



FortiAuthenticator™ REST API

Solution Guide



FortiAuthenticator™ REST API Solution Guide

November 18, 2013

Revision 4

Copyright© 2013 Fortinet, Inc. All rights reserved. Fortinet®, FortiGate®, and FortiGuard® are registered trademarks of Fortinet, Inc., and other Fortinet names herein may also be trademarks of Fortinet. All other product or company names may be trademarks of their respective owners. Performance metrics contained herein were attained in internal lab tests under ideal conditions, and performance may vary. Network variables, different network environments and other conditions may affect performance results. Nothing herein represents any binding commitment by Fortinet, and Fortinet disclaims all warranties, whether express or implied, except to the extent Fortinet enters a binding written contract, signed by Fortinet's General Counsel, with a purchaser that expressly warrants that the identified product will perform according to the performance metrics herein. For absolute clarity, any such warranty will be limited to performance in the same ideal conditions as in Fortinet's internal lab tests. Fortinet disclaims in full any guarantees. Fortinet reserves the right to change, modify, transfer, or otherwise revise this publication without notice, and the most current version of the publication shall be applicable.

Technical Documentation

<http://help.fortinet.com>

Knowledge Base

<http://kb.fortinet.com>

Forums

<https://support.fortinet.com/forums>

Customer Service & Support

<https://support.fortinet.com>

Training

<http://training.fortinet.com>

FortiGuard Threat Research & Response

<http://www.fortiguard.com>

License Agreement

<http://www.fortinet.com/doc/legal/EULA.pdf>

Document Feedback

Email: techdocs@fortinet.com

Table of contents

Change Log	5
Introduction.....	6
Software versions	6
The FortiAuthenticator API.....	7
Introduction to REST	7
Initializing the REST API.....	7
Accessing the REST API.....	9
Filtering query results.....	9
Field filters	9
Supported API Methods	9
Supported Data Formats	10
Resource Summary	10
Example API Calls.....	12
General API Usage.....	12
View Available Endpoint Resources.....	12
SSO/Remote Groups (/ssogroup/).....	14
Supported Fields	14
View SSO Group Configuration.....	14
Create SSO Group	15
FortiGate Group Filter (/fgtgroupfilter/).....	16
Supported Fields	16
View FortiGate Group Filter Configuration	16
Local Users (/localusers/).....	17
Supported Fields	17
Allowed Methods.....	18
List Local Users.....	18
Create Local User.....	18
Local Users (/usergroups/).....	20
View User Groups.....	20
Create a User Group	20
SSO Auth (/ssoauth/)	22
Supported Fields	22
Allowed Methods.....	22
Response Codes.....	22
FSSO User Login.....	23
Overwrite FSSO user login with different user.....	23
Logout FSSO User	24
Logging.....	25
Authentication (/auth/).....	26

Supported Fields	26
Allowed Methods.....	26
Response Codes.....	27
Validate a user password	27
Validate a users token code.....	27
Error States.....	28
Appendix A –API Response Codes.....	29
General API Response Codes.....	29

Change Log

Revision	Date	Change Description
1	2012-07-12	Initial Release
2	2012-11-01	Updated API to reflect changes since 2.0 release
3	2013-10-21	Updated API to reflect changes since 3.0 release
4	2013-11-18	Added missing group config in /localusers/

Introduction

This document introduces the FortiAuthenticator REST API and details how it can be configured and utilized.

Software versions

The API described within this document is as supported by FortiAuthenticator 3.0 and upwards.

The FortiAuthenticator API

Introduction to REST

An API (Application Programming Interface) is a set of defined interfaces to accomplish a task, such as retrieving or modifying data. FortiAuthenticator provides a Representational State Transfer (REST) API for interaction with components of the system. Programs communicate with the REST API over HTTP, the same protocol that your web browser uses to interact with web pages.

The REST API is based on interactions with a web page; data is treated like a static web page:

- Add data by POSTing a web page
- Fetch data by GETing a web page
- Update data by PUTing a web page
- Partial updates supported by PATCHing a web page
- Delete data by DELETEing a web page

After receiving the request, the FortiAuthenticator API sends back an HTTP response code. These error codes are summarized in [Appendix A – API Response Codes](#).

Initializing the REST API

Unlike most other vendors, the FortiAuthenticator API is accessible without additional cost or licensing. The server however is disabled by default and needs to be configured.

To enable the API, enable a user with administrator rights and select Web Service Access.

You must configure an e-mail address for the user at this point to as the API challenge key will be emailed to the address specified.



Note: The option *Allow RADIUS authentication* is removed when elevating a user to an FortiAuthenticator administrator.

Create a new user or edit an existing one. In the example shown, the admin account is used.

- Select *User Role: Administrator*
- Enable *Web Service Access*
- Enter a valid email address. The API Key will be forwarded to this address so ensure it is valid and email routing is working beforehand.

FortiAuthenticator

Logged in as admin

Help Logout

System

Authentication

- User Account Policies
- Lockouts
- Passwords
- Custom User Fields
- User Management
 - Local Users
 - Remote Users
 - Remote User Sync Rules
 - User Groups
 - FortiTokens
- MAC Devices
- Self-service Portal
- Remote Auth. Servers
- RADIUS Service
- LDAP Service
- FortiAuthenticator Agent

Fortinet SSO Methods

Edit User

Username: admin

☐ Disabled

☒ Password-based authentication [Change Password]

☐ Token-based authentication

☐ Enable account expiration

User Role

Role: ☒ Administrator ☐ User

Access: ☐ Custom ☒ Full

☒ Web service access

☐ Allow LDAP browsing

User Information

First name: Last name:

Email address: admin@example.com Phone number:

Mobile number: SMS gateway: Use default Test SMS

Street address:

City: State/Province:

Country:

Language: Use default

Alternative e-mail addresses

Password Recovery Options

Groups

E-mail Routing

Radius Attributes

Certificate Bindings

OK Cancel

Click **OK** to save the details

The API Web Service Access Key used to authenticate to the API will be e-mailed to the API administrator.

FortiAuthenticator

Logged in as admin

Help Logout

System

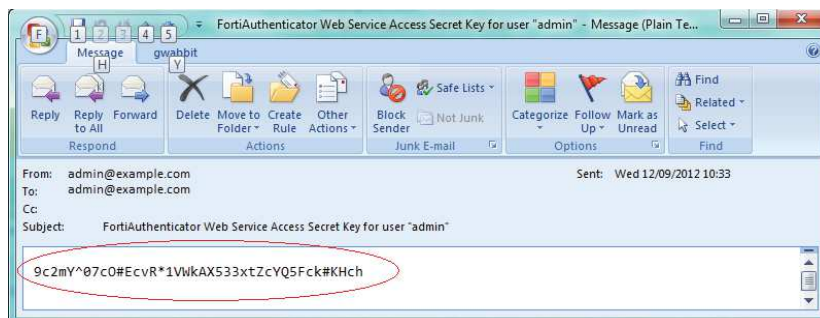
Authentication

Create New Import Delete Edit 0 of 2 selected Search for users Search

Message: User "admin" has been given an access to the web service. An e-mail containing the web service secret key has been sent to the user.

Username	First Name	Last Name	E-Mail Address	Admin	Status	Token	Groups	Authentication Method
admin			admin@example.com	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>			

Make a note of the API Web Service Access Key





Note: Should the API Web Service Access Key be lost, access can be recovered by disabling the Web Service feature for the user, saving and then re-enabling the feature. A new key will be generated (and all code using it will need to be updated with the new credentials).

Accessing the REST API

The FortiAuthenticator API can be accessed from most browsers (GET) however browser add-ons may be required for extended operations (e.g. PUT). More complicated, scripted queries can be made using utilities such as cURL and most scripting languages such as Perl or Python have built in libraries for interacting with RESTful APIs.

Example shown within this document will be demonstrated with the cross platform utility cURL.

All of the resource URLs are in this form:

[https://\[server_name\]/api/\[api_version\]/\[resource\]/](https://[server_name]/api/[api_version]/[resource]/)

where:

server_name	=	Name or IP of the FortiAuthenticator
api_version	=	API version to be used (currently v1)
resource	=	Resource or part of config to be viewed
id	=	Resource ID to view, edit or delete.

Filtering query results

Queries to the API can be to modify the query/response format or to filter the results. Below are some arguments that can be passed to the REST API URL. Please refer to the specific resource documentation to find out which of these filter operations are allowed.

?format=[format_type]	where format_type= xml or json (default)
?limit=[integer]	where integer specifies number of records to return (default unlimited)
?offset=[integer]	where integer specifies number of items in resource list to skip e.g. if there are 10 items, to return item #5 - #10 only, specify offset=4
?order_by=[field]	order returned list by a known field name (e.g. ?order_by=name)

Field filters

- exact: search for an exact match (e.g. to return items that has a name matching "John Doe", name__exact=John Doe)
- in: search for items that matches specific filter criteria (e.g. to return items that has a name matching "John" or "Bill", ?name__in=John&name__in=Bill)

Supported API Methods

To list all of the available resource endpoints in the current software release, send a GET request to: [https://\[server_name\]/api/\[api_version\]/?format=xml](https://[server_name]/api/[api_version]/?format=xml)

Method	URL	Operation Description	Success Response Code
GET (list)	/[resource]/	Retrieve a list of all resources for the endpoint	200 OK
GET (detail)	/[resource]/[id]/	Retrieve a specific resource with ID <code>id</code> from the endpoint	200 OK
POST	/[resource]/	Create a new resource on the given endpoint. The data being POST-ed must follow the same format as the data returned by the GET parameter	201 CREATED
PUT (list)	/[resource]/	Update all of the resources for the given endpoint. Any existing items will be replaced with the new data. Data must follow the same format as the data returned by the GET parameter.	204 NO CONTENT
PUT (detail)	/[resource]/[id]/	Update an existing item specified with ID <code>id</code> . Data must follow the same format as the data returned by the GET parameter.	204 NO CONTENT
DELETE (list)	/[resource]/	Delete all resources from an endpoint	204 NO CONTENT
DELETE (detail)	/[resource]/[id]/	Delete an existing resource specified with ID <code>id</code> from an endpoint	204 NO CONTENT
PATCH (detail)	/[resource]/[id]/	Update specific fields on an existing item with ID <code>id</code>	202 ACCEPTED

Supported Data Formats

Currently, JSON and XML are supported. To specify a format on the request:

For a GET request, there are 2 options:

- Use the GET format parameter (e.g. `?format=json` or `?format=xml`)
- Specify an Accept HTTP header with a correct mimetype (e.g. `Accepts: application/json` for JSON)



Note: The GET format parameter takes precedence over the Accept header. Browsers will usually default to requesting for an XML data type when format is not specified for a GET request.

Resource Summary

There are currently 6 main resources and the root record which can be accessed via the API:

Resource	URL	Operation Description	Supported Methods
Root	/	Allows querying of available resources	GET
Local User Management	/localusers/	Allows the creation, modification and deletion of user accounts	GET, POST, PATCH
Local Group Management	/usergroups/	Allows the creation and deletion of user groups and specify users within that group.	GET, POST, PUT, DELETE
User Authentication	/auth/	Allows validation of user authentication credentials	POST
SSO Group	/ssogroup/	Enables remote configuration of the SSO & Dynamic Policies → SSO → SSO Groups table	GET, POST, DELETE
FortiGate Filter Group	/fgtgroupfilter/	Enables remote configuration of the SSO & Dynamic Policies → SSO → FortiGate Group Filtering table	GET, PUT
SSO Authentication	/ssoauth/	Adds/removes a user from the FSSO logged in users table.	POST

Example API Calls

For the purpose of these examples, cURL is being used to make the requests. cURL is more flexible than a browser alone, is cross platform and can be called from most scripts. It is not as flexible as native scripting languages but is a good clear example which can be used to understand how the API functions.

The following flags are used in the cURL query:

Ignore certificate errors. This can be overcome with use of a valid certificate.

Verbose. Increase the level of logging information (useful for debugging).

User login information in format



Note: When using PUT/POST with cURL on Windows, problems can be encountered with escaping of the required double quotes in the data content, leading to errors related to incomplete closed brackets. To avoid this, the code should be properly escaped (using \ before any double quotes) or the data text stored in a file and referenced using

Alternatively, it is highly recommended that this is run on a Linux OS, where escaping of characters in cURL is more predictable.

General API Usage

View Available Endpoint Resources

JSON Query

JSON specified via GET



JSON specified via Accept Header



Response



XML Query

XML specified via GET

```
https://192.168.0.123/api/v1/?format=xml
```

XML specified via Accept Header



Response



SSO/Remote Groups (/ssogroup/)

URL:

In the FortiAuthenticator GUI, this resource corresponds to *Fortinet SSO Methods* > *SSO SSO Groups*.

Supported Fields

Field	Display Name	Type	Required	Other Restriction
name	Name	string	Yes	max length=50, unique

View SSO Group Configuration

XML Query

XML specified via GET



XML specified via Accept Header



Response



JSON Query

JSON specified via GET

JSON specified via Accept Header

Response

Create SSO Group

JSON Query

JSON specified via POST

XML Query

JSON specified via Accept Header

Response

Successful 201 CREATED response code. See Appendix A –API Response Codes for full details.

FortiGate Group Filter (/fgtgroupfilter/)

URL:

In the FortiAuthenticator GUI, this resource corresponds to *Fortinet SSO Methods* → *SSO FortiGate Group Filtering*.

Supported Fields

Field	Display Name	Type	Required	Other Restriction
name	Name	string	Yes	max length=32, unique
nasname	NAS name/IP	string	Yes	max length=128, unique
sso_groups	SSO Groups	string	No	

View FortiGate Group Filter Configuration

XML Query

XML specified via GET

XML specified via Accept Header

Response

Note there is a no entry in the fgtgroupfilter table (total_count=0)

Local Users (/localusers/)

URL:

This end-point represents local user resource i.e. a user account. In the FortiAuthenticator GUI, this resource corresponds to *Authentication > Local Users*. This API is for use by third party provisioning systems.

Supported Fields

Field	Display Name	Type	Required	Other Restriction
username	Username	string	Yes	max length = 30, contains only letters, numbers and @/./+/-/_ characters
password	Password	string	No	max length = 50
address	Address	string	No	max length = 80
city	City	string	No	max length = 40
country	Country	string	No	Must be a country code from ISO-3166 list
custom1	Custom user field 1	string	No	max length = 255
custom2	Custom user field 2	string	No	max length = 255
custom3	Custom user field 3	string	No	max length = 255
email	E-mail address	string	No	Must be a valid e-mail address
first_name	First name	string	No	max length = 30
last_name	Last name	string	No	max length = 30
mobile_number	Mobile number	string	No	max length = 25, must follow international number format: +[country_code]-[number]
phone_number	Mobile number	string	No	max length = 25
state	State or province	string	No	max length = 40
user_groups	Local user groups that this user is a member of	list	No	List of user groups URI

Allowed Methods

Type	Allowed Methods	Action	Note
List	GET	Get all regular users	
List	POST	Create a new user	If password is specified, it will be set. If password is not specified, the email field becomes required, and a random password will be created and e-mailed to the user.
Detail	GET	Get a specific user	
Detail	PATCH	Update specified fields for a specific user	

List Local Users

JSON Query

JSON specified via GET



JSON specified via Accept Header



Response



Here you will notice that there is only a single local user defined “dbane”. Note that admin users are not returned by the localusers query (hence why ID=4).

Create Local User

JSON Query

JSON specified via Accept Header

Response

Verify user creation

```
        {"address": "", "city": "",  
"country": "", "custom1": "", "custom2": "", "custom3": "", "email": "",  
"first_name": "", "id": 7, "last_name": "", "mobile_number": "", "phone_number": "",  
"resource_uri": "/api/v1/localusers/7/", "state": "", "user_groups": [], "username":  
"testuser"}]}
```

Local Users (/usergroups/)

URL:

This end-point represents the user group resource. In the FortiAuthenticator GUI, this resource corresponds to *Authentication > User Groups*. This API is for use by third party provisioning systems.

View User Groups

JSON Query

JSON specified via GET

```
[{"name": "Test", "users": []}]
```

JSON specified via Accept Header

```
[{"name": "Test", "users": []}]
```

Response

```
[{"name": "Test", "users": []}]
```

There is a single group named "Test" with no users currently assigned.

Create a User Group

JSON Query

JSON specified via Accept Header

```
{ "name": "Test", "users": [] }
```

Response



Verify group creation



SSO Auth (/ssoauth/)

URL:

This end-point represents the Fortinet SSO Authentication. In the FortiAuthenticator GUI, this resource corresponds to *Fortinet SSO Methods* > *SSO*. This API is for use by third party authentication systems for dynamic transparent user Single Sign-on to a Fortinet protected network.

Note that before attempting to authenticate, additional configuration is required under *Fortinet SSO Methods* > *Portal Services* > *SSO Web Service* to select which user directory is to be used for group embellishment.

Supported Fields

Field	Display Name	Type	Required	
event	Event type	integer/string	Yes	0=Login 1=Logout
username	User's username	string	Yes	max length=253
user_ip	User's workstation IP (Calling-Station-Id)	IPv4	Yes	
user_groups	Groups to send (Fortinet-Group-Name)	string	No	max length=253, list of groups must be separated with "+" character (group name cannot contain a "+" character)

Allowed Methods

Type	Allowed Methods	Action	Note
List	POST	Logon/logoff users to/from FSSO	

Response Codes

In addition to the general codes defined in [Appendix A –API Response Codes](#), a POST request to this resource can result in the following return codes:

Code	Response Content	Description
200 OK		FSSO login/logout request has been successfully sent to FSSO (but this doesn't mean that user has been logged-on/off, as the request is done asynchronously and is queued on FSSO side. Factors such as configuration and user not existing in LDAP may cause the entry to not

		populate FSSO).
404 Not Found	SSO web service is disabled	SSO web service has not been enabled so it can't be used in REST API
500 Internal Server Error		Failed to send logon/logoff request to FSSO

FSSO User Login

JSON Query

JSON specified via Accept Header



Response



Verify Login on FortiAuthenticator



Overwrite FSSO user login with different user

Note that if a login event is received with the same IP address but with a different username, the existing entry will be overwritten.

JSON Query

JSON specified via Accept Header



Response



Verify Login on FortiAuthenticator



Logout FSSO User

JSON Query

JSON specified via Accept Header



Response



Verify Logout on FortiAuthenticator

The screenshot shows the FortiAuthenticator web interface. The top navigation bar includes 'System', 'Authentication', 'Fortinet SSO Methods', and 'Monitor'. The 'Monitor' tab is selected, showing a search bar and a list of 0 SSO users. The left sidebar shows a tree view with 'SSO' expanded, containing 'Domains', 'SSO Users', 'Domain Controllers', and 'FortiGates'.

Logging

Note that SSO Login requests are logged regardless of whether the user details can be inserted into FSSO. For example logs may exist for SSO Logon for a user but an entry not appear in the monitor because when an LDAP lookup for group info was performed, no user existed.

The screenshot shows the FortiAuthenticator web interface with the 'Logging' tab selected. The interface displays a table of log records. The table has columns: ID, Timestamp, Level, Category, Sub category, Type id, NAS name/IP, and Short message. The table contains 26 log entries, including authentication requests, logoff requests, and system events.

ID	Timestamp	Level	Category	Sub category	Type id	NAS name/IP	Short message
2631	Fri Sep 20 09:45:40 2013	information	Event	Web Service	50501		Receiving an authentication request for user "cwindsor"
2630	Fri Sep 20 09:42:32 2013	information	Event	Web Service	50501		SSO logoff request sent for user "cwindsor" with IP 10.1.73.175
2629	Fri Sep 20 09:42:16 2013	information	Event	Web Service	50501		SSO logon request sent for user "cwindsor" with IP 10.1.73.175
2628	Fri Sep 20 09:41:47 2013	information	Event	Web Service	50501		SSO logon request sent for user "atano" with IP 10.1.73.175
2627	Fri Sep 20 09:41:25 2013	information	Event	Web Service	50501		SSO logoff request sent for user "cwindsor" with IP 192.1.73.175
2626	Fri Sep 20 09:41:22 2013	information	Event	Web Service	50501		SSO logon request sent for user "atano" with IP 10.1.73.175
2625	Fri Sep 20 09:35:48 2013	information	Event	Web Service	50501		SSO logon request sent for user "atano" with IP 10.1.73.175
2624	Fri Sep 20 09:35:36 2013	information	Event	Web Service	50501		SSO logon request sent for user "atano" with IP 10.1.73.175
2623	Fri Sep 20 09:35:10 2013	information	Event	Web Service	50501		SSO logon request sent for user "cwindsor" with IP 10.1.73.175
2622	Fri Sep 20 09:33:17 2013	information	Event	Web Service	50501		SSO logoff request sent for user "cwindsor" with IP 10.1.73.175
2621	Fri Sep 20 09:32:53 2013	information	Event	Web Service	50501		SSO logoff request sent for user "cwindsor" with IP 192.1.73.175
2620	Fri Sep 20 09:31:07 2013	information	Event	Web Service	50501		SSO logon request sent for user "cwindsor" with IP dave
2619	Fri Sep 20 09:28:54 2013	information	Event	Web Service	50501		SSO logon request sent for user "cwindsor" with IP 10.1.73.175
2618	Fri Sep 20 09:28:18 2013	information	Event	Authentication	20998		User 'admin' logged in
2617	Fri Sep 20 09:28:18 2013	information	Event	Authentication	20998		Local admin authentication with no token successful
2616	Fri Sep 20 09:27:30 2013	information	Event	Web Service	50501		SSO logon request sent for user "Carl" with IP 10.1.73.175
2615	Fri Sep 20 09:27:18 2013	information	Event	Web Service	50501		Receiving an authentication request for user "cwindsor"
2614	Fri Sep 20 09:22:51 2013	information	Event	Web Service	50501		Receiving an authentication request for user "cwindsor"
2613	Fri Sep 20 09:21:54 2013	information	Event	Web Service	50501		Receiving HTTP GET request at "/api/v1/ftggroupfilter/" from "192.168.0.154" (query p
2612	Fri Sep 20 09:21:16 2013	information	Event	Web Service	50501		Receiving HTTP GET request at "/api/v1/ftggroupfilter/" from "192.168.0.154" (query p
2611	Fri Sep 20 09:20:03 2013	information	Event	Web Service	50501		Receiving HTTP POST request at "/api/v1/ssogroup/" from "192.168.0.154" (query par
2610	Fri Sep 20 09:19:39 2013	information	Event	System	30101		radiusd running in full edition
2609	Fri Sep 20 09:19:38 2013	notice	Event	System	30100		Restarting RADIUS server to apply SSO group changes
2608	Fri Sep 20 09:19:37 2013	information	Event	System	30101		radiusd running in full edition
2607	Fri Sep 20 09:19:37 2013	information	Event	Admin Configuration	10001		Added SSO Group: Group1
2606	Fri Sep 20 09:19:37 2013	notice	Event	System	30100		Restarting RADIUS server to apply SSO group changes
2605	Fri Sep 20 09:19:36 2013	information	Event	Web Service	50501		Receiving HTTP POST request at "/api/v1/ssogroup/" from "192.168.0.154" (query par
2604	Fri Sep 20 09:17:09 2013	information	Event	Web Service	50501		Receiving HTTP GET request at "/api/v1/ssogroup/" from "192.168.0.154" (query para
2603	Fri Sep 20 02:00:05 2013	information	Event	Authentication	20150		Performing a regular check on users with expiring password

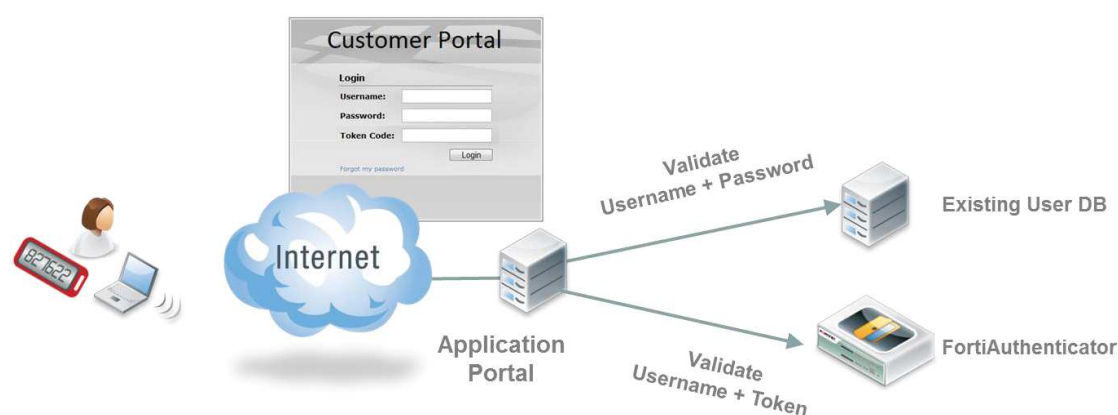
Authentication (/auth/)

URL:

The Authentication API is for validation of user credentials. Either the password, token or both can be validated. This is useful for adding an additional factor authentication (e.g. token) to web portals where the first factor is already being validated locally e.g. via LDAP or local DB or a proprietary, unsupported authentication method as is common in the banking industry.



Note: This API is for the validation of local user password and token passcode or remote user passcode only. Validation of remote user password is not supported. This is by design as most systems have an established mechanism for authentication via e.g. LDAP or some other proprietary mechanism as shown below.



To authenticate a user, you need to POST to **Error! Hyperlink reference not valid.** with the following key-value pair (in JSON format, but XML also possible):

```
{"username": "<username>", "token_code": "<token_code>", "password": "<password>"}
```

with "token_code" and "password" being optional fields i.e. you can just validate the token only or the password only. If password and token are specified, the password will be validated first before token code.

Supported Fields

Field	Display Name	Type	Required	Other Restriction
username	Username	string	Yes	
password	Password	string	No	
token_code	Security token code (only FortiToken? is supported)	string	No	

Allowed Methods

Type	Allowed	Action	Note
------	---------	--------	------

	Methods		
List	POST	Validate user's credentials	<p>Either password or token_code needs to be specified.</p> <p>If both are specified, password will be validated first, then token_code.</p> <p>If only one is specified (either password or token_code), only that credential will be validated.</p> <p>If a user doesn't have two-factor authentication configured, validation for that user with any token_codewill always pass.</p>

Response Codes

In addition to the general codes defined in [Appendix A –API Response Codes](#), a POST request to this resource can result in the following return codes:

Code	Response Content	Description
200 OK		User is successfully authenticated.
401 Unauthorized	User authentication failed	Credential is incorrect
401 Unauthorized	Token is out of sync	The security token requires synchronization
401 Unauthorized	User account is temporarily locked	User has reached maximum failed authentication attempts and is temporarily locked
404 Not Found on System	User does not exist	The given username does not exist in the system
200 OK		User is successfully authenticated.

Validate a user password

Query

JSON specified via Accept Header



Response



Validate a users token code

Query

JSON specified via Content-Type Header

```
[REDACTED]
```

Response

```
[REDACTED]
```

Error States

Response (incorrect password)

```
[REDACTED]
```

Response (incorrect token code)

```
[REDACTED]
```

Response (incorrect username)

```
[REDACTED]
```

Appendix A –API Response Codes

General API Response Codes

Code	Description
200 OK	The request was successfully completed.
201 Created	The request successfully created a new resource and the response body does not contain the newly created resource.
204 No Content	The server fulfilled the request, but does not need to return a response message body.
400 Bad Request	The request could not be processed because it contains missing or invalid information (i.e. the data in the request does not validate).
401 Not Authorized	The supplied credential is incorrect.
500 Internal Server Error	The server encountered an unexpected condition which prevented it from fulfilling the request.