

# Lab: Generating SSH Keys on MacOS / Linux

## Overview



SSH (or Secure *SH*ell) keys provide us with a high-security protocol that allows us to authenticate without using a password. Some roles require SSH keys in order to access different servers/services. In order to connect to those services you'll need to create and enroll your own set of keys.

Each set of keys consists of two long strings of characters, a public key and a private key, which will be protected with a passphrase (or password). You place the public key on remote servers you need to access while the private key remains on your local system. When the public and private key matches, the system is unlocked.

To learn more about public-private key cryptography, you can watch the [SysComm Deep Dive: OpenSSL video](#) (1:01).

## Objectives

Upon successful completion of this lab, you will have successfully

1. generated public and private SSH key pairs, and
2. uploaded your public keys and had them verified by the NOCC, and
3. add your keys to your session.

## Prerequisites

### Security Awareness Policy

Before we can begin you'll need to review and accept the [Akamai Security Awareness Policy](#) (if you have not done already). This document covers important information about protecting Akamai data, and acceptance is required in order to obtain SSH keys.

### Key Management Policy

Because SSH keys can give individuals access to potentially sensitive information and systems you'll need to review and accept the [Key Management Policy](#) (if you have not done already) before continuing.

## Kinds of Key Pairs

There are four different types of key pairs that we use at Akamai. Depending on your role, you may need more than one set.

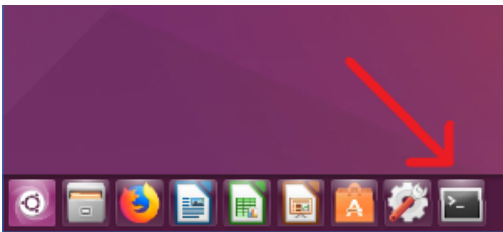
Internal	External	Deployed	Protected
----------	----------	----------	-----------

<p>Provides access to machines within the corporate network. This includes Perforce, lab machines, and other hosts that are managed internally.</p> <p>Everyone should generate this kind of key.</p>	<p>Allows access to gateways:</p> <ul style="list-style-type: none"><li>• callahan.akamai.com (East US)</li><li>• caldecot.akamai.com (West US)</li><li>• chowk.akamai.com (Bangalore)</li></ul>	<p>Used to access LSGs and to gwsh to machines on the deployed network, additionally they can be used in config. other* as allowed by authgate grants.</p>	<p>Protected keys are for NOCC personnel only, and used for the same type of access as a Deployed key. Protected keys can only be used at a NOCC workstation.</p>
---	--	--	---

Your manager will inform you if you should generate keys other than internal.

## Generate Public and Private Key Pairs

1. Open a terminal window. (Need MacOS and Ubuntu screen shots)

MacOs	Ubuntu
<screen shot here>	

2. Create the directories where the keys will be stored. The tilde (~) is a shortcut for your home directory. The period in front of .ssh indicates that directory is normally hidden.

```
$ mkdir -p ~/.ssh/internal
```

The next commands are executed if you are creating other key sets (external, deployed or protected).

```
$ mkdir -p ~/.ssh/external
$ mkdir -p ~/.ssh/deployed
$ mkdir -p ~/.ssh/protected
```

3. Use the `ssh-keygen` command to generate the keys.

The placeholder [key-type] represents the type of key and will either be “internal”, “external”, “deployed”, or “protected”.

Repeat the step below for each type of pair you need to generate, substituting the proper value for placeholder [key-type].

```
$ ssh-keygen -t rsa -b 2048 -C "`whoami`-[key-type]-`date +%Y-%m-%d`" -f ~/.ssh/[key-type]/`whoami`-[key-type]-`date +%Y-%m-%d`
```

4. List the contents of each key directory to verify you have keys.

```
$ ls ~/.ssh/internal
<username>-internal-2099-01-01  <username>-internal-2099-01-01.pub
```

Repeat as necessary.

```
$ ls ~/.ssh/external
<username>-external-2099-01-01  <username>-external-2099-01-01.pub
$ ls ~/.ssh/deployed
<username>-deployed-2099-01-01  <username>-deployed-2099-01-01.pub
$ ls ~/.ssh/protected
<username>-protected-2099-01-01  <username>-protected-2099-01-01.pub
```

The public half of each key pair is the file that ends with .pub. This is the half that will live on the remote server you’re looking to access. When you present your private half, the server will unlock because the two halves will match. In order to get the public half of your key on remote servers, you’ll need to first upload and verify your keys with the NOCC.

## Upload and Verify Keys

### Upload

1. Access the key upload page, either
  - a. From Aloha Quick Links Akamai Contacts Upload SSH Keys (link under your photo)
  - b. Or directly here: <https://sshkeys.akamai.com/>
2. Choose the key type (internal, external or deployed). Protected keys have a separate procedure.
3. Use the choose file dialog box to choose the applicable public key (file ending in .pub)
4. Click on the “Upload and request approval” button.

## Akamai SSH Upload Portal

Enter search term:   [My Profile](#)

### SSH Keys

Deployed: (No Key Uploaded)  
External: (No Key Uploaded)  
Internal: (No Key Uploaded)

#### New SSH Key

Choose  
key type:   
New public  
key:  No file chosen

If you have any issues in uploading or viewing the SSH keys, please file an issue [here](#).

## Akamai SSH Upload Portal

Enter search term:   [My Profile](#)

### SSH Keys

Deployed: (No Key Uploaded)  
External: (No Key Uploaded)  
Internal: (No Key Uploaded)

#### New SSH Key

Choose  
key type:   
New public  
key:  No file chosen

If you have any issues in uploading or viewing the SSH keys, please file an issue [here](#).

## Verify

Once your key has been uploaded you'll receive two emails. The first email will be a confirmation email, the second is a message that indicates your key is ready for verification.

Your key has a unique *fingerprint* that you will need to provide to the NOCC (or the automated key rotation service for subsequent key generations) in order to verify your identity and have the key approved. The fingerprint is shown in your terminal window when you create your key, but you may need to find the fingerprint again. To display your fingerprint use the following command, where like above, the placeholder [key-type] is either “internal”, “external” or “deployed”, and the <key filename> is the filename of the private key.

```
$ ssh-keygen -l -E md5 -f ~/.ssh/[key-type]/<key filename>
2048 32:99:94:57:1a:2b:60:49:b9:72:48:22:5d:d3:91:3d <key filename> (RSA)
```

where the fingerprint is the hexadecimal string between but not including “2048” and <key filename> (32:99:94:57:1a:2b:60:49:b9:72:48:22:5d:d3:91:3d in this example). Some versions of the command prepend the string “MD5:” to the fingerprint. This should be ignored.

Some versions of the command do not need the `-E md5` options. If generating keys fails, try again without this.

You will receive a call from the NOCC and be asked to read the fingerprint to them, which completes the verification. It may take a few hours for your keys to become effective as your keys propagate through the network.

**NEED TO SEE HOW THIS WORKS SINCE THE TELEPHONES HAVE BEEN REMOVED.**

## Add Keys to Session

In order for your keys to be effective, they need to be added to your session, that is, made available for use. Use `ssh-add` to add keys to your session. **Need to work on this one. Find out about the MacOS HostKeyAlgorithms flag for the ssh command / sshd daemon. Also need to be able to verify this and put that here.**

## Conclusion

### Review Objectives

Please review the lab’s objectives and verify you have met them by checking off each one. If you were unable to meet any of the objectives, please describe which one(s) you could not meet and why in the Feedback section.

- ☐ Generate public and private SSH key pairs.
- ☐ Upload your public keys and have them verified by the NOCC.
- ☐ Add your keys to your session.

### Feedback

If you have any questions, comments or feedback, please send an e-mail to <EngLearnFeedback>@akamai.com.