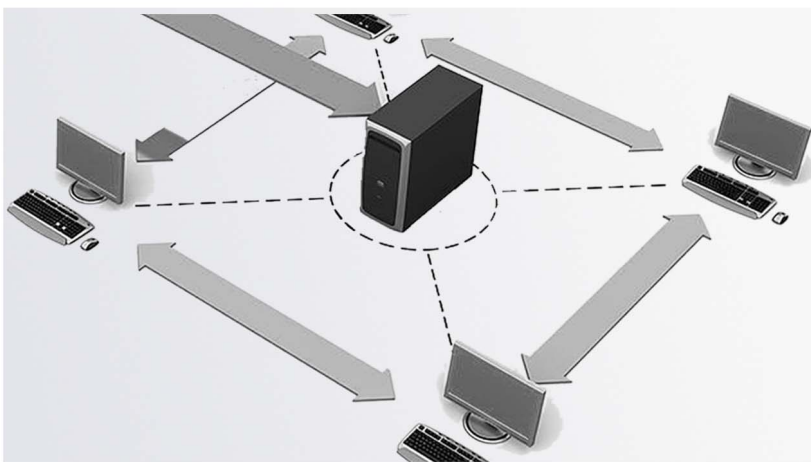
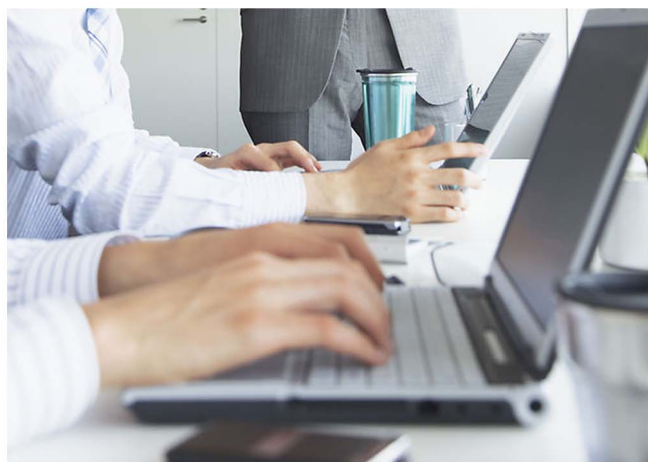


UniOTP Server Manual



SecuTech Solution Inc.

<http://www.esecutech.com>

Date	Version	Modification
2011-5-11	V1.0	First version

Summary

What is this document used for	1
How to use C/C++ SDK.....	1
Preparation	1
Add support for UniOTP dynamic password authentication	2
SDK Operation procedure and Important points.....	2
Functions interfaces	3
SECUOTP_INITENV	3
SECUOTP_CLEANUP().....	3
SECUOTP_GETLASTERROR()	4
SECUOTP_AUTH()	4
SECUOTP_SYNCHRONIZE()	5
SECUOTP_SETUSERPIN()	6
SECUOTP_RESETPIN().....	6
SECUOTP_SETAUTHTYPE()	7
SECUOTP_ADDUSER()	7
SECUOTP_DELETEUSER().....	8
SECUOTP_SETUSERINVALID().....	9
SECUOTP_SETUSERLOCK()	9
SECUOTP_FREE()	10
SECUOTP_TOKENIMPORT()	11
SECUOTP_USERIMPORT()	11
SECUOTP_TOKENDELETE	12
SECUOTP_MAKELOG()	13
Operation code list	14
SECUOTP_REBIND()	14
Error code list.....	15
Static data definition	18
PHP SDK Operation procedure.....	18
Preparation	18
Add support for UniOTP dynamic password	18
Function interfaces	19
SECUOTP_AUTH()	19
SECUOTP_SYNCHRONIZE()	20
SECUOTP_SETUSERPIN()	20
SECUOTP_RESETPIN().....	21
SECUOTP_SETAUTHTYPE()	21

SECUOTP_ADDUSER()	22
SECUOTP_DELETEUSER()	23
SECUOTP_SETUSERINVALID()	23
SECUOTP_SETUSERLOCK()	24
SECUOTP_TOKENIMPORT()	24
SECUOTP_USERIMPORT()	25
SECUOTP_TOKENDELETE()	26
SECUOTP_MAKELOG()	26
SECUOTP_REBIND()	27
PHP extension SDK operation procedure	28
Preparation	28
Add support for UniOTP dynamic password	28
SDK Operation procedure and important points	28
Function interfaces	29
SECUOTP_INITENV()	29
SECUOTP_CLEANUP()	29
SECUOTP_GETLASTERROR()	30
SECUOTP_AUTH()	30
SECUOTP_SYNCHRONIZE()	31
SECUOTP_SETUSERPIN()	31
SECUOTP_RESETPIN()	32
SECUOTP_SETAUTHTYPE()	32
SECUOTP_ADDUSER()	33
SECUOTP_DELETEUSER()	34
SECUOTP_SETUSERINVALID()	34
SECUOTP_SETUSERLOCK()	35
SECUOTP_TOKENIMPORT()	36
SECUOTP_USERIMPORT()	36
SECUOTP_TOKENDELETE()	37
SECUOTP_MAKELOG()	37
SECUOTP_REBIND()	38

What is this document used for

This document introduces how to use UniOTP Server SDK. It is for Developers and Maintenance Engineer who wish to use UniOTP Server SDK to do some second development or build some systems using UniOTP dynamic password authentication. It includes some explanations about how to use Agent SDK and the points that you should pay attention to.

How to use C/C++ SDK

Preparation

1. The SDK server needs to operate a database. Inside the C/C++ version SDK uses ODBC to access to the database. That's why, before beginning to develop, it is required to install a database (for now, UniOTP supports Oracle/MySQL/Postgresql/SQL Server) and to configure the ODBC driver for database products access.
2. In the distributable package, you can find a file named initsqlscript that contains the SQL initialization script for the 4 types of database described before. You must use your database administration tool (for example SQLPlus, MySQL Client, etc.) to create a new database, or you can use one of your own databases. Use the SQL file inside the folder initsqlscript to initialize the database tables for UniOTP for UniOTP dynamic password authentication system.
3. Once you finish preparing the database, create an ODBC data source for this newly created database.(Use the database prepared in step 2)
4. Before beginning developing using the SDK, you need to create a configuration file (use the file otp_conf.com as an example)
You just need to use the configuration file created in step 2 to reconfigure the information related to the settings file. Use the function secuinitenv(char *pszConfpath) to add settings file (such as otp_conf.con) and to complete initialization of information in the SDK.

Add support for UniOTP dynamic password authentication

1. Add the Server SDK library files (dynamic or static library) and its header files (otp_corec32.h) to the project.
2. Copy the database configuration file created in the “preparation” part step 4 to the appropriate place in your project.(you can actually put the configuration file anywhere , but it is recommended to put it in the project directory)
3. Use API introduced later in this document to perform operations related to authentication.

SDK Operation procedure and Important points

Operation procedure

1. Use the API secuotp_initenv(char *pszConfpath) to initialize the SDK Settings information.
2. Use other interface to complete business related treatments.
3. Use Secuotp_cleanup()API to perform maintenance operations such as releasing resources.
For more details about other interfaces way of use, please see the sample.

Important points

1. The function secuotp_initenv(char *pszConfpath) is used to initialize settings information, when using the SDK, you only need to call the function once.
2. Once you finish using UniOTP SDK, call the secuotp_cleanup() function to perform cleaning operations.
3. secuotp_tokenimport(char*pszTokenfile, char *pszSecret, char **pstrErrinfo) function is used to import token in batch , pstrErrinfo is assigned a value by the internal API. If you don't need this memory area any longer, please use secuotp_free(void **spsbuf) to release it.
4. secuotp_userimport(char *pszUserfile, char *pszSecret, char **pstrErrinfo) this function is used to import users in batch , pstrErrinfo is assigned a value by the internal API. If you don't need this memory area any longer , please use secuotp_free(void **spsbuf) to release it
5. secuotp_userimport(char *pszUserfile, char *pszSecret, char **pstrErrinfo) this function is used to import users in batch, pszSecret points to the user record file decryption key, if pszSecret is NULL or the buffer data is null, then it shows the file isn't encrypted . We recommend not to use encryption.

Functions interfaces

SECUOTP_INITENV

Prototype

```
int secuotp_initenv(char *pszConfpath);
```

Description

Initialize environment, set the path for the configuration file.

Parameters

PARAMETERS	DESCRIPTION
char *pszConfpath	[IN] configuration file path

Return Value

RETURN VALUE	DESCRIPTION
SECUOTP_ERROR_SUCCESS	the function was executed successfully
SECUOTP_ERROR_INVALID_PARAMETER	Invalid parameters
SECUOTP_ERROR_ALLOCHANDLE_FAILED	Failed to allocate database handle (ODBC)
SECUOTP_ERROR_SETENVATTR_FAILED	Failed to configure ODBC driver version
SECUOTP_ERROR_GETCONFIGFILE_FAILED	Configuration file error or failed to read configuration information

SECUOTP_CLEANUP()

Prototype

```
int secuotp_cleanup();
```

Description

Complete a few cleaning operations, before the process ends or before uninstalling dynamic libraries, you need to call this function to release handles that haven't been unreleased.

Parameters

none

Return value

RETURN VALUE	DESCRIPTION
SECUOTP_ERROR_SUCCESS	function executed successfully
SECUOTP_ERROR_FREEHANDL E_FAILED	error while releasing handles

SECUOTP_GETLASTERROR()**Prototype**

```
unsigned long secuotp_getlasterror();
```

Description

Get last error code

Parameters:

None

Return value:

Code of the last error that happened.

SECUOTP_AUTH()**Prototype**

```
int secuotp_auth(char *pszUsername, char *pszPwd, unsigned int  
*pChallenge);
```

Description

Perform dynamic password user authentication

Parameters

PARAMETERS	DESCRIPTION
char *pszUsername	[IN] Username for the user waiting to be authenticated
char *pszPwd	[IN] password (OTP or OTP+PIN)
unsigned int *pChallenge	[OUT] challenge code returned by the server

Return Value

RETURN VALUE	DESCRIPTION
SECUOTP_ERROR_SUCCESS	Authentication succeeded
SECUOTP_ERROR_CHALLENGE	the challenge information from the client is needed

Others (See the [error code list](#) for more details) Authentication failed or error happened when authenticating.

SECUOTP_SYNCHRONIZE()

Prototype

```
int secuotp_synchronize(char *pszUsername, char *pszOtp1, char *pszOtp2) ;
```

Description

Perform user token synchronization

Parameters

PARAMETERS	DESCRIPTION
char *pszUsername	[IN] username of the user's token that needs to be synchronized
char *pszOtp1	[IN] For the synchronization to be completed, you need to provide two consecutive dynamic passwords. This parameter is the first of the two passwords
char *pszOtp2	[IN] The second of the two passwords

Return Value

RETURN VALUE	DESCRIPTION
SECUOTP_ERROR_SUCCESS	Synchronization succeeded.

Others (See the [error code list](#) for more details): Synchronization failed or error happened while synchronizing.

SECUOTP_SETUSERPIN()

Prototype

```
int secuotp_setuserpin(char *pszUsername, char *pszPin);
```

Description

Configure user static PIN.

Parameters

PARAMETERS	DESCRIPTION
char *pszUsername	[IN] Username
char *pszPin	[IN] new static PIN for this user

Return Value

RETURN VALUE	DESCRIPTION
SECUOTP_ERROR_SUCCESS	Static PIN configured correctly

Others (See the [error code list](#) for more details): Failed to configure the static PIN

SECUOTP_RESETPIN()

Prototype

```
int secuotp_resetpin(char *pszUsername, char *pszOldpin, char *pszNewpin);
```

Description

Reset user static PIN

Parameters

PARAMETERS	DESCRIPTION
char *pszUsername	[IN] Username
char *pszOldpin	[IN] old static PIN
char *pszNewpin	[IN] new static PIN

Return value

RETURN VALUE	DESCRIPTION
SECUOTP_ERROR_SUCCESS	New PIN set successfully

Others(See the [error code list](#) for more details) Failed to set the new static PIN

SECUOTP_SETAUTHTYPE()

Prototype

```
int secuotp_setauthtype(char *pszUsername, int nAuthtype);
```

Description

Set authentication type

Parameters

PARAMETERS	DESCRIPTION
char *pszUsername	[IN] Username
int nAuthtype	[IN] Authentication type(1- use only OTP, 2-use OTP+PIN)

Return value

RETURN VALUE	DESCRIPTION
SECUOTP_ERROR_SUCCESS	authentication type set successfully

Others (See the [error code list](#) for more details) failed to set authentication type

SECUOTP_ADDUSER()

Prototype

```
int secuotp_adduser(char *pszUsername, char *pszUserid, char *pszContacttel, char *pszSerialnum, char *pszRemark, char *pszEmail, char *pszDomain, char *pszPin, char *pszPreusedate, int nAuthtype);
```

Description

Add a new user

Remark:

Before adding a user into the UniOTP Authentication System, Please make sure that the token allocated to the user has been imported into the database using the `secuotp_tokenimport()` function.

Parameters

PARAMETERS	DESCRIPTION
char *pszUsername	[IN] username of the user you want to add
char *pszUserid	[IN] User ID, for example Identity card number
char *pszContacttel	[IN] Phone number
char *pszSerialnum	[IN] serial number of the token allocated to the user
char *pszRemark	[IN] remarks/comments about the user
char *pszEmail	[IN] user Email address
char *pszDomain	[IN] <i>reserved</i> , NULL
char *pszPin	[IN] User static PIN
char *pszPreusedate	[IN] User registration date
int nAuthtype	[IN] Authentication date

Return value

RETURN VALUE	DESCRIPTION
SECUOTP_ERROR_SUCCESS	User registration successful

Others (See the [error code list](#) for more details) failed to register user

SECUOTP_DELETEUSER()

Prototype

```
int secuotp_deleteuser(char *pszUsername);
```

Description

Delete a user

Parameters

PARAMETERS	DESCRIPTION
char *pszUsername	[IN] username of the user you want to delete

Return value

RETURN VALUE	DESCRIPTION
SECUOTP_ERROR_SUCCESS	user deleted successfully

Others (See the [error code list](#) for more details) failed to delete user

SECUOTP_SETUSERINVALID()

Prototype

```
int secuotp_setuserinvalid(char *pszUsername, int nvalid);
```

Description

Set user validity status (used to report a token as “lost”)

Parameters

PARAMETERS	DESCRIPTION
char *pszUsername	[IN] Username
int nvalid	[IN] User validity status (0-valid 1-invalid)

Return value

RETURN VALUE	DESCRIPTION
SECUOTP_ERROR_SUCCESS	function called successfully

Others (See the [error code list](#) for more details) failed to call the function

SECUOTP_SETUSERLOCK()

Prototype

```
int secuotp_setuserlock(char *pszUsername, int nlock);
```

Description

Lock a user

Remark:

If nlock is set 1, then the user will be locked for 10 minutes

Parameters

PARAMETERS	DESCRIPTION
char *pszUsername	[IN] Username
int nlock	[IN] lock (0-not locked 1-locked)

Return value

RETURN VALUE	DESCRIPTION
SECUOTP_ERROR_SUCCESS	function called successfully

Others (See the [error code list](#) for more details) failed to call the function

SECUOTP_FREE()

Prototype

```
void secuotp_free(void **psbuf)
```

Description

Release the buffer (release the space created by the caller)

Parameters

PARAMETERS	DESCRIPTION
void **psbuf	[IN] first address of the area that you want to release

Return value

none

SECUOTP_TOKENIMPORT()

Prototype

```
int secuotp_tokenimport(char *pszTokenfile, char *pszSecret, char
**pstrErrinfo);
```

Description

Import tokens by batch

Parameters

PARAMETERS	DESCRIPTION
char *pszTokenfile	[IN] Token file path
char *pszSecret	[IN] decryption secret key
char *pstrErrinfo	[OUT] invalid records occur in batch import

note: this function for internal using assigns space used for storing invalid record for the caller , when caller no longer uses the space , please use secuotp_free(void **psbuf) to release the space pointed by pstrErrinfo

Return Value

RETURN VALUE	DESCRIPTION
SECUOTP_ERROR_SUCCESS	token batch import succeeded.
SECUOTP_ERROR_SUCCESS_WI THINFO	token batch import partially succeeded.(use pstrErrinfo to search for invalid entries)

Others (See the [error code list](#) for more details) token batch import failed

SECUOTP_USERIMPORT()

Prototype

```
int secuotp_userimport(char *pszUserfile, char *pszSecret, char
**pstrErrinfo);
```

Description

User batch import

Parameters

PARAMETERS	DESCRIPTION
char *pszUserfile	[IN] User file path
char *pszSecret	[IN] Decryption secret key
char **pstrErrinfo	[OUT] invalid records occur in batch import

note: this function for internal using assigns space used for storing invalid record for the caller , when caller no longer uses the space , please use secuotp_free(void **psbuf) to release the space pointed by pstrErrinfo

Return value

RETURN VALUE	DESCRIPTION
SECUOTP_ERROR_SUCCESS	user batch import successful
SECUOTP_ERROR_SUCCESS_WI THINFO	user batch import partially succeeded.(use pstrErrinfo to search for invalid entries)

Others (See the [error code list](#) for more details) user batch import failed

SECUOTP_TOKENDELETE

Prototype

```
int secuotp_tokendelete(char *pszTokenserialnum);
```

Description

Delete a token

Parameters

PARAMETERS	DESCRIPTION
char *pszTokenserialnum	[IN] serial number of the token you want to delete

Return value

RETURN VALUE	DESCRIPTION
SECUOTP_ERROR_SUCCESS	token deleted successfully

Others (See the [error code list](#) for more details) failed to delete the token

SECUOTP_MAKELOG()

Prototype

```
int secuotp_makelog(int nLoglev, char *pszOperator, int nOtype,
char *pszContent, char *pszSrvip);
```

Description

write logs

Parameters

PARAMETERS	DESCRIPTION
int nLoglev	[IN] logs level 1 or SECUOTP_PRI_SUPER super administrator level; 4 or SECUOTP_PRI_ADMIN administrator level; 10 or SECUOTP_PRI_OPERATOR operator level; 100 or SECUOTP_PRI_USER user level
char *pszOperator	[IN] operator ID, if no user is authenticated, set the username.
int nOtype	[IN] operator type code (see operations list)
char *pszContent	[IN] logs contents
char *pszSrvip	[IN] IP address of the computer performing the operation, if the user is authenticated, set the IP address of the authenticated client.

Return value

RETURN VALUE	DESCRIPTION
SECUOTP_ERROR_SUCCESS	log file written successfully

Others (See the [error code list](#) for more details) Failed to create log file

Operation code list

Operation code	Operation
1	User authentication
11	Add user
12	Delete user
13	Token synchronization
15	Set User static PIN
16	Set User authentication method
17	Set User usability
18	Set User lock
23	User batch import
25	Token batch import
26	Token deletion

SECUOTP_REBIND()

Prototype

```
int secuotp_rebind(const char *pszUsername, const char
*pszNewserialnum);
```

Description

Rebind a token to an user

Parameters

PARAMETERS	DESCRIPTION
const char *pszUsername	[IN] Username
const char *pszNewserialnum	[IN] new token serial number used

Return value

RETURN VALUE	DESCRIPTION
SECUOTP_ERROR_SUCCESS	token rebound successfully

Others (See the [error code list](#) for more details) Failed to rebind the token

Error code list

Error code	value	meaning
SECUOTP_ERROR_SUCCESS	0	success
SECUOTP_ERROR_ALLOCHANDLE_FAILED	11	Failed to request ODBC source
SECUOTP_ERROR_FREEHANDLE_FAILED	12	Failed to release ODBC source
SECUOTP_ERROR_SETENVATTR_FAILED	13	Failed to set ODBC properties
SECUOTP_ERROR_GETCONFIGFAILED	14	Failed to get setting information
SECUOTP_ERROR_CONNECTDBFAILED	15	Failed to connect to the database
SECUOTP_ERROR_SQLEXFAILED	16	Failed to execute SQL query
SECUOTP_ERROR_NORELATEITEM	17	unrelated record
SECUOTP_ERROR_GETUSERINFO_FAILED	18	Failed to get user information
SECUOTP_ERROR_USERREGLOST	19	User token reported lost
SECUOTP_ERROR_USERLOCKED	20	User is locked
SECUOTP_ERROR_UNLOCK_FAILED	21	failed to unlock the user
SECUOTP_ERROR_UPDATEUSEDATE_FAILED	22	Failed to update operation time
SECUOTP_ERROR_UNKNOWNTKTYPE	23	Unknown token type
SECUOTP_ERROR_UPCHALLENG_FAILED	24	Failed to update user challenge information
SECUOTP_ERROR_INVALID_PASSWORD	25	Authentication password error
SECUOTP_ERROR_SYNCHRONIZE_FAILED	26	Synchronization failed
SECUOTP_ERROR_INVALID_OLDPWD	27	Wrong old password

SECUOTP_ERROR_INVALID_AUTHTYPE	28	Unable to use this authentication method
SECUOTP_ERROR_USEREXIST	29	Username already exists
SECUOTP_ERROR_INVALID_TOKEN	30	Invalid Token (the Token doesn't exist, is already bound or is disabled.)
SECUOTP_ERROR_INVALID_PARAM	31	Invalid parameters
SECUOTP_ERROR_ADDUSER_FAILED	32	Failed to add user
SECUOTP_ERROR_UPDATETOKEN_FAILED	33	Failed to update token state
SECUOTP_ERROR_DELETETOKEN_FAILED	34	Failed to delete token
SECUOTP_ERROR_GETTOKENSTATE_FAILED	35	Failed to get token state
SECUOTP_ERROR_TOKENINUSE	36	Token is already bound
SECUOTP_ERROR_INVALIDKEY	37	Invalid secret key
SECUOTP_ERROR_OPENTOKENFILE_FAILED	38	Failed to open token file
SECUOTP_ERROR_AESMKKEY_FAILED	39	AES process private key error
SECUOTP_ERROR_AESDECRYPT_FAILED	40	Failed to decrypt file
SECUOTP_ERROR_AESGETPLAIN_FAILED	41	Failed to read plain text
SECUOTP_ERROR_AUTHSUCCESS_WITHINFO	42	Authentication succeeded but with warning message
SECUOTP_ERROR_UPDATEKEYINFO_FAILED	43	Failed to upgrade counter
SECUOTP_ERROR_NOTNEEDSYNCHRONIZE	44	Try to

		synchronize the challenge/response Token.
SECUOTP_ERROR_SETUSERPIN_FAILED	45	Failed to set static PIN
SECUOTP_ERROR_UPDATEPIN_FAILED	46	Failed to update static PIN
SECUOTP_ERROR_SUCCESS_WITHINFO	47	Token/User batch import partially succeeded. There are some invalid records returned
SECUOTP_ERROR_OPENUSERFILE_FAILED	48	Failed to open the user file for batch import
SECUOTP_ERROR_GETFILESIZE_FAILED	49	Failed to get the size of the file.
SECUOTP_ERROR_SRVIPNULL	50	Client IP address error
SECUOTP_ERROR_GETERRINFOBUF_FAILED	51	Failed to apply for space
SECUOTP_ERROR_GETFILECONT_FAILED	52	Failed to get the file contents.
SECUOTP_ERROR_UPDATEUSERBINDINFO_FAILED	53	Failed to rebind the token with the user
SECUOTP_ERROR_UPDATENEWTKSTATE_FAILED	54	Failed to update the token state after rebinding
SECUOTP_ERROR_INVALIDLOGLEV	55	Invalid log level
SECUOTP_ERROR_CHALLENGE	100	Challenge error for the user requesting authentication

Static data definition

Static data	value	meaning
MAX_PATH	260	Path maximum length
MAXLEN_56	56	
MAXLEN_32	32	
MAXLEN_10	10	
MAX_DATEBUF_LEN	56	Maximum size for the buffer
COUNTER_SIZE	8	Counter(number of bytes)

PHP SDK Operation procedure

Preparation

1. Use the sql file inside the distributable package to create the database for UniOTP dynamic password authentication.
2. The PHP version SDK uses PHP to access to the database. Before beginning, we should make sure that the appropriate PHP database modules are running.
3. Create a configuration file (or modify the existing configuration file) and set the parameters to access the database.

Add support for UniOTP dynamic password

1. Add the SDK files (include otp_const.php, otp_corefunc.php, folder config, folder dbinterface and folder encrypt) to the project in which you want to implement dynamic password authentication.
2. Modify the configuration file.
3. Use the provided API to perform authentication related operations and others.

SDK Operation procedure and important points

Call procedure

When you need to perform some business treatments, you just need to call the related API. (for details about calls, please see the sample)

Important points

1. You need to set the database type inside the configuration file. (This is because, depending on the database type, the connection interface used by the database is not the same)
2. Call the function `secuotp_userimport($sUserfile, &$sErrinfo, $sSecret="")` to perform batch import for user. If `$sSecret` value is not empty, this means that the file is encrypted. We recommend not to use file encryption feature.

Function interfaces

SECUOTP_AUTH()

Prototype

Function `secuotp_auth($susername, $sPassword, &$challenge);`

Description

Dynamic password authentication.

Parameters

PARAMETERS	DESCRIPTION
<code>\$sUsername</code>	name of user to be authenticated
<code>\$sPassword</code>	authentication password(OTP or OTP+PIN. Depends on the user's authentication mode)
<code>&\$Challenge</code>	challenge number. if SECUOTP_ERROR_CHALLENGE is returned, challenge number will be set in <code>\$Challenge</code>

Return value

RETURN VALUE	DESCRIPTION
SECUOTP_ERROR_SUCCESS	authentication successful
SECUOTP_ERROR_CHALLENGE	a challenge should be sent to the user

Others: failed

SECUOTP_SYNCHRONIZE()

Prototype

Function secuotp_synchronize(\$sUsername, \$sOtp1, \$sOtps)

Description

Synchronize the token

Parameters

PARAMETERS	DESCRIPTION
\$sUsername	name of user whose token is to be synchronized
\$sOtp1	the first OTP
\$sOtp2	the second OTP

Return value

RETURN VALUE	DESCRIPTION
SECUOTP_ERROR_SUCCESS	synchronized successfully

Others: failed

Remark: the two OTPs must be consecutive

SECUOTP_SETUSERPIN()

Prototype

Function secuotp_setuserpin(\$sUsername, \$sPin)

Description

Set PIN for user

Parameters

PARAMETERS	DESCRIPTION
\$sUsername	user name
\$sPin	PIN for \$sUsername

Return value

RETURN VALUE	DESCRIPTION
SECUOTP_ERROR_SUCCESS	New PIN set successfully

Others: failed

SECUOTP_RESETPIN()

Prototype

Function secuotp_resetpin(\$sUsername, \$sOldpin, \$sNewpin)

Description

Reset PIN

Parameters

PARAMETERS	DESCRIPTION
\$sUsername	Username of the user you want to reset the PIN for
\$sOldpin	old PIN
\$sNewpin	new PIN

Return value

RETURN VALUE	DESCRIPTION
SECUOTP_ERROR_SUCCESS	PIN reset successfully

Others: failed

SECUOTP_SETAUTHTYPE()

Prototype

Function secuotp_setauthtype(\$sUsername, \$nAuthtype)

Description

Set authentication type (1—use OTP only 2—user OTP+PIN)

Parameters

PARAMETERS	DESCRIPTION
\$sUsername	Username of the user you want to reset the PIN for
\$nAuthtype	authentication type (1-OTP 2-OTP+PIN)

Return value

RETURN VALUE	DESCRIPTION
SECUOTP_ERROR_SUCCESS	Authentication type set successfully

Others: failed

SECUOTP_ADDUSER()

Prototype

```
Function secuotp_adduser($sUsername, $sUserid, $sContacttel,
$sSerialnum, $sRemark, $sEmail,$sDomain, $sPin, $sPreusedate,
$nAuthtype)
```

Description

Add a new user

Parameters

PARAMETERS	DESCRIPTION
\$sUsername	username
\$sUserid	identity number such as personal ID
\$sContacttel	contact phone number
\$sSerialnum	serial number of a usable token allocated to the user
\$sRemark	remark
\$sEmail	email address
\$sDomain	reserved, you should set it null string(“”)
\$sPin	static PIN
\$nAuthtype	authentication type (1-OTP 2-OTP+PIN)

Return value

RETURN VALUE	DESCRIPTION
SECUOTP_ERROR_SUCCESS	User added successfully

Others: failed

SECUOTP_DELETEUSER()

Prototype

Function secuotp_deleteuser(\$sUsername)

Description

Delete a user

Parameters

PARAMETERS	DESCRIPTION
\$sUsername	username of user to be deleted

Return value

RETURN VALUE	DESCRIPTION
SECUOTP_ERROR_SUCCESS	User deleted successfully

Others: failed

SECUOTP_SETUSERINVALID()

Prototype

Function secuotp_setuserinvalid(\$nUsername, \$nValid)

Description

Set user validity status (0 valid, 1 invalid)

Parameters

PARAMETERS	DESCRIPTION
\$sUsername	username
\$nValid	validity status (0 valid, 1 invalid)

Return value

RETURN VALUE	DESCRIPTION
SECUOTP_ERROR_SUCCESS	Validity status set successfully

Others failed

SECUOTP_SETUSERLOCK()

Prototype

Function secuotp_setuserlock(\$sUsername, \$nLock)

Description

Set user's lock status

Parameters

PARAMETERS	DESCRIPTION
\$sUsername	username
\$nLock	lock status (0 unlocked, 1 locked)

Return value

RETURN VALUE	DESCRIPTION
SECUOTP_ERROR_SUCCESS	Lock status set successfully

Others: failed

Remark:

If nLock is set to 1, the user will be locked for 10 minutes

SECUOTP_TOKENIMPORT()

Prototype

Function secuotp_tokenimport(\$sTokenfile, \$sSecret, &\$sErrinfo)

Description

Import token items into the UniOTP system

Parameters

PARAMETERS	DESCRIPTION
\$sTokenfile	token file path
\$sSecret	decrypt share key
&\$sErrinfo	error information. It often contains invalid token items if SECUOTP_ERROR_SUCCESS_WITHINFO is returned

Return value

RETURN VALUE	DESCRIPTION
SECUOTP_ERROR_SUCCESS	Token imported successfully
SECUOTP_ERROR_SUCCESS_WITHINFO	the token file has some invalid token item

Others: failed

SECUOTP_USERIMPORT()

Prototype

Function secuotp_userimport(\$sUserfile, &\$sErrinfo, \$sSecret)

Description

Batch user import

Parameters

PARAMETERS	DESCRIPTION
\$sUserfile	path of file which contains user items
&\$sErrinfo	If SECUOTP_ERROR_SUCCESS_WITHINFO is returned, it contains the bad user items
\$sSecret	decrypt secret, if a null string is given, the user file will be deal with as a plain text

Return value

RETURN VALUE	DESCRIPTION
SECUOTP_ERROR_SUCCESS	User(s) imported successfully
SECUOTP_ERROR_SUCCESS_WITHINFO	The user file has some invalid user items. Invalid items have been stored in \$sErrinfo

SECUOTP_TOKENDELETE()

Prototype

Function secuotp_tokendelete(\$sTokenserialnum)

Description

Remove a token from the database

Parameters

PARAMETERS	DESCRIPTION
\$sTokenserialnum	serial number of the token to be removed

Return value

PARAMETERS	DESCRIPTION
SECUOTP_ERROR_SUCCESS	Token deleted from database successfully

Others: failed

SECUOTP_MAKELOG()

Prototype

Function secuotp_makelog(\$nLoglev, \$sOperator, \$nOtype, \$sContent, \$sSrvip)

Description

Add log information

Parameters

PARAMETERS	DESCRIPTION
\$nLoglev	log level 1 super administrator 4 administrator 10 operator 100 user
\$sOperator	operator id or name of user

\$nOptype	operation type (see the operation type definition in file otp_const.php)
\$sContent	log content
\$sSrvip	IP address of operator's computer or user's computer

Return value

RETURN VALUE	DESCRIPTION
SECUOTP_ERROR_SUCCESS	Log created successfully

Others: failed

SECUOTP_REBIND()

Prototype

Function secuotp_rebind(\$sUsername, \$sNewserialnum)

Description

Allocated a new token to the user

Parameters

PARAMETERS	DESCRIPTION
\$sUsername	username
\$sNewserialnum	new token serial number

Return value

RETURN VALUE	DESCRIPTION
0	success

PHP extension SDK operation procedure

Preparation

1. Use the SQL file inside the distributable package to create the database for the UniOTP dynamic password authentication system.
2. Install the driver corresponding to the database you are using and choose to use this driver.
3. Create an ODBC data source for the newly created database.
4. Create a configuration file (or modify the existing configuration file) and define the database access settings.

Add support for UniOTP dynamic password

1. Copy UniOTP SDK extension modules to the PHP extension modules directory.
2. Modify the PHP configuration file and add support for UniOTP extension modules.
3. Restart the server to enable extension modules.
4. Modify the configuration file. the configuration file contains the database connection and authentication parameters
5. Use API to perform operations related to authentication.

SDK Operation procedure and important points

Call procedure

1. Call the function `secuotp_initenv($Confpath)` to initialize database settings.
2. Use other interfaces to perform business-related operations.
3. Call the function `secuotp_cleanup()` to perform cleaning operations such as releasing the memory.

Important points

1. The first time you call functions from the SDK, you should first call the function `secuotp_initenv($Confpath)` to initialize database settings. You only need to call it once in each PHP page where you will use functions of the SDK.
2. When you finish using the SDK, use `secuotp_cleanup()` to perform cleaning operations.

Function interfaces

SECUOTP_INITENV()

Prototype

```
function secuotp_initenv($ConfpPath);
```

Description

Initialize configuration file. Only need to be called once.

Parameters

PARAMETERS	DESCRIPTION
\$ConfpPath	[IN] configuration file path

Return value

RETURN VALUE	DESCRIPTION
0	success

Others: failure, use ErrorCode2Msg(\$index) to check the error.

SECUOTP_CLEANUP()

Prototype

```
function secuotp_cleanup();
```

Description

This function is in charge of freeing assigned global resources (apply initialization settings)

Parameters

None

Return value

RETURN VALUE	DESCRIPTION
0	success

Others: failure, use ErrorCode2Msg(\$index) to check the error.

SECUOTP_GETLASTERROR()

Prototype

```
function secuotp_getlasterror();
```

Description

Return the last error message

Parameters

None

Return value

RETURN VALUE	DESCRIPTION
0	success

Others: failure, use ErrorCode2Msg(\$index) to check the error.

SECUOTP_AUTH()

Prototype

```
function secuotp_auth($Username, $Pwd, &$Challenge);
```

Description

Dynamic password authentication

Parameters

PARAMETERS	DESCRIPTION
\$Username	[IN] Username
\$Pwd	[IN] Authentication password for this time
\$Challenge	[OUT] Use to save the challenge message returned by the server

Return value

RETURN VALUE	DESCRIPTION
0	success

Others: failed, use `ErrorCode2Msg($index)` to check the error.

SECUOTP_SYNCHRONIZE()

Prototype

```
function secuotp_synchronize($Username, $Otp1, $Otp2);
```

Description

Synchronize the token

Parameters

PARAMETERS	DESCRIPTION
\$Username	[IN] Username
\$Otp1	[IN] For the synchronization to be completed, you need to provide 2 consecutive dynamic passwords. This parameter is the first of the two passwords
\$Otp2	[IN] The second of the two passwords

Return Value

RETURN VALUE	DESCRIPTION
0	success

Others: failure, use `ErrorCode2Msg($index)` to check the error

SECUOTP_SETUSERPIN()

Prototype

```
function secuotp_setuserpin($Username, $Pin);
```

Description

Set the user PIN

Parameters

PARAMETERS	DESCRIPTION
\$Username	[IN] Username
\$Pin	[IN] User PIN

Return Value

RETURN VALUE	DESCRIPTION
0	success

Others: failure, use ErrorCode2Msg(\$index) to check the error

SECUOTP_RESETPIN()

Prototype

```
function secuotp_resetpin($Username, $Oldpin, $Newpin);
```

Description

Set a new PIN

Parameters

PARAMETERS	DESCRIPTION
\$Username	[IN] Username
\$Oldpin	[IN] old PIN
\$Newpin	[IN] new PIN

Return Value

RETURN VALUE	DESCRIPTION
0	success

Others: failure, use ErrorCode2Msg(\$index) to check the error

SECUOTP_SETAUTHTYPE()

Prototype

```
function secuotp_setauthtype($Username, $nAuthtype);
```

Description

Set authentication type

Parameters

PARAMETERS	DESCRIPTION
\$Username	[IN] Username
\$nAuthtype	[IN] Authentication type 1- Use only OTP to perform authentication, 2- Use both OTP and PIN to perform authentication

Return Value

RETURN VALUE	DESCRIPTION
0	success

Others: failure, use `ErrorCode2Msg($index)` to check the error

SECUOTP_ADDUSER()

Prototype

```
function secuotp_adduser($Username, $Userid, $Contacttel,
$Serialnum, $Remark, $Email, $Domain, $Pin, $Preusedate,
$nAuthtype);
```

Description

Add a new user; this operation will bind a new user and a new token.

Remark:

Before adding a user, make sure the token allocated to the user has been imported into the database using the function `secuotp_tokenimport()`.

Parameters

PARAMETERS	DESCRIPTION
\$Username	[IN] Username
\$Userid	[IN] User ID (Unique)
\$Contacttel	[IN] User phone number
\$Serialnum	[IN] Token serial number attributed to this user

\$Remark	[IN] Remark/Comment
\$Email	[IN]Email address
\$Domain	[IN]Reserved, always NULL
\$Pin	[IN]User PIN
\$Preusedate	[IN]User registration date
\$nAuthtype	[IN]Authentication type,1-only use OTP to perform authentication 2- Use OTP+PIN to perform authentication.

Return value

RETURN VALUE	DESCRIPTION
0	success

Others: failure, check error through ErrorCode2Msg(\$index)

SECUOTP_DELETEUSER()

Prototype

```
function secuotp_deleteuser($Username);
```

Description

Delete a user

Parameters

PARAMETERS	DESCRIPTION
\$Username	[IN] Username

Return value

RETURN VALUE	DESCRIPTION
0	success

Others: failure, check error through ErrorCode2Msg(\$index)

SECUOTP_SETUSERINVALID()

Prototype

```
function secuotp_setuserinvalid($Username, $nValid);
```

Description

Set user validity status

Parameters

PARAMETERS	DESCRIPTION
\$Username	[IN] Username
\$nValid	[IN]User validity status 0-valid 1-invalid

Return value

RETURN VALUE	DESCRIPTION
0	success

Others: failure, check error through ErrorCode2Msg(\$index)

SECUOTP_SETUSERLOCK()

Prototype

```
function secuotp_setuserlock($Username, $nLock);
```

Description

Set the user lock state

Remark:

If nLock has been set to 1, the user will be locked for 10 minutes.

Parameters

PARAMETERS	DESCRIPTION
\$Username	[IN] Username
\$nLock	[IN]lock state 0-unlocked 1-locked

Return value

RETURN VALUE	DESCRIPTION
0	success

Others: failure, check error through ErrorCode2Msg(\$index)

SECUOTP_TOKENIMPORT()

Prototype

```
function secuotp_tokenimport($Tokenfile, $Secret, $invalid_items);
```

Description

import token

Parameters

PARAMETERS	DESCRIPTION
\$Tokenfile	[IN]Token file path
\$Secret	[IN] secret key used for decryption
\$invalid_items	[OUT] invalid token record inside the token file

Return value

RETURN VALUE	DESCRIPTION
0	success

Others: failure, check error through ErrorCode2Msg(\$index)

SECUOTP_USERIMPORT()

Prototype

```
function secuotp_userimport($Userfile, $Secret, $invalid_items);
```

Description

User import

Parameters

PARAMETERS	DESCRIPTION
\$Userfile	[IN] User file
\$Secret	[IN] Decryption secret key, let empty if the file is not encrypted
\$invalid_items	[OUT] invalid token record inside

	the user file	
--	---------------	--

Return value

RETURN VALUE	DESCRIPTION
0	success

Others: failure, check error through ErrorCode2Msg(\$index)

SECUOTP_TOKENDELETE()

Prototype

```
function secuotp_tokendelete($Tokenserialnum);
```

Description

Delete a token

Parameters

PARAMETERS	DESCRIPTION
\$Tokenserialnum	[IN] token that you want to delete

Return value

RETURN VALUE	DESCRIPTION
0	success

Others: failure, check error through ErrorCode2Msg(\$index)

SECUOTP_MAKELOG()

Prototype

```
function secuotp_makelog($nLoglev, $pszOperator, $nOtype, $pszContent, $pszSrvip);
```

Description

Write logs

Parameters

PARAMETERS	DESCRIPTION
\$nLoglev	[IN]logs level

	The following values are usable: 1 (super administrator) 2 (administrator) 3 (operator) 4 (user)
\$pszOperator	[IN]Operator
\$nOptype	[IN]Operation type
\$pszContent	[IN]logs data
\$pszSrvip	[IN]Server IP

Return value

RETURN VALUE	DESCRIPTION
0	success

Others: failure, check error through ErrorCode2Msg(\$index)

SECUOTP_REBIND()

Prototype

```
function secuotp_rebind($Username, $Newtokenserial);
```

Description

Rebind a token with a user

Parameters

PARAMETERS	DESCRIPTION
\$Username	[IN] Username
\$Newtokenserial	[IN]New Token serial number

Return value

RETURN VALUE	DESCRIPTION
0	success

Others: failure, check error through ErrorCode2Msg(\$index)