CS 35L

Software Construction Laboratory

Lecture 8.1

20th May, 2019

Logistics

- Assignment 10 Signup Sheet
  - https://docs.google.com/spreadsheets/d/19bPoaFoi9rWZ-05hTJgUAqZPKWlAetRjFMniljwmZBs/edit?usp=sharing
- Assignment 7 Deadline
  - 20th May, 2019 – 11:55pm
- Hardware requirement for Week 8
  - Seeed Studio BeagleBone Green Wireless Development Board
  - Today's class

Review - Previous Lab

- Static Linking
- Dynamic Linking
- Dynamic Loading
  - Examples

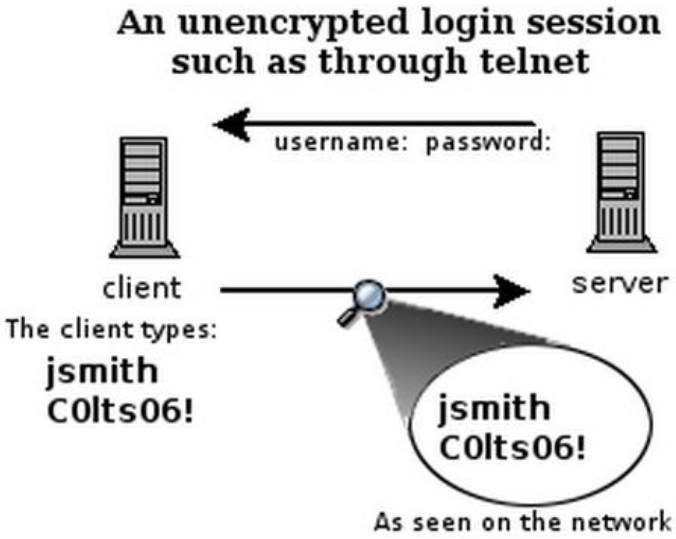
SSH

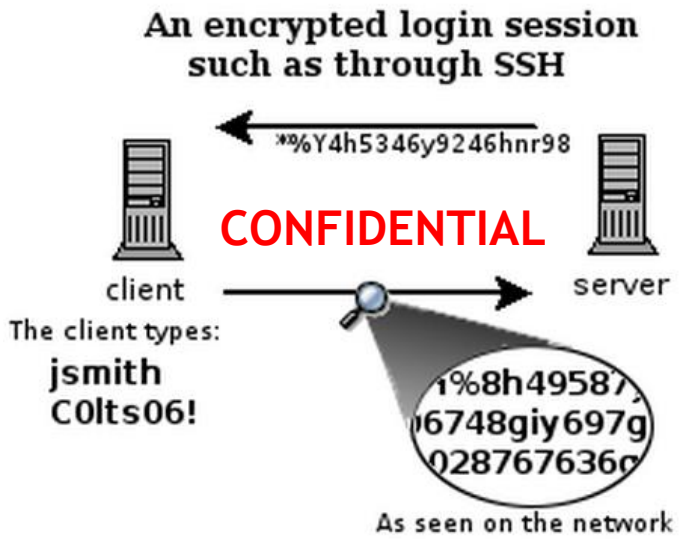
Communication Over the Internet

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

What is SSH?

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

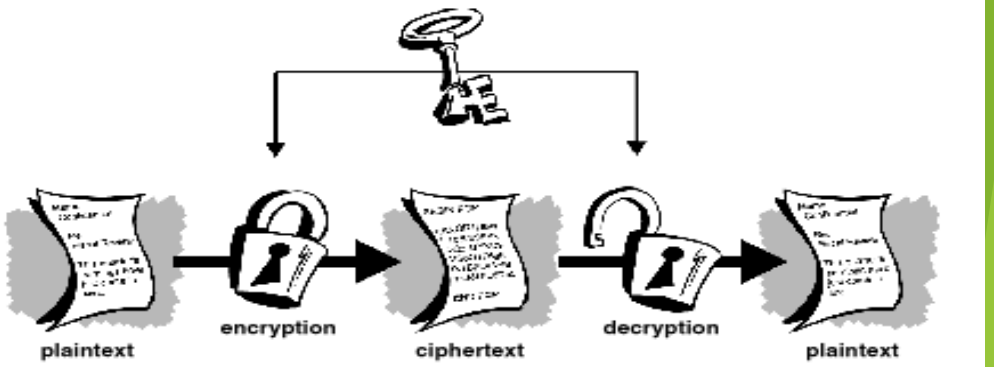




Encryption Types

- 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
    - 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!!!

Symmetric-key Encryption



- Also called Shared Key/Shared Secret encryption
- Same secret key used for encryption and decryption
- Key distribution is a problem
  - The secret key has to be delivered in a safe way to the recipient
  - Chance of key being compromised
- Key Generation
  - Uses the **Key Exchange Algorithm**
    - AES (Advanced Encryption Standard), CAST128, Blowfish
  - Both client and server derive this key using this algorithm
  - The two machines share public pieces of data and then manipulate it to independently calculate the secret key
- Secure because the key is never transmitted between the client and the host.

Symmetric-key Encryption

- Example: Data Encryption Standard (DES)
- Example: Caesar's Cipher
  - Map the alphabet to a shifted version
    - ABCDEFGHIJKLMNOPQRSTUVWXYZ
    - DEFGHIJKLMNOPQRSTUVWXYZABC
  - Plaintext – SECRET; CIPHERTEXT – VHFUHW
  - Key is 3 (Number of shifts)

Public Key Encryption (Asymmetric)

- Uses a pair of keys for encryption
  - Public Key – Published and known to everyone
  - Private Key – Secret key known only to the owner
  - Private key cannot be mathematically computed from the public key!
- Encryption
  - Use public key to encrypt messages
  - Anyone can encrypt message, but cannot decrypt the ciphertext
- Decryption
  - Use private key to decrypt messages
  - Example: RSA – Rivest, Shamir and Adleman
  - Property used: - Difficulty of factoring large integers to prime numbers
- One way relation
  - Cannot decrypt anything encrypted by the private key.
  - Public key cannot decrypt its own messages

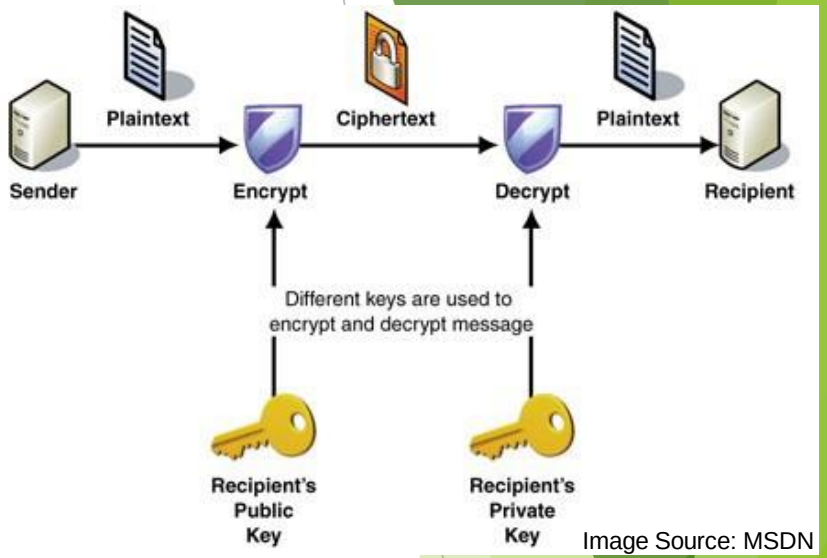


Image Source: MSDN

SSH Protocol

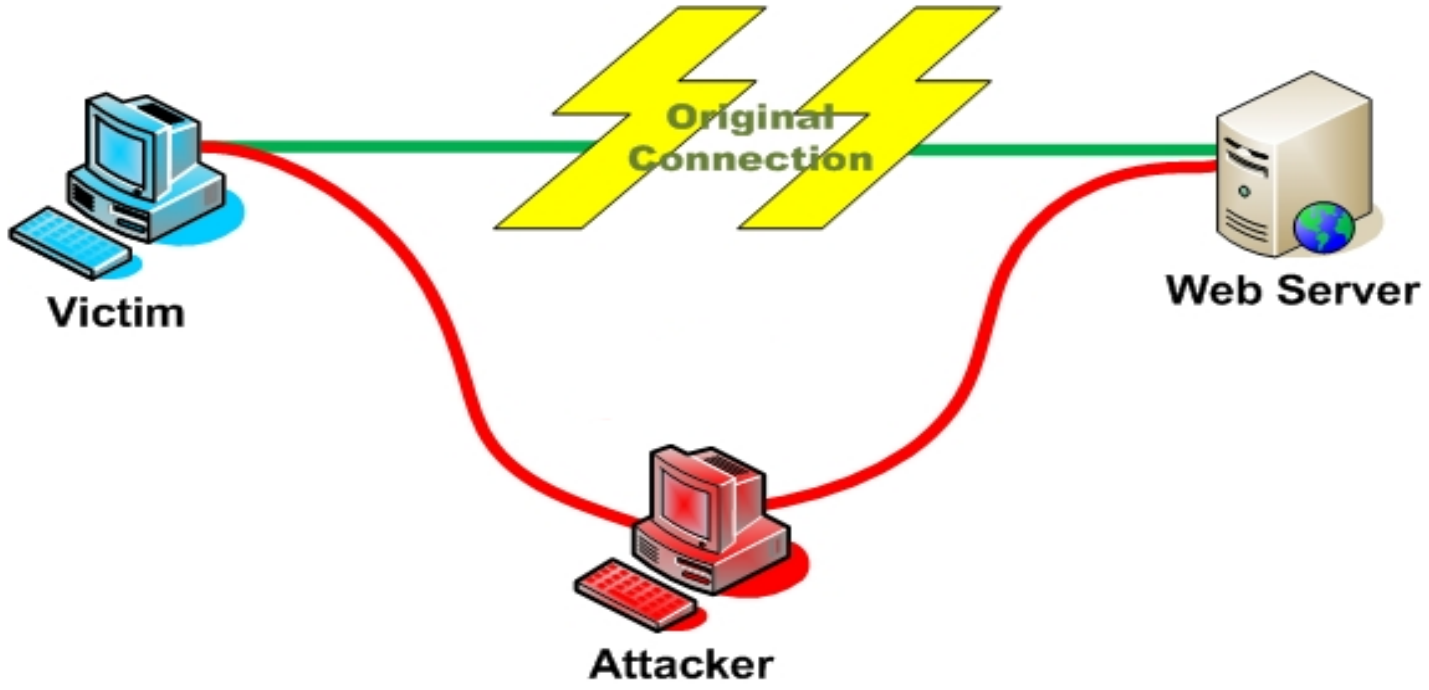
- Client ssh's to remote server
  - $ ssh username@somehost
  - If first time talking to server -> host/server validation

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

Server/Host Validation

- Next time client connects to server
  - Check server's public key against saved public key
  - If they don't match



Server Validation (cont'd)

- Client asks server to prove that it is the owner of the public key using asymmetric encryption
  - Create a challenge - Encrypt a message with public key
  - If server is true owner, it can decrypt the message with private key
- Server decrypts challenge with private key
  - Sends message back to client as validation
  - Client verifies message and if same, server is successfully validated
- Note that there is now "shared data" between the machines
- Client message = Challenge = shared data

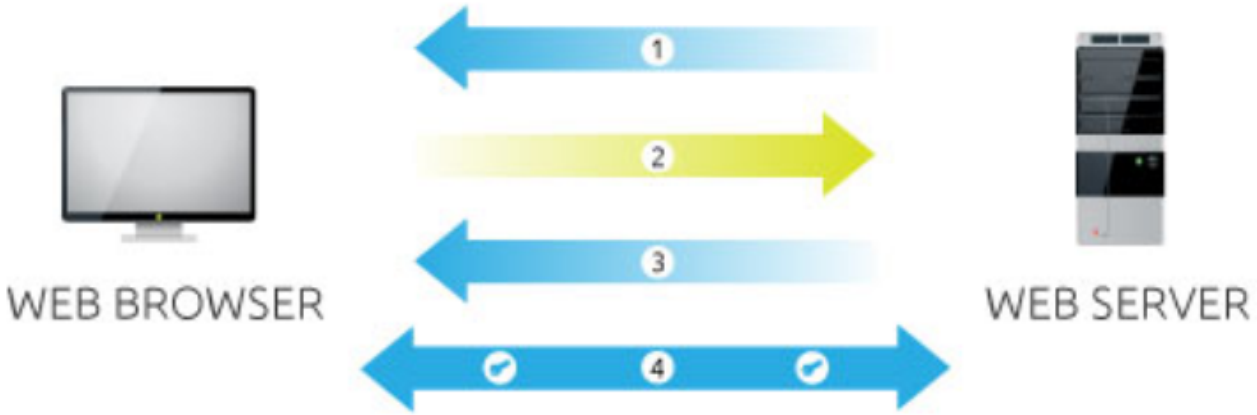
User Authentication

- **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!)

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

Session Encryption



1. **Server** sends a copy of its asymmetric public key.
2. **Browser** creates a symmetric session key and encrypts it with the server's asymmetric public key. Then sends it to the server.
3. **Server** decrypts the encrypted session key using its asymmetric private key to get the symmetric session key.
4. **Server** and **Browser** now encrypt and decrypt all transmitted data with the symmetric session key. This allows for a secure channel because only the browser and the server know the symmetric session key, and the session key is only used for that session. If the browser was to connect to the same server the next day, a new session key would be created.

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

X Window System

- Windowing system that forms the basis for most GUIs on UNIX
- 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 (X Session Forwarding)

Assignment 7 - Laboratory

- 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

Lab Environment Setup

- 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 installed,
  - Sudo apt-get install openssh-server
  - Sudo apt-get install openssh-client

Server Steps (for “root” user)

- 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 accunt for the client on the server
  - $ sudo useradd <username>
  - $ sudo passwd <username>
- Create .ssh directory for new user
  - $ sudo mkdir .ssh
- Change ownership and permission on .ssh directory
  - Sudo chown –R username .ssh
  - Sudo chmod 700 .ssh

Server Steps (for “debian” user)

- 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 (If still locked out of root)
  - 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 Address

- $ ifconfig
  - configure or display the current network interface configuration information (IP address, etc.)
- $ hostname –I
  - gives the IP address of your machine directly
- $ 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

Assignment 10 - Presentations

- Today's Presentation
  - Miles Wu
  - Daniel Adea
- Next Class
  - Chester Hulse
  - Jackie Lam

Questions?