Public Key Setup & Management

- SSH Key Authentication Create and Share
- Managing Multiple SSH Keys
- SSH Agent for Passphrase Management
- Disable Password Login

SSH Key Authentication Create and Share

Why SSH Keys are Important? SSH keys use asymmetric encryption—a public key is placed on the server, and a private key stays on your device.

Benefits over Passwords

- Security: Keys are far more resistant to brute-force attacks than passwords
- Convenience: No need to type passwords every time
- Automation: Ideal for scripts and remote tasks
- Granular Access: You can assign different keys to different users or devices

Pre requisite- Needs OpenSSH if not already installed.

Generating SSH Keys

1. On your local computer that will be used to establish connection with a server
2. In terminal type following command

```
ssh-keygen -t rsa -b 4096 -C "key name"
```

"-t rsa" - specifies the type of key RSA

"b 4096" - Sets the key length to 4096 bits

"-C" - Adds a comment, optional

or use ed25519 for stronger encryption

```
ssh-keygen -t rsa -t ed25519 -C "key name"
```

3. Copy Public Key to Server

```
ssh-copy-id username@server_ip
```

Or if you make multiple different keys

```
ssh-copy-id -i ~/.ssh/id_rsa_work.pub 'name of the key you want to copy' username@server_ip
```

To See the public key

```
cat ~/.ssh/id_rsa.pub
```

More Info on managing multiple keys check Managing Multiple SSH Keys

When using Passphrase, it would be a good idea to have SSH Agent to manege them. More info at SSH Agent Management

Managing Multiple SSH Keys

If you use different keys for different servers or services (e.g., GitHub, work, personal), here’s how to keep it organized. Using different keys for different services makes things way more organized and secure. This way if one key gets compromised, you only need to change it for that service or server.

Creating Multiple Keys

1. Login to machine you want to have access to server or services
2. Open Terminal and Create Keys

```
ssh-keygen -t rsa -b 4096 -f ~/.ssh/id_rsa_work
ssh-keygen -t rsa -b 4096 -f ~/.ssh/id_rsa_personal   'adding path will make the key unique
and wont override the previous one'
```

3. Send each key to it's designated server

```
ssh-copy-id -i ~/.ssh/id_rsa_work.pub username@server_ip
```

Using SSH Config File

For easier access adjust config file. This will tell which key goes to which session

```
# Work server
Host work-server
    HostName work.example.com
    User yourusername
    IdentityFile ~/.ssh/id_rsa_work
    IdentitiesOnly yes

# Personal server
Host personal-server
    HostName personal.example.com
    User yourusername
    IdentityFile ~/.ssh/id_rsa_personal
```

```
    IdentitiesOnly yes
```

Using SSH Agent is a good idea if using passphrase for each one, check out SSH Agent Management

SSH Agent for Passphrase Management

SSH Agent is a background process that keeps your keys unlocked during your session.

1. Start the agent and add keys:

```
eval "$(ssh-agent -s)"
ssh-add ~/.ssh/id_rsa_work
ssh-add ~/.ssh/id_rsa_personal
```

Per each key added to ssh you will need to type a phrase

Disable Password Login

Once we have ssh key setup we can disable password login on servers once keys are set up. This way our server and services are more secure.

1. SSH to server
2. Edit sshd_config file

```
sudo nano /etc/ssh/sshd_config
```

1. Set Authentication to No

```
PasswordAuthentication no
```

1. Restart SSH

```
sudo systemctl restart ssh
```