User and Host ssh keys, and configuration persistence:

This class will discuss ssh keys (both user and host). Then you will create a method to make configuration parameters persistent across reboots. Your method will probably involve a modified kickstart and one or more puppet modules.

|  |
| --- |
| For this discussion:  htc190 - Cobbler/puppet server  htc191 - Cobbler/puppet client |

User ssh keys:

S**ecure Shell (SSH)** is a cryptographic network protocol for operating network services securely over an unsecured network. The best known example application is for remote login to computer systems by users:

|  |
| --- |
| student> ssh htc191  student> ssh htc191 whoami  student> ssh -l root  student> ssh -l root htc191 whoami |

**SSH** uses a **public** key and **private** key **encryption**. If the client does not have a permanent key, ssh generates a temporary key (**session key**).

**Permanent** keys can be useful. If there is no **passphrase** protecting the keys, then you use **passwordless** ssh key **login**.

|  |
| --- |
| User keys are stored in **~/.ssh/**  **~** is shell symbol for the **user’s home directory**. For example:   **~** =  **/home/student** |

|  |
| --- |
| **WARNING**: The **~** **does not always “expand” to the user’s home directory**.  Be particularly careful when writing scripts. |

Generating permanent user ssh keys:

The command “**ssh-keygen**” is used to generate ssh keys. By **default** it will store them in **~/.ssh.**

The command below will generate **ssh keys** of type **RSA** (an encryption algorithm) using **2048** bits. The **more bits** in the keys, the **harder** they are to **crack** by **brute force** methods. **2048** is the **maximum** number of bits for **RSA**.

Accept all the defaults. **Do not enter a passphrase.**

|  |
| --- |
| student> ssh-keygen -t rsa -b 2048 |

|  |
| --- |
| Generating public/private rsa key pair.  Enter file in which to save the key (**/home/student/.ssh/id\_rsa**):  Enter passphrase (empty for no passphrase):  Enter same passphrase again:  Your identification has been saved in **/home/student/.ssh/id\_rsa**.  Your public key has been saved in **/home/student/.ssh/id\_rsa.pub**.  The key fingerprint is:  ae:a5:34:55:2f:8f:2d:de:b9:91:f4:ce:df:04:03:d8 reltest@htc190.colorado.edu  The key's randomart image is:  +--[ RSA 2048]----+  | |  | o |  | o E |  | . . . |  | S . o o |  | o \* o o |  | o o o = . .|  | . = . o = ..|  | o . +.o.o|  +-----------------+ |

You can now use the **public key** to set up **passwordless** login by adding your public key to the file below on the client:

|  |
| --- |
| ~/.ssh/authorized\_keys |

**ssh** provides a command to do this:

|  |
| --- |
| ssh-copy-id student@htc191 |

Now you should be able to login to htc191 without a password. Since you know the student password on all the machines in the room, you can install your student public key on every machine and then you have passwordless login to all those machines.

|  |
| --- |
| ssh-copy-id student@htc180  ssh-copy-id student@htc181  ssh-copy-id student@htc182  ssh-copy-id student@htc183  ..  ssh-copy-id student@htc200 |

Now you can login to any machine from your server without a password.

|  |
| --- |
| **NOTE**: You **can not login** to **any** **machine** in the room from any machine in the room **without a password**. |

|  |
| --- |
| **Question: Which machine can login from without and which require a password? Can you explain what is going?** |

|  |
| --- |
| **WARNING**: **Passwordless** login is a potential security issue. Can you explain why?  **Most computing systems disable remote**  **access using passwordless ssh key login.** |

Other keys and certificates:

There is another set of **keys** that we have talked about: the **host ssh keys.** The **host ssh keys** are store in:

|  |
| --- |
| **/etc/ssh:**  -rw-r----- 1 root ssh\_keys 227 Mar 28 2018 ssh\_host\_ecdsa\_key  -rw-r--r-- 1 root root 162 Mar 28 2018 ssh\_host\_ecdsa\_key.pub  -rw-r----- 1 root ssh\_keys 387 Mar 28 2018 ssh\_host\_ed25519\_key  -rw-r--r-- 1 root root 82 Mar 28 2018 ssh\_host\_ed25519\_key.pub  -rw-r----- 1 root ssh\_keys 1679 Mar 28 2018 ssh\_host\_rsa\_key  -rw-r--r-- 1 root root 382 Mar 28 2018 ssh\_host\_rsa\_key.pub |

**Notice the permissions on these files.** These are the files that change between installations.

|  |
| --- |
| **Question: Can you come up with a way to put these in Puppet? If you install them after the installation (overwriting the automatically generated keys) you will not get the “man in middle warning” any more.**  **Try to construct a Puppet Module to do this.** |

The Puppet certificates:

We talked in class about **Puppet Client certificates** and that if you **re-install** the operating system on a **Puppet Client** you will lose the **Puppet certificates**. These certificates are stored in:

|  |
| --- |
| /etc/puppetlabs/puppet/ssl |

Now there is a bit of problem. You **can not** put these in **Puppet** because these are needed to access Puppet, but you would like these to be installed **automatically** after an operating system **re-installation**.

|  |
| --- |
| **Real World Scenario:** You are running a cluster with many compute nodes and one of the system drives crashes on one of the compute nodes. You want to be able to replace that hard drive, turn the machine back on and it automatically returns to its previous configuration. You do not want to manually intervene during the re-installation and configuration process. You have enough to do taking care of the other **999** **compute nodes.** |

|  |
| --- |
| Project:   * Create a Puppet module to install the host ssh keys. * Create a means to save the Puppet Client certificates. * Create a means to reinstall the operating system such that:   + Puppet Client is automatically configured at installation time.   + The Puppet Client certificates do not change after a reinstallation. * **Goal:** The Puppet Client should be able to go through a reinstallation of the operating system and be identically configured to how it was prior to the reinstallation. This should all be done without ever logging into the Puppet Client. * **Hint:** There are some files that are already installed at installation time.   How is that done? * **Further hint:** Puppet Servers run a web server.   You may use any source you like to try to address this problem. |