Guide to API Authentication - API Tokens in WHM

Introduction

API tokens allow you to call WHM API 1 functions outside of a WHM session. You can use API tokens instead of a password or access hash key to call WHM API 1 functions over HTTPS. This is useful, for example, to allow a reseller user or third-party developer to run API function calls with your account's data.

Important:

- We introduced this functionality in cPanel & WHM version 64.
- API calls using a method that includes a URL **must** use one the following ports:
  - `2087` — Secure calls to WHM's APIs, or to cPanel's APIs via the WHM API.
  - `443` — Secure calls to WHM's APIs, or to cPanel's APIs via the WHM API through a service subdomain.

Creating an API token

To create an API token, use WHM's *Manage API Tokens* interface (*WHM >> Home >> Security Center >> Manage API Tokens*). Then, include that API token in your custom code.

Warning:
Make **certain** that you save your API token in a secure location on your workstation. You **cannot** access the token after navigating away from the interface or if you refresh the *API Tokens* table.

Using an API token

To call a WHM API 1 function with an API token, call it via HTTPS and include the following HTTP header in your request:

```
Authorization: whm username:token
```

Note:
In this example, `username` represents the `root` user or a reseller user and `token` represents the API token.

Example scripts

Use the following examples to create a script that includes the API token in your custom code:

curl Perl PHP

Note:

- Specify either the `root` user or a reseller user for the `username` entry.
- Replace `MYAPITOKEN` with your valid API token.
- Replace `127.0.0.1` with your server's IP address.

Related documentation

- Manage2 API Functions - Register Key-Based Authentication — This function registers a server to use keyed authentication.

```
curl -H'Authorization: whm
username:MYAPITOKEN'
'https://127.0.0.1:2087/json-api/applis
t?api.version=1'
```

Note:

- Specify either the `root` user or a reseller user in line 9.
- Replace `MYAPITOKEN` in line 10 with your valid API token.
- Replace `127.0.0.1` in line 18 with your server's IP address.

```
#!/usr/bin/perl
use strict;
use warnings;

use JSON       ();
use HTTP::Tiny ();


my $user  = 'root';
my $token = 'MYAPITOKEN';
my $ua = HTTP::Tiny->new(
    'verify_SSL'      => 0,
    'default_headers' => {
        'Authorization' => "whm
$user:$token",
    },
);

my $response =
$ua->get("https://127.0.0.1:2087/json-a
pi/listaccts?api.version=1");
if ( $response->{'success'} ) {
    my $json = JSON::decode_json(
$response->{'content'} );
    print "[+] Current cPanel users on
the system:\n";
    print "\t$_\n" for map {
$_->{'user'} } @{
$json->{'data'}->{'acct'} };
}
else {
    print "[!] Error:
$response->{'status'}
$response->{'reason'} returned\n";
}
```

- In line 10, the script declares the `$token` variable and assigns the API token hash to it as a value.
- In line 11, the script creates an `HTTP::Tiny` user agent. It also configures it

- Guide to External Authentication - OpenID Connect — External authentication allows your server's users to log in to WHM, cPanel, and Webmail through OpenID Connect-compliant identity providers.

to send the token authorization headers with every request.

- Line 18 invokes the WHM API 1 `listaccts` function via the `HTTP::Tiny` user agent and saves the response in the `$response` variable.
- The script prints the following output:
  - If the `listacct` function succeeds, it parses the JSON data and returns the account's usernames.
  - If the `listacct` function fails, it returns an error message.

Notes:

- Specify either the `root` user or a reseller user in line 2.
- Replace `MYAPITOKEN` in line 3 with your valid API token.
- Replace `127.0.0.1` in line 5 with your server's IP address.

- Guide to External Authentication — External Authentication modules allow users to log in through OpenID Connect-compliant identity providers.
- Guide to Testing Custom Code - API Authentication — This guide explains the basics of how to test your custom code's authentication with the cPanel & WHM server.

```
<?
    $user = "root";
    $token = "MYAPITOKEN";

    $query =
"https://127.0.0.1:2087/json-api/listac
cts?api.version=1";

    $curl = curl_init();
    curl_setopt($curl,
CURLOPT_SSL_VERIFYHOST,0);
    curl_setopt($curl,
CURLOPT_SSL_VERIFYPEER,0);
    curl_setopt($curl,
CURLOPT_RETURNTRANSFER,1);

    $header[0] = "Authorization: whm
$user:$token";

curl_setopt($curl,CURLOPT_HTTPHEADER,$h
eader);
    curl_setopt($curl, CURLOPT_URL,
$query);

    $result = curl_exec($curl);

    $http_status = curl_getinfo($curl,
CURLINFO_HTTP_CODE);
    if ($http_status != 200) {
        echo "[!] Error: " .
$http_status . " returned\n";
    } else {
        $json = json_decode($result);
        echo "[+] Current cPanel users
on the system:\n";
        foreach
($json->{'data'}->{'acct'} as
$userdetails) {
            echo "\t" .
$userdetails->{'user'} . "\n";
        }
    }

    curl_close($curl);
?>
```

- Line 2 sets the `$user` value as the `root` user.
- Line 3 sets the `$token` value as the contents of the appropriate API token.
- Line 5 initializes the WHM API 1 `listaccts` function via a `curl` call. It then configures the call to send the `Authorization: token $user:$token` h eaders with every request.

- Guide to API Authentication - Browser-Based Authentication — Browser-based authentication accesses cPanel & WHM's functionality via the browser.

- Line 16 performs the WHM API 1 `listaccts` function via a `curl` call and saves the response in the `$response` variable.
- The script prints the following output:
    - If the `listacct` function succeeds, it parses the JSON data and returns the account's usernames.
    - If the `listacct` function fails, it returns an error message.