SSH Users CLI Usage Examples

Overview

On your EC2 instances your VirtualHost's or "webroot's" (as we will refer to to them in this document) have their own user account for security purposes. This gives each application that runs under a webroot their own distinct permissions boundaries for files and running processes. Your SSH user is granted access to assume these users VIA SUDO NOPASSWD. In this document we will show you various examples how you can use the SUDO command to start an SSH Shell session, an sftp session etc... assuming a webroot user thus granting you permissions to file resources and processes running under their distinct permissions boundaries.

SSH Users CLI Usage Examples

Assuming a webroot user VIA SUDO

While logged into an EC2 instance you can use SUDO to assume a webroot user. Assuming a webroot user grants you access to their resources and processes. In addition to that access any resources you create or processes you start will be under the webroot's account.

The command:

```
sudo -u {WEBROOT} -s
```

Replace the bracketed value(s) with:

{WEBROOT} = The name of the webroot to assume

Example using the following values:

{WEBROOT} = example.com

```
sudo -u example.com -s
```

The above command will start a new shell session under the "example.com" user account.

SSH Users CLI Usage Examples

SSH Shell Session

You can start an SSH Shell session and instantly assume a webroot account.

The command:

```
ssh -t {SSH_USER}@{INSTANCE_IP} "sudo -u {WEBROOT} -s"
```

Replace the bracketed value(s) with:

{WEBROOT} = The name of the webroot to assume
{SSH_USER} = The username of your SSH account
{INSTANCE_IP} = The IP Address or DNS Name of your EC2 instance

Example using the following values:

{WEBROOT} = example.com
{SSH_USER} = developer
{INSTANCE_IP} = 172.13.14.15

```
ssh -t developer@172.13.14.15 "sudo -u example.com -s"
```

SSH Users CLI Usage Examples

SFTP

To start an SFTP session VIA cli as a webroot user we will set the remote SFTP subsystem using SUDO assuming the webroot user.

Comand:

```
sftp -s "sudo -u {WEBROOT} -s /bin/sftp-server"
{SSH_USER}@{INSTANCE_IP}:/var/www/html/{WEBROOT}
```

Replacing the bracketed values with:

{WEBROOT} = The name of the webroot to assume
{SSH_USER} = The username of your SSH account
{INSTANCE_IP} = The IP Address or DNS Name of your EC2 instance

Example using the following values:

{WEBROOT} = example.com
{SSH_USER} = developer
{INSTANCE_IP} = 172.13.14.15

```
sftp -s "sudo -u example.com -s /bin/sftp-server"
developer@172.13.14.15:/var/www/html/example.com
```

Your SFTP session will now start in the webroot's directory and the files you "PUT" will be owned by the webroot user.

SSH Users CLI Usage Examples

RSYNC

The "rsync" command exposes an option flag (--rsync_path) to set the "rsync" path on the remote system. We can use this option flag to configure the remote session to assume a webroot VIA SUDO thus files synchronized during the session will be owned by the webroot user.

The command: (We will set some arbitrary options such as (-v) verbose, (-a) archive, (-delete) delete and (-z) GZIP however these options can be modified to fit your use-case.)

```
rsync -vaz --delete --rsync-path="sudo -u {WEBROOT} -s /usr/bin/rsync"
{LOCAL_PATH} {SSH_USER}@{INSTANCE_IP}:/var/www/html/{WEBROOT}
```

Replace the bracketed values with:

{WEBROOT} = The name of the webroot to assume
{SSH_USER} = The username of your SSH account
{INSTANCE_IP} = The IP Address or DNS Name of your EC2 instance
{LOCAL_PATH} = Path on our system to synchronize to remote

Example using the following values:

{WEBROOT} = example.com
{SSH_USER} = developer
{INSTANCE_IP} = 172.13.14.15
{LOCAL_PATH} = A folder named "example.com" on your Desktop

```
rsync -vaz --delete --rsync-path="sudo -u example.com -s /usr/bin/rsync"
~/Desktop/example.com developer@127.13.14.15:/var/www/html/example.com
```

You can swap the source and destination path to sync from remote to local as well.

SSH Users CLI Usage Examples