SSH

March 7, 2019

Colorado School of Mines Linux Users Group
Getting Started
What is SSH?

- **Secure SHell**.
- A cryptographic network protocol for operating network services securely over an unsecured network.
- Uses public-key cryptography for authentication.
- SSH clients allow you to access any SSH server remotely and securely.
- Common uses are logging into remote servers and accessing GitHub/GitLab repos.
What is SSH?

- Secure SHEll.
- A cryptographic network protocol for operating network services securely over an unsecured network.
- Uses public-key cryptography for authentication.
- SSH clients allow you to access any SSH server remotely and securely.
- Common uses are logging into remote servers and accessing GitHub/GitLab repos.
What is SSH?

- Secure SHell.
- A cryptographic network protocol for operating network services securely over an unsecured network.
  - Uses public-key cryptography for authentication.
  - SSH clients allow you to access any SSH server remotely and securely.
  - Common uses are logging into remote servers and accessing GitHub/GitLab repos.
What is SSH?

- **Secure SHell.**
- A cryptographic network protocol for operating network services securely over an unsecured network.
- Uses public-key cryptography for authentication.
  - SSH clients allow you to access any SSH server remotely and securely.
  - Common uses are logging into remote servers and accessing GitHub/GitLab repos.
What is SSH?

- **Secure SHell**.
- A cryptographic network protocol for operating network services securely over an unsecured network.
- Uses public-key cryptography for authentication.
- SSH clients allow you to access any SSH server remotely and securely.
  - Common uses are logging into remote servers and accessing GitHub/GitLab repos.
What is SSH?

- **Secure SHell.**
- A cryptographic network protocol for operating network services securely over an unsecured network.
- Uses public-key cryptography for authentication.
- SSH clients allow you to access any SSH server remotely and securely.
- Common uses are logging into remote servers and accessing GitHub/GitLab repos.
Using a SSH client
How do I get a SSH client?

- **Linux**: `openssh` (or similar) package in your package manager. It’s probably already already installed.
- **macOS**: SSH is already installed, but it may be an old version. Use Homebrew (a package manager for macOS) if you want the latest version.
- **Windows**: You can use PuTTY (https://www.putty.org/) or you can use PowerShell on Windows 10.
- **Your web browser**: there’s an SSH plugin for all the modern browsers.
- **Your phone**: Android and iOS apps exist.
The basics

- `ssh [user@]server[:port]`
  - user is defaulted to your local username
  - port defaults to 22 (you can specify a different one with `-p`)
- Enable X-Forwarding: use `-X` flag
- Exiting an SSH session: Ctrl + D or type `logout` or `exit` if your remote session is still running
- If you want to just run one command on the remote server: `ssh [flags] user@server[:port] command`
The basics

- `ssh [user@]server[:port]`
  - `user` is defaulted to your local username
  - `port` defaults to 22 (you can specify a different one with `-p`)
- Enable X-Forwarding: use `-X` flag
- Exiting an SSH session: Ctrl + D or type `logout` or `exit` if your remote session is still running
- If you want to just run one command on the remote server: `ssh [flags] user@server[:port] command`
The basics

- `ssh [user@]server[:port]`
  - `user` is defaulted to your local username
  - `port` defaults to 22 (you can specify a different one with `-p`)
- Enable X-Forwarding: use `-X` flag
- Exiting an SSH session: Ctrl + D or type `logout` or `exit` if your remote session is still running
- If you want to just run one command on the remote server: `ssh [flags] user@server[:port] command`
The basics

- ssh [user@]server[:port]
  user is defaulted to your local username
  port defaults to 22 (you can specify a different one with `-p`)
- Enable X-Forwarding: use `-X` flag
- Exiting an SSH session: Ctrl + D or type `logout` or `exit` if your remote session is still running
- If you want to just run one command on the remote server:
  ssh [flags] user@server[:port] command
SSH Keys
When logging into a server, you can authenticate using your password, or you can set up an SSH key to authenticate you without entering your password.
Configuring a SSH key for “normal” SSH

How to:

1. `ssh-keygen`\(^1\) and follow the steps. Definitely set a password!
2. `ssh-copy-id servername` and enter your password on the server.
3. `ssh servername` should now authenticate you without having to use a password.

\(^1\)more in-depth walkthrough: https://gitlab.com/help/ssh/README#generating-a-new-ssh-key-pair
Whenever you do operations involving git@gitlab.com:... it will authenticate with your SSH key if you configure one.

How to:

1. `ssh-keygen`, if you haven’t already done so.
2. Go to “SSH keys” in your GitLab/GitHub account page’s Settings and find the option to add a new key.
3. Copy/paste your public key (likely ~/.ssh/id_rsa.pub) into the box.
4. Give it a title. The hostname of the machine where you did the keygen is a good choice.
If you don’t like entering your SSH key password all the time, you can use `ssh-agent` and `ssh-add`.

The following in one’s `.bashrc` will set this up automatically.

```bash
if [ ! -S ~/.ssh/ssh_auth_sock ]; then
    eval `ssh-agent`
    ln -sf "$SSH_AUTH_SOCK" ~/.ssh/ssh_auth_sock
fi
export SSH_AUTH_SOCK=~/.ssh/ssh_auth_sock
ssh-add -l | grep "The agent has no identities" && ssh-add
```
SSH aliases

You can use aliases so you don’t have to type your full username and hostname every time you SSH.

(And we don’t mean aliasing isengard to jnunez@isengard.mines.edu \(-p \ 42\) in your .bashrc!)

You add them to ~/.ssh/config like so...

```
Host isengard
  HostName isengard.mines.edu
  User jnunez
  Port 42
...
```
SSH aliases

You can use aliases so you don’t have to type your full username and hostname every time you SSH.

(And we don’t mean aliasing isengard to jnunez@isengard.mines.edu -p 42 in your .bashrc!)

You add them to ~/.ssh/config like so...

```
Host isengard
  HostName isengard.mines.edu
  User jnunez
  Port 42
...
```
SSH aliases

You can use aliases so you don’t have to type your full username and hostname every time you SSH.

(And we don’t mean aliasing isengard to jnunez@isengard.mines.edu -p 42 in your .bashrc!)

You add them to ~/.ssh/config like so...

```
Host isengard
    HostName isengard.mines.edu
    User jnunez
    Port 42
...
```
Setting up a SSH Server
How do I install a SSH server?

- Arch Linux: `openssh` package.
- Other Linux: you may need to install `openssh-server` or similar.
- macOS: You can enable Remote Login\(^2\) in System Settings.
- Windows: Read this ServerFault article. Good luck. [Link](https://serverfault.com/questions/8411/what-is-a-good-ssh-server-to-use-on-windows)

\(^2\)[Link](https://www.techwalla.com/articles/how-to-use-ssh-on-mac-os-x)
Enabling SSH to your computer

On Arch, just start an enable sshd via systemctl.

You can configure your SSH daemon via the /etc/ssh/sshd_config file (note the d).

Here are some of the things you can configure:

- **AllowUsers** - allows you to set which users can log in
- **PermitRootLogin** - if yes, you can SSH into the computer as root. Not a great idea.
- **AllowGroups** - allows you to set which groups can log in
- **PasswordAuthentication** - set to no if you want to force authentication using SSH key
- **X11Forwarding** - yes enables this. xauth must be installed.
What can I do with SSH?
What can I do with SSH?

- SSH from your laptop into a powerful server is really nice for running/compiling code.
- At Mines, SSH is how you access Isengard and the supercomputers.
- Jumpbox is a server who allows you to SSH servers on the Mines network from off-campus, without a VPN. `ssh yourmultipass@jumpbox.mines.edu` and then from Jumpbox, you can SSH Isengard, etc.
- `scp` allows you to `cp` between computers like so:
  - From A to B as A: `scp path/to/file username@b:path/to/destination`
  - From A to B as B: `scp username@a:path/to/file path/to/destination`
  (Good luck trying this on Windows, just use WinSCP instead.)
What can I do with SSH?

- SSH from your laptop into a powerful server is really nice for running/compiling code.
- At Mines, SSH is how you access Isengard and the supercomputers
  - Jumpbox is a server who allows you to SSH servers on the Mines network from off-campus, without a VPN. `ssh yourmultipass@jumpbox.mines.edu` and then from Jumpbox, you can SSH Isengard, etc.
  - scp allows you to cp between computers like so:
    - From A to B as A: `scp path/to/file username@b:path/to/destination`
    - From A to B as B: `scp username@a:path/to/file path/to/destination`
    (Good luck trying this on Windows, just use WinSCP instead.)
What can I do with SSH?

- SSH from your laptop into a powerful server is really nice for running/compiling code.
- At Mines, SSH is how you access Isengard and the supercomputers.
- Jumpbox is a server who allows you to SSH servers on the Mines network from off-campus, without a VPN. ssh yourmultipass@jumpbox.mines.edu and then from Jumpbox, you can SSH Isengard, etc.
- `scp` allows you to `cp` between computers like so:
  - From A to B as A: `scp path/to/file username@b:path/to/destination`
  - From A to B as B: `scp username@a:path/to/file path/to/destination`
  (Good luck trying this on Windows, just use WinSCP instead.)
What can I do with SSH?

- SSH from your laptop into a powerful server is really nice for running/compiling code.
- At Mines, SSH is how you access Isengard and the supercomputers.
- Jumpbox is a server who allows you to SSH servers on the Mines network from off-campus, without a VPN. Use `ssh yourmultipass@jumpbox.mines.edu` and then from Jumpbox, you can SSH Isengard, etc.
- `scp` allows you to cp between computers like so:
  - From A to B as A: 
    
    ```bash
    scp path/to/file username@b:path/to/destination
    ```
  - From A to B as B:
    
    ```bash
    scp username@a:path/to/file path/to/destination
    ```
  
  (Good luck trying this on Windows, just use WinSCP instead.)
Questions?
References

- The Arch Wiki: https://wiki.archlinux.org/index.php/Secure_Shell
- The SSH manpage
- https://medium.com/@shazow/ssh-how-does-it-even-9e43586e4ffc#.uwmcu64az
- https://lani78.com/2008/08/08/generate-a-ssh-key-and-disable-password-authentication-on-ubuntu-server/

Thanks to Sumner Evans for letting us use his slides, and to Keith Hellman for inspiring the original talk.
This presentation was from the Mines Linux Users Group. A mostly-complete archive of our presentations can be found online at https://lug.mines.edu.

Individual authors may have certain copyright or licensing restrictions on their presentations. Please be certain to contact the original author to obtain permission to reuse or distribute these slides.