

Guide to Standardized Hooks - Script Hooks

Guide to Standardized Hooks

Hookable Events

[Convert Addon Functions](#)

[Cpanel Functions](#)

[PkgAcct Functions](#)

[Passwd Functions](#)

[RPM::Versions Functions](#)

[Stats Functions](#)

[System Functions](#)

[Whostmgr Functions](#)

[Hook Action Code](#)

[The describe\(\) Method](#)

[The manage_hooks Utility](#)

[Debug Mode](#)

Advanced Use

[Rollbacks](#)

[Checks](#)

[Exceptions](#)

[Privilege Escalation](#)

[Hookable Events in Custom Modules](#)

Deprecated Systems

[Custom Event Handlers](#)

[Function Hooks](#)

[Script Hooks](#)

[Universal Password Trap](#)

Introduction

Script hooks execute custom code before and after certain system-level operations. For example, script hooks can run before or after account creation, account modification, server software installation, or backup runs.

To view a list of scripts for which you can create script hooks, read our [WHM Scripts documentation](#).

Warning:

This hook method is **deprecated**. To convert script hooks to use the Standardized Hooks system, use [System](#) or [Whostmgr events](#) in your Hook Action Code.

Basic usage

To create a script hook, perform the following steps:

1. In any programming language, write a shell script to perform the desired actions.
 - Data passes through script hooks via the `@ARGV` array.
 - The system **always** executes script hook scripts as the `root` user.
2. Store the script in the `/usr/local/cpanel/scripts/` directory.

Note:

Before you create a new script hook, check whether the hook exists in our list of available script hooks.

Convert @ARGV hashes

Because the system passes some script hooks to a hash through the `@ARGV` array, you **must** convert the `@ARGV` hash into a usable data structure.

Select a tab to view information for that programming language:

[Perl](#)[PHP](#)[Python](#)

To convert `@ARGV` into a usable data structure in Perl, assign it to a hash:

```
my %OPTS = @ARGV
```

To access the data in the `%OPTS` hash, assign it to a variable:

```
my $username = $OPTS{'user'};
```

To convert `@ARGV` into a usable data structure in PHP, convert it into an associative array:

Standardized Hooks

Related Documentation

- [Manage Hooks](#)
- [Track Settings - Development](#)

```
function argv2array ($argv) { $opts =  
array(); $argv0 = array_shift($argv);  
  
while(count($argv)) { $key =  
array_shift($argv); $value =  
array_shift($argv); $opts[$key] =  
$value; } return $opts; }
```

Pass the `$argv` variable through the function to assign the data to the `$opts` variable:

```
$opts = argv2array($argv);
```

To access the data in the `$opts` variable, you could use the following code:

```
$opts['user'];
```

To convert `@ARGV` into a useable data structure in Python, convert it into a dictionary:

```
#!/usr/bin/env python  
import sys  
opts =  
dict(zip(*[iter(sys.argv[1:])] * 2))
```

To access the data in the `opts` dictionary, you could use the following code:

```
print opts['user']
```

Available script hooks

cPanel & WHM ships with hooks already available for the following scripts and system actions:

`/scripts/cpbackup`

The `/scripts/cpbackup` script's hooks trigger each time that cPanel & WHM runs a backup.

For these script hooks to function correctly, you **must** add the following lines to the `/etc/cpbackup.conf` file:

```
PREBACKUP 1
POSTBACKUP 1
```

By default, the system triggers the `pre` hook after it performs checks and validations. To cause the hook to run **before** the checks and validations (for example, regardless of whether backups are up-to-date), add the following line to the `/etc/cpbackup.conf` file:

```
precpbackup -1
```

pre hook script:

`/scripts/precpbackup`

post hook

script:

`/scripts/postcpbackup`

`/scripts/killacct`

You can select the following options:

-
-
-
-
-

This setting defaults to *Debug mode is off*.

Note: If, instead of `/var/cpanel/cpsrvd`, you use `/var/cpanel/cpsrvd`, the file to create is `daemc`.

The `/scripts/killacct` script's hooks trigger each time that the system deletes an account.

pre hook script:

```
/scripts/prekillacct
```

post hook

script:

```
/scripts/postkillacct
```

When the system runs the `/scripts/postkillacct` script, it passes in the `%OPTS` hash, which contains the following parameters:

Parameter	Type	Description	Possible values	Example
<code>user</code>	<i>string</i>	The terminated account's username.	A valid username.	<code>username</code>
<code>killdns</code>	<i>boolean</i>	Whether the system deleted the account's zone files during termination.	<ul style="list-style-type: none">• 1 — The system deleted the zone files.• 0 — The system did not delete the zone files.	0

`/scripts/suspendacct`

The `/scripts/suspendacct` script's hooks trigger each time that the system suspends an account.

pre hook script:

```
/scripts/presuspendacct
```

post hook

script:

```
/scripts/postsuspendacct
```

These scripts accept the following arguments:

- `username` — The suspended account's username.
- `reason` — The reason for account suspension.
- `disallow` — Whether to allow only the `root` user to unsuspend the account.

Important:

These arguments **must** maintain the following order: `username`, `reason`, `disallow`.

`/scripts/unsuspendacct`

- WHM API 1 Functions - `reorder_hooks`

This function changes the order of script hooks.

- WHM API 1 Functions - `delete_hook`

This function removes a script hook.

- WHM API 1 Functions - `edit_hook`

This function edits a script hook.

The `/scripts/unsuspendacct` script's hooks trigger each time that the system un suspends an account.

pre hook script:

`/scripts/preunsuspendacct`

This script accepts the following argument:

- `username` — The suspended account's username.

post hook

script:

`/scripts/postunsuspendacct`

This script accepts the following argument:

- `user` — The suspended account's username.

`/scripts/upcp`

The `/scripts/upcp` script's hooks trigger each time that cPanel & WHM updates.

pre hook script:

`/scripts/preupcp`

post hook

script:

`/scripts/postupcp`

`/scripts/updateuserdomains`

The `/scripts/updateuserdomains` script's hooks trigger each time that the system generates a domain list.

pre hook script:

`none`

post hook

script:

`/scripts/postupdateuserdomains`

`/scripts/enXim-pkgacct`

The `/scripts/enXim-pkgacct` script's hooks trigger each time that the system packages an account into a `cpmove` file via this script. Generally, this occurs when an account transfers from the Ensim® control panel to a cPanel & WHM server.

Note:

These hooks **do not** trigger for cPanel & WHM-generated `cpmove` files.

pre hook script:

`/scripts/prepkgacct`

This script accepts the following argument:

- `user` — The account's username.

post hook

script:

`/scripts/postpkgacct`

This script accepts the following argument:

- `user` — The account's original (remote) username.
- `split` — The `cpmove` file consists of multiple files.
- `nosplit` — The `cpmove` file consists of a single file.
- `cpuser` — The account's new (local) username.
- `splitdir` — The directory that contains the `cpmove` file.

`/scripts/restoreacct`

The `/scripts/restoreacct` script's hooks trigger each time that the system restores an account.

pre hook script:

`/scripts/prerestoreacct`

This script accepts the following arguments:

- `cpuser` — The account's new (local) username.
- `olduser` — The account's old (remote) username.
- `extractdir` — The directory to which the system will extract the `cpmove` file.

post hook

script:

`/scripts/postrestoreacct`

This script accepts the following arguments:

- `user` — The account's new (local) username.
- `olduser` — The account's old (remote) username.
- `domain` — The account's main domain.
- `user_homedir` — The absolute path to the account's home directory.

`/scripts/wwwacct`

The `/scripts/wwwacct` script's hooks trigger each time that the system creates a cPanel account.

For more information, read the [/scripts/wwwacct script](#) documentation.

pre hook script:

`/scripts/prewwwacct`

post hook

script:

`/scripts/postwwwacct`

In these scripts, the `@ARGV` array passes in a hash that contains the following data:

Argument	Type	Description	Possible values	Example
<code>domain</code>	<i>string</i>	The account's main domain name.	A valid domain name.	<code>example.com</code>
<code>user</code>	<i>string</i>	The account's username.	A valid username.	<code>user</code>
<code>pass</code>	<i>string</i>	The account's password.	A secure password.	<code>12345luggage</code>

quota	<i>integer</i>	The account's disk space quota.	<ul style="list-style-type: none"> • An integer value between one and 999999 that represents a disk space quota in Mega bytes (MB). • 0 — The account possesses unlimited disk space. 	0
cpmod	<i>string</i>	The account's cPanel theme.	<ul style="list-style-type: none"> • paper_lantern • Another valid theme on the server. 	paper_lantern

ip	<i>string</i>	<p>Whether the account has a dedicated IP address.</p> <p>This parameter defaults to n.</p>	<ul style="list-style-type: none"> • y — The account possesses a dedicated IP address. • n — The account does not possess a dedicated IP address. 	n
cgi	<i>string</i>	Whether the account has CGI access.	<ul style="list-style-type: none"> • y — CGI access. • n — No CGI access. 	y
frontpage	<i>string</i>	Whether Microsoft® FrontPage® Extensions exist on the account.	<ul style="list-style-type: none"> • y — Installed. • n — Not installed. <div style="border: 1px solid orange; width: 40px; height: 40px; margin: 10px 0;"></div> <p>Note: In cPanel & WHM version 11.46 and later, this always</p>	n

maxftp	<i>string</i>	The account's maximum number of FTP accounts.	<ul style="list-style-type: none"> • A positive integer between one and 999,999. • 0, unlimited, or null — Unlimited. 	0
maxsql	<i>string</i>	The account's maximum number of SQL databases	<ul style="list-style-type: none"> • A positive integer between one and 999,999. • 0, unlimited, or null — Unlimited. 	0
maxpop	<i>string</i>	The account's maximum number of email addresses.	<ul style="list-style-type: none"> • A positive integer between one and 999,999. • 0, unlimited, or null — Unlimited. 	0

maxlst	<i>string</i>	The account's maximum number of Mailman mailing lists.	<ul style="list-style-type: none"> • A positive integer between one and 999,999. • 0, unlimited, or null — Unlimited. 	0
maxsub	<i>string</i>	The account's maximum number of subdomains.	<ul style="list-style-type: none"> • A positive integer between one and 999,999. • 0, unlimited, or null — Unlimited. 	0
bwlimit	<i>string</i>	The account's bandwidth quota.	<ul style="list-style-type: none"> • A positive integer between one and 999,999. • 0, unlimited, or null — Unlimited. 	0

hasshell	<i>string</i>	Whether the account has shell access.	<ul style="list-style-type: none"> • y — Shell access. • n — No shell access. 	y
owner	<i>string</i>	The WHM account that owns this account.	<ul style="list-style-type: none"> • A valid reseller's username. • root 	root
plan	<i>string</i>	The account's hosting plan (package).	A valid package name.	reseller_gold
maxpark	<i>string</i>	The account's maximum number of parked domains (aliases). This parameter defaults to unlimited.	<ul style="list-style-type: none"> • A positive integer between one and 999,999. • 0, unlimited, or null — The account possesses unlimited parked domains. 	unlimited

maxaddon	<i>string</i>	<p>The account's maximum number of addon domains.</p> <p>This parameter defaults to unlimited.</p>	<ul style="list-style-type: none"> • A positive integer between one and 999,999. • 0, unlimited, or null — The account possesses unlimited addon domains. 	unlimited
featurelist	<i>string</i>	<p>The account's feature list.</p> <p>If you do not use this parameter, the function assigns the default feature list to the account.</p>	A valid feature list name on the server.	feature_list
contactemail	<i>string</i>	The account's contact email address.	A valid email address.	user@example.com

use_registered_nameservers	<i>string</i>	Whether to use the domain's registered nameservers instead of the server's nameservers.	<ul style="list-style-type: none"> • y — Use the domain's name servers. • n — Use the server's name servers. 	0
language	<i>string</i>	The account's locale.	A valid locale name.	en

spamassassin	<i>string</i>	<p>Whether Apache SpamAssassin™ is enabled on the account.</p> <div data-bbox="748 359 857 1745" style="border: 1px solid orange; padding: 5px; text-align: center;"> <p>N o t e : W e a d d e d t h i s p a r a m e t e r i n c P a n e l & W H M v e r s i o n 7 0 .</p> </div>	<ul style="list-style-type: none"> • y — Enabled. • n — Disabled. 	y
max_emailacct_quota	<i>integer</i>			unlimited

The maximum size that the account can define when it creates an email account.

- unlimited — The account possesses an unlimited quota.

I m p o r t a n t : If you supply a

plan

value with this parameter, or

- A positive integer between one and 4,294,967,296.

s e t h i s p a r a m e t e r w i t h o u t t h e p l a n v a l u e , i t o v e r r i d e s t h e h o s t i n g p l a n ' s d e

fi
n
e
d
v
a
l

ue . We recommend that you allow the account's splantodetermine this value .

**N
o
t
e
:**

- The system uses the plan parameter's value if you use it with `max_emailacct_quota` parameter.
- We introduced this parameter in cPanel & WHM version 70.

This parameter defaults to unlimited.

User creation

The following script hooks trigger each time that the system creates a user.

pre hook script:

none

post hook

script:

`/scripts/postwwwacctuser`

This script accepts the following argument:

- `user` — The new account's username.

Account modification

The following script hooks trigger each time that the system modifies an account.

pre hook script:

`/scripts/premodifyacct`

post hook

script:

`/scripts/postmodifyacct`