DRONACHARYA
College of Engineering

Computer Science & Engineering

Data Communication and Computer Networks

(MTCSE-101-A)

Secure Shell (SSH)

Agenda

- Why SSH ?
- History of SSH
- What is SSH ?
- Protocol Architecture
- Functionality
- Quiz / Questions

The Need For SSH

- With the evolution of the internet, services such as file transfers, remote logins, and remote command executions became possible.
- Existing implementations of protocols that supported these services included ftp, rcp, telnet, rlogin, and rsh.
- Problem existed with these protocols:
  - They lacked security ! (r-commands)
  - Possible for an intruder to intercept and read data.
- Telnet was especially risky:
  - Plaintext user name and password was easily intercepted over the network.
- A new protocol was needed to fix these security problems.

History of the Protocol

- Event & Result
  - 1995 Finland University network compromised via a password-sniffing attack.
  - Tatu Ylönen, a researcher at the university develops the SSH1 product for himself to improve security.

- SSH1 quickly grew popular and its use increased:
  - SSH1 released with a free license
  - Ylönen founded a company (SSH Communications Security/SCS).
  - Ylönen submits the SSH-1 protocol to the IETF.
  - Problems were discovered that were not fixable without losing backwards compatibility.

- In 1996, a new version of the protocol was released:
  - New Protocol named SSH 2.0 or SSH-2.
  - It improved both security and features of SSH-1.
  - Multiple shell sessions over a single SSH connection and improved security through the Diffie-Hellman (D-H) key exchange.

- IETF formed the SECSH group to standardize the protocol and the group submitted the protocol SSH-2 in 1997.

Continuation of History

- SCS released SSH2, a software product based on the SSH-2 protocol, in 1998.
  - Restrictive license (only education and non-profit) slowed acceptance/usage
  - Continued use of SSH1 with an unrestricted license to everyone

- 2000, SCS eased their restrictive licenses:
  - Allowed several operating systems to implement them including Linux, NetBSD, FreeBSD, and OpenBSD.

- OpenBSD developed OpenSSH, another SSH implementation
  - Based on the 1.2.12 free licensed version of the original SSH.
  - Freely available under the OpenBSD license
  - Presently used in several operating systems.

- In 2006, SSH-2 protocol became the proposed internet standard by the IETF. Today, SSH is supported by several operating systems including Linux, Mac, and Windows.

About SSH

- SSH is both a program and a protocol
  - Allows users to securely log into another computer over an insecure network, executes commands and transfers files
  - Created as a replacement for TELNET, ftp, and rlogin, rsh, and rcp
  - Uses TCP and provides authentication, confidentiality (both data and command), integrity, authorization, data compression, and with SSH-2, multiplexing
  - Has transparent client/server communication over encrypted network connections
  - Can be implemented on most Operating Systems (Win, Mac, Unix/Linux)

- What it's Not ?
  - It is not a shell / Command Interpreter (e.g. wildcard expansion)
    - A channel to run shell on a remote computer

SSH Features

- *Authentication*
  - Proof of identity of users and servers, typically password and public-key signature, but other methods are available
- *Privacy*
  - Via strong standard encryption algorithms
- *Integrity*
  - Cryptographic integrity checking via MD5 and SHA-1 keyed hash algorithms
- *Authorization / Access*
  - Server configurable access
- *Forwarding* or *Tunnelling*
  - Encrypt other TCP/IP-based sessions
- *Data Compression*

Advantages

- SSH is available on most platform
  - Clients are available for many platforms (besides major Operating System – OS/2, BeOS, Java, etc.)
- Free for noncommercial use
  - The open source version has gone through many improvements with patches, bug fixes, and addition of functionalities.
  - lsh is the General Public License (GPL) version of SSH-2 – currently being standardized by the IETF SECSH working group.
- SSH can multiplex services over the same connection
  - One of the most powerful function of multiplexing is port forwarding or tunneling
  - SSH can securely tunnel insecure applications like POP3, SMTP, IMAP, and CVS.

Protection

- Perhaps, the most important advantage of SSH is its protection against packet spoofing, IP/host spoofing, password sniffing, and eavesdropping.
  - SSH uses user and host key (discuss later in encryption) rather than IP address.
    - SSH is less susceptible to packet spoofing and IP/host spoofing
  - SSH implements cryptography for both authentication and communication.
    - Strong encryption make password sniffing and eavesdropping virtually impossible.

- E.g. Electronic Frontier Foundation (EFF) in 1998 successful attacked a single message encrypted with DES symmetric cipher – the process took 56 hours on a $250,000 machine containing more than 18,000 custom chips. Because of security risk, users are advised to use newer 3DES.

Disadvantages

- Only support known port number
  - Dynamic port not supported
  - Port Number can be exploited.
- SSH cannot fix all TCP's problems since TCP run below SSH
  - Can minimize attack types with authentication and security
  - Network hijacking – SSH is vulnerable to DoS
- SSH cannot protect users from attack made through other protocols.
  - E.g. NFS mounting can allow malicious access to root on UNIX/LINUX systems
- SSH provides no protection against Trojan horses or viruses

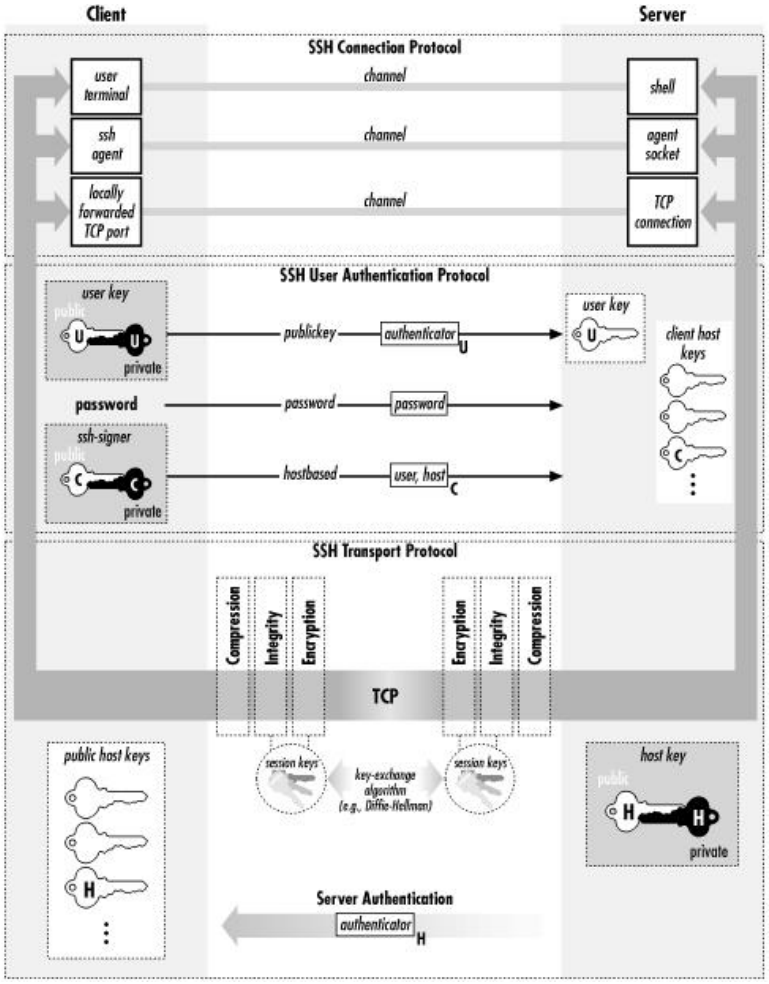
SSH-2 Protocol Architecture

- SSH-2 is separated into modules and consists of three protocols working together
  - SSH Transport Layer Protocol (SSH-TRANS)
    - server authentication, confidentiality, and integrity.
    - runs over a TCP/IP connection or some other reliable data stream.
  - SSH Authentication Protocol (SSH-AUTH)
    - authenticates the client-side user to the server.
    - runs over the transport layer protocol.
  - SSH Connection Protocol (SSH-CONN)
    - multiplexes the encrypted tunnel into several logical channels.
    - runs over the user authentication protocol.

- Port 22 used over TCP/IP

- Described in depth in RFC 4251: http://www.ietf.org/rfc/rfc4251.txt

SSH-2 Architecture (cont.)



SSH, The Secure Shell: The Definitive Guide Daniel J. Barrett, Richard Silverman, Publisher: O'Reilly, January 2001

SSH-2 Architecture Encryption

- Authentication Protocol – users to server by asymmetric public-key
  - one-time Password or Kerberos.
  - Uses RSA or DSA

- Transport Protocol - data by symmetric secret-key
  - Encryption type can be specified by user
    - based on random keys that are securely negotiated by client and server for each session
      - Diffie-Hellman key agreement algorithm
  - standard ciphers: 3DES, Blowfish, AES, Arcfour
  - Host to client asymmetric public key

SSH Functionality

- Copy files (scp & sftp)
- Remote terminal (ssh, slogin)
- Remote Commands (ssh)
- Keys and agents
- Port Forwarding and Xforwarding
- SOCKS - Proxies

SCP and SFTP

- SCP or Secure Copy allows files to be copied between hosts on a network.
  - scp fileToCopy user@host:directory/newFileName
  - scp user@host:directory/fileToCopy ./newFileName

- Created as a replacement for rcp
  - Authentication and security done by underlying SSH

- SFTP vs SCP ?
  - SFTP has more functions than SCP – e.g. directory listing, interrupted transfers resuming, and remote deletion.
  - SCP transfer file(s) only
  - SCS provides SCP-2
    - Uses SSH2 for data transfer
    - Uses SFTP-2 for data exchange between client and server

SCP and SFTP (cont.)

- SSH File Transfer Protocol
  - New Protocol designed by IETF SECSH working group
  - It is not FTP running over SSH
  - Sender: sftp –f d:\uploads*.*
  - Receiver: sftp –r f:\downloads
- SFTP can be secure replacement for FTP
  - FTP does not take any precautions measure to protect data
  - There are flaws inherit within the protocol that is susceptible to attacks – e.g. FTP bounce attack.
- SFTP typically run as a subsystem of SSH-2; but it can run over SSH-1

Remote Terminal

- Secure channel between client and server is established
  - Password supplied by client is encrypted
  - Password is sent over the network to the server
  - Server then checks the password and allows login
  - Data exchange between the two parties is secure

- Note:
  - Secure channel is established between the client and the server. If telnet is used to go to a third machine, that communication channel is not secure. SSH must again be used to establish a secure connection with the third machine.

Remote Terminal Example

To log into an account with the username smith on the remote computer merlin.csun.ecs.edu, use this command:

```
$ ssh smith@merlin.csun.ecs.edu or
$ ssh –l smith merlin.csun.ecs.edu
```

The command invokes the ssh client on the local computer which contacts the ssh server running on merlin.csun.ecs.edu and asks to be logged in as smith

The following message may be seen if the SSH client encounters a new remote machine.

```
Host key not found from the list of known hosts.
Are you sure you want to continue connecting (yes/no)?
```

If the user responds with a yes, the client continues:

```
Host ‘merlin.csun.ecs.edu’added to the list of known hosts.
```

The known hosts database can be found at $HOME/.ssh/known_hosts

Known-hosts mechanism helps minimize the “man-in-the-middle” attack

Remote Terminal Example (cont.)

- How ?
  - Hypothetical example → DNS/NIS hack
  - Public Key Cryptography / Host Key
  - @@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@ @@@@@@@@@@@@ @ WARNING: HOST IDENTIFICATION HAS CHANGED! @ @@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@ @@@@@@@@@@@@ IT IS POSSIBLE THAT SOMEONE IS DOING SOMETHING NASTY! Someone could be eavesdropping on you right now (man-in-the-middle attack)! It is also possible that the host key has just been changed. Please contact your system administrator. Add correct host key in <path>/known_hosts to get rid of this message. Agent forwarding is disabled to avoid attacks by corrupted servers. X11 forwarding is disabled to avoid attacks by corrupted servers. Are you sure you want to continue connecting (yes/no)
  - SSH Secure → Even a yes will prohibit some features
  - No Host Update Takes Place →Manually Done

Remote Command Execution

- Execute a command on a remote machine

- What does this do ?

```
$ ssh username@k200.ecs.csun.edu /usr/bin/date
  username@k200.ecs.csun.edu's password: ***
  Sun Apr 16 14:43:33 PDT 2006
```

Keys and Agents

- Need for Public-Key Authentication:
  - Passwords have several drawbacks
    - Good passwords must be random/long – hard to memorize !
    - Passwords sent on network may be intercepted
    - Password changes must be communicated
  - Keys are more secure then passwords !

- What is a Key ?
  - Digital Identity (sequence of bits)
  - SSH uses a private and public key
  - Private key (client) vs Public key (server) = key pair
  - Challenge and Authenticator

Keys and Agents (cont.)

- Generating Key pairs:
  - ssh-keygen creates a public and private key
  - A pass-phrase is supplied to protect the private key
  - OpenSSH can use either the RSA or DSA algorithm
  - Public key and private key are stored on the local machine after they are mathematically generated
    - ~/.ssh (SSH1/OpenSSH) or ~/.ssh2 (SSH2)
    - Private key SSH1 → identity
    - Public key SSH1→ identity.pub
    - Private key SSH2 → id_dsa_1024_a
    - Public key SSH2 → id_dsa_1024_a.pub
  - Private key is encrypted by pass-phrase and is only viewable by the person that generated it.
  - SSH2 allows a collection of private keys

Keys and Agents (cont.)

- Installing Public Key on Remote Machines
  - Public key must be installed on ssh server machine for the user account
    - *~/.ssh/authorized_keys*
    - *~/.ssh/authorization for ssh2*

- Benefits ?
  - Two components to capture → file and passphrase
  - No secret information is transmitted from client
  - Human passwords can be cracked while cryptographic functions are harder to break

Port Forwarding / Tunneling

- Port forwarding, also called tunneling, reroutes a TCP/IP connection to pass through an SSH connection
  - client side splicing is called *local* port forwarding (-L option)
  - server side splicing is called *remote* port forwarding (-R option)

- Not completely transparent, occurs at the application level, not the network level like VPN

- Connect to servers such as SMTP, IMAP, POP, and LDAP across a firewall that does not allow direct access while encrypting those sessions and passwords.

Port Forwarding / Tunneling (cont.)

- Local forwarding command line - forwards a local port on the local machine across an encrypted channel to a server port (remote-port) on the remote machine

```
ssh -L local-port:remote-machine:remote-port remote-machine
```

- Remote forwarding command line - remote host act as a proxy for a local port.
  - Server may want to enforce all connections from a specific port on remote machines.

```
ssh -R remote-port:remote-machine:local-port remote-machine
```

Port Forwarding Local Example

- IMAP mail servers listen on port 143.
- You want to connect to your IMAP server from *mymachine* and encrypt your session (contents and password).
- Your IMAP server is also running an SSH server.
- Forward *mymachine* port 1962 to *IMAPhost* 143

```
$ ssh -L 1962:localhost:143 username@IMAPhost
   -L specifies local forwarding (client spliced)
   1962 is port ssh listens on mymachine
   localhost:143 is the socket connect for the IMAP server
   IMAPhost is the IMAP server host name or IP
```

- Set your email client to connect to mymachine port 1962 to receive mail and the request is forwarded through an SSH Tunnel.
- Client config file can also be used with LocalForward keyword.
- Example shows the SSH server and IMAP server on the same host - hence the use of localhost for the IMAP server.

Port Forwarding Local -through FW

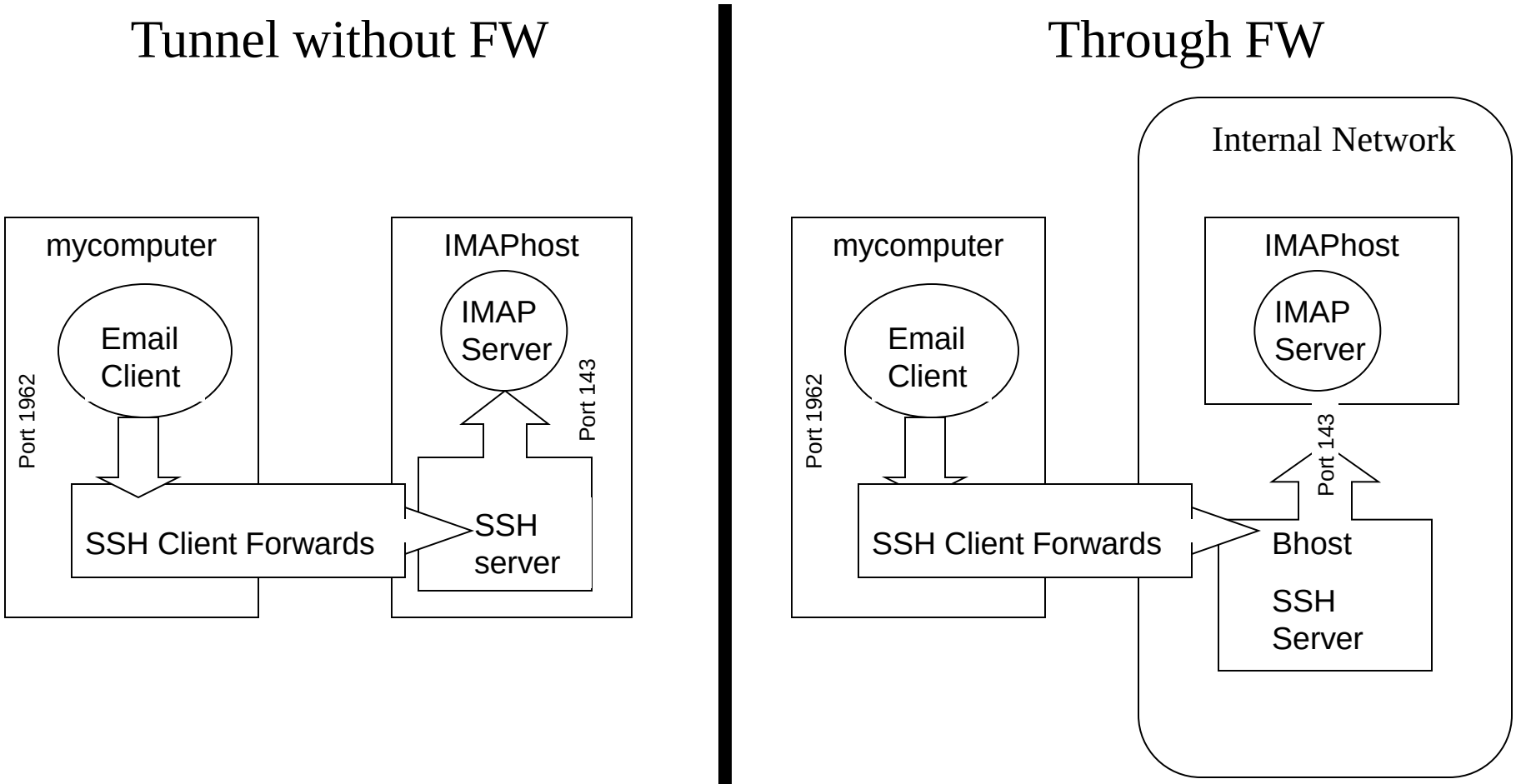
- Similar to last example except IMAP server is only accessible by bastion host (FW).
- Forward *mymachine* port 1962 to *IMAPhost* 143 through *Bhost*

```
$ ssh -L 1962:IMAPhost:143 username@Bhost
   -L specifies local forwarding (client spliced)
   1962 is port ssh listens on mymachine
   IMAPhost:143 is the socket connect for the IMAP
     server
   Bhost is location of SSH server
```

Bhost accepts SSH connections and forwards it on to IMAPhost unencrypted. - OK since protected by FW

Port Forwarding Local Example (cont.)

- The resulting connections look like this:



X Forwarding

- X-apps are run on a remote machine and appear securely on a local display.
  - X graphical display system for Unix consists of clients and servers.

- X-forwarding in SSH must be enabled by SSH client and server
  - *ForwardX11 yes* in ssh_config file on local machine
  - *X11Forwarding yes* in sshd_config file on remote machine
  - Creates and x-proxy and x-client upon login

```
$ ssh merlin.ecs.csun.edu
 Welcome to Darwin!
 merlin$ echo $DISPLAY
 merlin:10.0
 merlin$ xterm
```

The "xterm" X client appears on my local screen

X Forwarding (cont.)

- The DISPLAY value is an X-proxy created by SSH when you logged in.
- Now any X-app will connect to the X-proxy
  - forwards the program output to your SSH client which behaves like as a proxy X-client
- Authentication done by authentication spoofing
  - SSH client modifies x-server public-keys on local machine
    - local machine keeps private-key (.Xauthority)
  - Sends public x-server public-key to remote machine
    - remote machine keeps public-key (SXAUTHORITY)
  - When X-app tries to display to local machine X-authentication is done by verifying keys match.

SOCKS - Proxies

- SOCKetS is an application-layer network protocol for proxies.
- A proxy is a gateway that hides or protects a private network from the internet. Sometimes a firewall.
- For example: you might want to keep private internal network IPs from being exposed, so users connect to proxy server to get to the internet.
- SSH can create connections passing through a SOCKS proxy server.
  - OpenSSH, SSH1, and SSH2 all have slightly different implementations. SSH1 supports socks5 and SS2 supports socks4.
  - You must install SOCKS-aware SSH.
  - Lacks transparency - programs must be written to support a specific proxy configuration

SSH Summary

- Software solution to network security.
  - Provides secure alternatives to ftp, rcp, telnet, rlogin, and rsh.
- Available free as OpenSSH and a commercial product.
- Architecture consists of 3 modules on top of TCP/IP and uses strong standard encryption algorithms for authentication and data
- Preserves data integrity though MD5 and SHA-1 keyed hash algorithms
- Powerful public-keys for more than just password authentication
- Powerful port forwarding capability
- Proxy servers can be used to hide private network information

Questions / Quiz

- Why can’t we use port forwarding on our poker game programming project to secure our connection ?
- What are the three layers to the SSH-2 protocol architecture ?
- Name 3 SSH features.
- What is a benefit of using public-key authentication ?
- Why was there are need for the ssh protocol even though rcp, rsh, and rlogin were in existence ?