SSH Deploy Key Documentation

Release 0.3.0

Travis Bear

March 24, 2014

CHAPTER 1

Overview

SSH deploy-key is an easy-to-use tool that deploys ssh keys to remote hosts. It makes deploying to a single host simple. It makes deploying to many hosts simple and fast.

CHAPTER 2

Source Code

The source code for ssh-deploy-key code is hosted on Bitbucket at this location:

https://bitbucket.org/travis_bear/ssh-deploy-key

CHAPTER 3

Contents

3.1 Alternatives

There are lots of ways to copy out an ssh key to a remote host. It can be done by hand. It can be done with existing tools like ssh-copy-id. It can be done using configuration management tools. So why use SSH Deploy Key?

Although ssh-deploy-key is not ideal for every scenario, its speed and simplicity make it a good choice in many situations. It is a tool that is tightly focused on one task only – moving ssh keys out to remote hosts as easily and as quickly as possible.

3.1.1 Alternatives

Here is a comparison of ssh-deploy-key with some other common ways to deploy a key.

Manual Deployment

Deploying ssh keys by hand is a time-honored technique that in general works pretty well. However, in almost all cases, using ssh-deploy-key is a better option. It's faster, easier, more reliable, and more repeatable. When deploying to more than one host at a time, these advantages only multiply with ssh-deploy-key's bulk deployment abilities.

There is one use case where deploying by hand is a better bet: when the remote host is on a different network, behind a jump box. ssh-deploy-key does not handle that scenario.

ssh-copy-id

ssh-copy-id is a great tool, but it's not the ideal solution for every scenario.

- ssh-copy-id is not installed by default on all systems, notably on Mac OS.
- ssh-copy-id has no concept of 'smart append'. It will append a key to a remote host's authorized keys file regardless of whether that key is already present.
- Scripting the use of ssh-copy-id for deploying to multiple remote hosts can be challenging:
  - The password is entered interactively for each host.
  - In the case where there are numerous remote hosts that have not seen before, you'd need to interactively allow each host to be added to your known_hosts file.

Configuration Management Tools

Configuration management tools (like Puppet, Ansible, etc.) can do a terrific job deploying ssh key(s). But if you are not already set up to use them for key distribution, these general-purpose solutions can be overkill, especially when compared with a dedicated tool like ssh-deploy-key that only does one thing.

3.2 Compatibility

3.2.1 Python Versions Supported

ssh-deploy-key is currently only compatible with cpython 2.7.

3.2.2 Unsupportable Python Versions

SSH Deploy Key uses the Paramiko ssh library. Paramiko is not a "pure python" solution. It has binary dependencies that make it incompatible with non-cpython implementations, including:

- jython
- pypy

3.2.3 Python 3

Getting Python 3 support in ssh-deploy-key is a priority. Unfortunately, the ssh library we use – Paramiko – does not yet support Python 3. When this is available, we'll update ssh-deploy-key right away to include Python 3 support.

See this thread for more info on the status of running Paramiko on Python 3:

https://github.com/paramiko/paramiko/issues/16

3.3 Installation Notes

ssh-deploy-key is normally installed via pip. However, on some systems, there are source files that must be installed first.

3.3.1 Install the Prerequisites

ssh-deploy-key depends on the Paramiko ssh library, which requires the Python sources. You can install these using the normal package managers for your OS.

Debian/Ubuntu (apt-get)

```
sudo apt-get install python-dev
```

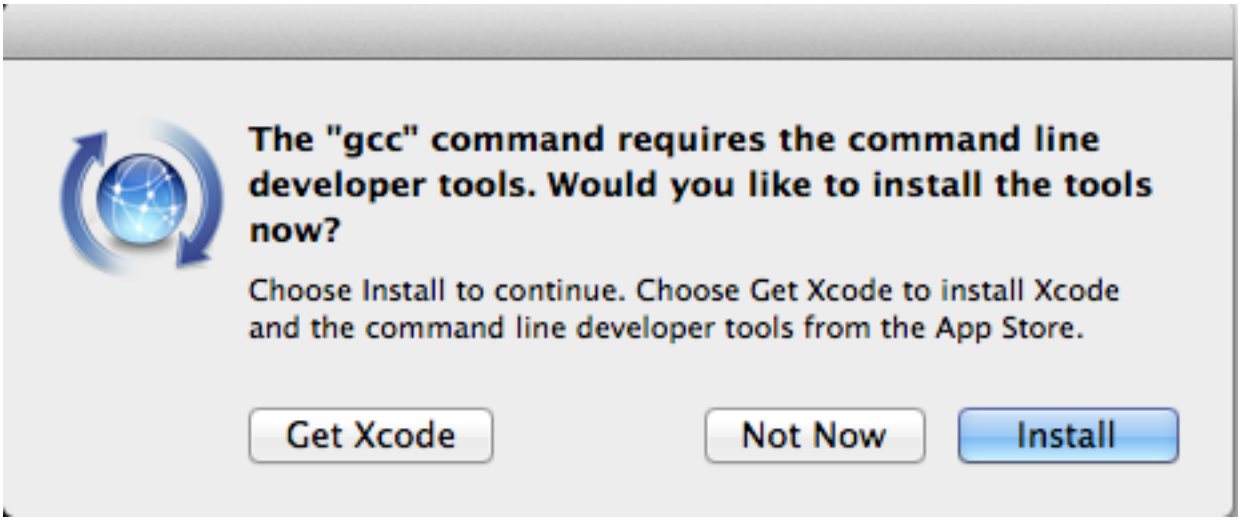
Red Hat/Centos (yum)

```
sudo yum install python-devel
```

OS X

On macs, the Python sources do not need to be installed, but a compiler is required. And compilers are not pre-installed by default.

Normally, you can just run the pip command to install ssh-deploy-key (see below), and if no compiler is currently on your system, you will be prompted to install one:



If this happens, click the 'install' button, then run the pip command again.

3.3.2 Install ssh-deploy-key via Pip

Once the development stuff is in place, ssh-deploy-key is installed with the pip command. Pip is the standard Python package installation tool. To get it, see http://www.pip-installer.org/en/latest/installing.html

Then,

```
sudo pip install ssh-deploy-key
```

Boom! Done!

3.4 Usage

ssh-deploy-key [options] [remote host[s]]

3.4.1 Options

```
usage: ssh-deploy-key [-h] [-a AUTHORIZED_KEYS] [-d] [-k KEY_FILE]
                      [-m TIMEOUT_SECONDS] [-o PORT] [-p PASSWORD]
                      [-s SSH_DIR] [-t THREADS] [-u USERNAME]
                      [hosts [hosts ...]]

Distribute an ssh key to remote hosts.

positional arguments:
  hosts                 Zero or more remote hosts to receive the ssh key. If
```

```
                        this value is unspecified, remote hosts will be read
                        from standard in.

optional arguments:
-h, --help              show this help message and exit
-a AUTHORIZED_KEYS, --authorized_keys AUTHORIZED_KEYS
                        Name of the remote authorized keys file. (Changing
                        this setting is uncommon.)
-d, --append            Add the ssh key to the end of the remote authorized
                        keys file instead of overwriting it. Default is
                        false.  Key will only be added if it's not already
                        present.
-k KEY_FILE, --key_file KEY_FILE
                        Path to the local public ssh key file. Default is
                        ~/.ssh/id_rsa.pub
-m TIMEOUT_SECONDS, --timeout_seconds TIMEOUT_SECONDS
                        Timeout value (in seconds) for connecting to each
                        host. Default is 3
-o PORT, --port PORT    The ssh port to connect to the remote hosts on.
                        Default is 22
-p PASSWORD, --password PASSWORD
                        Password to use on remote hosts. If not specified
                        here, you will be prompted for this interactively.
-s SSH_DIR, --ssh_dir SSH_DIR
                        Directory to copy the key into on the remote host.
                        Default is ~/.ssh
-t THREADS, --threads THREADS
                        Number of threads to use for simultaneous key
                        distribution. Default is 100.
-u USERNAME, --username USERNAME
                        Username to use on remote hosts. Default is <current user>
```

Remote hosts can be given in either the 'hostname' or 'user@hostname' format. If 'user@hostname' format is used, this value for 'user' will override anything specified in the -u/–username arguments.

3.4.2 Examples

These are some of the common ways to use ssh-deploy-key

Specifying remote hosts interactively

ssh-deploy-key can run interactively. The user will be prompted for additional hosts until typing 'exit' or ^D.

```
[~/git/ssh-deploy-key/bin]$ ssh-deploy-key
Enter common password for remote hosts:
Distributing key '/Users/travis/.ssh/id_rsa.pub' to remote hosts in overwrite mode.
Enter one hostname per line.  Terminate with 'exit' or ^D.
192.168.1.112
  copying key to travis@192.168.1.112:~/.ssh/authorized_keys... [SUCCESS!]
192.168.1.113
  copying key to travis@192.168.1.113:~/.ssh/authorized_keys... [SUCCESS!]
exit
```

Note that if you do not specify a password for the remote host on the command line, you will prompted for it interactively.

Specifying remote hosts on the command line

```
[~/git/ssh-deploy-key/bin]$ ssh-deploy-key 192.168.1.112 othello@192.168.1.101
Enter common password for remote hosts:
Distributing key '/Users/travis/.ssh/id_rsa.pub' to remote hosts in overwrite mode.
  copying key to travis@192.168.1.112:~/.ssh/authorized_keys... [SUCCESS!]
  copying key to othello@192.168.1.101:~/.ssh/authorized_keys... [SUCCESS!]
```

Note the mix of 'host' and 'user@host' formats for the remote host.

With Shell Redirection

ssh-deploy-key accepts piped input. For example, if you had a script to generate a list of hosts, you could run it this way

```
get_host_list.sh | ssh-deploy-key
```

From a data File

If you have a data file listing your hosts already, you can redirect standard in from the file

```
ssh-deploy-key < host_list
```

Specifying the username and password on the command line

```
ssh-deploy-key -u root -p p@ssw0rd host1
```

3.5 License

3.5.1 License Version

SSH deploy key is licensed under version 2.1 of the GNU Lesser GPL.

3.5.2 License Text

The full text of the license is available here:

http://www.gnu.org/licenses/lgpl-2.1.txt