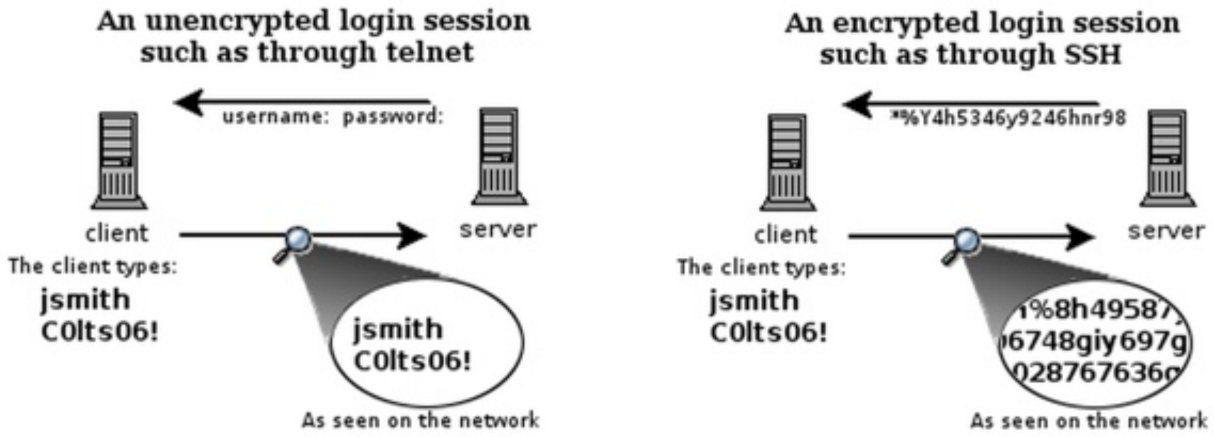
Week 6: SSH

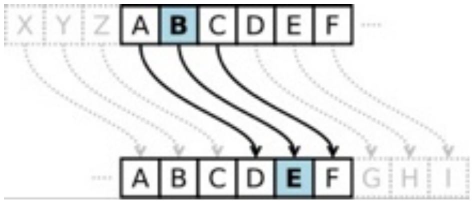
What is SSH?

- Secure Shell
- used to remotely access shell
- successor of telnet
- encrypted and better authenticated session



Encryption Types

Symmetric Key Encryption

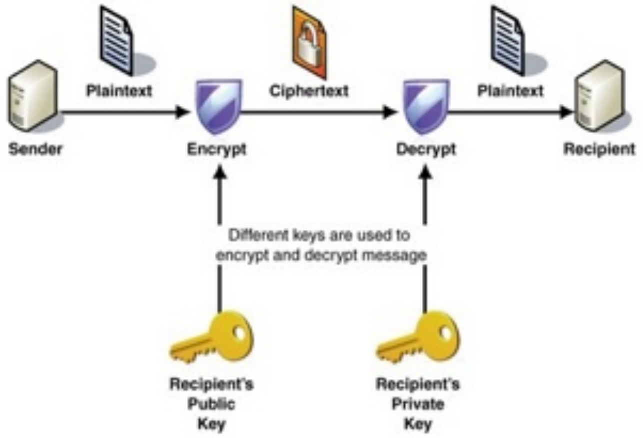
- shared/secret key
- key used to encrypt is the same key used to decrypt
- example: Data Encryption Standard (DES)
- Caesar's Cipher:
    - map the alphabet to a shifted version
    - 

- key distribution is a problem
    - secret key has to be delivered in a safe way to the recipient
    - chance of key being compromised

Asymmetric Key Encryption: Public/Private

- 2 different (but related) keys: public and private
    - only creator knows the relation

    - private key cannot be derived from public key
- data encrypted with public key can only be decrypted by private key and vice versa
- public key can be seen by anyone
    - anyone can encrypt message, but cannot decrypt the ciphertext
- **never** publish private key
- example: RSA - Rivest, Shamir, & Adleman
    - property used: difficulty of factoring large integers to prime umbers
    - N = p * q | 3233 = 61 * 53
    - N is a large integer and p, q are prime numbers
    - N is part of the public key



High-Level SSH Protocol

- client ssh's to remove server
    - `$ ssh username@somehost`
- on first time talking to server, requires host validation

```
The authenticity of host 'somehost (192.168.1.1)' can't be established. RSA key fingerprint is 90:9c:46:ab:03:1d:30:2c:5c:87:c5:c7:d9:13:5d:75. Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added 'somehost' (RSA) to the list of known hosts.
```

- ssh doesn't know about this host yet
- shows hostname, IP address, and fingerprint of the server's public key, so you can be sure you're talking to the correct computer
- after accepting, public key is saved in `~/.ssh/known_hosts`

Host Validation

- next time client connects to server
    - check hosts public key against saved public key to see if the host is the actual host that is trying to be reached
- client asks server to prove that it is the owner of the public key using **asymmetric encryption**

    - encrypt a message with a public key
    - if server is true owner, it can decrypt the message with private key
- if everything works, host is successfully validated

Session Encryption

- client and server agree on a **symmetric encryption key** (session key)
- all messages sent between client and server are
    - encrypted at the sender with session key
    - decrypted at the receiver with session key
- anybody who doesn't know the session key (hopefully, no one but client and server) doesn't know any of the contents of those messages

Client Authentication

- **password-based authentication**
    - prompt for passwords on remote server
    - if username specified exists and remote password for it is correct, system lets you in
- **key-based authentication**
    - generate a key pair on the client
    - copy public key to the server (~/.ssh/authorized_keys)
    - server authenticates client if it can demonstrate that it has the private key
    - private key can be protected with a passphrase
    - every time you ssh to a host, you will be asked for the passphrase (inconvenient!)

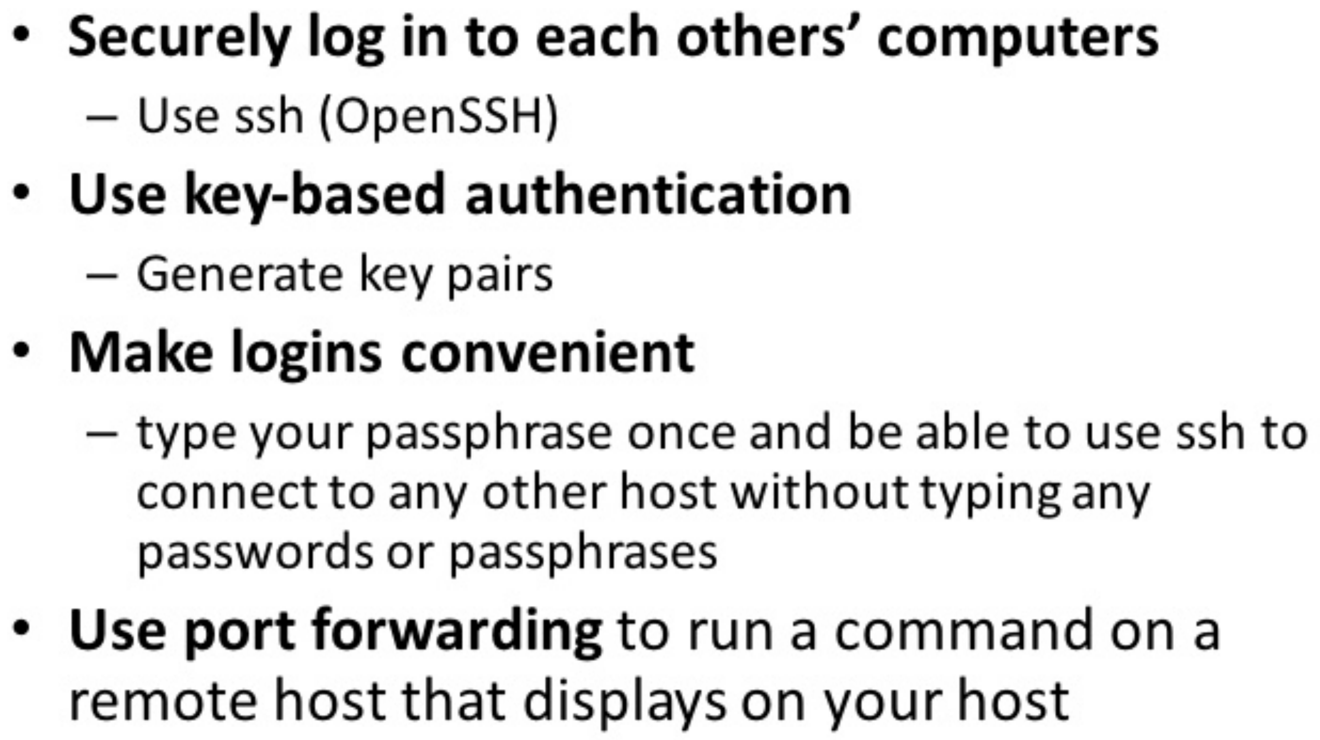
ssh-agent (paraphrase-less ssh)

- a program used with OpenSSH that provides a secure way of storing the private key
- `ssh-add` prompts user for the passphrase once and adds it to the list maintained by `ssh-agent`
- once paraphrase is added to `ssh-agent`, the user will not be prompted for it again when using SSH
- OpenSSH will talk to the local ssh-agent daemon and retrieve the private key from it automatically

X Window System

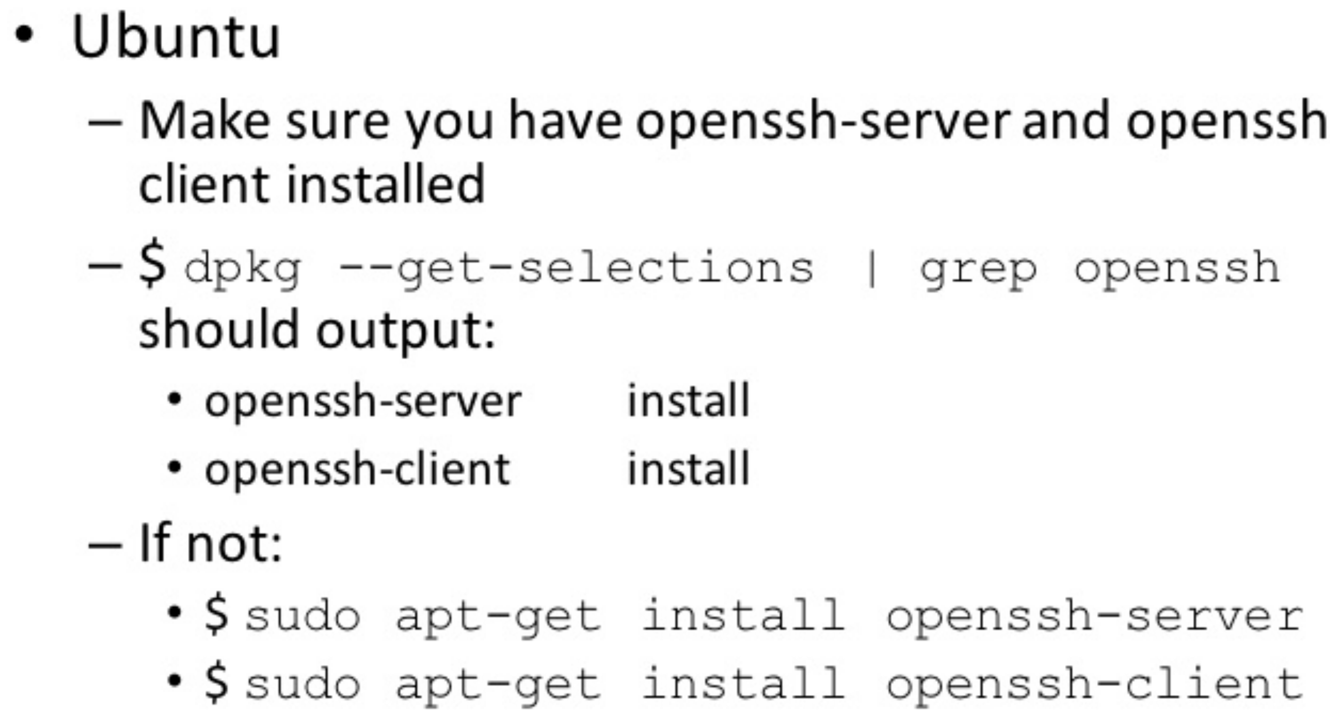
- windowing system that forms the basis for most GUIs on UNIX
- X is a network-based system
    - based upon a network protocol such that a program can run on one computer but be displayed on another (X Session Forwarding)

Lab 6



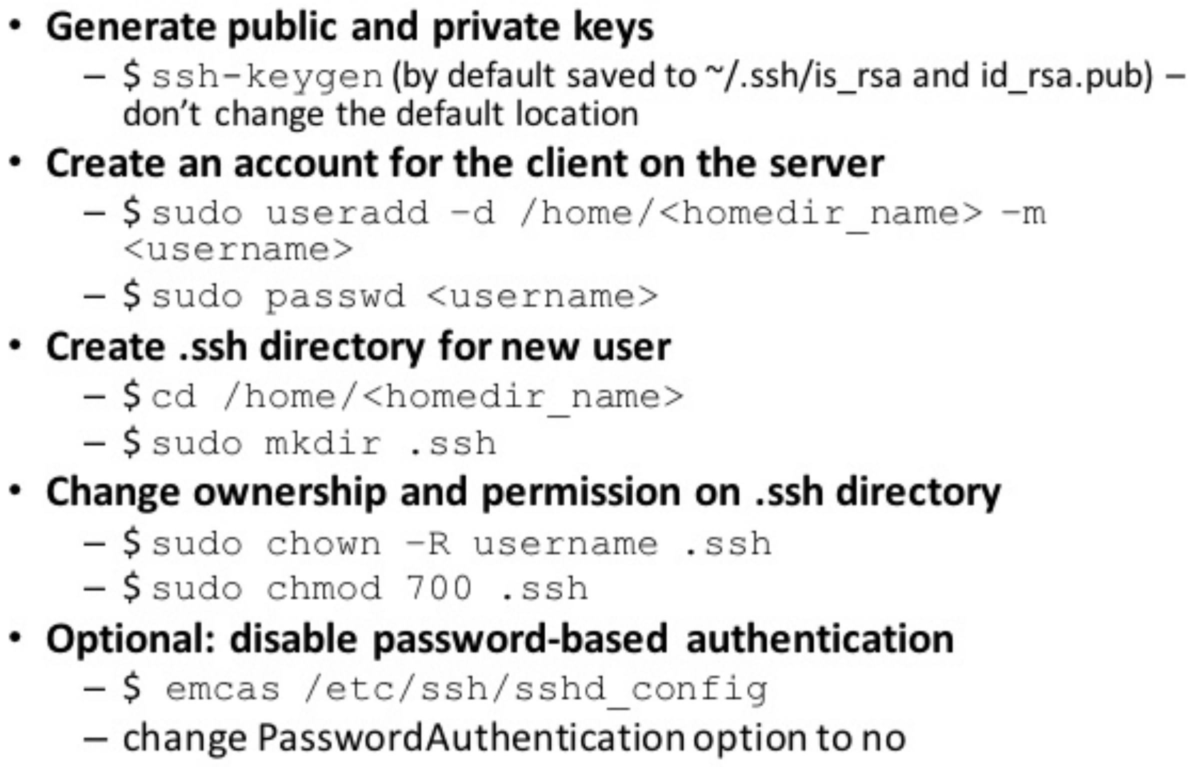
- **Securely log in to each others' computers**
  - Use ssh (OpenSSH)
- **Use key-based authentication**
  - Generate key pairs
- **Make logins convenient**
  - type your passphrase once and be able to use ssh to connect to any other host without typing any passwords or passphrases
- **Use port forwarding** to run a command on a remote host that displays on your host

Environment Setup



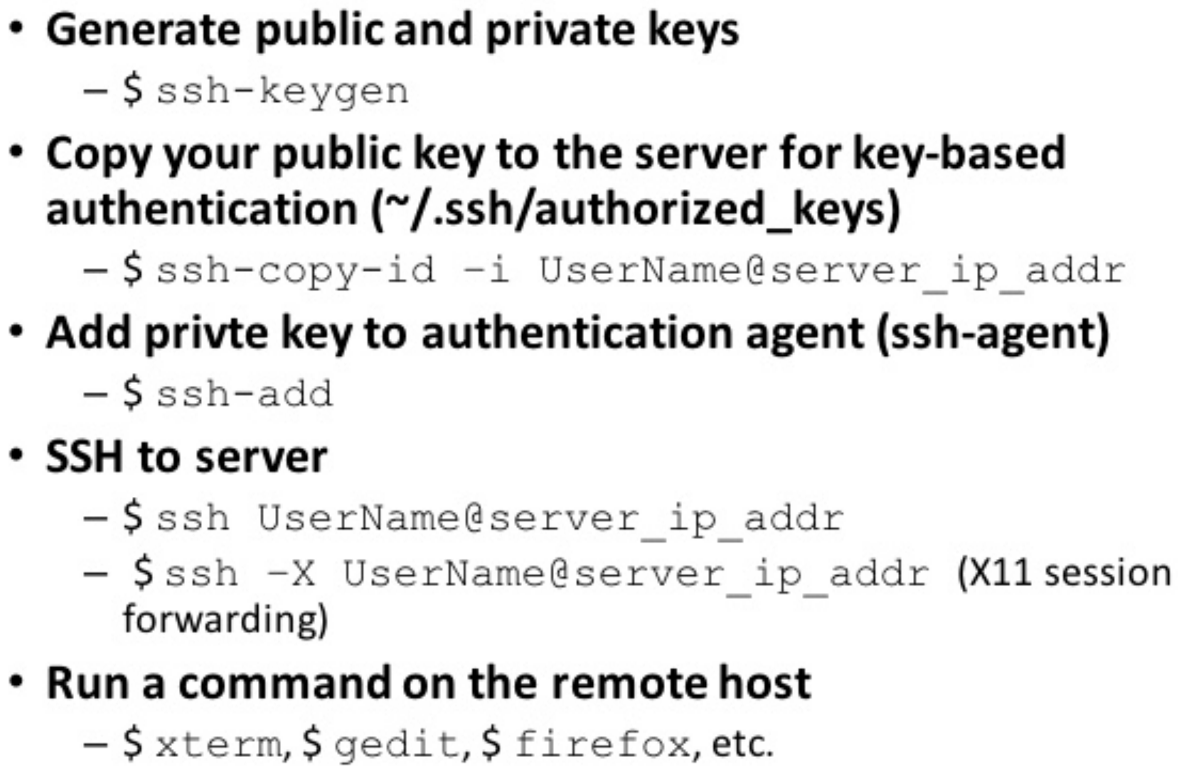
- Ubuntu
  - Make sure you have openssh-server and openssh-client installed
  - $ `dpkg --get-selections | grep openssh` should output:
    - openssh-server install
    - openssh-client install
  - If not:
    - $ `sudo apt-get install openssh-server`
    - $ `sudo apt-get install openssh-client`

Server Steps



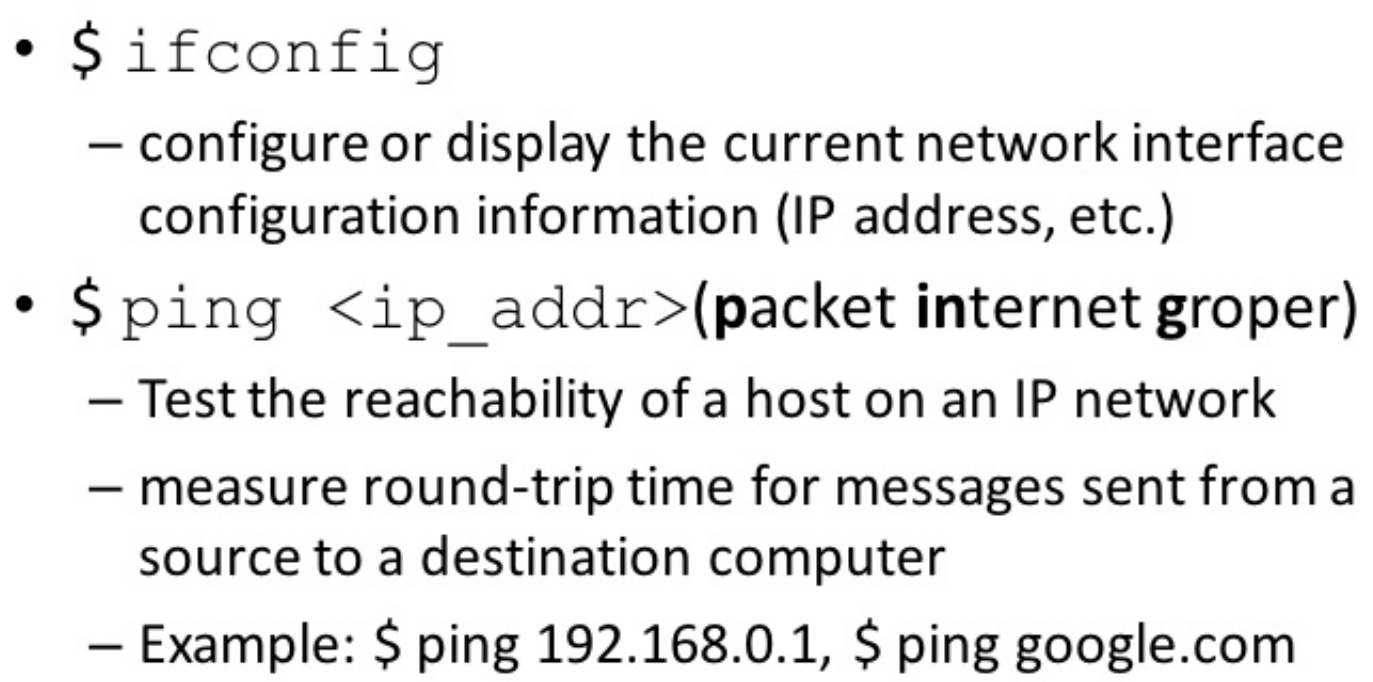
- **Generate public and private keys**
  - $ `ssh-keygen` (by default saved to ~/.ssh/is_rsa and id_rsa.pub) – don't change the default location
- **Create an account for the client on the server**
  - $ `sudo useradd -d /home/<homedir_name> -m <username>`
  - $ `sudo passwd <username>`
- **Create .ssh directory for new user**
  - $ `cd /home/<homedir_name>`
  - $ `sudo mkdir .ssh`
- **Change ownership and permission on .ssh directory**
  - $ `sudo chown -R username .ssh`
  - $ `sudo chmod 700 .ssh`
- **Optional: disable password-based authentication**
  - $ `emcas /etc/ssh/sshd_config`
  - change PasswordAuthentication option to no

Client Steps



- **Generate public and private keys**
  - $ `ssh-keygen`
- **Copy your public key to the server for key-based authentication (~/.ssh/authorized_keys)**
  - $ `ssh-copy-id -i UserName@server_ip_addr`
- **Add privte key to authentication agent (ssh-agent)**
  - $ `ssh-add`
- **SSH to server**
  - $ `ssh UserName@server_ip_addr`
  - $ `ssh -X UserName@server_ip_addr` (X11 session forwarding)
- **Run a command on the remote host**
  - $ `xterm`, $ `gedit`, $ `firefox`, etc.

Checking IP Address



- $ `ifconfig`
  - configure or display the current network interface configuration information (IP address, etc.)
- $ `ping <ip_addr>`(**p**acket **in**ternet **g**roper)
  - Test the reachability of a host on an IP network
  - measure round-trip time for messages sent from a source to a destination computer
  - Example: $ ping 192.168.0.1, $ ping google.com