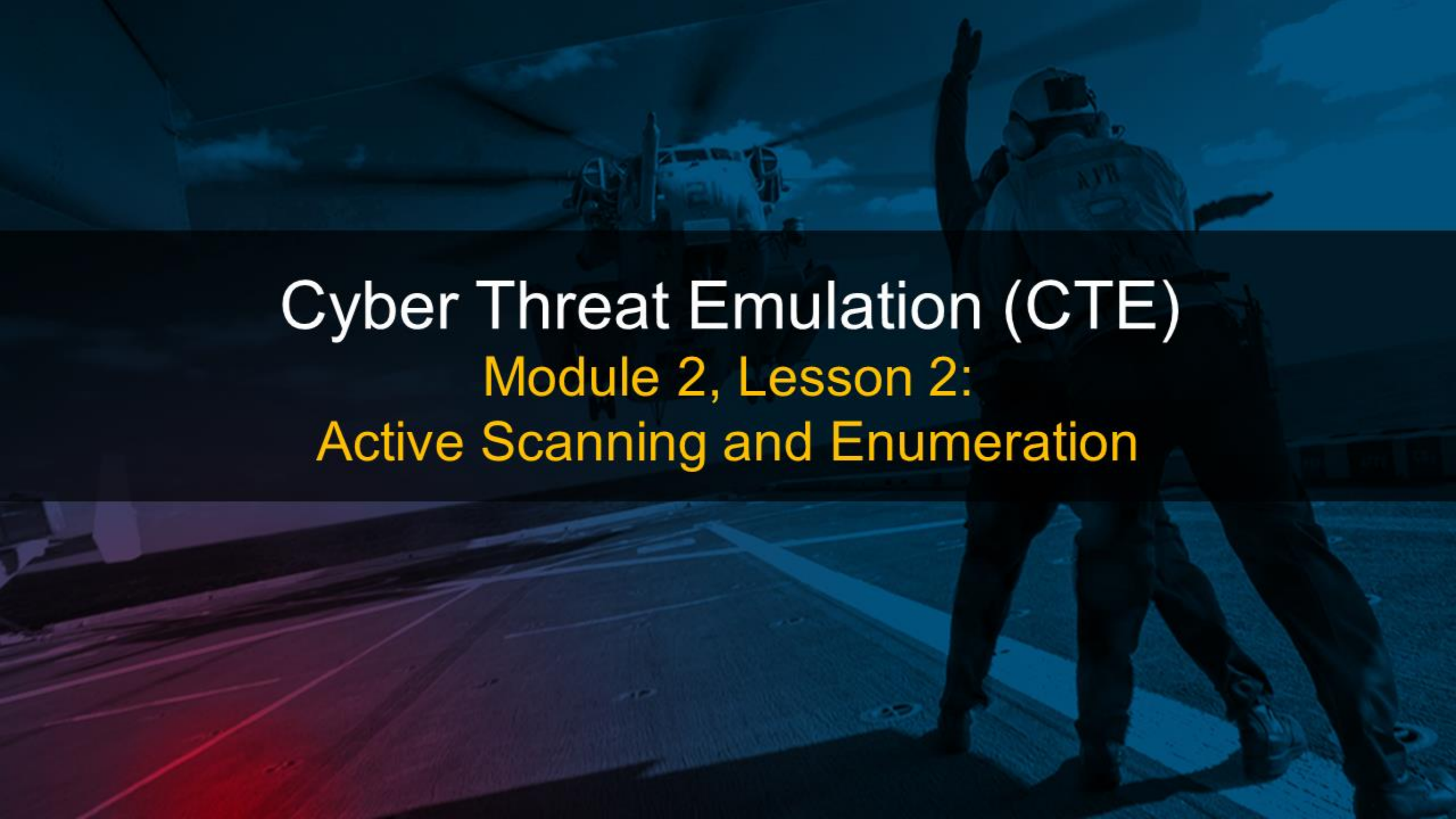


Cyber Threat Emulation (CTE)

Module 2, Lesson 2:
Active Scanning and Enumeration

Course Objectives

After completing this course, students will be able to:

- Summarize the CTE squad's responsibilities, objectives, and deliverables from each CPT stage
- Analyze threat information
- Develop a Threat Emulation Plan (TEP)
- Generate mitigative and preemptive recommendations for local defenders
- Develop mission reporting
- Conduct participative operations
- Conduct reconnaissance
- Analyze network logs for offensive and defensive measures

Course Objectives (Continued)

Students will also be able to:

- Analyze network traffic and tunneling protocols for offensive and defensive measures
- Plan non-participative operations using commonly used tools, techniques and procedures (TTPs)

Module 2: Threat Emulation (Objectives)

- Conduct reconnaissance
- Generate mission reports from non-participative operations
- Plan a non-participative operation using social engineering
- Plan a non-participative operation using Metasploit
- Analyze network logs for offensive and defensive measures
- Analyze network traffic and tunneling protocols for offensive and defensive measures
- Plan a non-participative operation using Python
- Develop fuzzing scripts
- Develop buffer overflow exploits

Module 2 – Lesson 2: Active Scanning and Enumeration (Objectives)

- Conduct active reconnaissance
- Develop mission reports from results of exploitation

Methods of Scanning

Passive discovery techniques

- Monitor communications
- Transparent
- Take more time

Active discovery techniques

- Fast
- Provide a lot of information
- Can trigger alerts

Methods of Scanning

Port Scanning

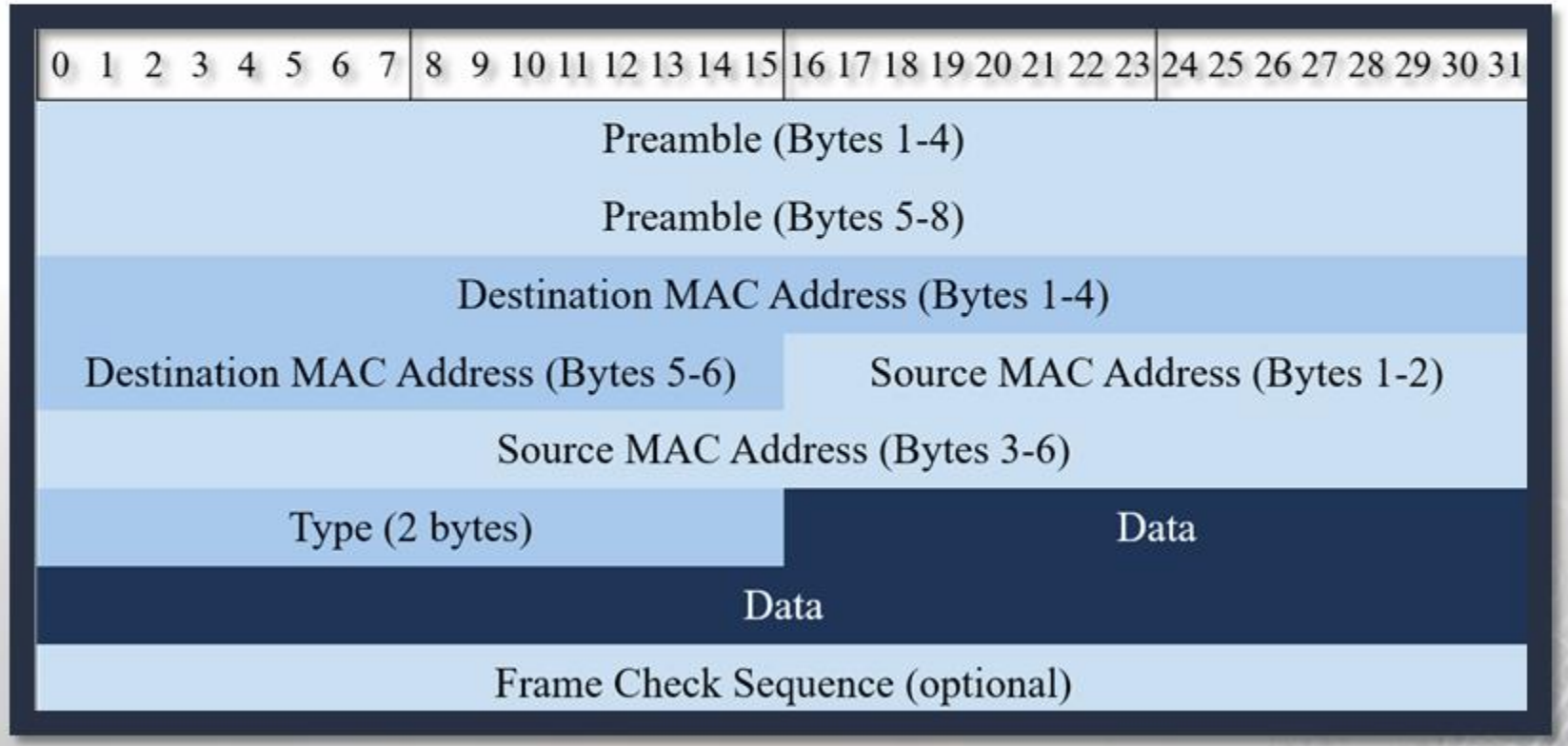
- Determining ports that are open
- Reveals presence of devices
- Reconnaissance tool for attackers

Vulnerability Scanning

- Combines port scanning
- Reveals hosts and servers for known vulnerabilities
- Provides report

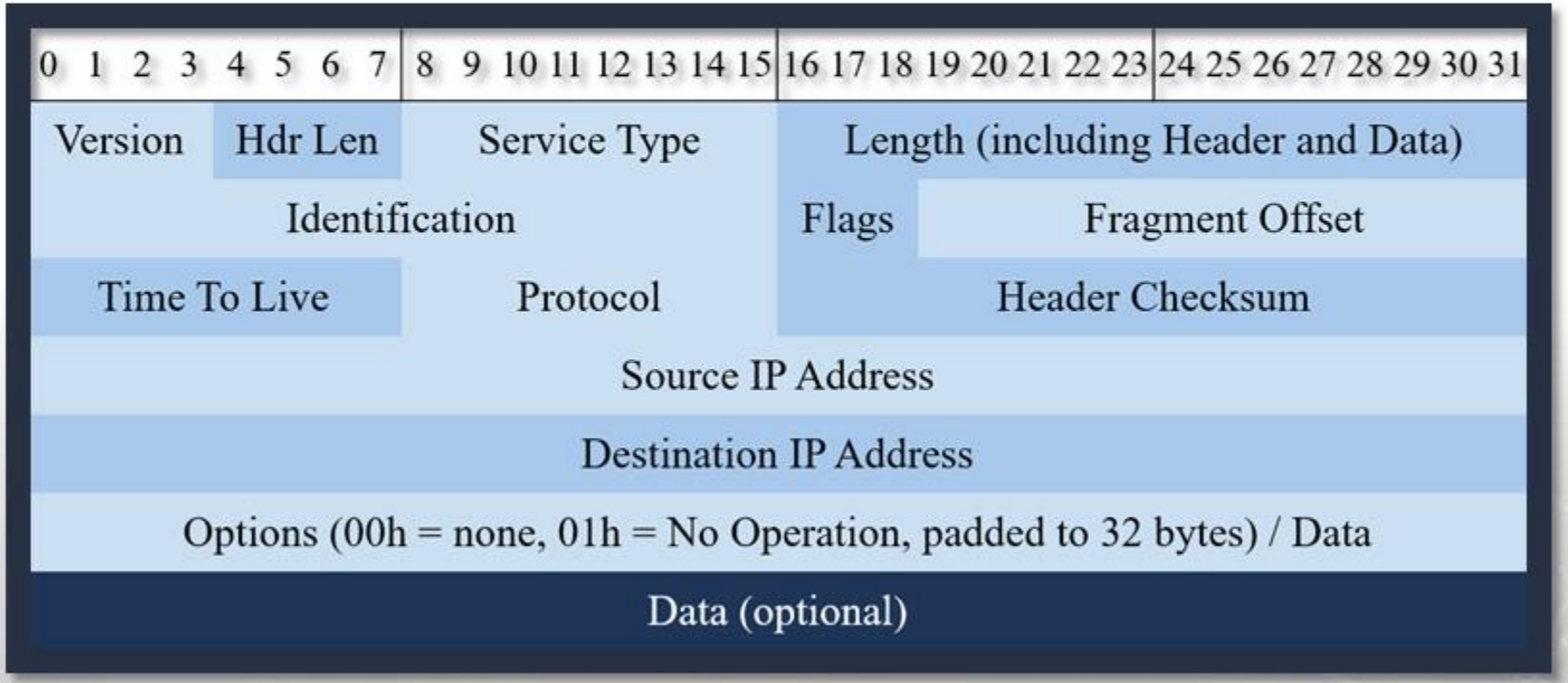
Major Protocols Review

- Ethernet



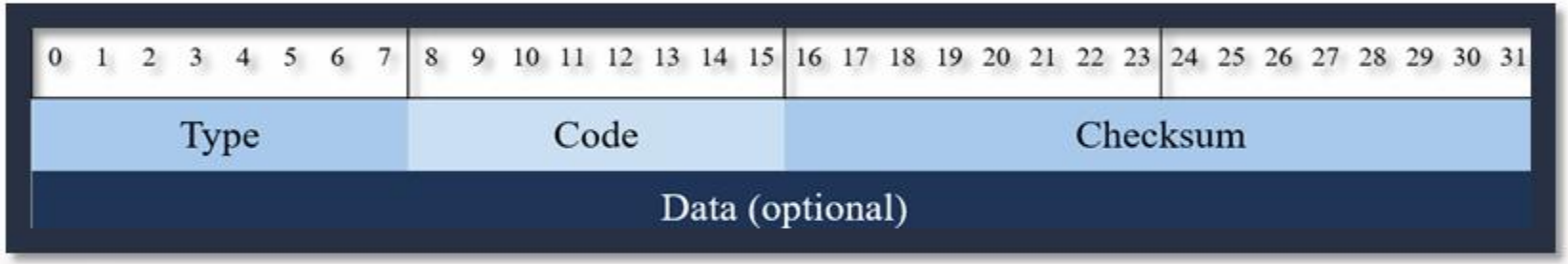
Major Protocols Review

- IPv4



Major Protocols Review

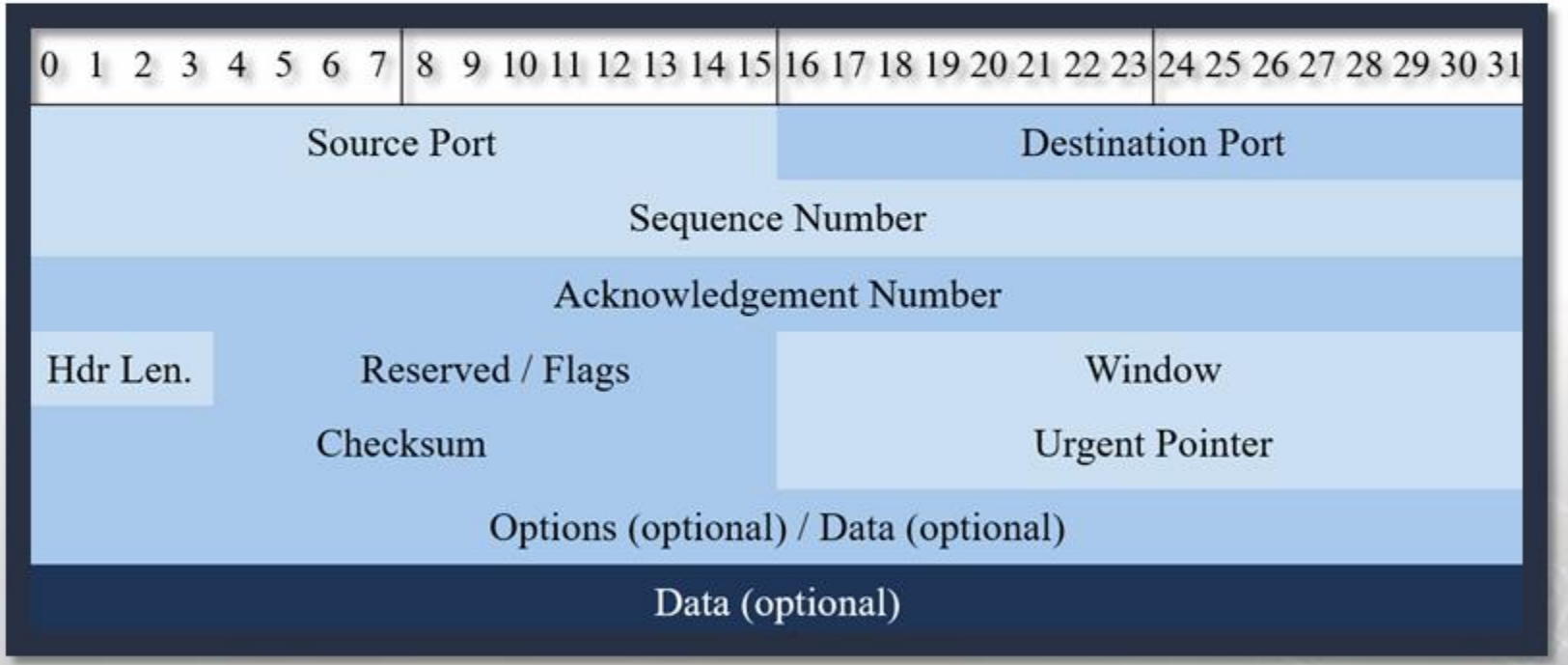
- ICMP



ICMP Type	Purpose
0x08	ICMP Echo Request
0x00	ICMP Echo Reply
0x03	ICMP Destination Unreachable
0x0B	ICMP TTL Exceeded

Major Protocols Review

- TCP



Major Protocols Review

- UDP

Active Scanning Techniques

- Discovering Hosts (Network Mapping)
  - MAC and IP addresses
  - Host names
  - Operating systems (OSs)
  - Services running

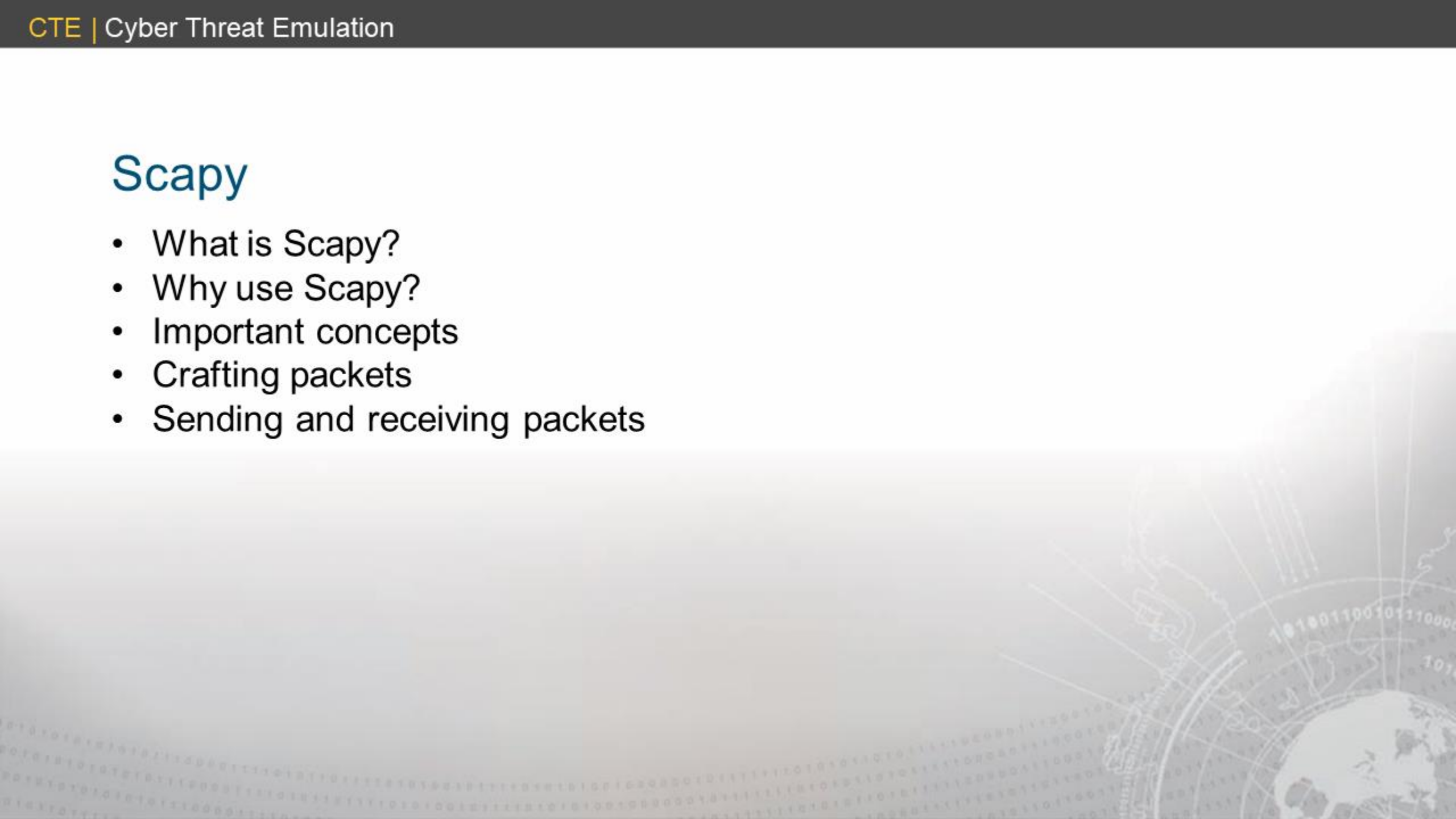
- Broadcast pings and ping sweeps
- ARP scans
- ICMPv6 neighbor discovery

Active Scanning Techniques

- Scanning ports
  - What will happen when connecting to a TCP port?
  - What about UDP?
- OS detection
- Service and version detection
- Timing and optimization
- Firewall and IDS evasion
- Packet manipulation

Scapy

- What is Scapy?
- Why use Scapy?
- Important concepts
- Crafting packets
- Sending and receiving packets



What is Scapy?

- Program for manipulating packets
- Capable of sniffing and transmitting packets
- Can handle many tasks:
  - Scanning
  - Traceroute
  - Host discovery
  - Probing
  - And more

Scapy

- Very useful tool
- Cross platform
- Scripting in Python
- Replay packets

Crafting Packets with Scapy

- Create a packet

```
>>> i=IP()
>>> i.src="192.168.229.55"
>>> i.dst="192.168.229.13"
>>> icmp=ICMP()
>>> icmp.type=8
>>> icmp.code=0
```

- Combine the layers

```
>>> packet=i/icmp
```

- Show the contents of the packet

```
>>> packet.show()
```

Sending Packets with Scapy

- Sending packets is easy!

```
>>> send(packet)
.
Sent 1 packets.
>>>
```

- Or you could use

```
>>> send(IP(src="192.168.229.55", dst="192.168.229.13")/ICMP(type=8, code=0))
.
Sent 1 packets.
```

Sending and Receiving Packets

Functions:

- Send and Receive
  - sr()
- Send and Receive one packet
  - sr1()

```
>>> sr1(IP(dst="192.168.229.13")/ICMP()/"Hello World")
Begin emission:
.Finished sending 1 packets.
*
Received 2 packets, got 1 answers, remaining 0 packets
<IP  version=4 ihl=5 tos=0x0 len=39 id=102 flags= frag=0 ttl
=128 proto=icmp chksum=0xeef3 src=192.168.229.13 dst=192.168
.229.29 options=[] |<ICMP  type=echo-reply code=0 chksum=0xa
e31 id=0x0 seq=0x0 |<Raw  load='Hello World' |<Padding  load
='\x00\x00\x00\x00\x00\x00\x00' |>>>>
```

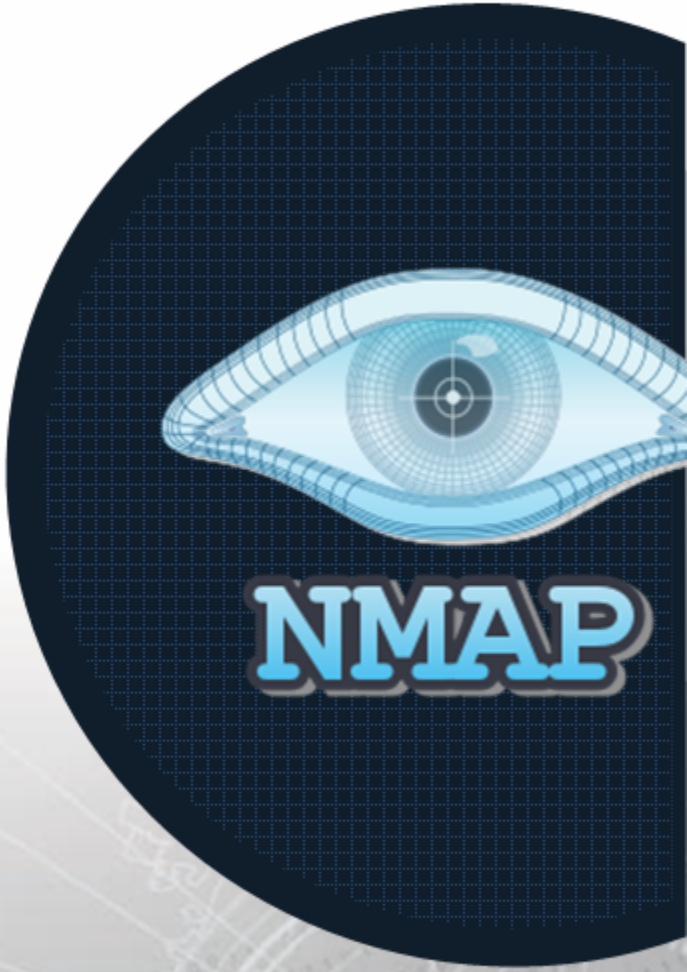
Nmap

Features include

- Host and port scanning
- OS detection
- Detecting versions
- Scriptable

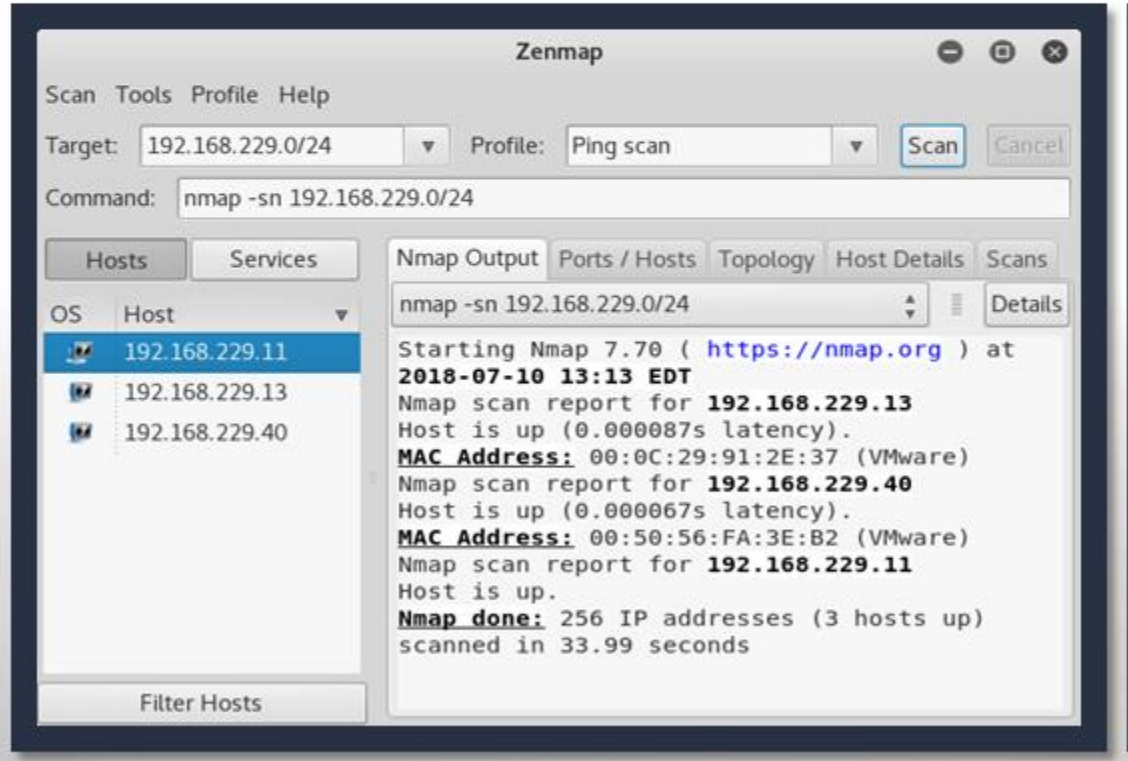
Uses include

- Mapping networks
- Identifying open ports
- Security Auditing



Nmap

Graphical User Interface (Zenmap):



Command Line:

```
root@kali:~# nmap -sn 192.168.229.0/24
Starting Nmap 7.70 ( https://nmap.org ) at 2018-07-25 10:21 EDT
Nmap scan report for 192.168.229.13
Host is up (0.00017s latency).
MAC Address: 00:0C:29:E8:6F:4E (VMware)
Nmap scan report for 192.168.229.40
Host is up (0.000050s latency).
MAC Address: 00:50:56:FA:3E:B2 (VMware)
Nmap scan report for 192.168.229.29
Host is up.
Nmap done: 256 IP addresses (3 hosts up) scanned in 33.84 seconds
root@kali:~#
```

Nmap Options

```
root@kali:~# nmap
Nmap 7.70 ( https://nmap.org )
Usage: nmap [Scan Type(s)] [Options] {target specification}
TARGET SPECIFICATION:
  Can pass hostnames, IP addresses, networks, etc.
  Ex: scanme.nmap.org, microsoft.com/24, 192.168.0.1; 10.0.0-255.1-254
  -iL <inputfilename>: Input from list of hosts/networks
  -iR <num hosts>: Choose random targets
  --exclude <host1[,host2][,host3],...>: Exclude hosts/networks
  --excludefile <exclude_file>: Exclude list from file
HOST DISCOVERY:
  -sL: List Scan - simply list targets to scan
  -sn: Ping Scan - disable port scan
  -Pn: Treat all hosts as online -- skip host discovery
```

Mapping the Environment – Discovering Hosts

Discovery options

- List scan (-sL)
- No port scan (-sn)
- No ping (-Pn)
- TCP SYN ping (-PS)
- TCP ACK ping (-PA)
- UDP ping (-PU)
- SCTP INIT ping (-PY)
- ICMP ping Types (-PE, -PP, -PM)
- IP protocol ping (-PO)
- ARP ping (-PR)

Mapping the Environment - Outputting Results

```
OUTPUT:
  -oN/-oX/-oS/-oG <file>: Output scan in normal, XML, s|<rIpt kIddi3,
     and Grepable format, respectively, to the given filename.
  -oA <basename>: Output in the three major formats at once
  -v: Increase verbosity level (use -vv or more for greater effect)
  -d: Increase debugging level (use -dd or more for greater effect)
  --reason: Display the reason a port is in a particular state
  --open: Only show open (or possibly open) ports
  --packet-trace: Show all packets sent and received
  --iflist: Print host interfaces and routes (for debugging)
  --append-output: Append to rather than clobber specified output files
  --resume <filename>: Resume an aborted scan
  --stylesheet <path/URL>: XSL stylesheet to transform XML output to HTML
  --webxml: Reference stylesheet from Nmap.Org for more portable XML
  --no-stylesheet: Prevent associating of XSL stylesheet w/XML output
```

Mapping the Environment – Port Scanning

Option	Scan Type
-sP	Ping scan only
-sS	TCP SYN
-sT	TCP connect
-sA	TCP ACK
-sN / -sF / -sX	TCP NULL, FIN, and Xmas
-sW	TCP window
-sM	TCP Miamon
-sU	UDP
-sI	Idle
-b	FTP bounce
-sO	IP protocol

OS Fingerprinting and Version Detection

```
OS DETECTION:
  -O: Enable OS detection
  --osscan-limit: Limit OS detection to promising targets
  --osscan-guess: Guess OS more aggressively
```

```
SERVICE/VERSION DETECTION:
  -sV: Probe open ports to determine service/version info
  --version-intensity <level>: Set from 0 (light) to 9 (try all probes)
  --version-light: Limit to most likely probes (intensity 2)
  --version-all: Try every single probe (intensity 9)
  --version-trace: Show detailed version scan activity (for debugging)
```

Nmap Scripting Engine (NSE)

Automates networking tasks

- Network discovery
- More sophisticated version detection
- Vulnerability detection
- Backdoor detection
- Vulnerability exploitation

Nmap Scripting Engine (NSE)

Types of Scripts:

- Prerule
- Host
- Service
- Postrule

Categories:

- Auth
- Broadcast
- Brute
- Default
- Discovery
- DOS
- Exploit
- External
- Fuzzer
- Intrusive
- Malware
- Safe
- Version
- Vuln

Using Nmap Scripting Engine (NSE)

- Selecting scripts
  - locate *.nse
  - find / -name “*.nse”
- Performing Script Scans

```
SCRIPT SCAN:
  -sC: equivalent to --script=default
  --script=<Lua scripts>: <Lua scripts> is a comma separated list of
           directories, script-files or script-categories
  --script-args=<n1=v1,[n2=v2,...]>: provide arguments to scripts
  --script-args-file=filename: provide NSE script args in a file
  --script-trace: Show all data sent and received
  --script-updatedb: Update the script database.
  --script-help=<Lua scripts>: Show help about scripts.
           <Lua scripts> is a comma-separated list of script-files or
           script-categories.
```

Ncat

```
root@kali:~# ncat -h
Ncat 7.70 ( https://nmap.org/ncat )
Usage: ncat [options] [hostname] [port]

Options taking a time assume seconds. Append 'ms' for milliseconds,
's' for seconds, 'm' for minutes, or 'h' for hours (e.g. 500ms).
  -4                         Use IPv4 only
  -6                         Use IPv6 only
  -U, --unixsock             Use Unix domain sockets only
  -C, --crlf                 Use CRLF for EOL sequence
  -c, --sh-exec <command>    Executes the given command via /bin/sh
  -e, --exec <command>       Executes the given command
      --lua-exec <filename>  Executes the given Lua script
  -g hop1[,hop2,...]         Loose source routing hop points (8 max)
  -G <n>                     Loose source routing hop pointer (4, 8, 12, ...)
```

Nping

```
root@kali:~# nping
Nping 0.7.70 ( https://nmap.org/nping )
Usage: nping [Probe mode] [Options] {target specification}

TARGET SPECIFICATION:
  Targets may be specified as hostnames, IP addresses, networks, etc.
  Ex: scanme.nmap.org, microsoft.com/24, 192.168.0.1; 10.0.*.1-24
PROBE MODES:
  --tcp-connect                    : Unprivileged TCP connect probe mode.
  --tcp                            : TCP probe mode.
  --udp                            : UDP probe mode.
  --icmp                           : ICMP probe mode.
  --arp                            : ARP/RARP probe mode.
  --tr, --traceroute               : Traceroute mode (can only be used with
                                     TCP/UDP/ICMP modes).
TCP CONNECT MODE:
   -p, --dest-port <port spec>     : Set destination port(s).
   -g, --source-port <portnumber>  : Try to use a custom source port.
TCP PROBE MODE:
   -g, --source-port <portnumber>  : Set source port.
```

Nping

- Use `man nping` to find syntax examples

```
      A typical Nping execution is shown in Example 1. The only Nping arguments used in this example are
      -c, to specify the number of times to target each host, --tcp to specify TCP Probe Mode, -p 80,433
      to specify the target ports; and then the two target hostnames.

      Example 1. A representative Nping execution

          # nping -c 1 --tcp -p 80,433 scanme.nmap.org google.com

          Starting Nping ( https://nmap.org/nping )
          SENT (0.0120s) TCP 96.16.226.135:50091 > 64.13.134.52:80 S ttl=64 id=52072 iplen=40  seq=1077657
88 win=1480
```

Nping

```
root@kali:~# nping -c 1 --tcp -p 80,433 192.168.229.13 192.168.229.80

Starting Nping 0.7.70 ( https://nmap.org/nping ) at 2018-07-25 13:50 EDT
SENT (0.0818s) TCP 192.168.229.29:1837 > 192.168.229.13:80 S ttl=64 id=4814 iplen=40  seq=145641095 win=1480
SENT (1.0827s) TCP 192.168.229.29:1837 > 192.168.229.80:80 S ttl=64 id=4814 iplen=40  seq=145641095 win=1480
RCVD (1.0834s) TCP 192.168.229.80:80 > 192.168.229.29:1837 SA ttl=128 id=891 iplen=44  seq=3928731479 win=8192 <mss 1460>
SENT (2.0840s) TCP 192.168.229.29:1837 > 192.168.229.13:433 S ttl=64 id=4814 iplen=40  seq=145641095 win=1480
SENT (3.0857s) TCP 192.168.229.29:1837 > 192.168.229.80:433 S ttl=64 id=4814 iplen=40  seq=145641095 win=1480
RCVD (3.0860s) TCP 192.168.229.80:433 > 192.168.229.29:1837 RA ttl=128 id=892 iplen=40  seq=0 win=0

Statistics for host 192.168.229.13:
 |  Probes Sent: 2 | Rcvd: 0 | Lost: 2  (100.00%)
 |_ Max rtt: N/A | Min rtt: N/A | Avg rtt: N/A
Statistics for host 192.168.229.80:
 |  Probes Sent: 2 | Rcvd: 2 | Lost: 0  (0.00%)
 |_ Max rtt: 0.567ms | Min rtt: 0.177ms | Avg rtt: 0.372ms
Raw packets sent: 4 (160B) | Rcvd: 2 (92B) | Lost: 2 (50.00%)
Nping done: 2 IP addresses pinged in 3.12 seconds
```

Ndiff

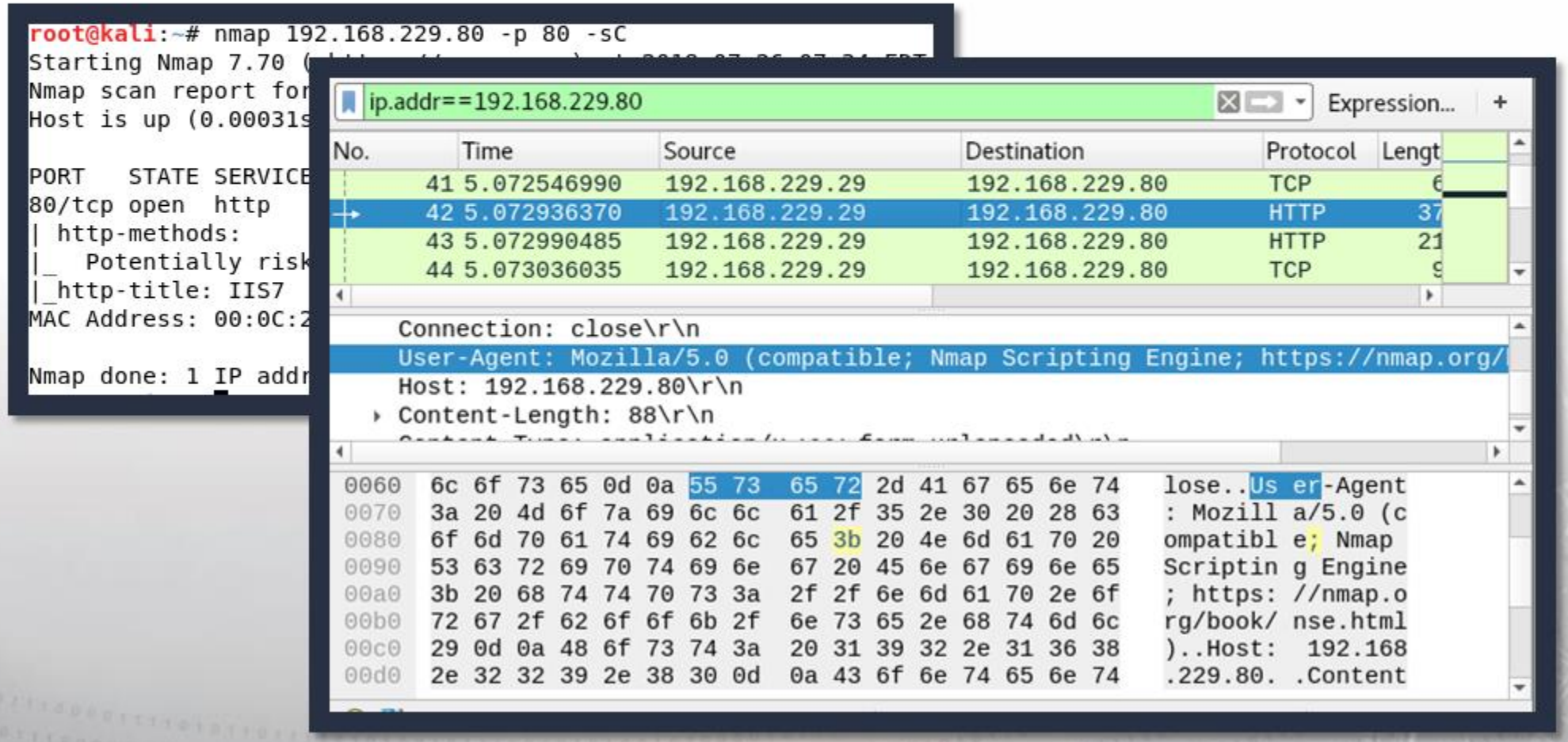
```
root@kali:~# ndiff scan1.xml scan2.xml
-Nmap 7.70 scan initiated Mon Jul 16 09:12:34 2018 as: nmap -sS -oX scan1.xml 19
2.168.229.13
+Nmap 7.70 scan initiated Mon Jul 16 08:40:37 2018 as: nmap -sS -oX scan2.xml 19
2.168.229.13

 192.168.229.13, 00:0C:29:91:2E:37:
-Not shown: 990 closed ports
+Not shown: 989 closed ports
 PORT   STATE SERVICE VERSION
+80/tcp open  http
root@kali:~#
```

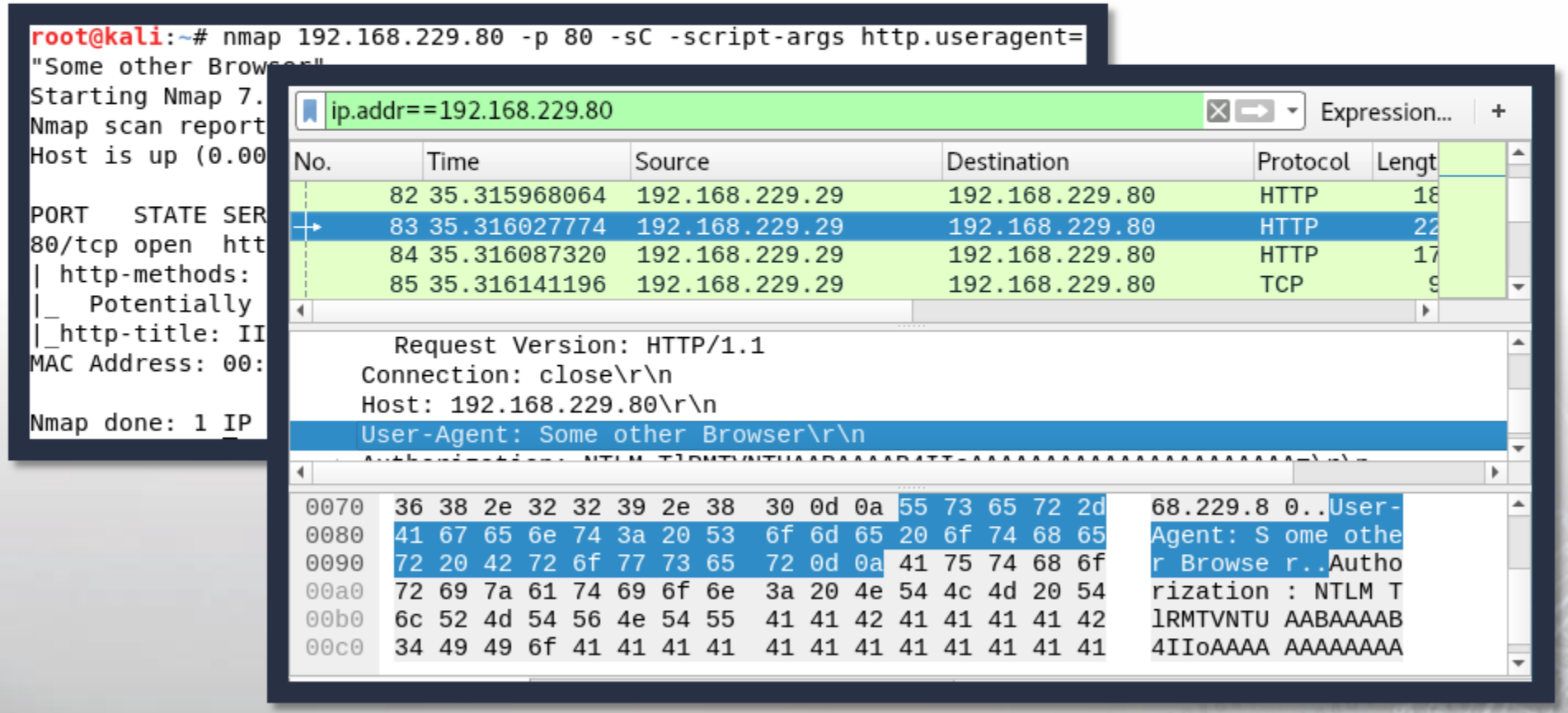
Evasion Techniques

- Customizing TCP scan flag (--scan-flags)
- Fragmentation (-f / --mtu)
- Adding decoy IP addresses (-D)
- Idle Scan (-sI)
- Changing the Source port (-g)
- Spoofing IP (-S) and MAC address (--spoof-mac)
- Randomize target scan order (--randomize-hosts)
- Add random data to packets (--data-length)
- Manipulating the time-to-live field (--ttl)
- Send packets with bogus TCP or UDP checksums (--badsums)
- Firewalk (--script firewalk)

Evasion Techniques



Evasion Techniques

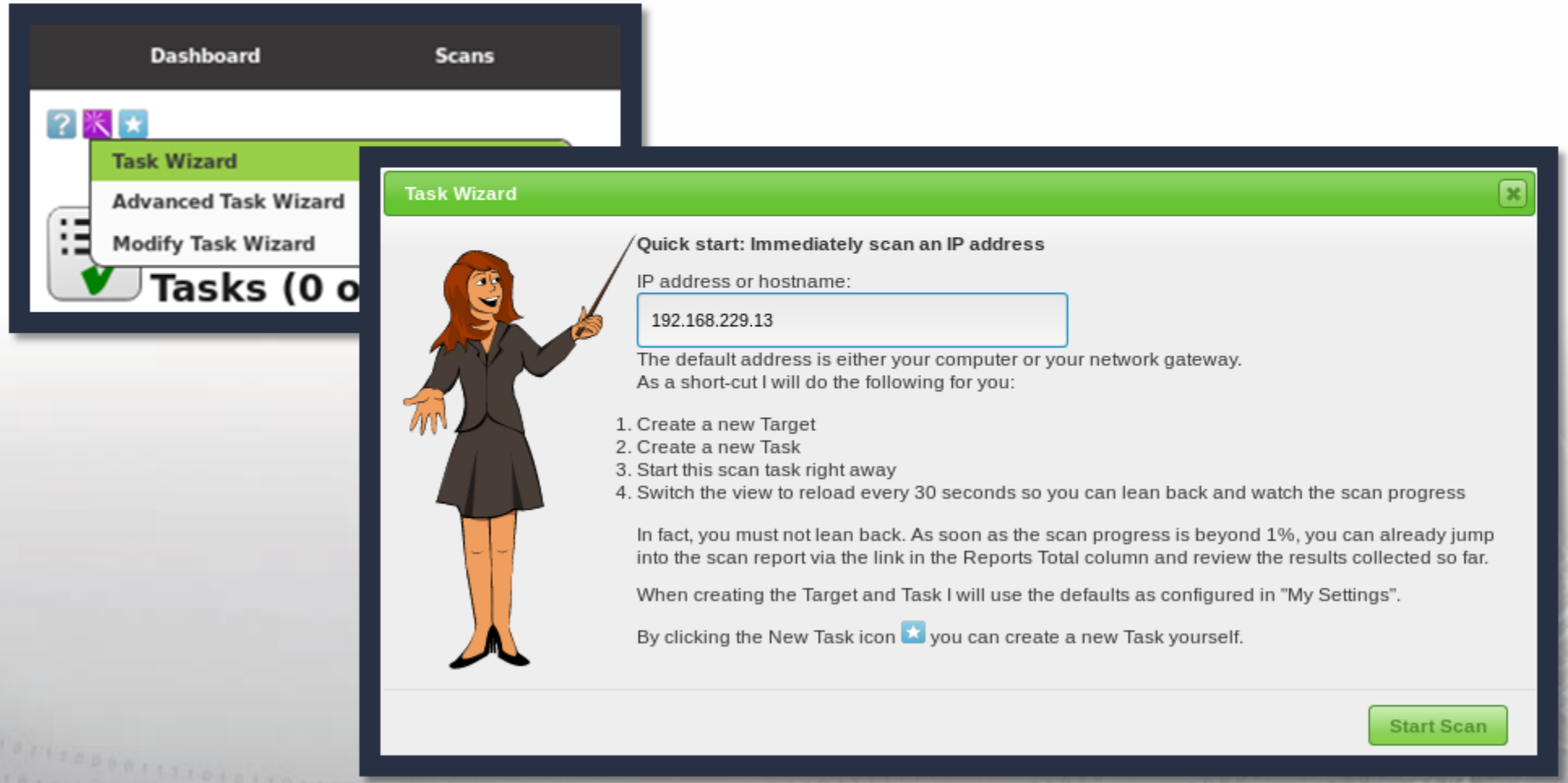


OpenVAS

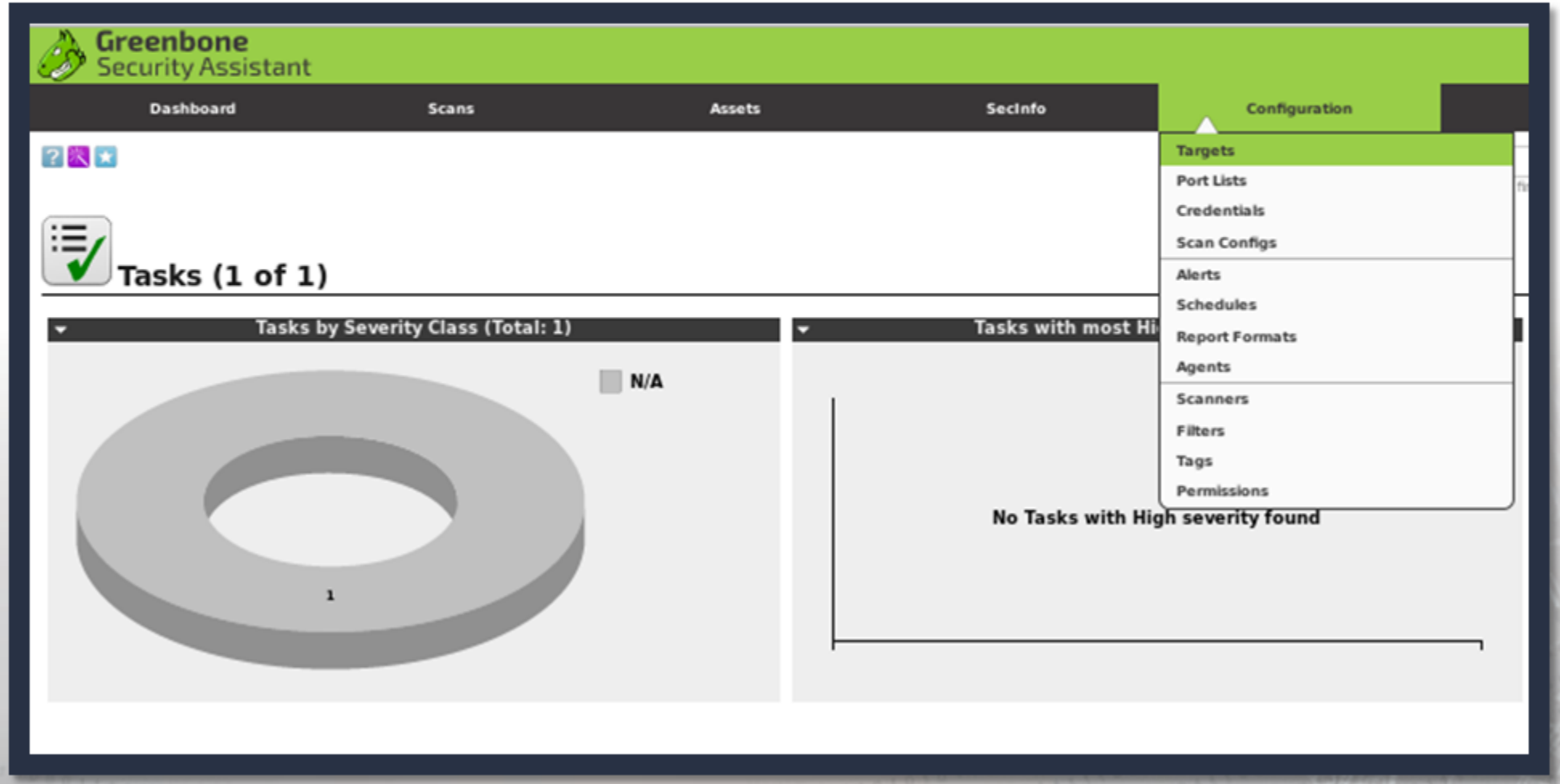
- Open source vulnerability scanner
- It's free!
- Over 50,000 network vulnerability tests
- Identifies vulnerabilities - Does not fix them
- Allows customized:
  - Targets
  - Scans
  - Depth of scans, and scan regions
  - Reports

OpenVAS – Accessing the Web Interface

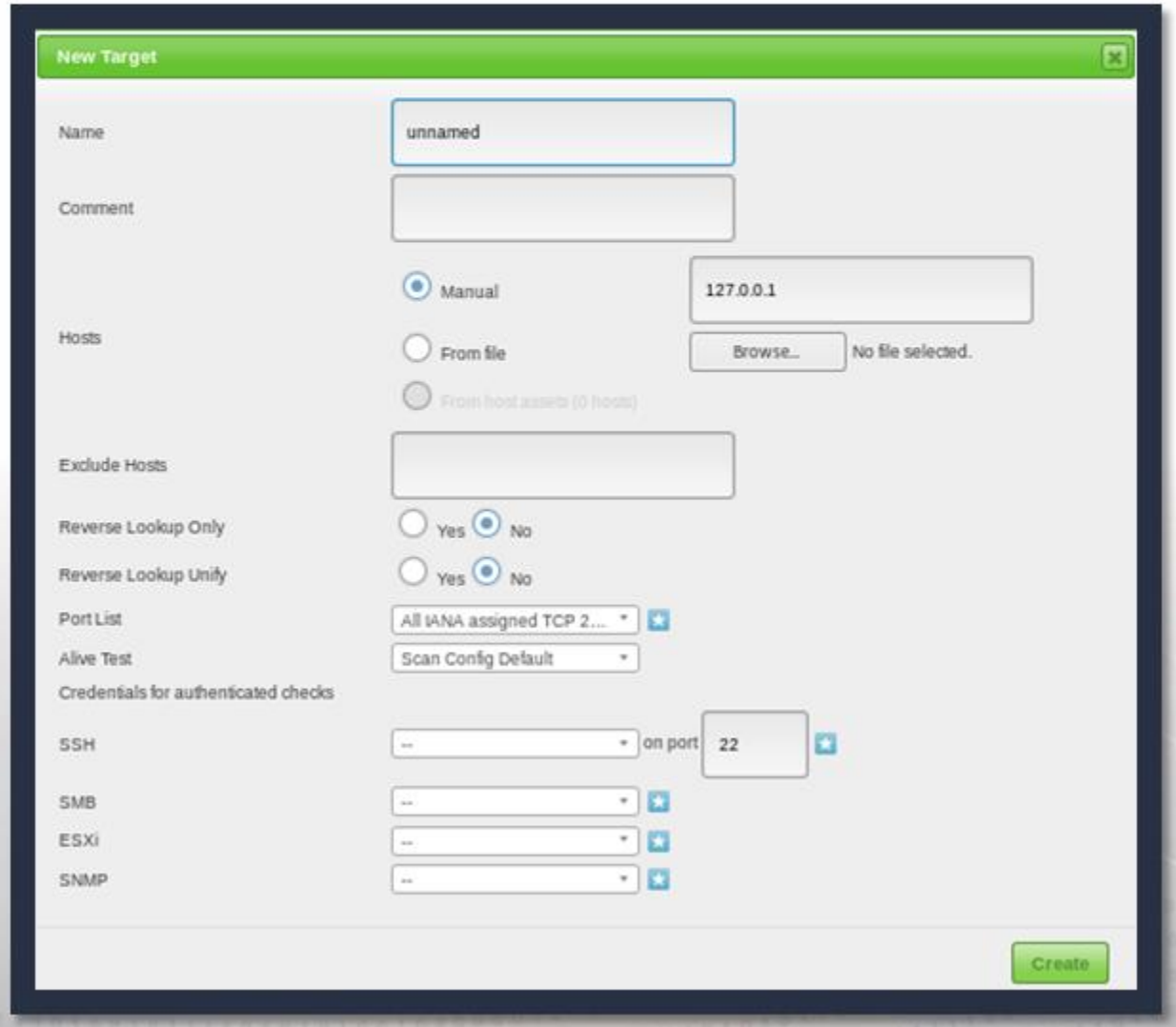
OpenVAS – Configuring a Scan



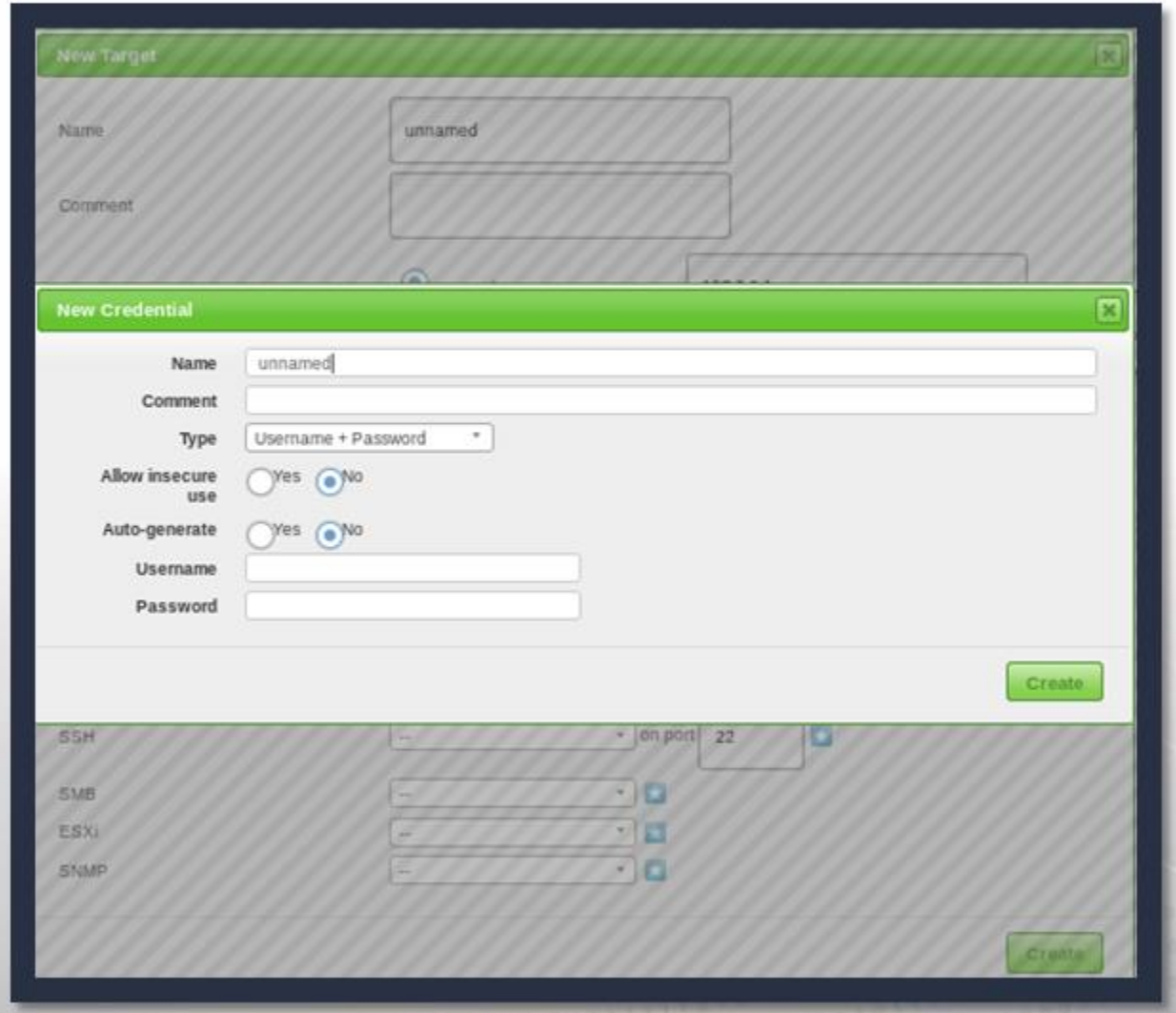
OpenVAS – Adding Targets



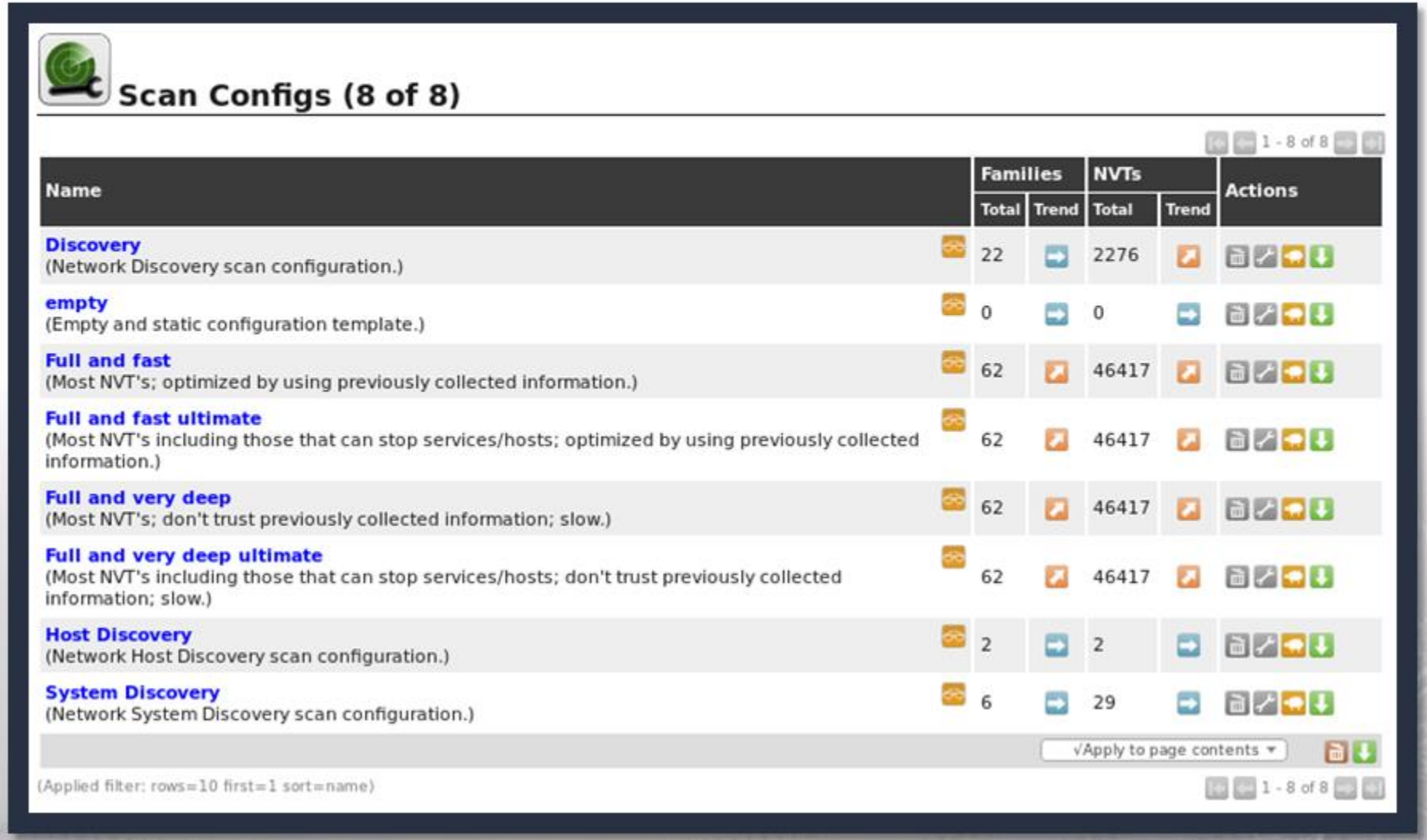
OpenVAS – Adding Targets



OpenVAS – Adding Credentials



OpenVAS – Changing the Network Vulnerability Tests (NVTs)



Scan Configs (8 of 8)

1 - 8 of 8

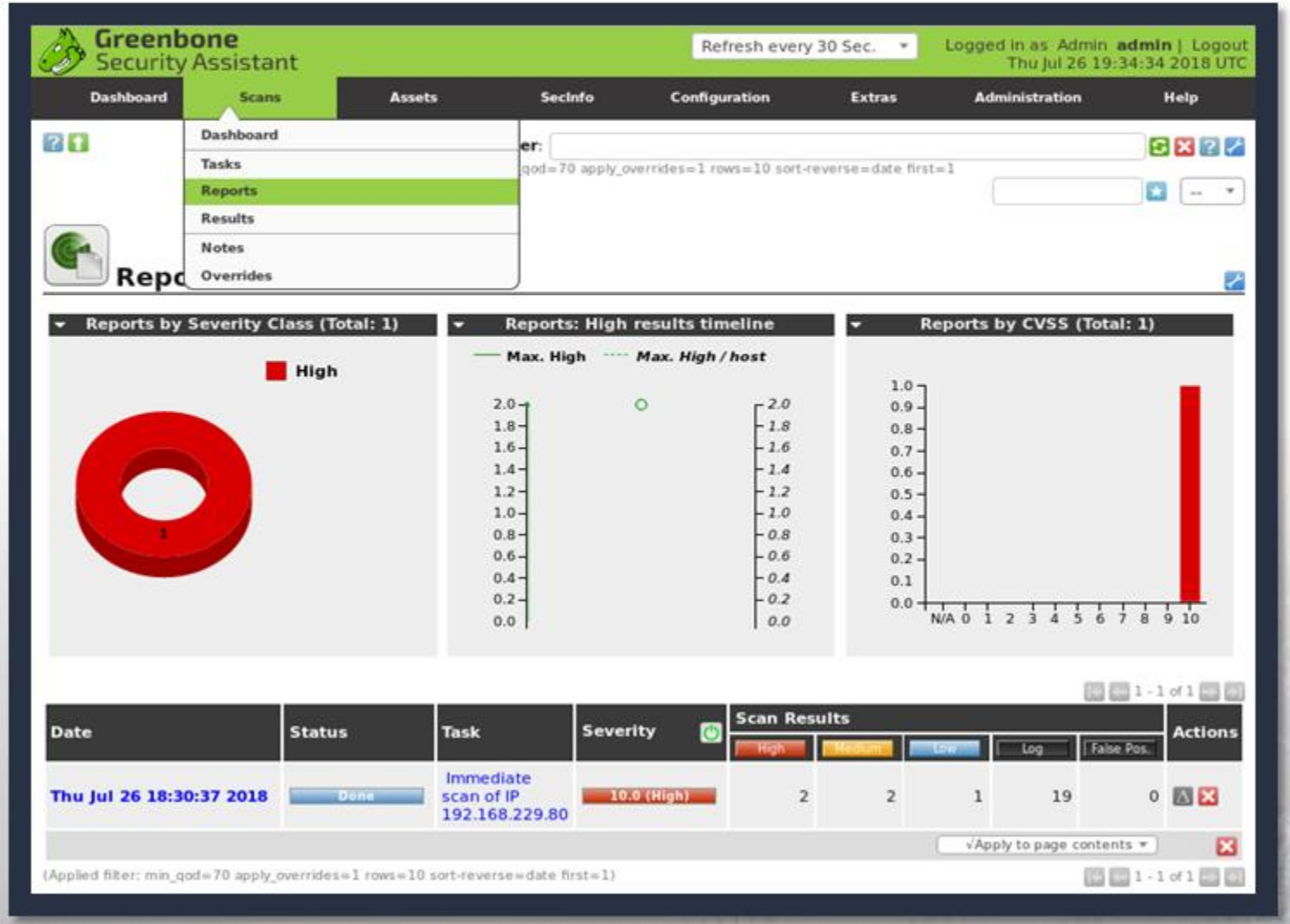
Name	Families		NVTs		Actions
	Total	Trend	Total	Trend	
Discovery (Network Discovery scan configuration.)	22		2276		
empty (Empty and static configuration template.)	0		0		
Full and fast (Most NVT's; optimized by using previously collected information.)	62		46417		
Full and fast ultimate (Most NVT's including those that can stop services/hosts; optimized by using previously collected information.)	62		46417		
Full and very deep (Most NVT's; don't trust previously collected information; slow.)	62		46417		
Full and very deep ultimate (Most NVT's including those that can stop services/hosts; don't trust previously collected information; slow.)	62		46417		
Host Discovery (Network Host Discovery scan configuration.)	2		2		
System Discovery (Network System Discovery scan configuration.)	6		29		

√Apply to page contents

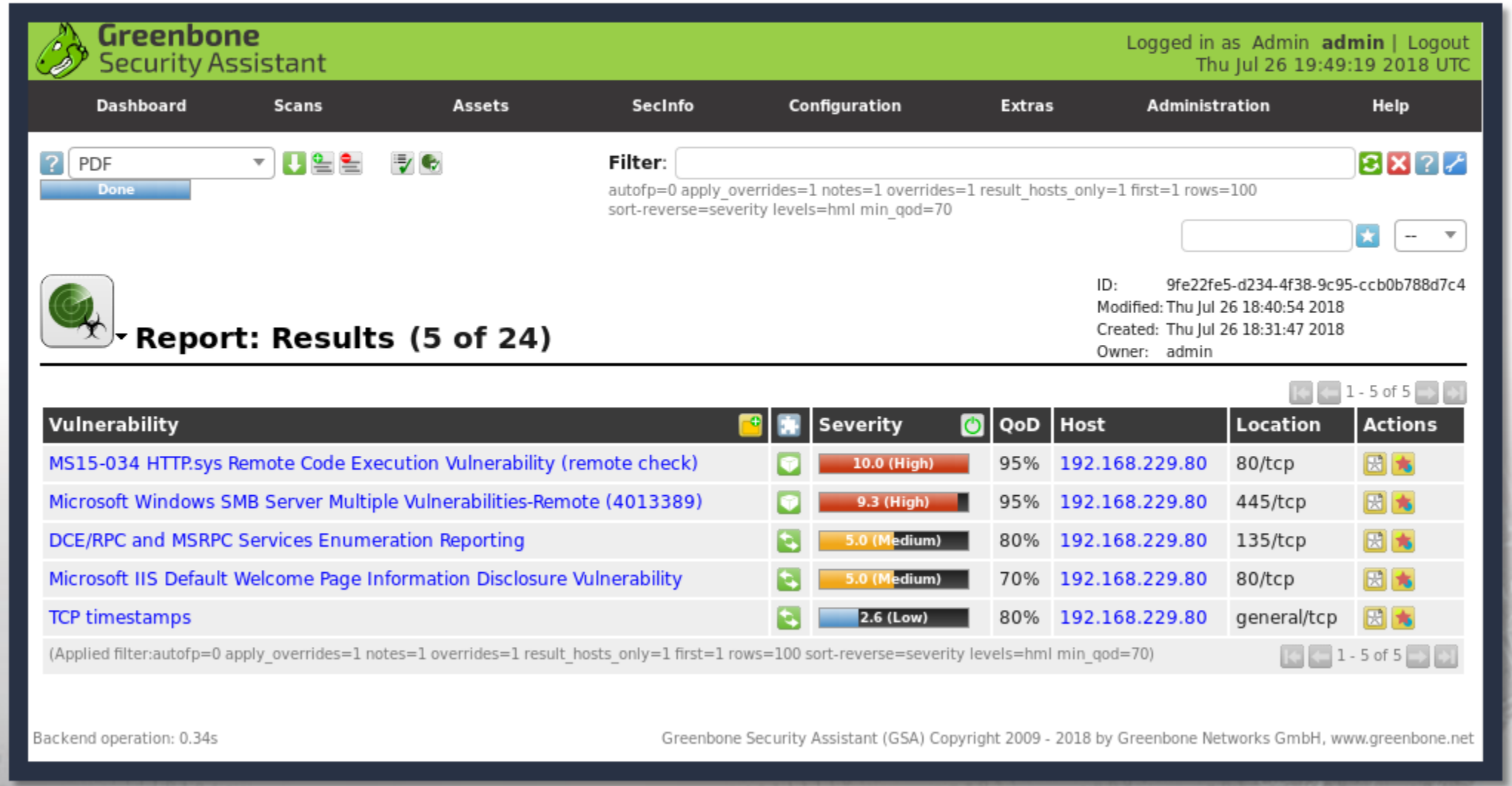
(Applied filter: rows=10 first=1 sort=name)

1 - 8 of 8

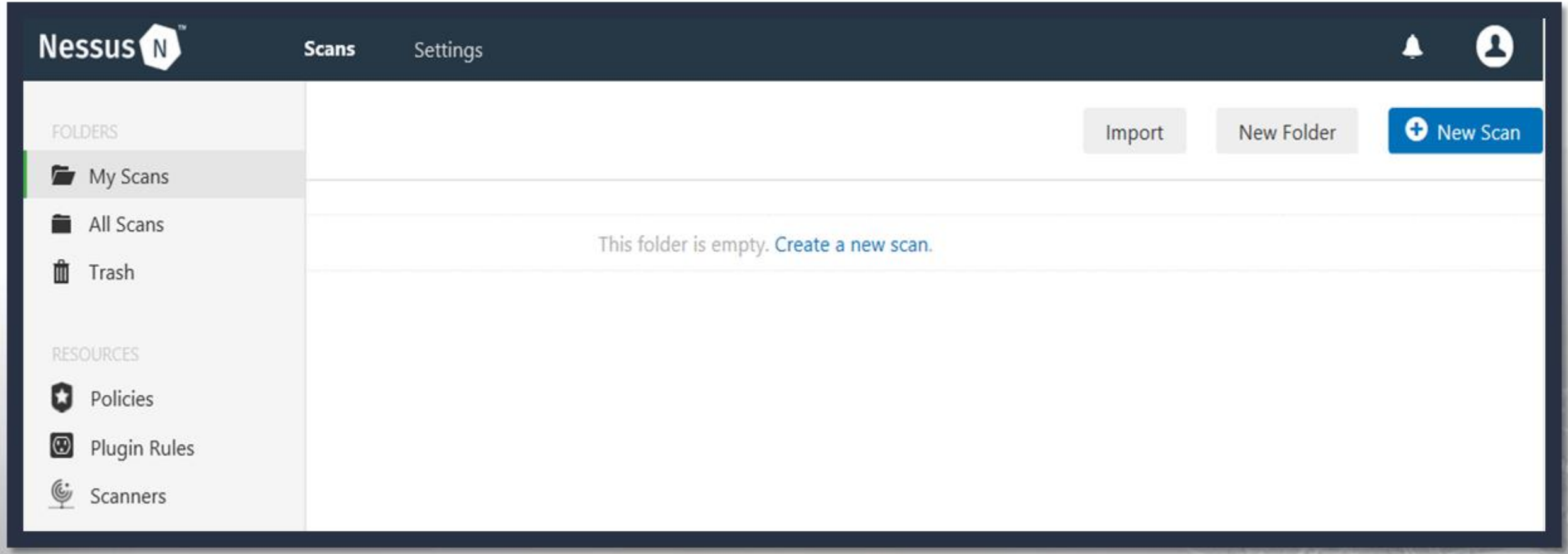
OpenVAS – Reviewing Scan Results



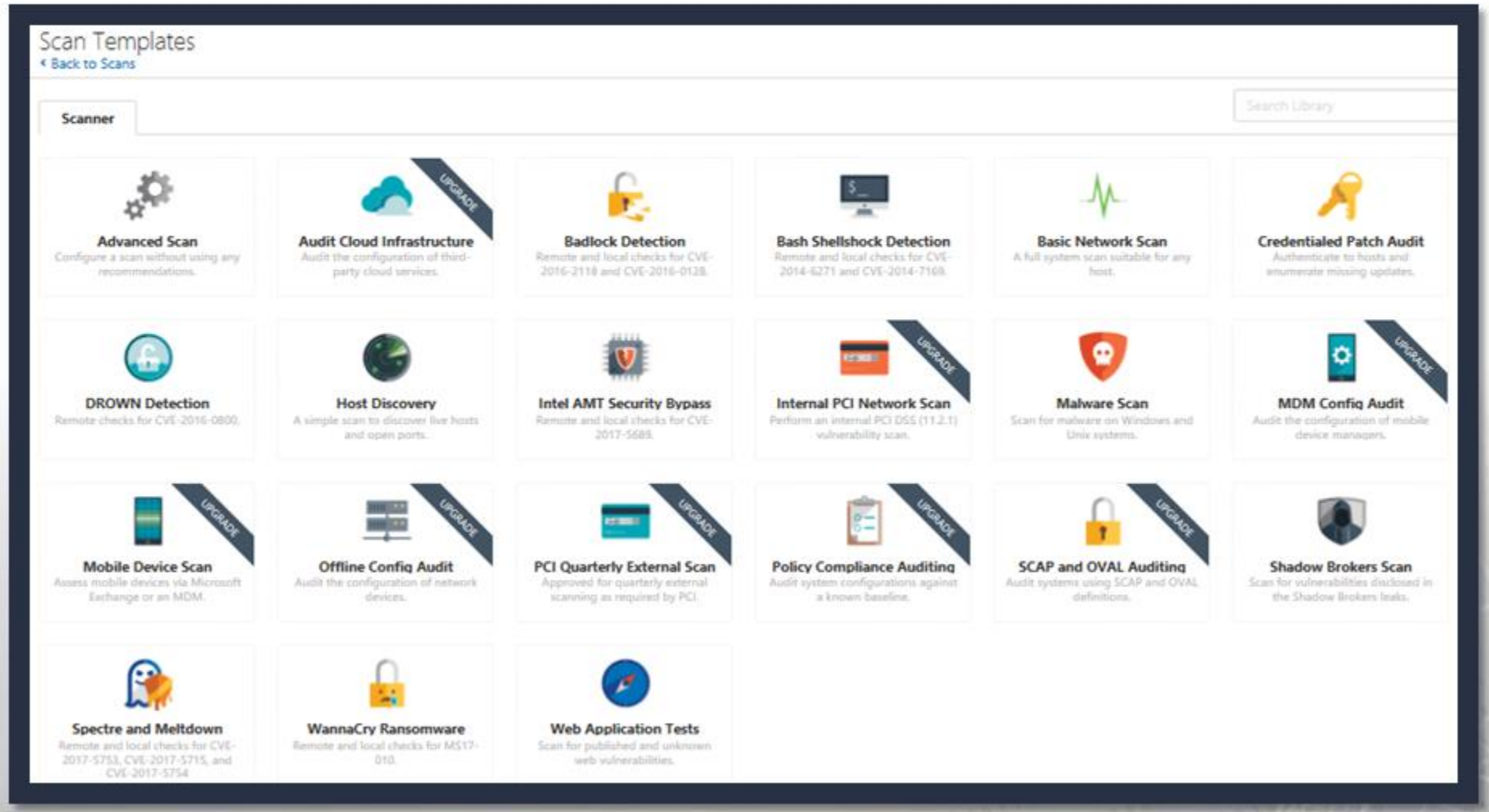
OpenVAS – Reviewing Scan Results



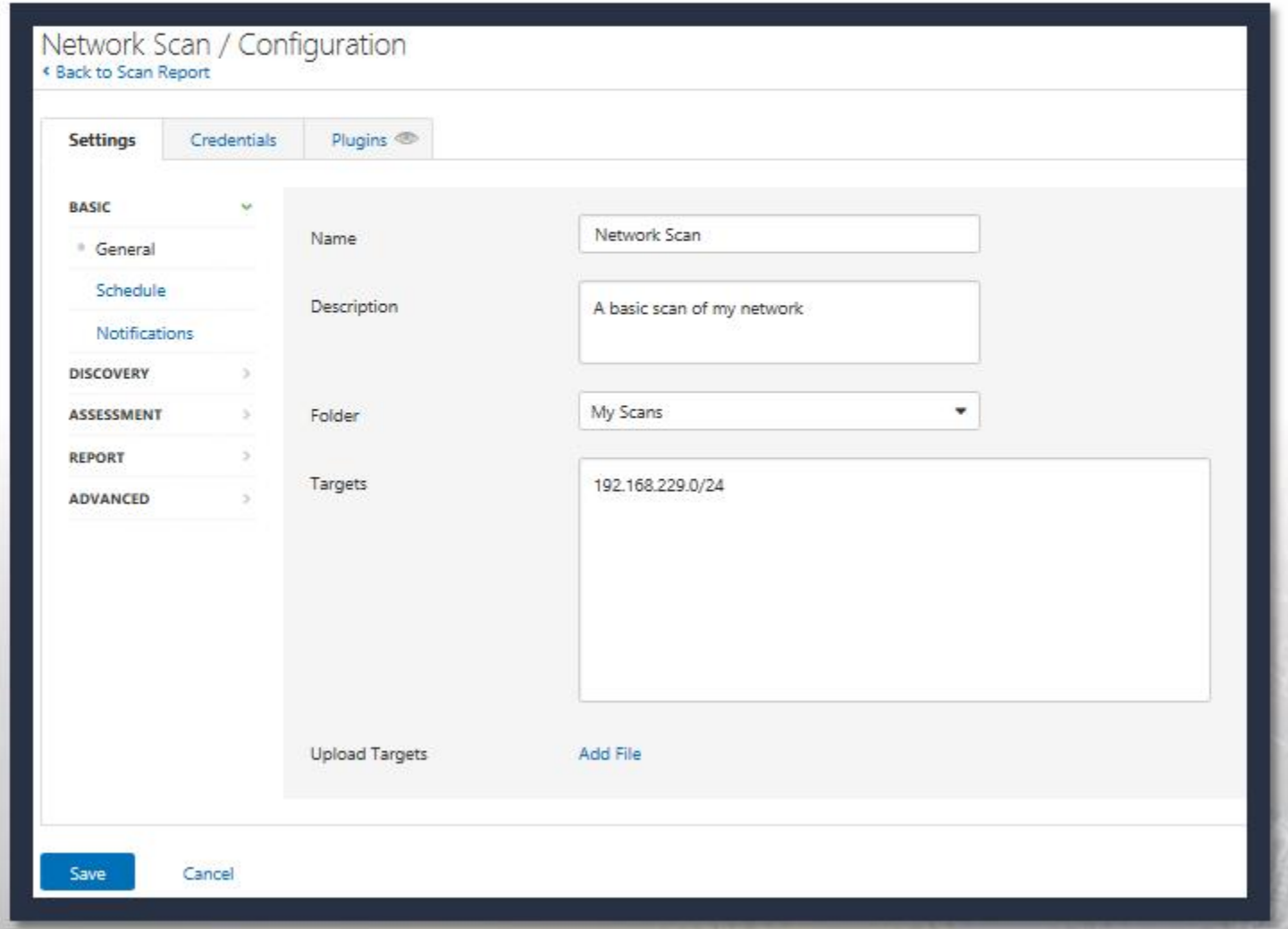
Nessus – Accessing



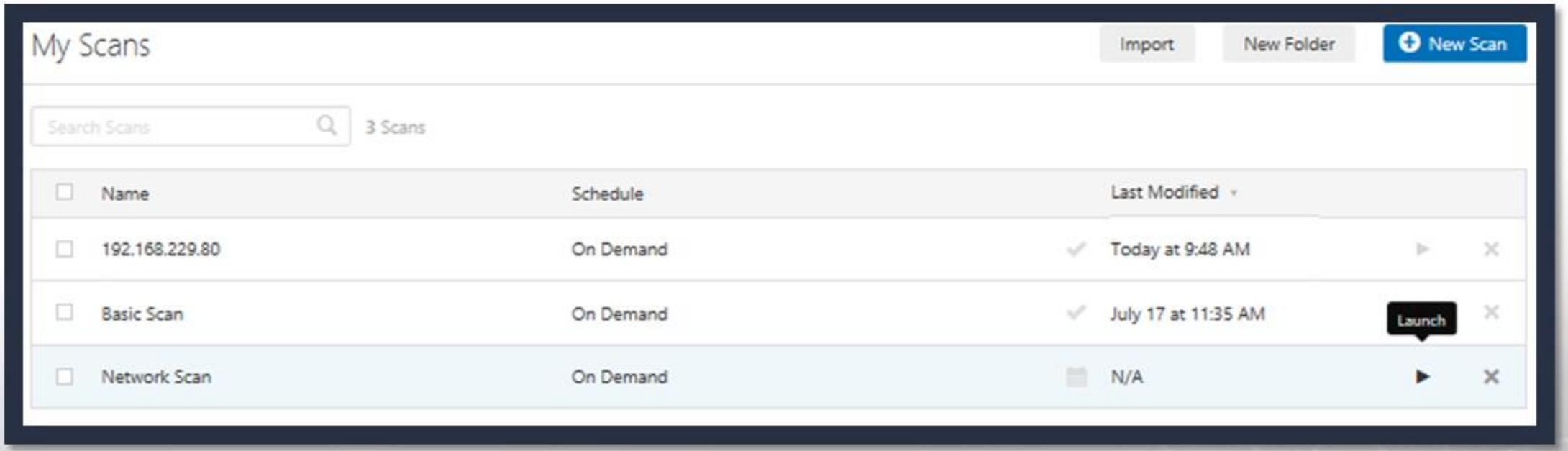
Nessus – Configuring a Scan



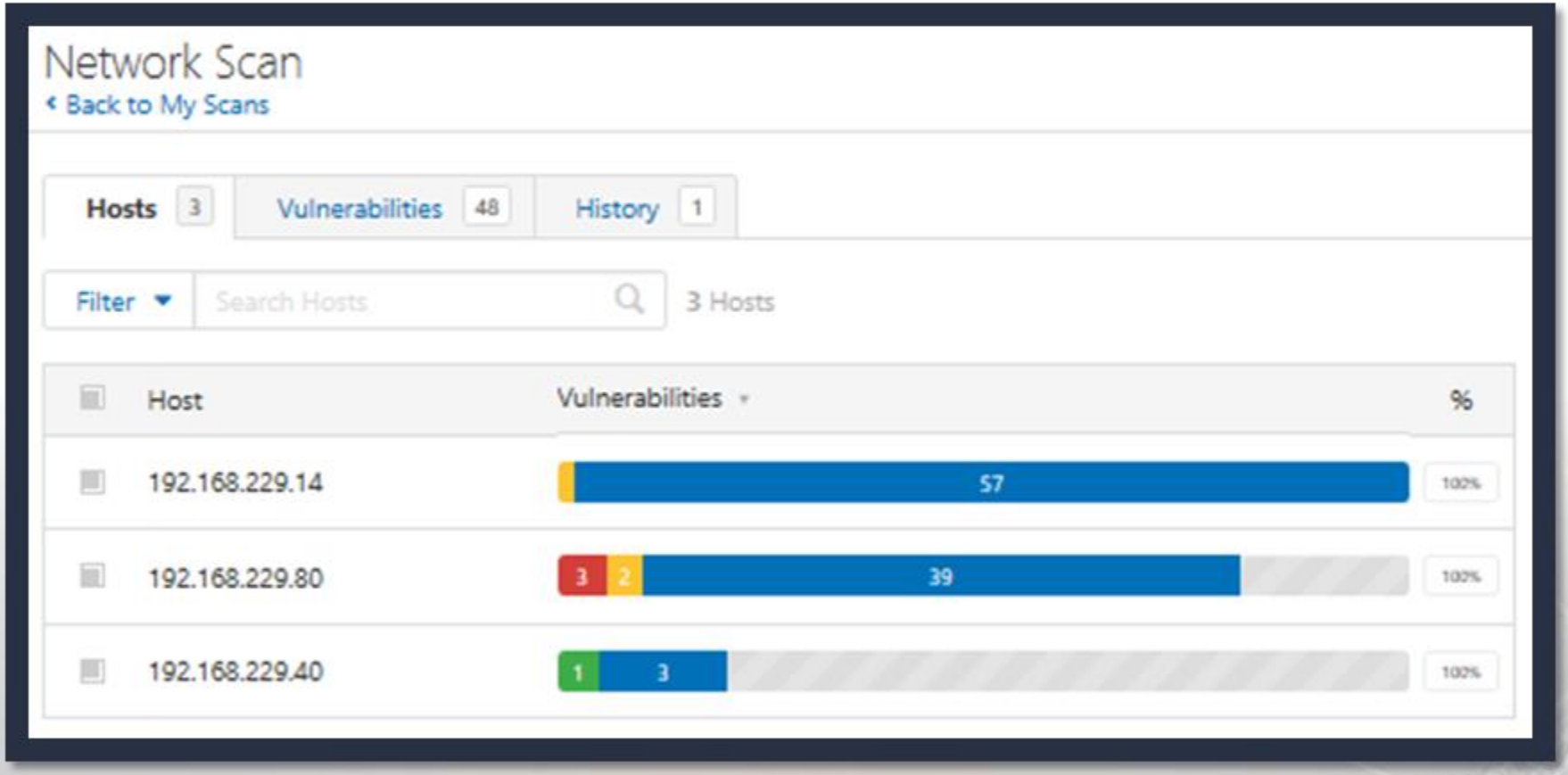
Nessus – Configuring a Scan



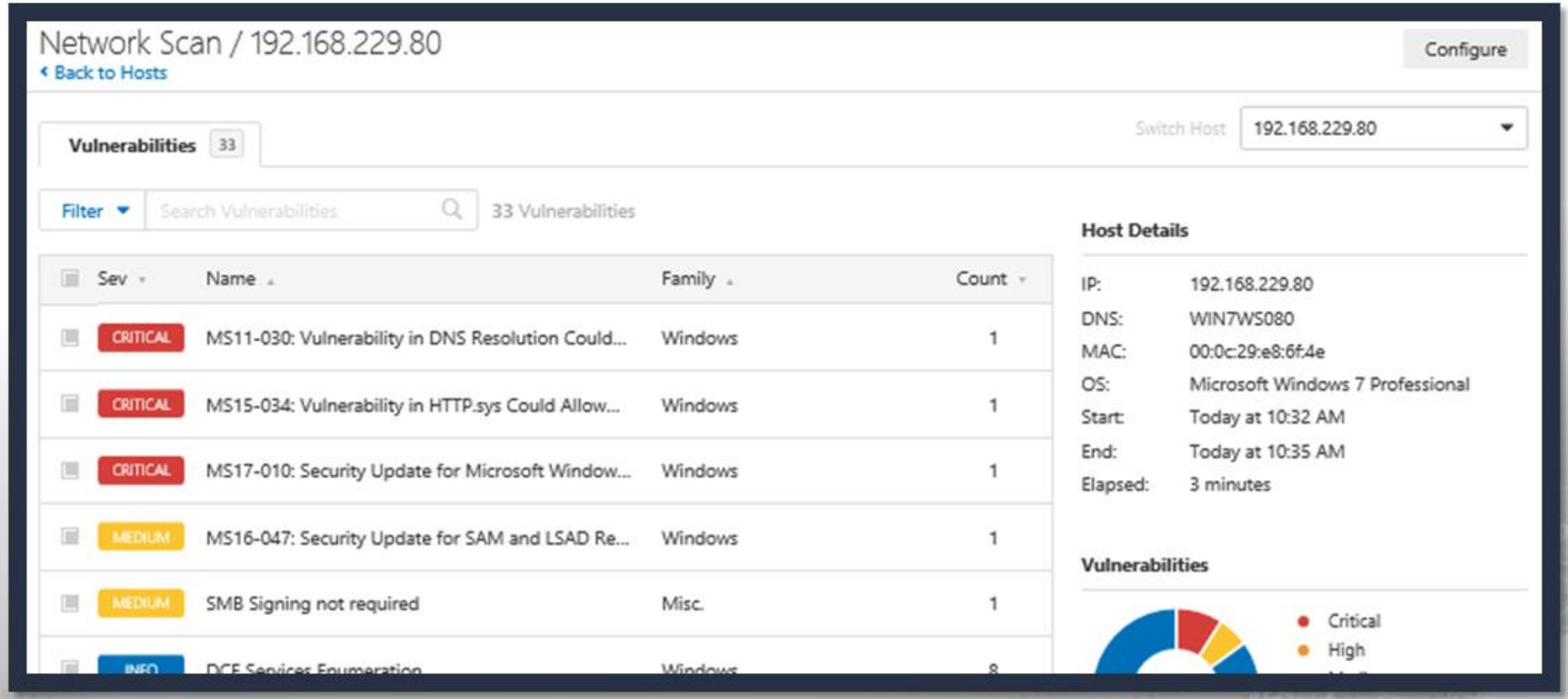
Nessus – Launching a Scan



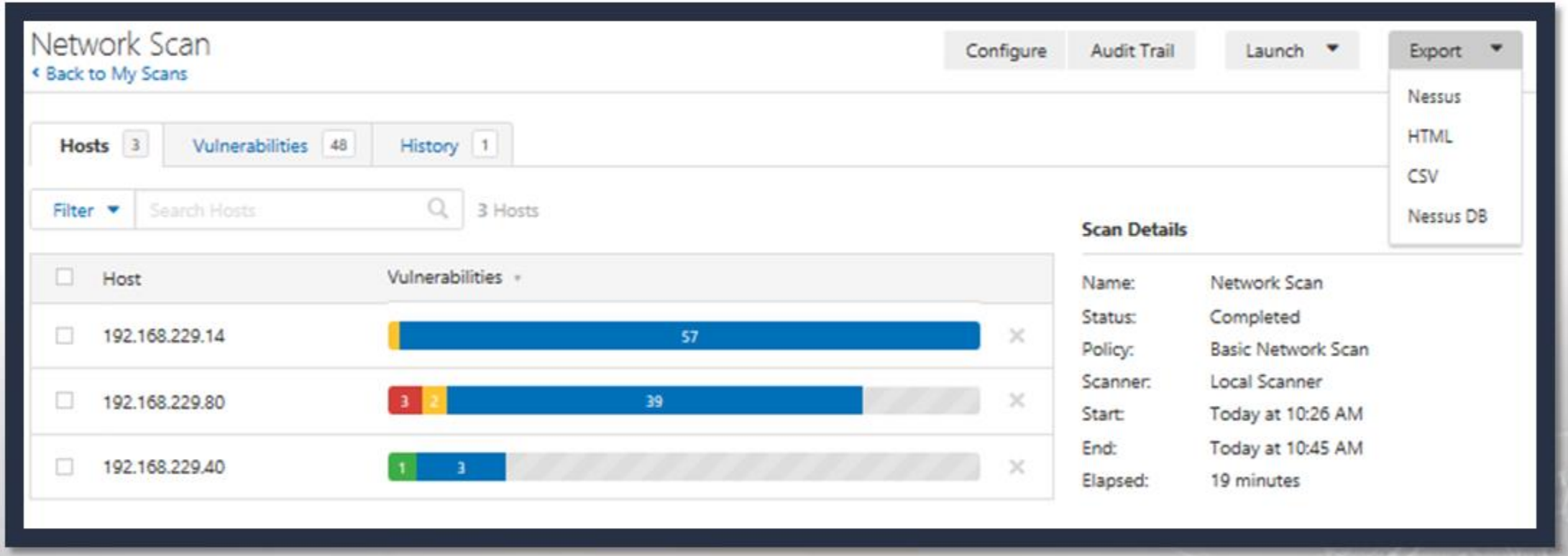
Nessus – Reviewing Scan Results



Nessus – Reviewing Scan Results



Nessus – Reviewing Scan Results



Exercise: Discovery Scanning & Enumerating Hosts

Objectives

After completing this exercise, students will be able to:

- Conduct active reconnaissance
- Develop mission reports from results of exploitation

Duration

This exercise will take approximately **2** hours to complete.

Debrief

Specific Questions

- What are some other ways in which Scapy or a similar tool could be used?

Debrief

General Questions

- How did you feel about this procedure?
- Were there any areas in particular where you had difficulty?
- Do you understand how this relates to the work you will be doing?

Exercise: Advanced Scanning & Evasion Techniques

Objectives

After completing this exercise, students will be able to:

- Conduct active reconnaissance
- Develop mission reports from results of exploitation

Duration

This exercise will take approximately **2.5** hours to complete.

Debrief

General Questions

- How did you feel about this procedure?
- Were there any areas in particular where you had difficulty?
- Do you understand how this relates to the work you will be doing?

Specific Questions

- Which other options did you use along with your scans and why did you choose to use them?

Exercise: Vulnerability Scanning

Objectives

After completing this exercise, students will be able to:

- Conduct active reconnaissance
- Develop mission reports from results of exploitation

Duration

This exercise will take approximately **1** hour to complete.

Debrief

General Questions

- How did you feel about this procedure?
- Were there any areas in particular where you had difficulty?
- Do you understand how this relates to the work you will be doing?

Specific Questions

- What is the difference between credentialed and non-credentialed scans?
- How do the Nessus and OpenVAS vulnerability scans compare to the results of previous port scans using Nmap?

Lesson Summary

In this lesson we learned about:

- Preparing custom packets for scanning
- Using Nmap scans to target ports, detect services, versions of operating systems and applications on a remote host and interpret the results
- Using available scripts to automate networking tasks (such as identifying vulnerabilities, testing controls and detecting backdoors)
- Using Nmap utilities (ncat, ndiff or nping) to analyze a network, generate network packets, connect to other hosts, or compare existing scans
- Performing suitable Nmap scans designed to evade the basic rules of firewalls or Intrusion detection systems
- Completing a vulnerability scan using Nessus and OpenVAS
- Assessing the vulnerability risks to a system and assemble results for inclusion in reporting

End of Module 2, Lesson 2