HOTPin

celestix

Authentication API Documentation

Contents

1 Authentication API

The Authentication API SDK extends HOTPin functionality. It includes examples and sample code to help you use two-factor authentication in your projects.

The first section provides an overview to help acclimate to the SDK structure. The second section provides information about the Authentication API. The third section provides information about the Challenge-Response API to use QR codes for authentication.

1.1 SDK Overview

The API components are included in the following directories contained in the **Authentication API SDK**:

Agent – Documentation, COM component help and samples

AgentAuthConsole – Sample C++ wrapper application to the Agent COM component.

AgentAuthWebService – Sample Agent authentication proxy web service.

bin – Win32 authentication service libraries

Documentation – Authentication service documentation and help

DotNet – Microsoft .NET based samples

ClientAspNetWeb – Sample client library ASP.Net web site.

Clx.HOTPin.AuthService.Client – Authentication service C# client source code

Java – Java based samples

Mobile – Mobile device samples

\HOTPin.WinPhone.AuthService.Client – Windows Phone sample

AndroidHotpinClientLibSample.zip – Android sample

1.2 Authentication API Reference

The authentication web service has SOAP (WS-*), SOAP (Basic) and REST endpoints. You can use one of the included .NET or Java authentication client libraries to connect to the service or create your own client proxy from the metadata endpoints.

1.2.1 Service Metadata Endpoints

WSDL address:

https://{address}:10603/hotpinserver/ws/authsvc/authenticate.svc?wsdl

MEX address:

https://{address}:10603/hotpinserver/ws/authsvc/authenticate.svc/mex

1.2.2 SOAP (WS-*) Service Endpoint

Service Address:

https://{address}:10603/hotpinserver/ws/authsvc/authenticate.svc/soap

Note: See the SOAP help information for additional definitions:
Authentication API SDK|Documentation|AuthAPIHelp.chm.

1.2.3 SOAP (Basic, non WS-*) Service

Service Address:

https://{address}:10603/hotpinserver/ws/authsvc/authenticate.svc/basic

1.2.4 REST Service

The REST service uses JSON-formatted objects for authentication, set pin parameters, and returned info; plain values (i.e. true, 1, "string") are returned for all other methods. For simplicity and security, we are using just POSTs and GETs; user info is not passed in any query vars.

The service will normally return a status success code of 200 for all calls and a 500 code for any server error as well as the normal HTTP codes such as not found (404).

Base address: https://{address}:10603/hotpinserver/authsvc/rest

Method	URI Template	Description
POST	/authenticate	Authenticate a user.
POST	/pin	Set the PIN of a user.
GET	/arraymember	Is an array member or not.
GET	/nodetype	Get the node type
GET	/productversion	Get the HOTPin version.
GET	/ping	Ping the authentication service

1.2.4.1 Authenticate Method

Authentication failure will not return a status code of 401 (unauthorized), so you need to check the "result" property from the returned object.

Property	Description
userName	The user's name as it appears in HOTPin.
pwd	The user's HOTPin OTP, PIN, PINOTP, send command or PIN,send command based on the HOTPin settings and user settings and state.
userAddress	(optional) The user's IP address if known and want logged to HOTPin.
client	(optional) The client application name/type making the call and want logged to HOTPin.
clientAddress	(optional) The client application IP address and want logged to HOTPin.
properties	(optional) Extended properties object sent to the HOTPin server. The properties object is a collection of name/value string pairs. Not currently used; will allow additional functionality in the future.

Example POST

The JSON object format requires "pwd" and "userName". All others are optional and should only be set if available.

Example:

```
{
	"pwd":"1234",
	"userName":"test",

	"userAddress":"000.000.000.000",
	"client":"RADIUS Client",
	"clientAddress":"111.222.333.444",
	"properties":
		[
		{"name":"extProp1","value":"value1"},
		{"name":"extProp2","value":"value2"}
		]
}
```

Authenticate Return

The result will always be returned and all others will only be returned if they have a value.

Property	Description
result	The authentication result code (see Example Return below).
handle	Set PIN handle if the user is in new PIN mode. The handle is required to set the user's PIN and is valid for a limited amount of time (approx. 2 minutes).
fullName	The user's full name is returned only on a successful login.
properties	Extended properties object sent to the HOTPin server. The properties object is a collection of name/value string pairs. Not currently used; will allow additional functionality in the future.

Example Return

```
{
	"result":1,
	"handle":"12345678",
	"fullName":"test user",
	"properties":
		[
		{"name":"extProp1","value":"value1"},
		{"name":"extProp1","value":"value2"}
		]
}
```

Reference authentication result codes	
0	Fail
1	Pass
2	NewPinMode
3	NextCode
4	LockedOut
5	CodeSent
6	CodeSentNewPinMode
7	CodeSentNoPinRequired

1.2.4.2 pin Method

This is used to set a user's PIN if they are in new pin mode. The handle value is required from the previous authentication call in order to set the pin.

Property	Description
userName	The user's name as it appears in HOTPin.
pin	The user's new PIN of 4-8 numerals and must differ from the user's previous PIN.
handle	The set PIN handle from the previous authentication call.
userAddress	(optional) The user's IP address if known and want logged to HOTPin.
client	(optional) The client application name/type making the call and want logged to HOTPin.
clientAddress	(optional) The client application IP address and want logged to HOTPin.
properties	(optional) Extended properties object sent to the HOTPin server. The properties object is a collection of name/value string pairs. Not currently used; will allow additional functionality in the future.

Example POST

JSON object format, "userName", "pin" and "handle" are required; all others are optional and should only be set if available.

Example:

```
{
	"userName":"test",
	"pin":"1234",
	"handle":"123456",

	"userAddress":"000.000.000.000",
	"client":"RADIUS Client",
	"clientAddress":"111.222.333.444",
	"properties":
		[
		{"name":"extProp1","value":"value1"},
		{"name":"extProp1","value":"value2"}
		]
}
```

Return

The return is an integer: see codes below.

Reference set pin response codes	
0	Fail
1	Pass
2	InvalidPin
3	PinSetTtlExpired
4	NewPinSameAsOld
5	PassTokenOverride

1.2.4.3 arraymember Method

Returns true or false if the server is an array member, not standalone or primary.

1.2.4.4 nodetype Method

The return is an integer: see codes below.

Reference response node type code	
0	Standalone
1	Manager
2	Member

1.2.4.5 productversion Method

The return is a string that represents the HOTPin version (i.e. "3.7.00")

1.2.4.6 ping Method

The return is either **true** if the authentication service is up on the HOTPin server, or **false** if not.

1.3 Challenge-Response for QR Code Authentication API Reference

The challenge-response authentication web service, used for QR code authentication, has SOAP (WS-*), SOAP (Basic) and REST endpoints. You can use one of the included .NET or Java authentication client libraries to connect to the service or create your own client proxy from the metadata endpoints.

1.3.1 Service Metadata Endpoints

WSDL address:

https://{address}:10603/hotpinserver/ws/crauthsvc/authenticate.svc?wsdl

MEX address:

https://{address}:10603/hotpinserver/ws/crauthsvc/authenticate.svc/mex

1.3.2 SOAP (WS-*) Service Endpoint

Service Address:

https://{address}:10603/hotpinserver/ws/crauthsvc/authenticate.svc/soap

Note: See the SOAP help information for additional definitions:
Authentication API SDK|Documentation|AuthAPIHelp.chm.

1.3.3 SOAP (Basic, non WS-*) Service

Service Address:

https://{address}:10603/hotpinserver/ws/crauthsvc/authenticate.svc/basic

1.3.4 REST Service

The REST service uses JSON-formatted objects for challenge, response and authenticatesession for parameter and returned info; plain values (i.e. true, 1, "string") are returned for all other methods. For simplicity and security, we are using just POSTs and GETs; user info is not passed in any query vars.

The service will normally return a status success code of 200 for all calls and a 500 code for any server error as well as the normal HTTP codes such as not found (404).

Base address: https://{address}:10603/hotpinserver/crauthsvc/rest

Method	URI Template	Description
POST	/challenge	Start a challenge authentication response session.
POST	/response	Set the response value for an authentication session at the server.
POST	/authenticatesession	Authenticate the user after the response has been received.
GET	/status/{sessionID}	Get authentication status.
GET	/ping	Ping the service
GET	/cancel/{sessionID}	Cancel the challenge-response authentication session.

1.3.4.1 Challenge Method

Initiates a challenge-response authentication session for a specific user. A session times out after about 3 minutes and only one session per user can exist at any one time. If a new session is started, all previous user sessions are deleted.

Property	Description
userName	The user's name as it appears in HOTPin.
responseUrl	The URL to a web page or handler that the response will be sent to. This will be encoded in the returned QR code and used by the client software.
applicationName	(optional) The web application name as you want it to appear in the client software; it will be encoded in the QR code.
applicationMessage	(optional) The web application message or description as you want it to appear in the client software; it will be encoded into the QR code.
applicationIconUrl	(optional) A URL to a web application icon graphic (gif, png, jpeg) that will appear in the client software and will be encoded in the QR code.
alwaysReturnChallenge	Set to "true" or "false" to always return challenge information regardless if the passed user name exists or is enabled. Use to prevent someone from learning valid user names from a public site logon page.
returnQRInfo	Set to "true" or "false" to return QR code text and optionally the QR code image.
returnQRImage	Set to "true" or "false" to return the QR code image (png format) base64 encoded.
qrImageSize	Set the size of the returned QR image size. The default is 320 pixels and can be from 200 to 1024.
userAddress	(optional) The user's IP address if known and want logged to HOTPin.
client	(optional) The client application name/type making the call and want logged to HOTPin.
clientAddress	(optional) The client application IP address and want logged to HOTPin.
properties	(optional) Extended properties object sent to the HOTPin server. The properties object is a collection of name/value string pairs. Not currently used; will allow additional functionality in the future.

Example POST

```
{
	"userName":"test",
	"responseUrl":"https://host/qrhandler.ashx",
	"applicationName":"Your Web App",
	"applicationMessage":"Your Web App Description",
	"applicationIconUrl":"https://host/icon.gif",
	"alwaysReturnChallenge":true,
	"returnQRInfo":true,
	"returnQRImage":true,
	"qrImageSize":320,
	"userAddress":"000.000.000.000",
	"client":"Web Client",
	"clientAddress":"111.222.333.444",
	"properties":
		[
		{"name":"extProp1","value":"value1"},
		{"name":"extProp2","value":"value2"}
		]
}
```

Challenge Return

The return will be a JSON-formatted object with the following properties.

Property	Description
userName	The user's name.
sessionID	The challenge-response session ID.
sessionHandle	The challenge-response session handle used for authentication.
ocraSuiteString	Defines the format of the challenge-response authentication values for reference.
pinSalt	User's PIN salt required by the client to generate the challenge value for reference.
challenge	An arbitrary challenge value used by the client for reference.
qrAuthVersion	Internal QR code format version.
qrImage	Base64 encoded QR code image (png format) to display to the user.
qrImageSize	The returned QR code image size.
qrText	The returned QR code text if you want to generate your own QR code image.
createdUtc	The UTC time when the session was created.
expiresUtc	The UTC time when the session will expire.
properties	Extended properties object sent to the HOTPin server. The properties object is a collection of name/value string pairs. Not currently used; will allow additional functionality in the future.

Example Return


```
{
	"challenge":"24031432",
	"createdUtc":"\/Date(1371756902050)\/",
	"expiresUtc":"\/Date(1371757082050)\/",
	"ocraSuiteString":"OCRA-1:HOTP-SHA1-6:C-QN08",
	"qrAuthVersion":"CHQA-1",
	"qrImage":"(Base64 encoded PNG Image)",
	"qrImageSize":320,
	"qrText":"v=CHQA-1&s=117e69e0-7d44-4ddc-9177-
	   71997f4f4843&q=24031432&o=OCRA-1%3aHOTP-SHA1-6%3aC-
	   QN08&n=App+Name&m=App+message.&e=2013-06-
	   20T19%3a38%3a02.0508359Z",
	"sessionHandle":"77263811",
	"sessionID":"117e69e0-7d44-4ddc-9177-71997f4f4843",
	"userName":"user1"
}
```


1.3.4.2 Response Method

Set the response value for an authentication session at the server. Returns true if the session was found and the response set.

Property	Description
sessionID	The challenge-response session ID.
response	The response value sent by the client to your application.
clientAddress	(optional) The address of the client device if known.
properties	Extended properties object sent to the HOTPin server. The properties object is a collection of name/value string pairs. Not currently used; will allow additional functionality in the future.

Example POST


```
{
	"sessionID":"117e69e0-7d44-4ddc-9177-71997f4f4843",
	"response":"05837528",
	"clientAddress":"111.222.333.444"
}
```


1.3.4.3 Authenticate Session Method

Authenticate the user after the response has been received and return the result in JSON object.

Property	Description
userName	The user name for the session.
sessionID	The challenge-response session ID.
sessionHandle	The session handle returned from the "challenge" method call.
userAddress	(optional) The user's IP address if known and want logged to HOTPin.
client	(optional) The client application name/type making the call and want logged to HOTPin.
clientAddress	(optional) The client application IP address and want logged to HOTPin.
properties	(optional) Extended properties object sent to the HOTPin server. The properties object is a collection of name/value string pairs. Not currently used; will allow additional functionality in the future.

Example POST

```
{
	"userName":"user1",
	"sessionID":"117e69e0-7d44-4ddc-9177-71997f4f4843",
	"sessionHandle":"77263811",

	"userAddress":"000.000.000.000",
	"client":"Web Client",
	"clientAddress":"111.222.333.444",
	"properties":
		[
		{"name":"extProp1","value":"value1"},
		{"name":"extProp2","value":"value2"}
		]
}
```

Authenticate Session Return

The return will be a JSON-formatted object with the following properties.

Property	Description
fullName	The user's full name if successful authentication.
handle	A set PIN handle value if successful authentication and the user is in new PIN mode.
result	An integer value determining the result of the authentication (see below).
properties	Extended properties object sent to the HOTPin server. The properties object is a collection of name/value string pairs. Not currently used; will allow additional functionality in the future.

Reference authentication result codes	
0	Fail
1	Pass
2	NewPinMode
3	NextCode
4	LockedOut
5	CodeSent
6	CodeSentNewPinMode
7	CodeSentNoPinRequired

Example Return

```
{
	"fullName":"User Number1",
	"result":2,
	"handle":"0485673"
}
```

1.3.4.4 Status Method

Get the authentication status and return a status code.

Example GET:

https://{address}:10603/hotpinserver/crauthsvc/rest/status/117e69e0-7d44-4ddc-9177-71997f4f4843

Return

0	Initialized - The session is initialized; awaiting a response value.
1	ResponseReceived - The response for the session has been received; awaiting authentication.
2	NotFound - The session was not found either because of an invalid session ID, or the session expired and was deleted.
3	Expired - The session was found but has expired and has not been automatically deleted yet.

1.3.4.5 Ping Method

The return is either **true** if the authentication service is up on the HOTPin server, or **false** if not.

1.3.4.6 cancel Method

Cancel a challenge-response authentication session.

Example GET

https://{address}:10603/hotpinserver/crauthsvc/rest/cancel/117e69e0-7d44-4ddc-9177-71997f4f4843

Return

Returns **true** if the authentication session exists and was canceled, otherwise **false**.