

CS 35L

Week 6

TA: Tomer Weiss
Feb-09-2016

goo.gl/hTokZ1

Slides

Announcements

- Student presentations today:
 - Google DeepMind's AlphaGo Program defeats European Champion in Go
 - Robots Learn by Watching “How To” Videos

web.cs.ucla.edu/classes/winter16/cs35L/assign/assign10.html

- Next week:
 - Write your topic [here](#)
 - Not registering your topic beforehand may result in rescheduling of your presentation
 - For reference on presentation, grading, please refer to this [rubric](#).

SSH - Secure Shell

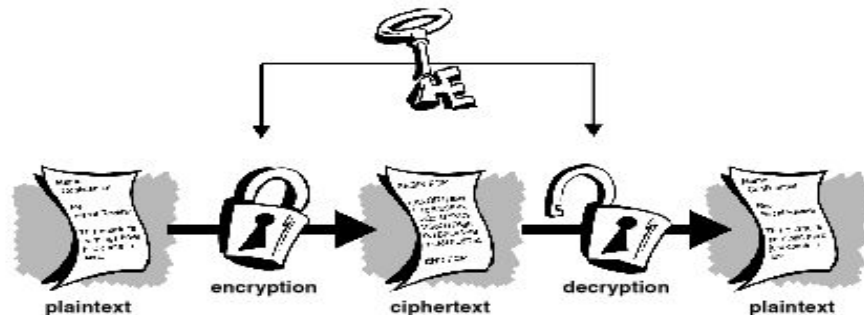
Week 6

Communication Over the Internet

- What type of guarantees do we want?
 - **Confidentiality**
 - Message secrecy
 - **Data integrity**
 - Message consistency
 - **Authentication**
 - Identity confirmation
 - **Authorization**
 - Specifying access rights to resources

Cryptography

- Plaintext - actual message
- Ciphertext - encrypted message (unreadable gibberish)
- Encryption - converting from plaintext to ciphertext
- Decryption - converting from ciphertext to plaintext
- Secret key
 - part of the mathematical function used to encrypt\decrypt
 - Good key makes it hard to get back plaintext from ciphertext



Symmetric-key Encryption

- Same secret key used for encryption and decryption
- **Example** : Data Encryption Standard (DES)
- **Caesar's cipher**
 - Map the alphabet to a shifted version
 - ABCDEFGHIJKLMNOPQRSTUVWXYZ
 - DEF_GHIJKLMNOPQRSTUVWXYZABC
 - Plaintext – SECRET. Ciphertext – VHFUHW
 - Key is 3 (number of shifts of the alphabet)
- **Key distribution** is a problem
 - The secret key has to be delivered in a safe way to the recipient
 - Chance of key being compromised

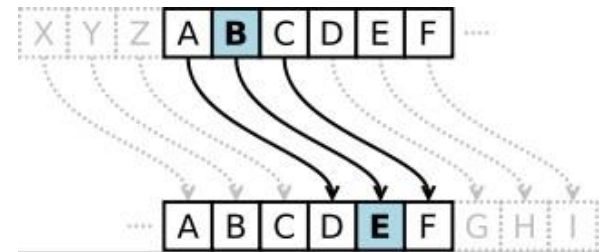


Image Source: wikipedia

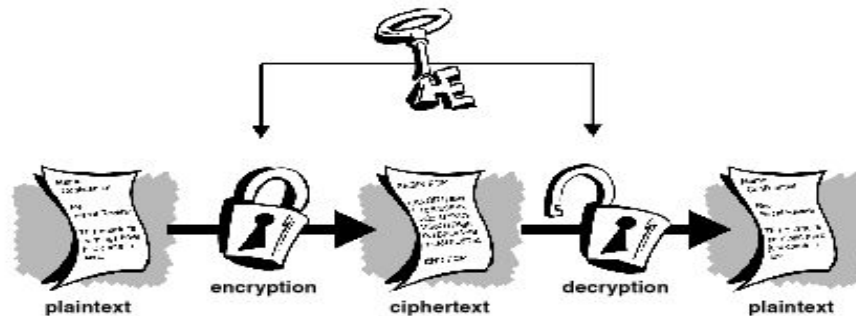


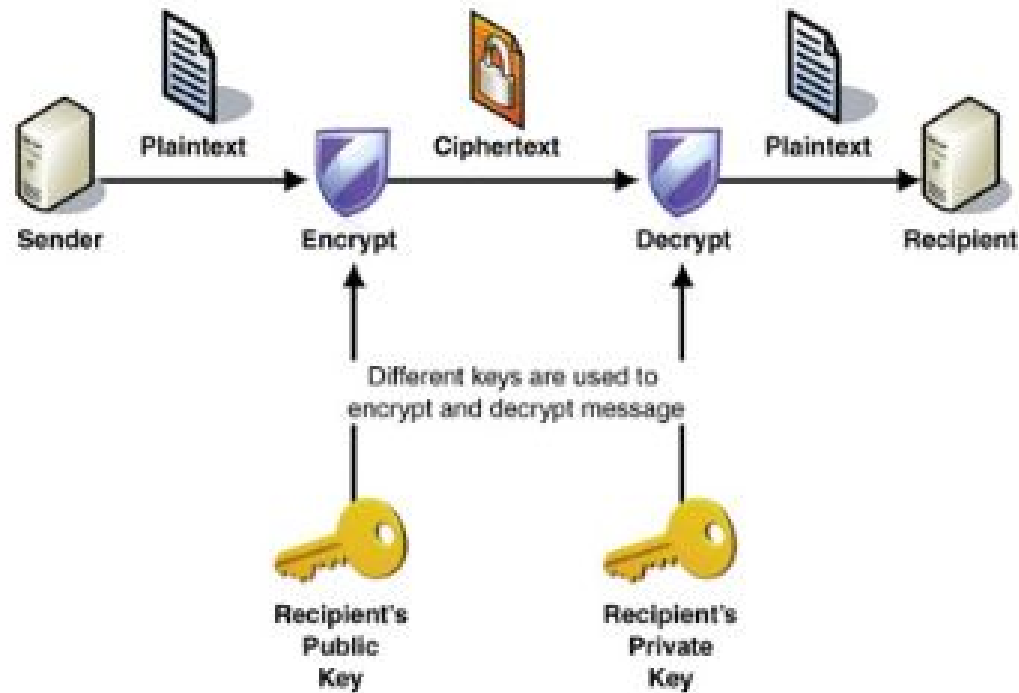
Image Source: gpgtools.org

Public-Key Encryption (Asymmetric)

- Uses a pair of keys for encryption
 - Public Key - published and well known to everyone
 - Private - Secret key known only to the owner
- Encryption
 - Use public key to encrypt messages
 - Anyone can encrypt message, but they cannot decrypt the ciphertext
- Decryption
 - Use private key to decrypt messages
- In what scheme is this encryption useful?

Public-Key Encryption (Asymmetric)

- Example: RSA (Rivest, Shamir & Adelman)
 - Property used: Difficulty of factoring large integers to prime numbers
 - $N = p * q$
 - M is a large integer.
 - p, q are prime numbers
 - N is part of the public key



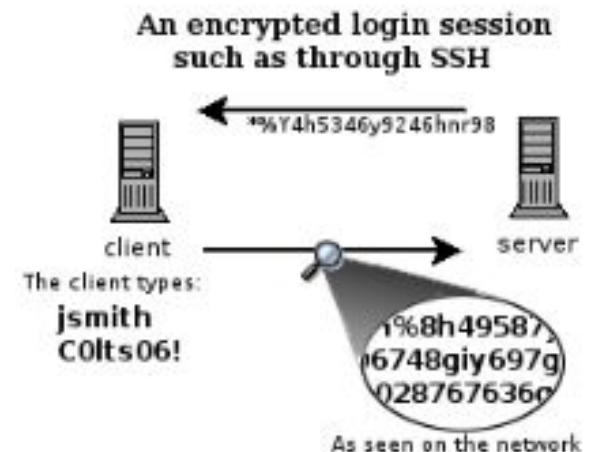
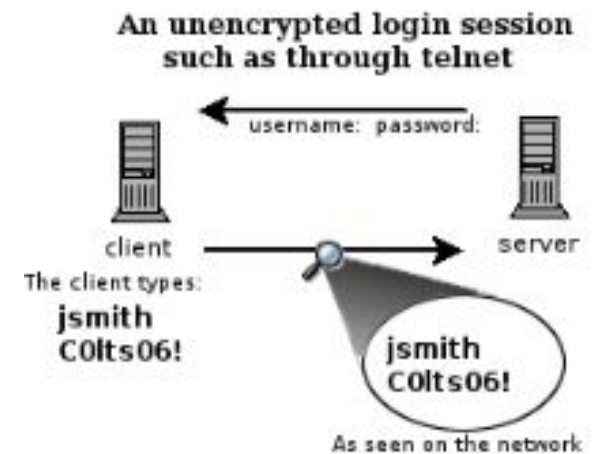
en.wikipedia.org/wiki/RSA_Factoring_Challenge

Encryption Types Comparison

- **Symmetric Key Encryption**
 - a.k.a shared/secret key
 - Key used to encrypt is the same as key used to decrypt
- **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
 - **Never** publish private key!!!

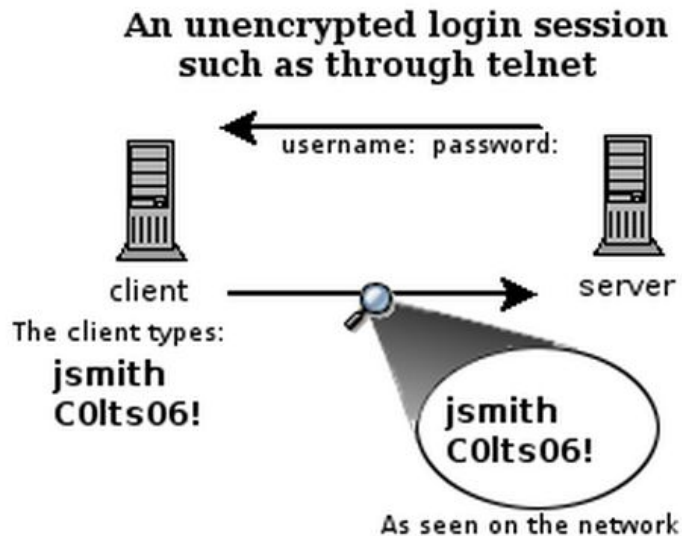
Secure Shell (SSH)

- Telnet
 - Remote access
 - Not encrypted
 - Packet sniffers can intercept sensitive information (username/password)
- SSH
 - run processes remotely
 - encrypted session
 - Session key (secret key) used for encryption during the session



What is SSH?

- **Secure Shell**
- Used to remotely access shell
- Successor of telnet
- Encrypted and better authenticated session



CONFIDENTIAL

Session Encryption

- Client and server agree on a **symmetric encryption key** (session key)
- All messages sent between client and server
 - 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

High-Level SSH Protocol

- Client ssh's to remote server
 - `$ ssh username@somehost`
 - If first time talking to server -> 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`

Secure Shell (SSH) - Client Authentication

- **Password** login
 - `ssh username@ugrad.seas.ucla.edu`
- **Passwordless** login with keys
 - Use private/public keys for authentication (server and client authentication)
 - `ssh-keygen`
 - Passphrase (longer version of a password/more secure)
 - Passphrase for protecting the private key
 - Passphrase needed whenever the keys are accessed
 - `ssh-copy-id username@ugrad.seas.ucla.edu`
 - Copies the public key to the server (`~/.ssh/authorized_keys`)
 - Login without password
 - `ssh username@ugrad.seas.ucla.edu`
 - Run scripts/commands on the remote machine
 - `ssh username@ugrad.seas.ucla.edu ls`
 - But you need to provide a passphrase to use a private key

Secure Shell (SSH) - Client Authentication

- **Passphrase-less** authentication
 - `ssh-agent` → authentication agent
 - Manages private key identities for SSH
 - To avoid entering the passphrase whenever the key is used
 - `ssh-add`
 - Registers the private key with the agent
 - Passphrase asked only once
 - `ssh` will ask the `ssh-agent` whenever the private keys are needed

Secure Shell (SSH) - Client Authentication

- **Session Encryption**
 - Symmetric encryption
 - Exchange secret key (Example - Diffie-Hellman)
- **Host/Client Validation**
 - Public-key Encryption
 - Challenge-Response
 - Host sends a “challenge” that has to be answered by the client
 - Similarly, client sends a “challenge” that has to be answered by the host

Account Administration

- Install OpenSSH (should be done on both server and client)
 - `sudo apt-get update`
 - `sudo apt-get install openssh-server`
 - `sudo apt-get install openssh-client`
- Server
 - `sudo useradd -d /home/<username> -m <UserName>`
 - `sudo passwd <username>`
 - `cd /home/<username>`
 - `sudo mkdir .ssh`
 - `sudo chown -R <username> .ssh`
 - `sudo chmod 700 .ssh`
 - `ifconfig` (this will give you the IP address of the server. Give this to your partner).
 - `ps aux | grep ssh`
 - This will give you the
 - or alternatively `pgrep -lf ssh`

Account Administration

- **Client**

- **Password** login

- `ping server_ip_addr` (just to check if the server responds)
 - `ssh <username>@server_ip_addr`

- **Password-less** login

- `ssh-keygen`
 - `ssh-copy-id -i <username>@server_ip_addr`
 - `ssh <username>@server_ip_addr` (should not ask for login password)

- **Passphrase-less** login

- `ssh-add`
 - `ssh -X <username>@server_ip_addr` (should not ask for key's passphrase)

- **X session forwarding** - running programs with GUI

- `ssh -X <UserName>@server_ip_addr`
 - `xterm`
 - `firefox`

X session forwarding

- **X** is the windowing system for GUI apps on linux
- **X** is a network-based system. It is based upon a network protocol such that a program can run on one computer but be displayed on another
 - i.e. you want to run such apps remotely, but the GUI should show up on the local machine
- Windowing system forms the basis for most GUIs on UNIX
 - `ssh -X username@ugrad.seas.ucla.edu`
 - `gedit`
 - `gimp`

Secure copy (scp)

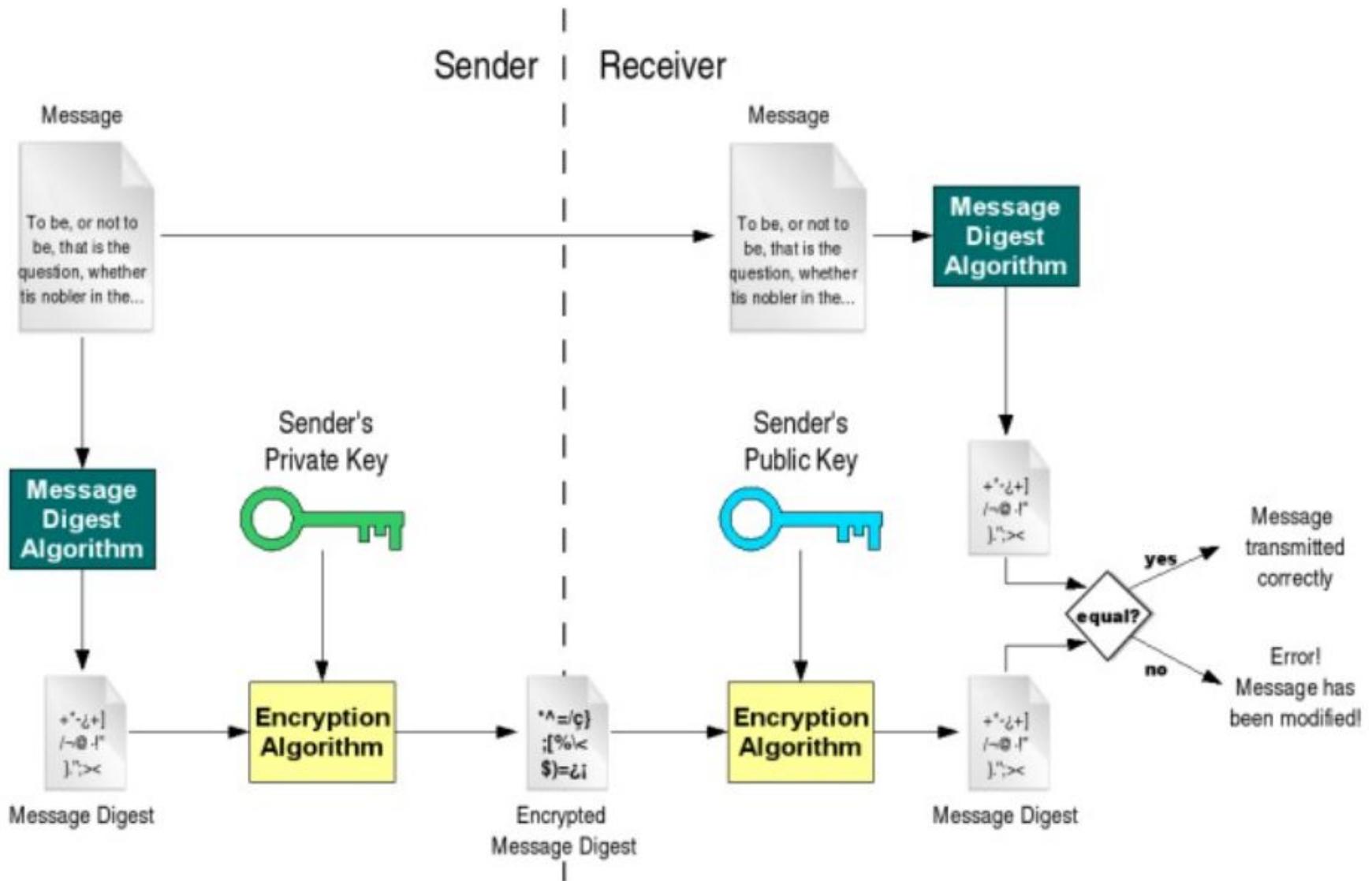
- Based on secure shell (ssh)
- Used for transferring files between hosts in a secure way (encrypted)
- Usage similar to cp
 - `scp [source] [destination]`
- Transferring to remote host
 - `scp /home/username/doc.txt username@ugrad.seas.ucla.edu:/home/user/docs`
 - Transferring from remote host
 - `scp username@ugrad.seas.ucla.edu:/home/user/docs/foo.txt /home/username`

Digital signature

- Protect **integrity** of the documents
 - Receiver received the document that the sender intended
- Digital signature is extra data attached to the document that can be used to check **tampering**
- Message digest
 - **Shorter** version of the document
 - Generated using **hashing** algorithms
 - Even a slight change in the original document will change the message digest with **high probability**

Digital signature

Verifies document integrity, but does it prove origin? and who is the Certificate Authority?



GNU privacy guard

> gpg [option]

--gen key

generating new keys

--armor

ASCII format

--export

exporting public key

--import

import public key

--detach-sign

creates a file with just the signature

--verify

verify signature with a public key

--encrypt

encrypt document

--decrypt

decrypt document

--list-keys

list all keys in the keyring

--send-keys

register key with a public server/-
keyserver option

--search-keys

search for someone's key

Client Authentication

- **Password-based authentication**
 - Prompt for password on remote server
 - If username specified exists and remote password for it is correct then the system lets you in
- **Key-based authentication**
 - Generate a key pair on the client
 - Copy the public key to the server (`~/.ssh/authorized_keys`)
 - Server authenticates client if it can demonstrate that it has the private key
 - The 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 (passphrase-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 passphrase 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

Lab 6

- **Securely login 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

Lab 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/id_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**
 - `$ emacs /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 private 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.`

How to Check IP Addresses

- `$ ifconfig`
 - configure or display the current network interface configuration information (IP address, etc.)
- `$ ping <ip_addr>`**(packet internet groper)**
 - 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`

Lab

web.cs.ucla.edu/classes/winter16/cs35L/assign/assign6.html