SSH-Agent

1 Overview

This lab illustrates the use of an SSH agent to manage private keys used to authenticate to SSH services on Linux computers. The goal is to allow a user to use SSH to securely authenticate from the client to a local server, and then from the local server to a remote server, without providing either a password or a passphrase, (after initial setup and initialization of an SSH Agent).

1.1 Background

A previous lab, *sshlab*, illustrated the use of public and private keys for SSH authentication. This lab will show how to protect those private keys with a passphrase, but without having to type in the passphrase each time you wish to access a remote system.

The student is expected to have some familiarity with the Linux command line, the basics of the file system, and the ability to locate and edit a file.

2 Lab Environment

This lab runs in the Labtainer framework, available at http://my.nps.edu/web/c3o/labtainers. That site includes links to a pre-built virtual machine that has Labtainers installed, however Labtainers can be run on any Linux host that supports Docker containers.

From your labtainer-student directory start the lab using:

```
labtainer ssh-agent
```

A link to this lab manual will be displayed.

3 Network Configuration

This lab includes a client computer, a local server and a remote server as shown in Figure **??**. When the lab starts, you will get one virtual terminal connected to the client.

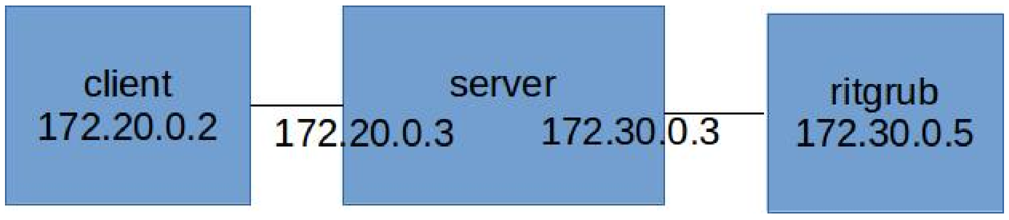

Figure 1: Network topology for the SSH-Agent lab

4 Lab Tasks

4.1 Explore

Use ssh to authenticate to the local server, and then from there to the remote server named `ritgrub`. The user ID and password are "ubuntu". Then use exit, twice, to return to the client's shell.

4.2 Generate and copy keys

Use the ssh-keygen command to generate a public and private key pair on the client computer.

```
ssh-keygen -t rsa
```

When prompted for a passphrase, provide one and make note if it. Recall that in the `sshlab`, you left the passphrase blank, which is convenient but not very secure.

Copy the public key to the server's `authorized_keys` file. This directs the server's SSH service to accept proof of possession of the corresponding private key as a basis for authenticating the user:

```
ssh-copy-id -i ~/.ssh/id_rsa.pub server
```

You should now be able to SSH from the client to the server without providing the password, though you will be prompted for the passphrase that you provided to protect the private key.

```
ssh server
```

While still in a session on the server, you will now copy your `authorized_keys` file to the `ritgrub` server.

```
scp -r ~/.ssh ritgrub:~/
```

This directs the `ritgrub` server's SSH service to accept proof of possession of the corresponding private key as a basis for authenticating the user. Note however that the private key will never exist on the local server.

4.3 Start an SSH Agent to manage your private key

Use `exit` to return to the client. Then use these commands to start an SSH agent on the client, and provide that agent with access to your new private key:

```
eval `ssh-agent`
ssh-add
```

Note the argument to the `eval` command is enclosed in back-tics (upper left on most keyboards.) When prompted, provide the passphrase used to protect your private key. Then SSH to the local server again. You should be able to do so without providing a passphrase.

```
ssh server
```

Accessing the `ritgrub` server from the local server still requires a password, because the local server is unable to prove that the requesting user possesses the private key. We can rectify this by exiting from the local server back to the client, and restarting the SSH session, but with the `-A` option:

```
ssh -A server
```

This option enables forwarding of the SSH agent connection. When you SSH to a remote host that recognizes the public key in its

```
~/.ssh/authorized\_keys
```

file, the SSH agent on the client will be used to prove possession of the private key. You should now be able to SSH from the local server to the gitrub server without a password. Try it, and then exit back to the client.

4.4 Use an SSH config file to enable agent forwarding

The SSH program references a file named

```
~/.ssh/config
```

(if it exists), to get defaults and configuration settings for each SSH session. This file can be configured so that you need not remember to add the `-A` option to the ssh command. Each entry in this file begins with a `Host` line that identifies the hostname you will use in the SSH command, and that is typically followed by a `HostName` line that includes the DNS name or IP address of that host. The entry then includes configuration information, in our case, an indication that we want to enable agent forwarding. Such an entry might look like:

```
Host server
   HostName server
   ForwardAgent yes
```

Create a `config` file and demonstrate your ability to SSH from the client to the server, and then on to the ritgrub server without passphrases, passwords, or use of an explicit `-A` option.

5 Submission

After finishing the lab, go to the terminal on your Linux system that was used to start the lab and type:

```
    stoplab
```

When you stop the lab, the system will display a path to the zipped lab results on your Linux system. Provide that file to your instructor, e.g., via the Sakai site.