user who is authenticating.
- **pass**: the password (consisting of OTP Pin and OTP value) of the user, who is authenticating.
- **realm**: an optional value, describing the instance of the UserIdResolver if several clients of UserIdResolvers are used.

2.1.1 check

Request:

```
1 https://linotpserver/validate/check?user=koelbel&pass=test354678
```

Response of missing parameter or invalid password:

```
2 {
3     * jsonrpc: "2.0"
4     * result: {
5         o status: true
6         o value: false
7     }
8     * id: 0
9 }
```

Response of any other error:

```
10 {
11     * jsonrpc: "2.0"
12     * result: {
13         o status: false
```

```
14     o value: false
15   }
16 * id: 0
```

Response of successful authentication:

```
17 {
18   * jsonrpc: "2.0"
19   * result: {
20     o status: true
21     o value: true
22   }
23   * id: 0
24 }
```

2.1.2 simplecheck

This interface can be used, when JSON parsing is not possible or suitable.

Request:

```
25 https://linotpserver/validate/simplecheck?user=koelbel&pass=test354678
```

Response of failed authentication:

```
26 :- (
```

Response of any other error:

```
27 :-/
```

Response of successful authentication:

```
28 :-)
```