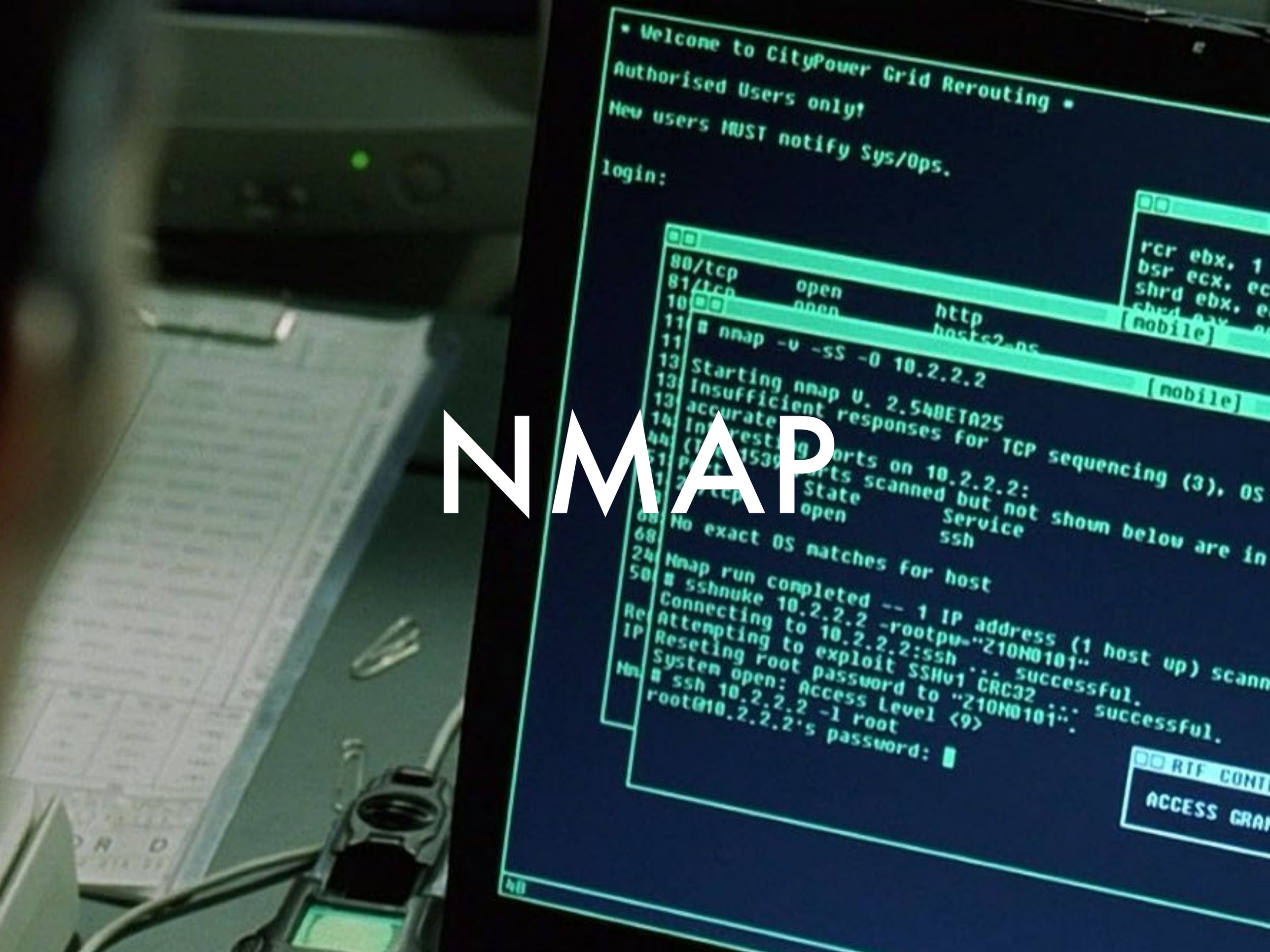
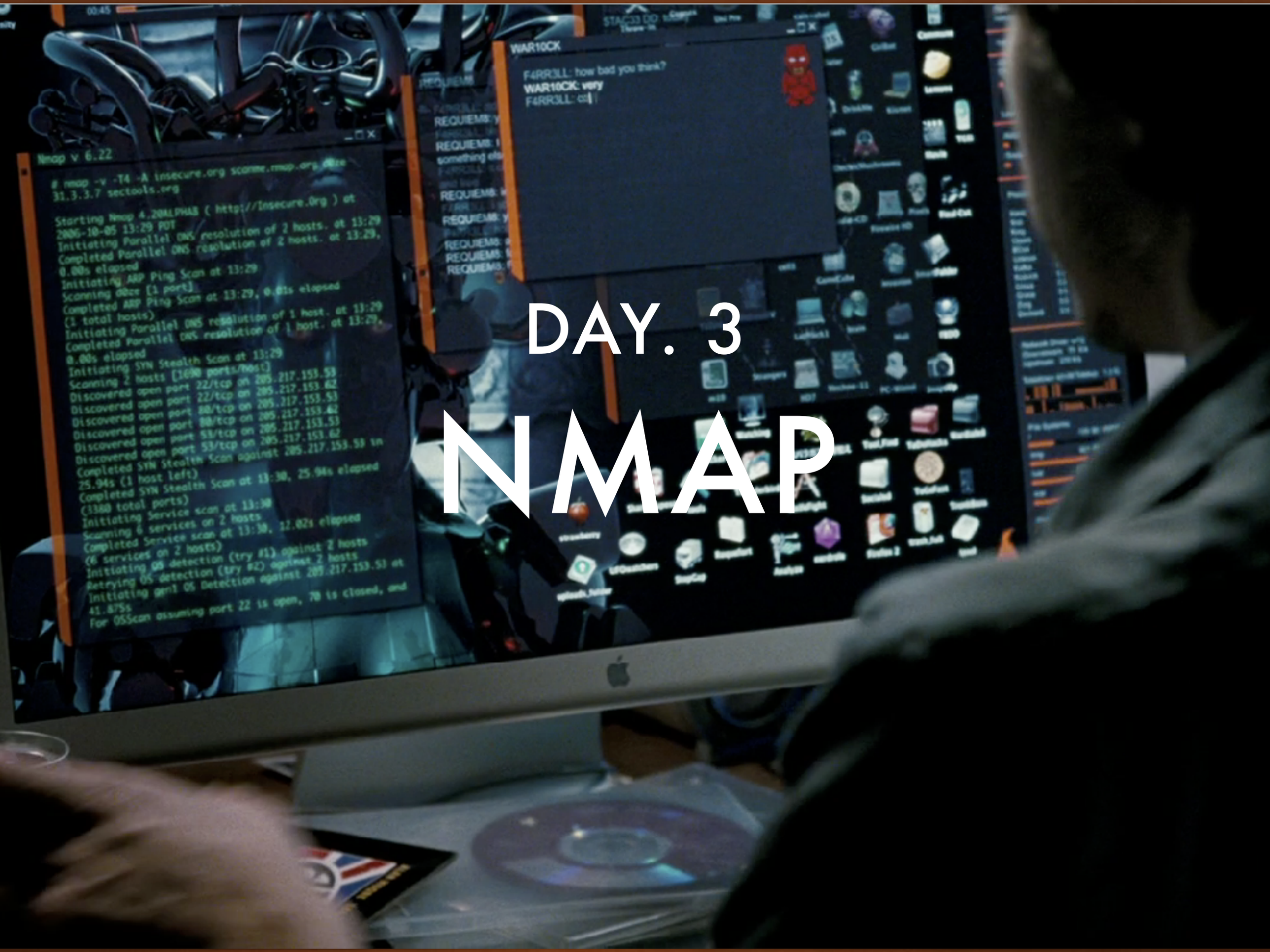
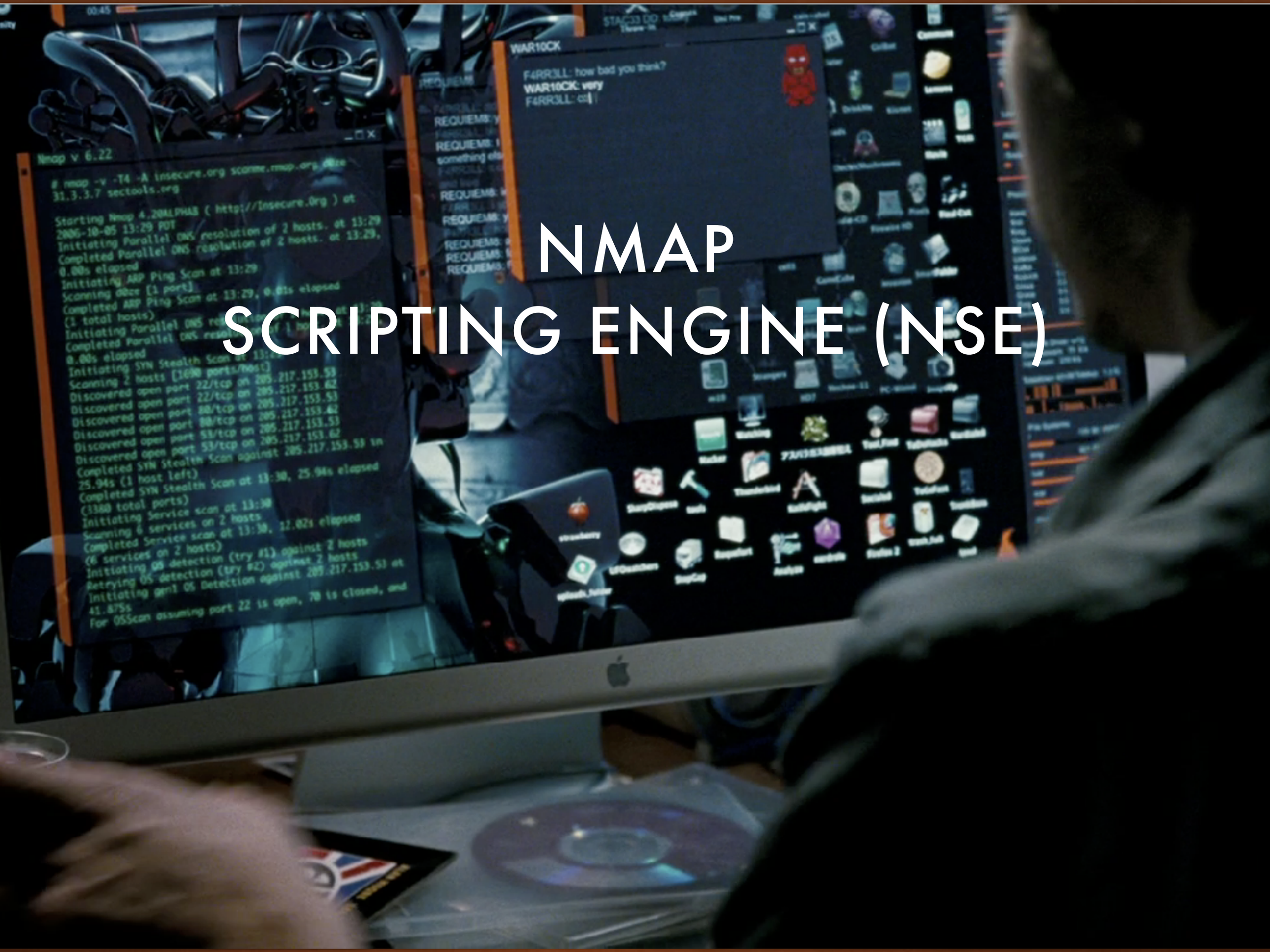

* Welcome to CityPower Grid Rerouting *
Authorised Users only!
New users MUST notify Sys/Ops.
login:
80/tcp open http
nmap -v -sS -O 10.2.2.2
Starting nmap V. 2.54BETA25
Insufficient responses for TCP sequencing (3), OS
accurate
Interesting ports on 10.2.2.2:
State Service
open ssh
No exact OS matches for host
Nmap run completed -- 1 IP address (1 host up) scann
sshnuke 10.2.2.2 -rootpw="Z1ON0101"
Connecting to 10.2.2.2:ssh ... successful.
Attempting to exploit SSHv1 CRC32
Reseting root password to "Z1ON0101".
System open: Access Level <9>
ssh 10.2.2.2 -l root
root@10.2.2.2's password:
ACCESS GRAN

NMAP


WAR10CK
F4RR3LL: how bad you think?
WAR10CK: very
F4RR3LL:
Nmap v 6.22
DAY. 3
NMAP

AGENDA DAY 3

- Nmap Scripting Engine (NSE)
  - Introduction
  - NSE Tasks
  - Script Categories
    - Usage and Examples
    - Develop Custom NSE
- Other Nmap bundle tools usage (Ncat, Ndiff, Nping)
- Zen map
  - Introduction
  - Usage and Example



NMAP SCRIPTING ENGINE (NSE)

NMAP SCRIPTING ENGINE

INTRODUCTION

- The Nmap Scripting Engine (NSE) is one of Nmap's most powerful and flexible features. It allows users to write (and share) simple scripts (using the Lua programming language) to automate a wide variety of networking tasks.

- The scripts are executed in parallel with the speed and efficiency you expect from Nmap. Users can rely on the growing and diverse set of scripts distributed with Nmap, or write their own to meet custom needs.

NMAP SCRIPTING ENGINE

INTRODUCTION

- NSE Scripts are written in the embedded Lua programming language.
- http://www.lua.org/
- Location in linux: /usr/share/nmap/scripts

NMAP SCRIPTING ENGINE

LOCATION

```
root@kali:~/Desktop/result# ls /usr/share/nmap/scripts/
acarsd-info.nse                         http-frontpage-login.nse                retobus-info.nse
address-info.nse                        http-general.nse                        retobus-version.nse
afp-brute.nse                           http-git.nse                            rexecbase-brute.nse
afp-ls.nse                              http-gitweb-projects-enum.nse           rfs-ls.nse
afp-path-vuln.nse                       http-google-malware.nse                 rfs-showmount.nse
afp-serverinfo.nse                      http-grep.nse                           rfs-statfs.nse
afp-showmount.nse                       http-headers.nse                        rje-node-brute.nse
ajp-auth.nse                            http-huawei-hg5xx-vuln.nse              rmi-dumpregistry.nse
ajp-brute.nse                           http-icloud-findmyiphone.nse            rmi-vuln-classloader.nse
ajp-headers.nse                         http-icloud-sendmsg.nse                 rpc-grind.nse
ajp-methods.nse                         http-iis-short-name-brute.nse           rpcap-info.nse
ajp-request.nse                         http-iis-webdav-vuln.nse                rpcap-brute.nse
allseeingeye-info.nse                   http-joomla-brute.nse                   rpcinfo.nse
amqp-info.nse                           http-litespeed-sourcecode-download.nse  rsa-vuln-roca.nse
asn-query.nse                           http-ls.nse                             rsync-brute.nse
auth-owners.nse                         http-majordomo2-dir-traversal.nse       rsync-list-modules.nse
auth-spoof.nse                          http-malware-host.nse                   rtsp-methods.nse
backorifice-brute.nse                   http-methods.nse                        rtsp-url-brute.nse
backorifice-info.nse                    http-method-tamper.nse                  s7-info.nse
bacnet-info.nse                         http-mobileversion-checker.nse          samba-vuln-cve-2012-1182.nse
banner.nse                              http-ntlm-info.nse                      sap-router-info.nse
bitcoin-getaddr.nse                     http-open-proxy.nse                     servicetags.nse
bitcoin-info.nse                        http-open-redirect.nse                  shodan-api.nse
bitcoinrpc-info.nse                     http-passwd.nse                         sip-brute.nse
bittorrent-discovery.nse                http-phpmyadmin-dir-traversal.nse       sip-call-spoof.nse
bjnp-discover.nse                       http-phpself-xss.nse                    sip-enum-users.nse
broadcast-ajp-discover.nse              http-php-version.nse                    sip-methods.nse
broadcast-avahi-dos.nse                 http-proxy-brute.nse                    skypev2-version.nse
broadcast-bjnp-discover.nse             http-put.nse                            smb-brute.nse
broadcast-db2-discover.nse              http-qnap-nas-info.nse                  smb-double-pulsar-backdoor.nse
broadcast-dhcp6-discover.nse            http-referer-checker.nse                smb-enum-domains.nse
broadcast-dhcp-discover.nse             http-rfi-spider.nse                     smb-enum-groups.nse
broadcast-dns-service-discovery.nse     http-robots.txt.nse                     smb-enum-processes.nse
broadcast-dropbox-listener.nse          http-robtex-reverse-ip.nse              smb-enum-services.nse
broadcast-eigrp-discovery.nse           http-robtex-shared-ns.nse               smb-enum-sessions.nse
broadcast-igmp-discovery.nse            http-server-header.nse                  smb-enum-shares.nse
broadcast-listener.nse                  http-shellshock.nse                     smb-ls.nse
```

NMAP SCRIPTING ENGINE

SCRIPT EXAMPLE

```
root@kali:/usr/share/nmap/scripts# cat smtp-strangeport.nse
description = [[
Checks if SMTP is running on a non-standard port.

This may indicate that crackers or script kiddies have set up a backdoor on the
system to send spam or control the machine.
]]

-- @output
-- 22/tcp  open   smtp
--   smtp-strangeport: Mail server on unusual port: possible malware

author = "Diman Todorov"

license = "Same as Nmap--See https://nmap.org/book/man-legal.html"

categories = {"malware", "safe"}

portrule = function(host, port)
  return port.service == 'smtp" and
    port.number ~= 25 and port.number ~= 465 and port.number ~= 587
    and port.protocol == 'tcp"
    and port.state == "open'
end

action = function()
  return 'Mail server on unusual port: possible malware"
end

root@kali:/usr/share/nmap/scripts#
```

NMAP SCRIPTING ENGINE

NSE TASKS

NMAP SCRIPTING ENGINE

NSE TASKS

- Network discovery

This is Nmap's bread and butter. Examples include looking up whois data based on the target domain, querying ARIN, RIPE, or APNIC for the target IP to determine ownership, performing identd lookups on open ports, SNMP queries, and listing available NFS/SMB/RPC shares and services.

- More sophisticated version detection

Neither of these tasks are well suited to traditional Nmap version detection, but both are easily accomplished with NSE. For these reasons, version detection now calls NSE by default to handle some tricky services.

- Vulnerability detection

When a new vulnerability is discovered, you often want to scan your networks quickly to identify vulnerable systems before the bad guys do. While Nmap isn't a comprehensive vulnerability scanner, NSE is powerful enough to handle even demanding vulnerability checks. Many vulnerability detection scripts are already available and we plan to distribute more as they are written.

NMAP SCRIPTING ENGINE

NSE TASKS

https://nmap.org/nsedoc/categories/discovery.html

Index
NSE Documentation

Categories

auth
broadcast
brute
default
discovery
dos
exploit
external
fuzzer
intrusive
malware
safe
version
vuln

Scripts (show 534)
Libraries (show 120)

Scripts

acarsd-info	Retrieves information from a listening acarsd daemon. Acarsd decodes ACARS (Aircraft Communication Addressing and Reporting System) data in real time. The information retrieved by this script includes the daemon version, API version, administrator e-mail address and listening frequency.
afp-ls	Attempts to get useful information about files from AFP volumes. The output is intended to resemble the output of ls.
afp-serverinfo	Shows AFP server information. This information includes the server's hostname, IPv4 and IPv6 addresses, and hardware type (for example Macmini or MacBookPro).
afp-showmount	Shows AFP shares and ACLs.
ajp-headers	Performs a HEAD or GET request against either the root directory or any optional directory of an Apache JServ Protocol server and returns the server response headers.
ajp-request	Requests a URI over the Apache JServ Protocol and displays the result (or stores it in a file). Different AJP methods such as; GET, HEAD, TRACE, PUT or DELETE may be used.
allseeingeye-info	Detects the All-Seeing Eye service. Provided by some game servers for querying the server's status.
amqp-info	Gathers information (a list of all server properties) from an AMQP (advanced message queuing protocol) server.
asn-query	Maps IP addresses to autonomous system (AS) numbers.
backorifice-info	Connects to a BackOrifice service and gathers information about the host and the BackOrifice service itself.
bacnet-info	Discovers and enumerates BACNet Devices collects device information based off standard requests. In some cases, devices may not strictly follow the specifications, or may comply with older versions of the specifications, and will result in a BACNET error response. Presence of this error positively identifies the device as a BACNet device, but no enumeration is possible.
banner	A simple banner grabber which connects to an open TCP port and prints out anything sent by the listening service within five seconds.
bitcoin-getaddr	Queries a Bitcoin server for a list of known Bitcoin nodes

NMAP SCRIPTING ENGINE

NSE TASKS

- Backdoor detection

Many attackers and some automated worms leave backdoors to enable later reentry. Some of these can be detected by Nmap's regular expression based version detection. For example, within hours of the MyDoom worm hitting the Internet, Jay Moran posted an Nmap version detection probe and signature so that others could quickly scan their networks for MyDoom infections. NSE is needed to reliably detect more complex worms and backdoors.

- Vulnerability exploitation

As a general scripting language, NSE can even be used to exploit vulnerabilities rather than just find them. The capability to add custom exploit scripts may be valuable for some people (particularly penetration testers), though we aren't planning to turn Nmap into an exploitation framework such as Metasploit.

NMAP SCRIPTING ENGINE

NSE TASKS

https://nmap.org/nsedoc/categories/exploit.html

NSEDoc

Index
NSE Documentation

Categories

auth
broadcast
brute
default
discovery
dos
exploit
external
fuzzer
intrusive
malware
safe
version
vuln

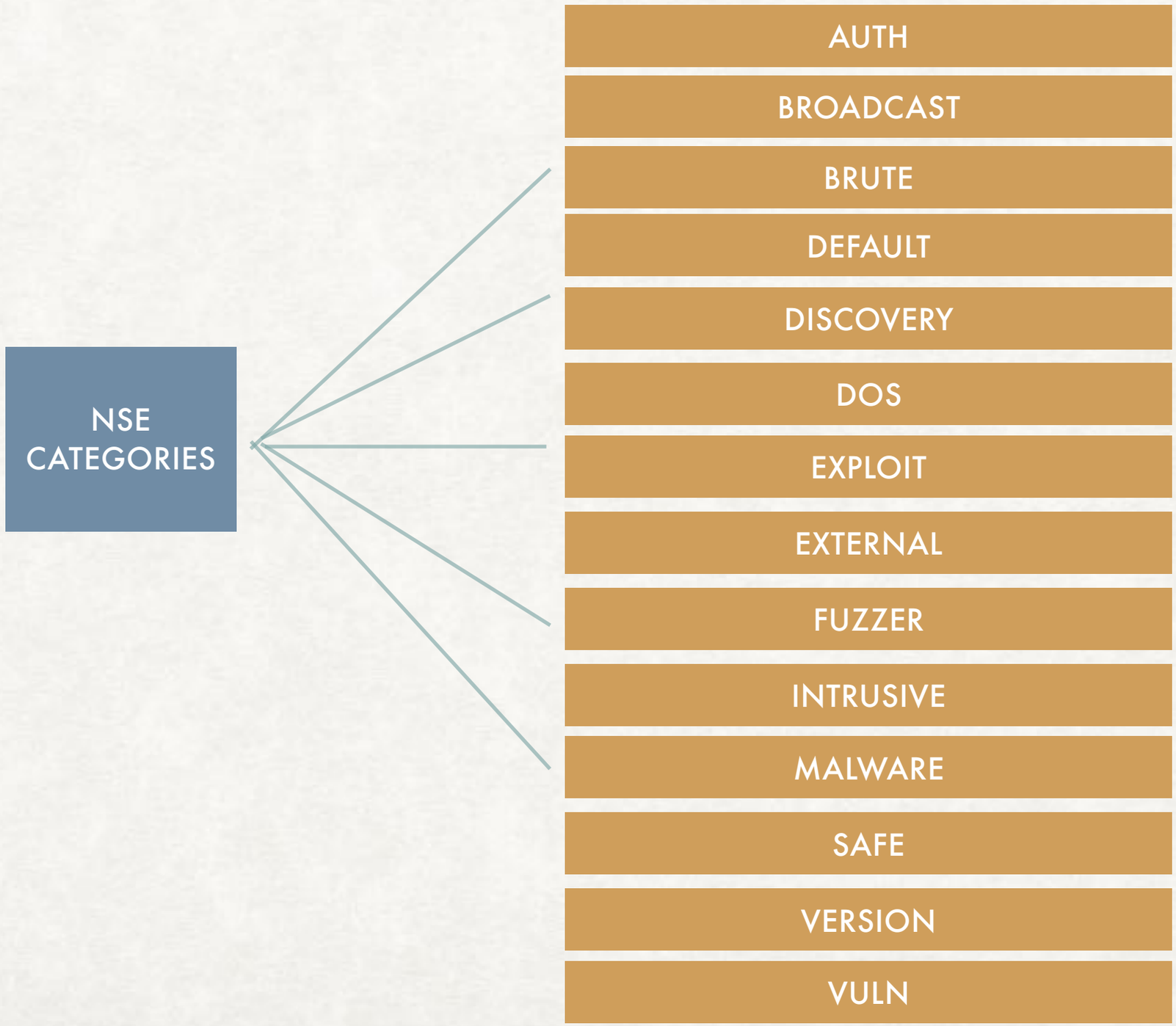
Scripts (show 534)
Libraries (show 120)

Scripts

afp-path-vuln	Detects the Mac OS X AFP directory traversal vulnerability, CVE-2010-0533.
clamav-exec	Exploits ClamAV servers vulnerable to unauthenticated clamav comand execution.
distcc-cve2004-2687	Detects and exploits a remote code execution vulnerability in the distributed compiler daemon distcc. The vulnerability was disclosed in 2002, but is still present in modern implementation due to poor configuration of the service.
ftp-proftpd-backdoor	Tests for the presence of the ProFTPD 1.3.3c backdoor reported as OSVDB-ID 69562. This script attempts to exploit the backdoor using the innocuous `id` command by default, but that can be changed with the `ftp-proftpd-backdoor.cmd` script argument.
ftp-vsftpd-backdoor	Tests for the presence of the vsFTPd 2.3.4 backdoor reported on 2011-07-04 (CVE-2011-2523). This script attempts to exploit the backdoor using the innocuous `id` command by default, but that can be changed with the `exploit.cmd` or `ftp-vsftpd-backdoor.cmd` script arguments.
http-adobe-coldfusion-apsa1301	Attempts to exploit an authentication bypass vulnerability in Adobe Coldfusion servers to retrieve a valid administrator's session cookie.
http-avaya-ipoffice-users	Attempts to enumerate users in Avaya IP Office systems 7.x.
http-awstatstotals-exec	Exploits a remote code execution vulnerability in Awstats Totals 1.0 up to 1.14 and possibly other products based on it (CVE: 2008-3922).
http-axis2-dir-traversal	Exploits a directory traversal vulnerability in Apache Axis2 version 1.4.1 by sending a specially crafted request to the parameter `xsd` (OSVDB-59001). By default it will try to retrieve the configuration file of the Axis2 service `'/conf/axis2.xml'` using the path `'/axis2/services/'` to return the username and password of the admin account.
http-barracuda-dir-traversal	Attempts to retrieve the configuration settings from a Barracuda Networks Spam & Virus Firewall device using the directory traversal vulnerability described at http://seclists.org/fulldisclosure/2010/Oct/119.

NMAP SCRIPTING ENGINE

NSE CATEGORIES



NMAP SCRIPTING ENGINE

SCRIPT CATEGORIES

- auth

These scripts deal with authentication credentials (or bypassing them) on the target system. Examples include x11-access, ftp-anon, and oracle-enum-users. Scripts which use brute force attacks to determine credentials are placed in the brute category instead.

- broadcast

Scripts in this category typically do discovery of hosts not listed on the command line by broadcasting on the local network. Use the newtargets script argument to allow these scripts to automatically add the hosts they discover to the Nmap scanning queue.

- brute

These scripts use brute force attacks to guess authentication credentials of a remote server. Nmap contains scripts for brute forcing dozens of protocols, including http-brute, oracle-brute, snmp-brute, etc.

NMAP SCRIPTING ENGINE

SCRIPT CATEGORIES

- default

These scripts are the default set and are run when using the -sC or -A options rather than listing scripts with --script. This category can also be specified explicitly like any other using --script=default.

- Many factors are considered in deciding whether a script should be run by default:

  Speed, Usefulness, Verbosity, Reliability, Intrusiveness, Privacy

NMAP SCRIPTING ENGINE

SCRIPT CATEGORIES

- discovery

These scripts try to actively discover more about the network by querying public registries, SNMP-enabled devices, directory services, and the like. Examples include html-title (obtains the title of the root path of web sites), smb-enum-shares (enumerates Windows shares), and snmp-sysdescr (extracts system details via SNMP).

- dos

Scripts in this category may cause a denial of service. Sometimes this is done to test vulnerability to a denial of service method, but more commonly it is an undesired by necessary side effect of testing for a traditional vulnerability. These tests sometimes crash vulnerable services.

- exploit

These scripts aim to actively exploit some vulnerability.

NMAP SCRIPTING ENGINE

SCRIPT CATEGORIES

- external

Scripts in this category may send data to a third-party database or other network resource. An example of this is whois, which makes a connection to whois servers to learn about the address of the target. There is always the possibility that operators of the third-party database will record anything you send to them, which in many cases will include your IP address and the address of the target. Most scripts involve traffic strictly between the scanning computer and the client; any that do not are placed in this category.

- fuzzer

This category contains scripts which are designed to send server software unexpected or randomized fields in each packet. While this technique can useful for finding undiscovered bugs and vulnerabilities in software, it is both a slow process and bandwidth intensive. An example of a script in this category is dns-fuzz, which bombards a DNS server with slightly flawed domain requests until either the server crashes or a user specified time limit elapses.

NMAP SCRIPTING ENGINE

SCRIPT CATEGORIES

- intrusive

These are scripts that cannot be classified in the safe category because the risks are too high that they will crash the target system, use up significant resources on the target host (such as bandwidth or CPU time), or otherwise be perceived as malicious by the target's system administrators. Examples are http-open-proxy (which attempts to use the target server as an HTTP proxy) and snmp-brute (which tries to guess a device's SNMP community string by sending common values such as public, private, and cisco). Unless a script is in the special version category, it should be categorized as either safe or intrusive.

- malware

These scripts test whether the target platform is infected by malware or backdoors. Examples include smtp-strangeport, which watches for SMTP servers running on unusual port numbers, and auth-spoof, which detects identd spoofing daemons which provide a fake answer before even receiving a query. Both of these behaviors are commonly associated with malware infections.

NMAP SCRIPTING ENGINE

SCRIPT CATEGORIES

- safe

Scripts which weren't designed to crash services, use large amounts of network bandwidth or other resources, or exploit security holes are categorized as safe. These are less likely to offend remote administrators, though (as with all other Nmap features) we cannot guarantee that they won't ever cause adverse reactions. Most of these perform general network discovery. Examples are ssh-hostkey (retrieves an SSH host key) and html-title (grabs the title from a web page). Scripts in the version category are not categorized by safety, but any other scripts which aren't in safe should be placed in intrusive.

- version

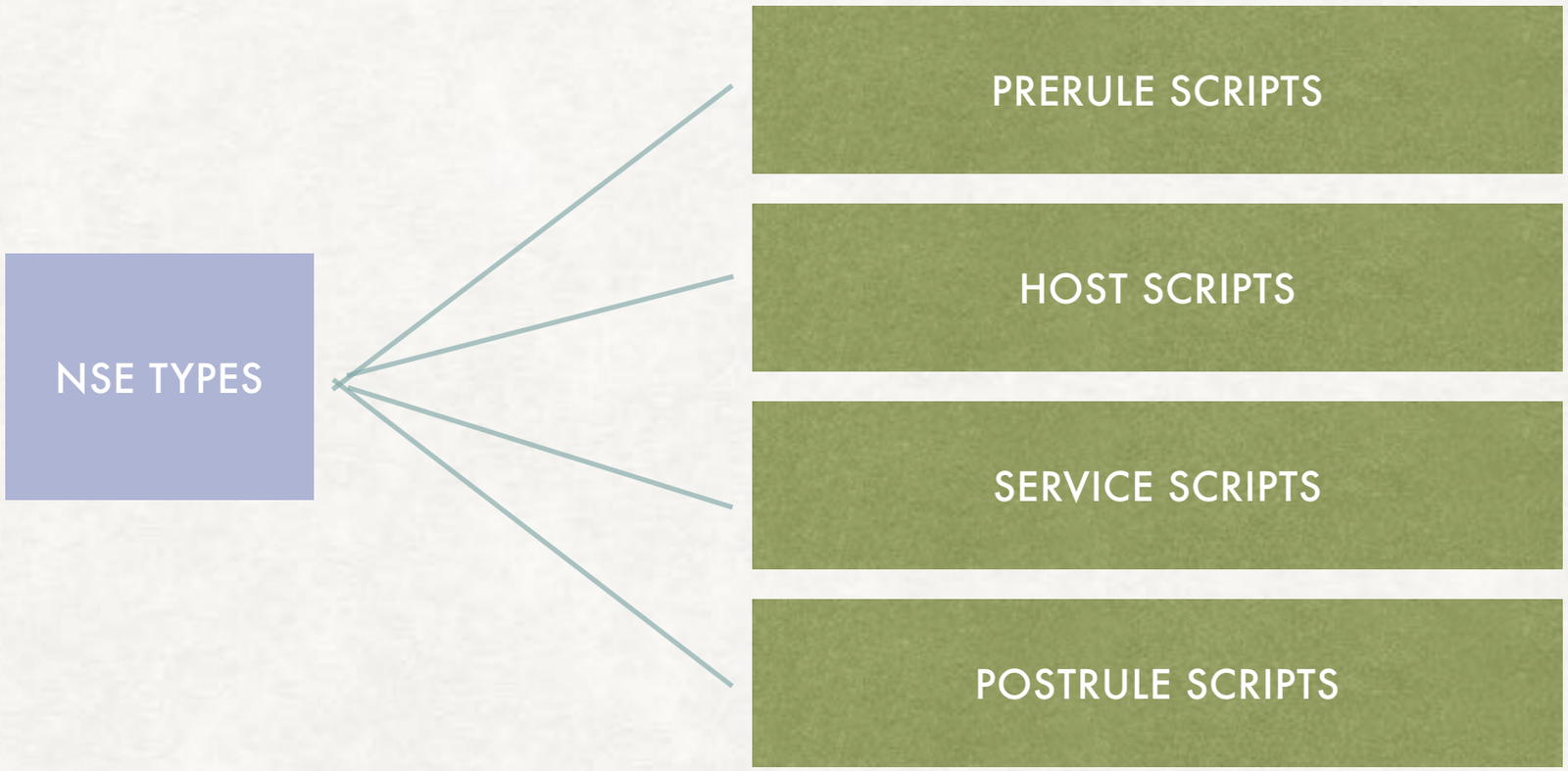
The scripts in this special category are an extension to the version detection feature and cannot be selected explicitly. They are selected to run only if version detection (-sV) was requested. Their output cannot be distinguished from version detection output and they do not produce service or host script results. Examples are skypev2-version, pptp-version, and iax2-version.

- vuln

These scripts check for specific known vulnerabilities and generally only report results if they are found. Examples include realvnc-auth-bypass and afp-path-vuln.

NMAP SCRIPTING ENGINE

NSE TYPES



NMAP SCRIPTING ENGINE

SCRIPT TYPES AND PHASES

- Prerule scripts

These scripts run before any of Nmap's scan phases, so Nmap has not collected any information about its targets yet. They can be useful for tasks which don't depend on specific scan targets, such as performing network broadcast requests to query DHCP and DNS SD servers. Some of these scripts can generate new targets for Nmap to scan (only if you specify the newtargets NSE argument). For example, dns-zone-transfer can obtain a list of IPs in a domain using a zone transfer request and then automatically add them to Nmap's scan target list. Prerule scripts can be identified by containing a prerule function (see the section called “Rules”).

- Host scripts

Scripts in this phase run during Nmap's normal scanning process after Nmap has performed host discovery, port scanning, version detection, and OS detection against the target host. This type of script is invoked once against each target host which matches its hostrule function. Examples are whois, which looks up ownership information for a target IP, and path-mtu which tries to determine the maximum IP packet size which can reach the target without requiring fragmentation.

NMAP SCRIPTING ENGINE

SCRIPT TYPES AND PHASES

- Service scripts

These scripts run against specific services listening on a target host. For example, Nmap includes more than 15 http service scripts to run against web servers. If a host has web servers running on multiple ports, those scripts may run multiple times (one for each port). These are the most commong Nmap script type, and they are distinguished by containing a portrule function for deciding which detected services a script should run against.

- Postrule scripts

These scripts run after Nmap has scanned all of its targets. They can be useful for formatting and presenting Nmap output. For example, ssh-hostkey is best known for its service (portrule) script which connects to SSH servers, discovers their public keys, and prints them. But it also includes a postrule which checks for duplicate keys amongst all of the hosts scanned, then prints any that are found. Another potential use for a postrule script is printing a reverse-index of the Nmap output—showing which hosts run a particular service rather than just listing the services on each host. Postrule scripts are identified by containing a postrule function.

NMAP SCRIPTING ENGINE

USAGE

- Simply specify **-sC** to enable the most common scripts.
- Performs a script scan using the default set of scripts. It is equivalent to --script=default. Some of the scripts in this category are considered intrusive and should not be run against a target network without permission.

NMAP SCRIPTING ENGINE

USAGE

```
root@kali:~/Desktop/result# nmap  T4  sC  p445 192.168.2.103

Starting Nmap 7.12 ( https://nmap.org ) at 2016-08-07 06:37 EDT
Nmap scan report for 192.168.2.103
Host is up (0.00051s latency).
PORT    STATE SERVICE
445/tcp open  microsoft-ds
MAC Address: 00:0C:29:83:ED:69 (VMware)

Host script results:
| nbstat: NetBIOS name: WIN-RE1NUHRDCNW, NetBIOS user: <unknown>, NetBIOS MAC: 00:0c:29:83:ec:69 (VMware)
| smb-os-discovery:
|   OS: Windows 7 Ultimate 7601 Service Pack 1 (Windows 7 Ultimate 6.1)
|   OS CPE: cpe:/o:microsoft:windows_7::sp1
|   Computer name: WIN-RE1N_HRDCNW
|   NetBIOS computer name: WIN-RE1NUHRDCNW
|   Workgroup: WORKGROUP
|_  System time: 2016-08-07T17:37:20+07:00
| smb-security-mode:
|   account_used: guest
|   authentication_level: user
|   challenge_response: supported
|_  message_signing: disabled (dangerous, but default)
|_smbv2-enabled: Server supports SMBv2 protocol

Nmap done: 1 IP address (1 host up) scanned in 18.91 seconds
root@kali:~/Desktop/result#
```

NMAP SCRIPTING ENGINE

USAGE

- Script scanning is normally done in combination with a port scan, because scripts may be run or not run depending on the port states found by the scan.
- To run a script scan with neither a host discovery nor a port scan, use the -Pn -sn options together with -sC or --script.
- Scripts are not run in a sandbox and thus could accidentally or maliciously damage your system or invade your privacy.
- Never run scripts from third parties unless you trust the authors or have carefully audited the scripts yourself.

NMAP SCRIPTING ENGINE

USAGE

```
root@kali:~/Desktop/result# nmap  T4  sC  p445 192.168.2.103

Starting Nmap 7.12 ( https://nmap.org ) at 2016-08-07 06:37 EDT
Nmap scan report for 192.168.2.103
Host is up (0.00051s latency).
PORT    STATE SERVICE
445/tcp open  microsoft-ds
MAC Address: 00:0C:29:83:ED:69 (VMware)

Host script results:
| nbstat: NetBIOS name: WIN-RE1NUHRDCNW, NetBIOS user: <unknown>, NetBIOS MAC: 00:0c:29:83:ec:69 (VMware)
| smb-os-discovery:
|   OS: Windows 7 Ultimate 7601 Service Pack 1 (Windows 7 Ultimate 6.1)
|   OS CPE: cpe:/o:microsoft:windows_7::sp1
|   Computer name: WIN-RE1N_HRDCNW
|   NetBIOS computer name: WIN-RE1NUHRDCNW
|   Workgroup: WORKGROUP
|_  System time: 2016-08-07T17:37:20+07:00
| smb-security-mode:
|   account_used: guest
|   authentication_level: user
|   challenge_response: supported
|_  message_signing: disabled (dangerous, but default)
|_smbv2-enabled: Server supports SMBv2 protocol

Nmap done: 1 IP address (1 host up) scanned in 18.91 seconds
root@kali:~/Desktop/result#
```

NMAP SCRIPTING ENGINE

USAGE

- --script <filename>|<category>|<directory>|<expression>[,...]

Runs a script scan using the comma-separated list of filenames, script categories, and directories. Each element in the list may also be a Boolean expression describing a more complex set of scripts. Each element is interpreted first as an expression, then as a category, and finally as a file or directory name. The special argument all makes every script in Nmap's script database eligible to run. The all argument should be used with caution as NSE may contain dangerous scripts including exploits, brute force authentication crackers, and denial of service attacks.

NMAP SCRIPTING ENGINE

USAGE

- nmap --script default,safe

Loads all scripts in the default and safe categories.

- nmap --script smb-os-discovery

Loads only the smb-os-discovery script. Note that the .nse extension is optional.

NMAP SCRIPTING ENGINE

USAGE

```
root@kali:/usr/share/nmap/scripts# nmap   script default,safe  Pn 192.168.2.103

Starting Nmap 7.12 ( https://nmap.org ) at 2016-08-07 08:55 IDT
Pre-scan script results:
  broadcast-dhcp-discover:
    Response 1 of 1:
      IP Offered: 192.168.2.108
      Server Identifier: 192.168.2.1
      Subnet Mask: 255.255.255.0
      Router: 192.168.2.1
_     Domain Name Server: 192.168.2.1
  broadcast-dns-service-discovery:
    192.168.2.103
      22/tcp ssh
        model=MacBookPro9,2
        osxvers=15
        Address=192.168.2.100 fe80:0:0:0:22c9:d0ff:fedb:939f
    22/tcp sftp-ssh
        model=MacBookPro9,2
        osxvers=15
        Address=192.168.2.100 fe80:0:0:0:22c9:d0ff:fedb:939f
    445/tcp smb
        model=MacBookPro9,2
        osxvers=15
        Address=192.168.2.100 fe80:0:0:0:22c9:d0ff:fedb:939f
    548/tcp afpovertcp
        model=MacBookPro9,2
        osxvers=15
        Address=192.168.2.100 fe80:0:0:0:22c9:d0ff:fedb:939f
    49421/tcp XeynoteControl
        name=raiser
        type=MacBookPro9,2
        Address=192.168.2.100 fe80:0:0:0:22c9:d0ff:fedb:939f
  192.168.2.104
    9/tcp workstation
      Address=192.168.2.104 fe80:0:0:0:20c:29ff:fe6e:8883
    22/tcp udisks-ssh
      Address=192.168.2.104 fe80:0:0:0:20c:29ff:fe6e:8883
```

NMAP SCRIPTING ENGINE

USAGE

```
root@kali:/usr/share/nmap/scripts# nmap --script smb-os-discovery -p445 -Pn 192.168.2.103

Starting Nmap 7.12 ( https://nmap.org ) at 2016-08-07 09:26 EDT
Nmap scan report for 192.168.2.103
Host is up (0.00069s latency).
PORT    STATE SERVICE
445/tcp open  microsoft-ds
MAC Address: 00:0C:29:83:ED:69 (VMware)

Host script results:
| smb-os-discovery:
|   OS: Windows 7 Ultimate 7601 Service Pack 1 (Windows 7 Ultimate 6.1)
|   OS CPE: cpe:/o:microsoft:windows_7::sp1
|   Computer name: WIN-RE1NUHRDONW
|   NetBIOS computer name: WIN-RE1NUHRDONW
|   Workgroup: WORKGROUP
|_  System time: 2016-08-07T20:26:52+07:00

Nmap done: 1 IP address (1 host up) scanned in 13.25 seconds
root@kali:/usr/share/nmap/scripts#
```

NMAP SCRIPTING ENGINE

USAGE

- nmap --script default,banner,/home/user/customscripts

Loads the script in the default category, the banner script, and all .nse files in the directory /home/user/customscripts.

- When referring to scripts from script.db by name, you can use a shell-style ‘*’ wildcard.

- nmap --script "http-*"

Loads all scripts whose name starts with http-, such as http-auth and http-open-proxy. The argument to --script had to be in quotes to protect the wildcard from the shell.

NMAP SCRIPTING ENGINE

USAGE

```
root@kali:/usr/share/nmap/scripts# nmap --script /root/Desktop/smb-os-discovery.nse -p445 192.168.2.103

Starting Nmap 7.12 ( https://nmap.org ) at 2016-08-07 09:45 EDT
Nmap scan report for 192.168.2.103
Host is up (0.00072s latency).
PORT    STATE SERVICE
445/tcp open  microsoft-ds
MAC Address: 00:0C:29:83:ED:69 (VMware)

Host script results:
| smb-os-discovery:
|   OS: Windows 7 Ultimate 7601 Service Pack 1 (Windows 7 Ultimate 6.1)
|   OS CPE: cpe:/o:microsoft:windows_7::sp1
|   Computer name: WIN-RE1NUHRDONW
|   NetBIOS computer name: WIN-RE1NUHRDONW
|   Workgroup: WORKGROUP
|_  System time: 2016-08-07T20:45:47-07:00

Nmap done: 1 IP address (1 host up) scanned in 13.26 seconds
root@kali:/usr/share/nmap/scripts#
```

NMAP SCRIPTING ENGINE

USAGE

- More complicated script selection can be done using the and, or, and not operators to build Boolean expressions. The operators have the same precedence as in Lua: not is the highest, followed by and and then or. You can alter precedence by using parentheses. Because expressions contain space characters it is necessary to quote them.

- nmap --script "not intrusive"

Loads every script except for those in the intrusive category.

- nmap --script "default or safe"

This is functionally equivalent to nmap --script "default,safe". It loads all scripts that are in the default category or the safe category or both.

NMAP SCRIPTING ENGINE

USAGE

- nmap --script "default and safe"

Loads those scripts that are in both the default and safe categories.

- nmap --script "(default or safe or intrusive) and not http-*"

Loads scripts in the default, safe, or intrusive categories, except for those whose names start with http-.

NMAP SCRIPTING ENGINE

USAGE

```
root@kali:/usr/share/nmap/scripts# nmap --script "smb-* and not intrusive" -p445 -Pn 192.168.2.103

Starting Nmap 7.12 ( https://nmap.org ) at 2016-08-07 09:39 EDT
Nmap scan report for 192.168.2.103
Host is up (0.00046s latency).
PORT    STATE SERVICE
445/tcp open  microsoft-ds
MAC Address: 00:0C:29:83:ED:69 (VMware)

Host script results:
| smb-enum-sessions:
|   Master Browser
|     WIN-RE1NUHRDONW  6.1
|   Potential Browser
|     WIN-RE1NUHRDONW  6.1
|   Server service
|     WIN-RE1NUHRDONW  6.1
|   Windows NT/2000/XP/2003 server
|     WIN-RE1NUHRDONW  6.1
|   Workstation
|     WIN-RE1NUHRDONW  6.1
| smb-os-discovery:
|   OS: Windows 7 Ultimate 7601 Service Pack 1 (Windows 7 Ultimate 6.1)
|   OS CPE: cpe:/o:microsoft:windows_7::sp1
|   Computer name: WIN-RE1NUHRDONW
|   NetBIOS computer name: WIN-RE1NUHRDONW
|   Workgroup: WORKGROUP
|_  System time: 2016-08-07T20:40:06+07:00
| smb-security-mode:
|   account_used: guest
|   authentication_level: user
|   challenge_response: supported
|_  message_signing: disabled (dangerous, but default)

Nmap done: 1 IP address (1 host up) scanned in 13.27 seconds
root@kali:/usr/share/nmap/scripts#
```

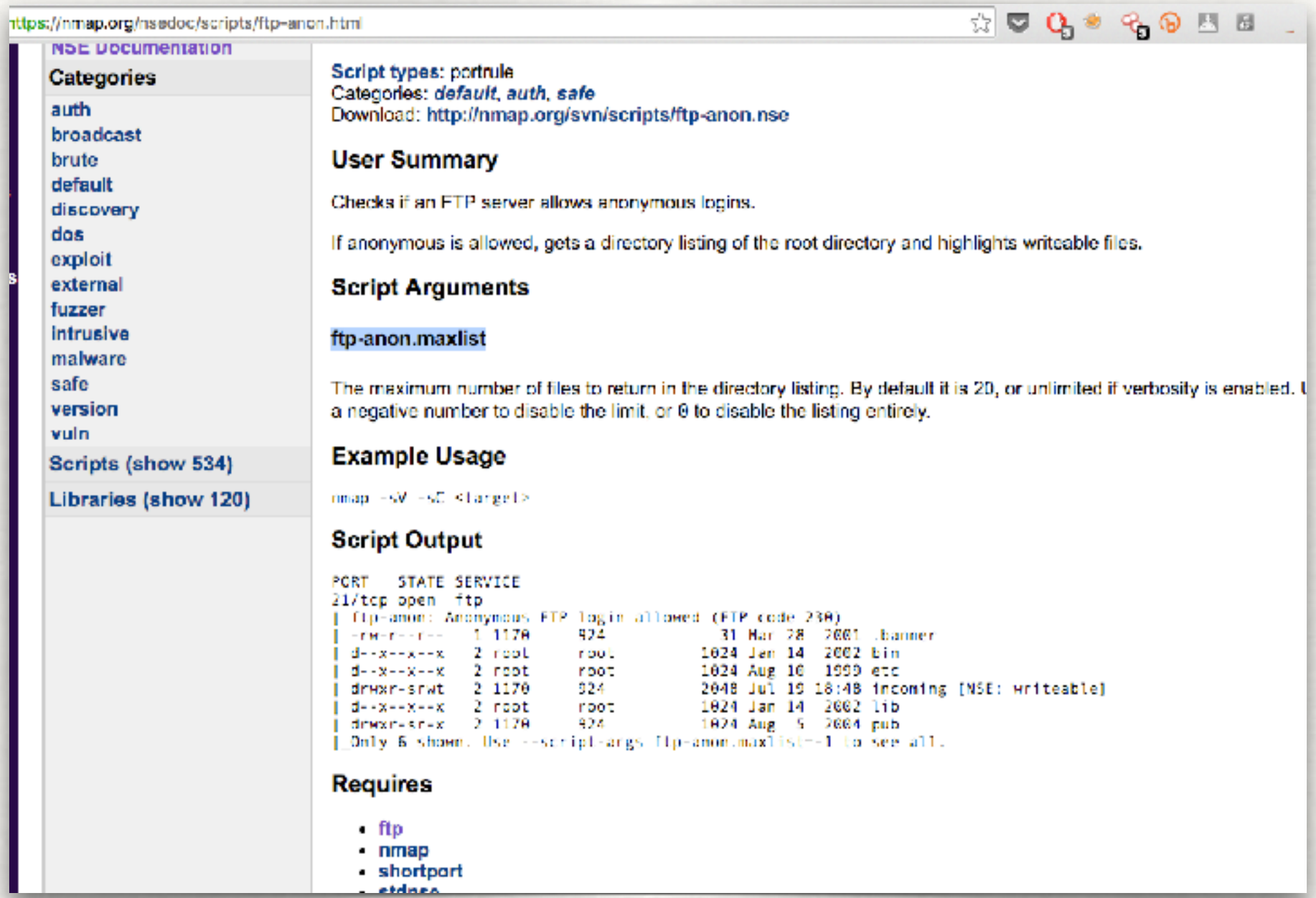
NMAP SCRIPTING ENGINE

ARGUMENTS TO SCRIPT

- Arguments may be passed to NSE scripts using the --script-args option or --script-args-file <filename>
- https://nmap.org/nsedoc/ lists the arguments that each script accepts.
- nmap --script snmp-sysdescr --script-args snmpcommunity=admin example.com

NMAP SCRIPTING ENGINE

ARGUMENTS TO SCRIPT



NMAP SCRIPTING ENGINE

ARGUMENTS TO SCRIPT

```
root@kali:/usr/share/nmap/scripts# nmap --script ftp-bounce --script-args ftp-b
ounce.checkhost=google.com -p21 192.168.2.103

Starting Nmap 7.12 ( https://nmap.org ) at 2016-08-07 10:33 EDT
Nmap scan report for 192.168.2.103
Host is up (0.00072s latency).
PORT   STATE SERVICE
21/tcp open  ftp
|_ftp-bounce: bounce working!
MAC Address: 00:0C:29:83:ED:69 (VMware)

Nmap done: 1 IP address (1 host up) scanned in 13.39 seconds
root@kali:/usr/share/nmap/scripts#
```

NMAP SCRIPTING ENGINE

SCRIPT HELP

- Shows help about scripts. For each script matching the given specification, Nmap prints the script name, its categories, and its description.

- The specifications are the same as those accepted by --script; so for example if you want help about the ftp-anon script, you would run nmap --script-help “smb-*”

NMAP SCRIPTING ENGINE

SCRIPT HELP

```
root@kali:/usr/share/nmap/scripts# nmap --script-help "smb-*"

Starting Nmap 7.12 ( https://nmap.org ) at 2016-08-07 10:04 EDT

smb-brute
Categories: intrusive brute
https://nmap.org/nsedoc/scripts/smb-brute.html
  Attempts to guess username/password combinations over SMB, storing discovered combinations
  for use in other scripts. Every attempt will be made to get a valid list of users and to
  verify each username before actually using them. When a username is discovered, besides
  being printed, it is also saved in the Nmap registry so other Nmap scripts can use it. That
  means that if you're going to run <code>smb-brute.nse</code>, you should run other <code>smb</code> scripts you want.
  This checks passwords in a case-insensitive way, determining case after a password is found,
  for Windows versions before Vista.

  This script is specifically targeted towards security auditors or penetration testers.
  One example of its use, suggested by Brandon Enright, was hooking up <code>smb-brute.nse</code> to the
  database of usernames and passwords used by the Conficker worm (the password list can be
  found at http://www.skullsecurity.org/wiki/index.php/Passwords, among other places.
  Then, the network is scanned and all systems that would be infected by Conficker are
  discovered.

  From the penetration tester perspective its use is pretty obvious. By discovering weak passwords
  on SMB, a protocol that is well suited for bruteforcing, access to a system can be gained.
  Further, passwords discovered against Windows with SMB might also be used on Linux or MySQL
  or custom Web applications. Discovering a password greatly beneficial for a pen tester.

  This script uses a lot of little tricks that I (Ron Bowes) describe in detail in a blog
  posting, http://www.skullsecurity.org/blog/?p=161. The tricks will be summarized here, but
  that blog is the best place to learn more.

  Usernames and passwords are initially taken from the unpwdb library. If possible, the usernames
  are verified as existing by taking advantage of Windows' odd behaviour with invalid username
  and invalid password responses. As soon as it is able, this script will download a full list
  of usernames from the server and replace the unpwdb usernames with those. This enables the
```

NMAP SCRIPTING ENGINE

SCRIPT TRACE

- --script-trace
- This option is similar to --packet-trace, but works at the application level rather than packet by packet. If this option is specified, all incoming and outgoing communication performed by scripts is printed.
- The displayed information includes the communication protocol, source and target addresses, and the transmitted data. If more than 5% of transmitted data is unprintable, hex dumps are given instead. Specifying --packet-trace enables script tracing too.

NMAP SCRIPTING ENGINE

SCRIPT TRACE

```
root@kali:/usr/share/nmap/scripts# nmap --script ftp-bounce --script-args ftp-b
ounce.checkhost=google.com --script-trace -p21 192.168.2.103

Starting Nmap 7.12 ( https://nmap.org ) at 2016-08-07 10:41 EDT
NSOCK INFO [13.2250s] nsock_trace_handler_callback(): Callback: CONNECT SUCCESS
 for EID 8 [192.168.2.103:21]
NSE: TCP 192.168.2.104:34769 > 192.168.2.103:21 | CONNECT
NSOCK INFO [13.2250s] nsock_read(): Read request from IOD #1 [192.168.2.103:21]
 (timeout: 10000ms) EID 18
NSOCK INFO [13.2290s] nsock_trace_handler_callback(): Callback: READ SUCCESS fo
r EID 18 [192.168.2.103:21] (25 bytes): 220 ALFTP Server ready...
NSE: TCP 192.168.2.104:34769 < 192.168.2.103:21 | 00000000: 32 32 30 20 41 4c 4
6 54 50 20 53 65 72 76 65 72 220 ALFTP Server
00000010: 20 72 65 61 64 79 2e 0d 0a                         ready.

NSE: TCP 192.168.2.104:34769 > 192.168.2.103:21 | 00000000: 55 53 45 52 20 61 6
e 6f 6e 79 6d 6f 75 73 0d 0a USER anonymous

NSOCK INFO [13.2300s] nsock_write(): Write request for 16 bytes to IOD #1 EID 2
7 [192.168.2.103:21]
NSOCK INFO [13.2300s] nsock_trace_handler_callback(): Callback: WRITE SUCCESS f
or EID 27 [192.168.2.103:21]
NSE: TCP 192.168.2.104:34769 > 192.168.2.103:21 | SEND
NSOCK INFO [13.2300s] nsock_read(): Read request from IOD #1 [192.168.2.103:21]
 (timeout: 5000ms) EID 34
NSOCK INFO [13.2320s] nsock_trace_handler_callback(): Callback: READ SUCCESS fo
r EID 34 [192.168.2.103:21] (38 bytes): 331 Password required for anonymous...
NSE: TCP 192.168.2.104:34769 < 192.168.2.103:21 | 00000000: 33 33 31 20 50 61 7
3 73 77 6f 72 64 20 72 65 71 331 Password req
```

NMAP SCRIPTING ENGINE

UPDATE DATABASE

- --script-updatedb
- This option updates the script database found in scripts/script.db which is used by Nmap to determine the available default scripts and categories.
- It is only necessary to update the database if you have added or removed NSE scripts from the default scripts directory or if you have changed the categories of any script.
- This option is used by itself without arguments: nmap --script-updatedb.

NMAP SCRIPTING ENGINE

UPDATE DATABASE

```
root@kali:/usr/share/nmap/scripts# nmap --script-updatedb

Starting Nmap 7.12 ( https://nmap.org ) at 2016-08-07 10:39 EDT
NSE: Updating rule database.
NSE: Script Database updated successfully.
Nmap done: 0 IP addresses (0 hosts up) scanned in 2.23 seconds
root@kali:/usr/share/nmap/scripts#
```


WAR10CK
F4RR3LL: how bad you think?
WAR10CK: very
F4RR3LL:
Nmap v 6.22
nmap -v -T4 -A insecure.org scanme.nmap.org 65ze
31.3.3.7 sectools.org
Starting Nmap 4.20ALPHA8 (http://Insecure.Org) at
2006-10-05 13:29 PDT
Initiating Parallel DNS resolution of 2 hosts. at 13:29
Completed Parallel DNS resolution of 2 hosts. at 13:29,
0.00s elapsed
Initiating ARP Ping Scan at 13:29
Scanning 65ze [1 port]
Completed ARP Ping Scan at 13:29, 0.01s elapsed
(1 total hosts)
Initiating Parallel DNS resolution of 1 host. at 13:29
Completed Parallel DNS resolution of 1 host. at 13:29,
0.00s elapsed
Initiating SYN Stealth Scan at 13:29
Scanning 2 hosts [1690 ports/host]
Discovered open port 22/tcp on 205.217.153.53
Discovered open port 22/tcp on 205.217.153.62
Discovered open port 80/tcp on 205.217.153.53
Discovered open port 80/tcp on 205.217.153.62
Discovered open port 53/tcp on 205.217.153.53
Discovered open port 53/tcp on 205.217.153.62
Completed SYN Stealth Scan against 205.217.153.53 in
25.94s (1 host left)
Completed SYN Stealth Scan at 13:30, 25.94s elapsed
(1380 total ports)
Initiating Service scan at 13:30
Scanning 6 services on 2 hosts
Completed Service scan at 13:30, 12.02s elapsed
(6 services on 2 hosts)
Initiating OS detection (try #1) against 2 hosts
Retrying OS detection (try #2) against 2 hosts
Initiating Nmap OS Detection against 205.217.153.53 at
41.87Ss
For OSScan assuming port 22 is open, 70 is closed, and

NCAT

NMAP SUITES: NCAT

INTRODUCTION

- ncat - Concatenate and redirect sockets
- Ncat is a feature-packed networking utility which reads and writes data across networks from the command line. Ncat was written for the Nmap Project and is the culmination of the currently splintered family of Netcat incarnations. It is designed to be a reliable back-end tool to instantly provide network connectivity to other applications and users. Ncat will not only work with IPv4 and IPv6 but provides the user with a virtually limitless number of potential uses.
- Among Ncat's vast number of features there is the ability to chain Ncats together; redirection of TCP, UDP, and SCTP ports to other sites; SSL support; and proxy connections via SOCKS4 or HTTP proxies (with optional proxy authentication as well). Some general principles apply to most applications and thus give you the capability of instantly adding networking support to software that would normally never support it.

NMAP SUITES: NCAT

USAGE

- ncat [<OPTIONS> ...] [<hostname>] [<port>]
- Ncat operates in one of two primary modes: Connect Mode (client) and Listen Mode (server)
- If port omitted, it defaults to 31337.

NMAP SUITES: NCAT

PROTOCOL OPTIONS

- -4 (IPv4 only): Force the use of IPv4 only.
- -6 (IPv6 only): Force the use of IPv6 only.
- -U, --unixsock (Use Unix domain sockets): Use Unix domain sockets rather than network sockets. This option may be used on its own for stream sockets, or combined with --udp for datagram sockets. A description of -U mode is in the section called “Unix Domain Sockets”.
- -u, --udp (Use UDP): Use UDP for the connection (the default is TCP).
- --sctp (Use SCTP): Use SCTP for the connection (the default is TCP). SCTP support is implemented in TCP-compatible mode.

NMAP SUITES: NCAT

GENERAL OPTIONS

- -l, --listen (Listen for connections) : Listen for connections rather than connecting to a remote machine

- -k, --keep-open (Accept multiple connections): This option makes Ncat to accept multiple simultaneous connections and wait for more connections after they have all been closed. It must be combined with --listen.

- --ssl (Use SSL) : In connect mode, this option transparently negotiates an SSL session with an SSL server to securely encrypt the connection. This is particularly handy for talking to SSL enabled HTTP servers, etc.

- --proxy <host>[:<port>] (Specify proxy address) : Requests proxying through <host>:<port>, using the protocol specified by --proxy-type and auth with --proxy-auth.

NMAP SUITES: NDIFF

USAGE

File Edit View Search Terminal Help

```
root@kali:/# ncat -l 192.168.2.106 3333
ls
id
chat-mode
```

root@kali: ~

File Edit View Search Terminal Help

```
root@kali:~# ncat 192.168.2.106 3333
ls
id
chat-mode
```

NMAP SUITES: NDIFF

USAGE

```
root@kali:/# ncat -l 192.168.2.106 3333 -k
test with -k
server still alive after connection closed
yes
```

root@kali: ~

File Edit View Search Terminal Help

```
root@kali:~# ncat 192.168.2.106 3333
test with -k
^C
root@kali:~# ncat 192.168.2.106 3333
server still alive after connection closed
^C
root@kali:~# ncat 192.168.2.106 3333
yes
```

NMAP SUITES: NCAT

GENERAL OPTIONS

- -e <command>, --exec <command> (Execute command)

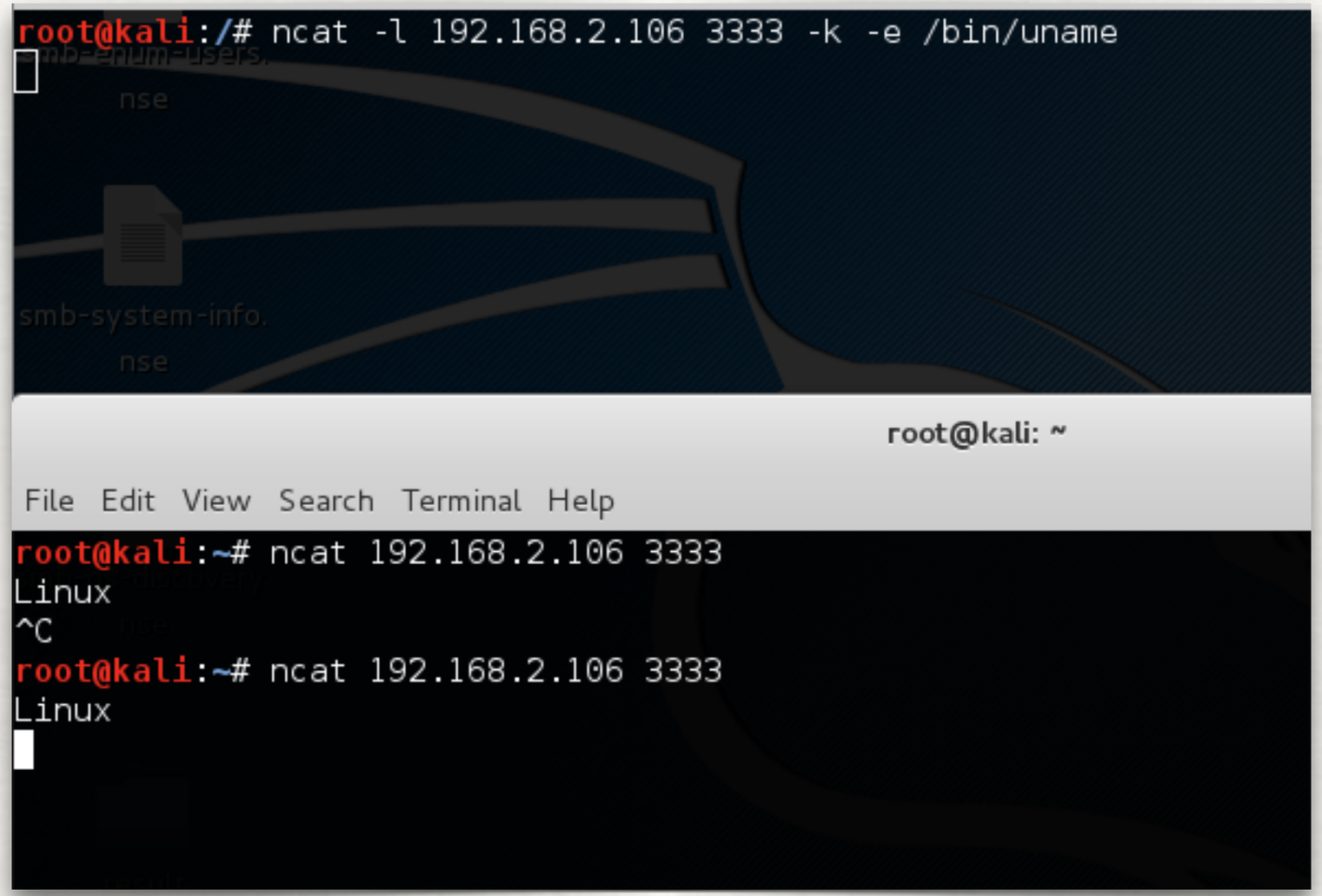
Execute the specified command after a connection has been established. The command must be specified as a full pathname. All input from the remote client will be sent to the application and responses sent back to the remote client over the socket, thus making your command-line application interactive over a socket. Combined with --keep-open, Ncat will handle multiple simultaneous connections to your specified port/application like inetd. Ncat will only accept a maximum, definable, number of simultaneous connections controlled by the -m option. By default this is set to 100 (60 on Windows).

- -c <command>, --sh-exec <command> (Execute command via sh)

Same as -e, except it tries to execute the command via /bin/sh. This means you don't have to specify the full path for the command, and shell facilities like environment variables are available.

NMAP SUITES: NCAT

USAGE



NMAP SUITES: NCAT

USAGE

```
root@kali:/# ncat -l 192.168.2.106 3333 -k -c /bin/sh
```

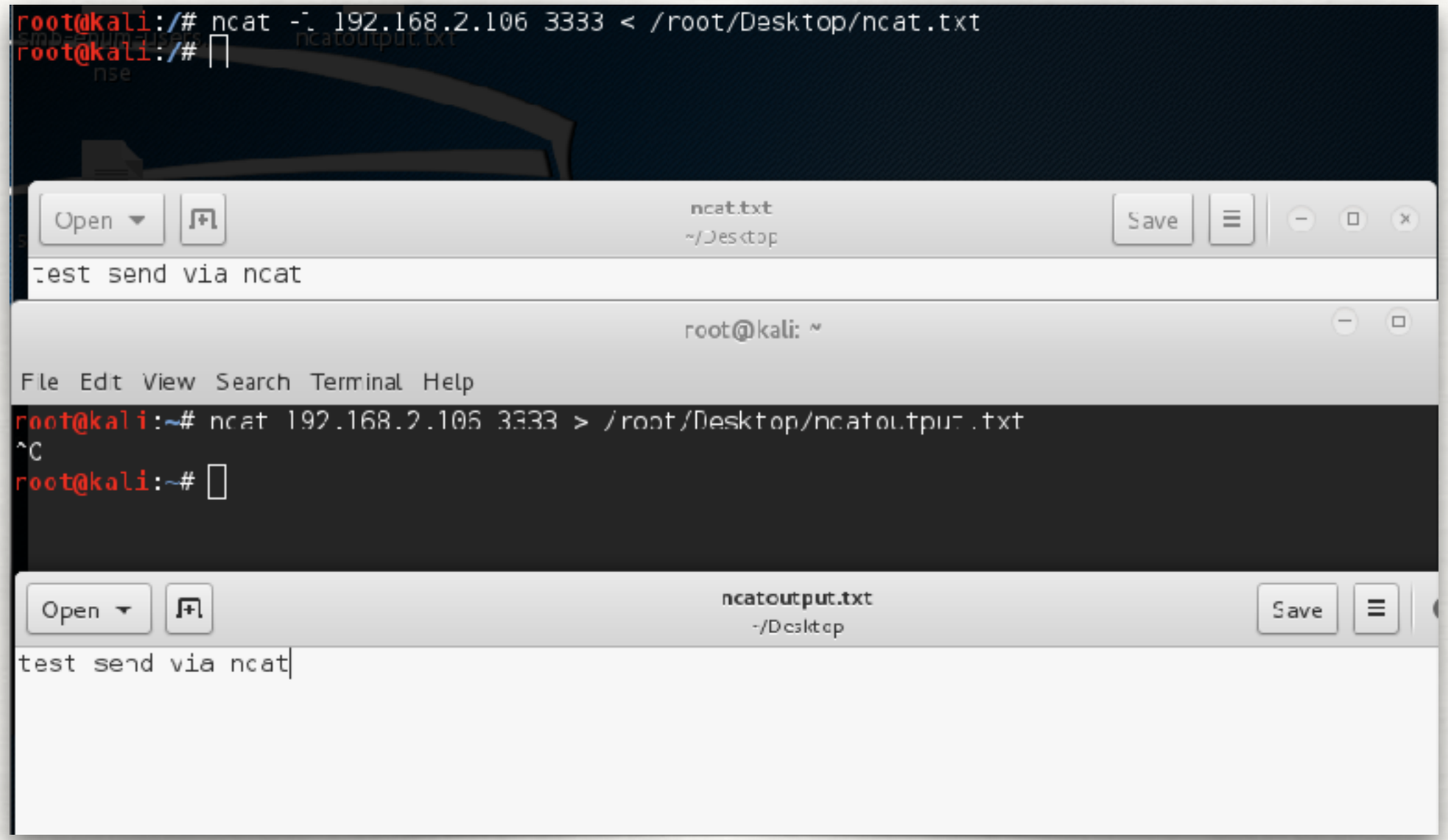
root@kali: ~

File Edit View Search Terminal Help

```
root@kali:~# ncat 192.168.2.106 3333
id
uid=0(root) gid=0(root) groups=0(root)
uname -a
Linux kali 4.0.0-kali1-amd64 #1 SMP Debian 4.0.4-1+kali2 (2015-06-03) x86_64 GNU/Linux
pwd
/
```

NMAP SUITES: NCAT

USAGE



DEVELOP CUSTOM NSE

NMAP SCRIPTING ENGINE

LUA:INTRODUCTION

- Lua is an extension programming language designed to support general procedural programming with data description facilities.
- It also offers good support for object-oriented programming, functional programming, and data-driven programming.
- Lua is intended to be used as a powerful, lightweight, embeddable scripting language for any program that needs one.
- Lua is implemented as a library, written in clean C, the common subset of Standard C and C++.

NMAP SCRIPTING ENGINE

LUA:INTRODUCTION

https://www.lua.org/pil/1.html

1 – Getting Started

To keep with the tradition, our first program in Lua just prints `"Hello World"`:

```
print("Hello World")
```

If you are using the stand-alone Lua interpreter, all you have to do to run your first program is to call the interpreter (usually named `lua`) with the name of the text file that contains your program. For instance, if you write the above program in a file `hello.lua`, the following command should run it:

```
prompt> lua hello.lua
```

As a slightly more complex example, the following program defines a function to compute the factorial of a given number, asks the user for a number, and prints its factorial:

```
-- defines a factorial function
function fact (n)
  if n == 0 then
    return 1
  else
    return n * fact(n-1)
  end
end

print("enter a number:")
a = io.read("*number")        -- read a number
print(fact(a))
```

If you are using Lua embedded in an application, such as CGILua or IUPLua, you may need to refer to the application manual (or to a "local guru") to learn how to run your programs. Nevertheless, Lua is still the same language; most things that we will see here are valid regardless of how you are using Lua. For a start, we recommend that you use the stand-alone interpreter (that is, the `lua` executable) to run your first examples and experiments.

NMAP SCRIPTING ENGINE

SCRIPT STRUCTURE

1. The Head Section

Contain meta information, this includes the fields: description, categories, dependencies, author, and license as well as initial NSEDoc information such as usage, args, and output tags

2. The Rule Section

Defines necessary conditions for the script to execute. This section must contain at least one function from this list: portrule, hostrule, prerule, postrule.

3. The Action Section

Defines the script logic.

NMAP SCRIPTING ENGINE

SCRIPT STRUCTURE

1. The Head Section

Contain meta information, this includes the fields: description, categories, dependencies, author, and license as well as initial NSEDoc information such as usage, args, and output tags

2. The Rule Section

Defines necessary conditions for the script to execute. This section must contain at least one function from this list: portrule, hostrule, prerule, postrule.

3. The Action Section

Defines the script logic.

NMAP SCRIPTING ENGINE

SCRIPT STRUCTURE: HEAD

description Field

The description field describes what a script is testing for and any important notes the user should be aware of. Depending on script complexity, descriptions may vary in length from a few sentences to a few paragraphs. The first paragraph should be a brief synopsis of the script function suitable for stand-alone presentation to the user. Further paragraphs may provide much more script detail.

categories Field

The categories field defines one or more categories to which a script belongs (see the section called “Script Categories”). The categories are case-insensitive and may be specified in any order. They are listed in an array-style Lua table as in this example:

```
categories = {"default", "discovery", "safe"}
```

author Field

The author field contains the script authors' names and can also contain contact information (such as home page URLs). We no longer recommend including email addresses because spammers might scrape them from the NSEDoc web site. This optional field is not used by NSE, but gives script authors their due credit or blame.

NMAP SCRIPTING ENGINE

SCRIPT STRUCTURE: HEAD

license Field

Nmap is a community project and we welcome all sorts of code contributions, including NSE scripts. So if you write a valuable script, don't keep it to yourself! The optional license field helps ensure that we have legal permission to distribute all the scripts which come with Nmap. All of those scripts currently use the standard Nmap license (described in the section called “Nmap Copyright and Licensing”). They include the following line:

license = "Same as Nmap--See https://nmap.org/book/man-legal.html”

dependencies Field

The dependencies field is an array containing the names of scripts that should run before this script, if they are also selected. This is used when one script can make use of the results of another. For example, most of the smb-* scripts depend on smb-brute, because the accounts found by smb-brute may allow the other scripts to get more information. Listing a script in dependencies doesn't cause that script to be run; it still has to be selected through the --script option or otherwise. dependencies merely forces an ordering among the scripts that are selected. This is an example of a dependencies table, from smb-os-discovery:

dependencies = {"smb-brute"}

The dependencies table is optional. NSE will assume the script has no dependencies if the field is omitted.

NMAP SCRIPTING ENGINE

SCRIPT STRUCTURE: RULE

Nmap uses the script rules to determine whether a script should be run against a target. A rule is a Lua function that returns either true or false. The script action function is only performed if the rule evaluates to true.

A script must contain one or more of the following functions that determine when the script will be run:

- prerule()
- hostrule(host)
- portrule(host, port)
- postrule()

NMAP SCRIPTING ENGINE

SCRIPT STRUCTURE: RULE

prerule scripts run once, before any hosts are scanned, during the script pre-scanning phase.

hostrule and portrule scripts run after each batch of hosts is scanned.

postrule scripts run once after all hosts have been scanned, in the script post-scanning phase. A script may run in more than one phase if it has several rules.

prerule and postrule do not accept arguments. hostrule accepts a host table and may test, for example, the IP address or hostname of the target. portrule accepts both a host table and a port table for any port in the open, openlfiltered, or unfiltered port states.

NMAP SCRIPTING ENGINE

SCRIPT STRUCTURE: ACTION

The action is the heart of an NSE script.

It contains all of the instructions to be executed when the script's prerule, portrule, hostrule or postrule triggers. It is a **Lua** function which accepts the same arguments as the rule.

The return value of the action value may be a table of name–value pairs, a string, or nil.

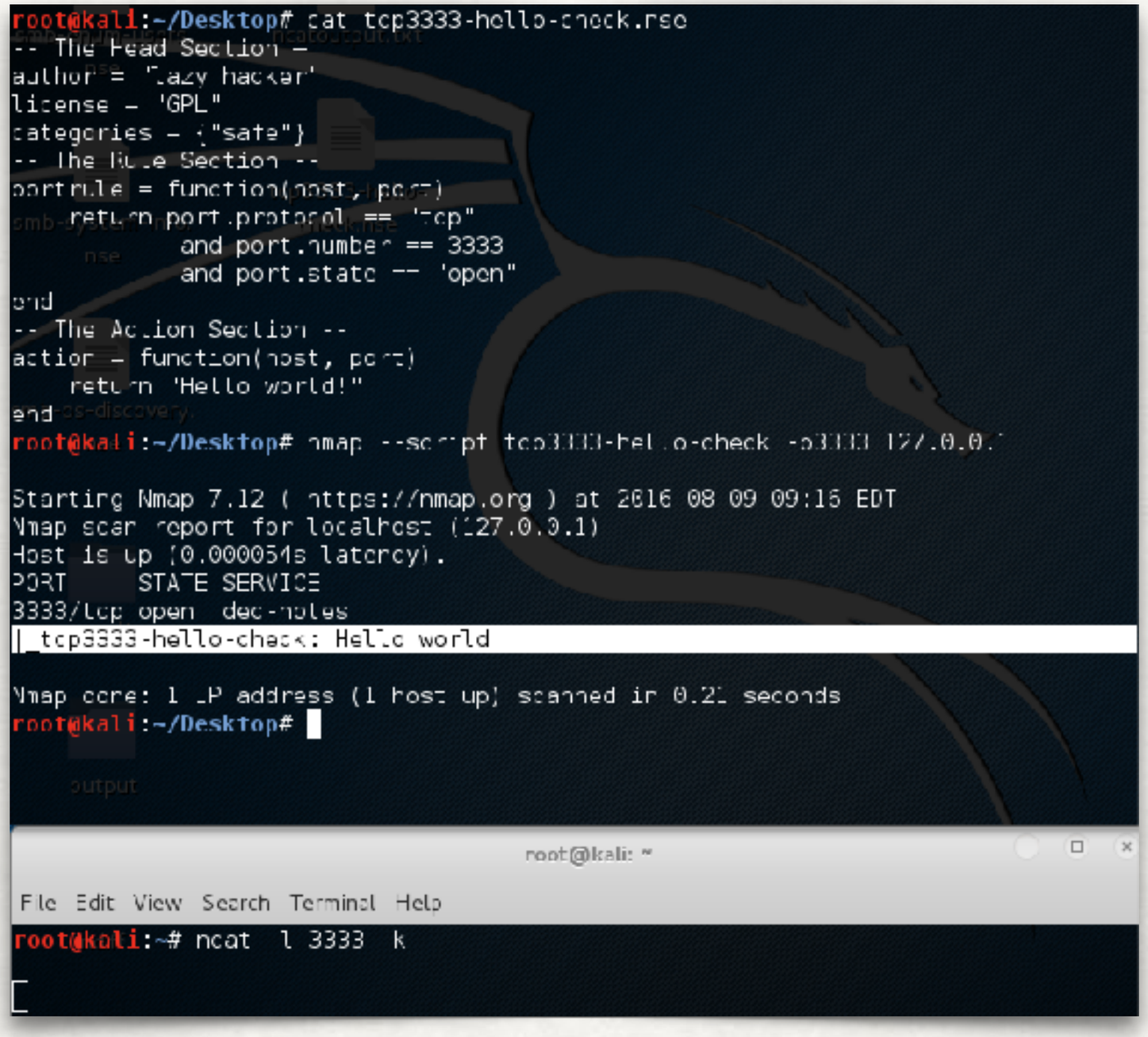
NMAP SCRIPTING ENGINE

SCRIPT EXAMPLE

```
-- The Head Section —

author = “lazy hacker"

license = "GPL"

categories = {"safe"}

-- The Rule Section --

portrule = function(host, port)

    return port.protocol == "tcp"

            and port.number == 3333

            and port.state == "open"

end

-- The Action Section --

action = function(host, port)

    return "Hello world!"

end
```

NMAP SCRIPTING ENGINE

CUSTOM NSE SCRIPT ON ACTION



NMAP SCRIPTING ENGINE

USING NSE LIBRARIES

- NSE become more powerfull with the NSE libraries
- https://nmap.org/book/nse-library.html#nse-library-list
- eg, using shortport library for the portrule
- https://nmap.org/nsedoc/lib/shortport.html

NMAP SCRIPTING ENGINE

SCRIPT WITH NSE LIBRARIES EXAMPLE

```
-- The Head Section —

author = “lazy hacker"

license = "GPL"

categories = {“safe"}

local shortport = require “shortport"

portrule = shortport.portnumber(3333)

-- The Action Section --

action = function(host, port)

    return "Hello world!"

end
```

NMAP SCRIPTING ENGINE

CUSTOM NSE ON ACTION

https://nmap.org/nsedoc/lib/shortport.html#service

- ports: A single port number or a list of port numbers.
- services: Service name or a list of names to run against.
- protos: The protocol or list of protocols to match against, default `"tcp"`.
- states: A state or list of states to match against, default `{"open", "open|filtered"}`.

Usage:

```
portrule = shortport.port_or_service(22,"ssh").
```

Return value:
Function for the portrule.

---

portnumber (ports, protos, states)

Return a portrule that returns true when given an open port matching a single port number or a list of port numbers.

Parameters

- ports: A single port number or a list of port numbers.
- protos: The protocol or list of protocols to match against, default `"tcp"`.
- states: A state or list of states to match against, default `{"open", "open|filtered"}`.

Usage:

```
portrule = shortport.portnumber({80, 443})
```

Return value:
Function for the portrule.

---

service (services, protos, states)

Return a portrule that returns true when given an open port with a service name matching a single service name or a list of service names.

NMAP SCRIPTING ENGINE

CUSTOM NSE SCRIPT ON ACTION

```
root@kali:~/Desktop# cat tcp3333-hello-check1.nse
-- The Head Section --
author = "lazy hacker"
license = "GPL"
categories = {"safe"}
local shortport = require "shortport"
-- The Rule Section --
portrule = shortport.portnumber(3333)
-- The Action Section --
action = function(host, port)
    return "Hello world!"
end
root@kali:~/Desktop# nmap --script tcp3333-hello-check1 -p3333 127.0.0.1

Starting Nmap 7.12 ( https://nmap.org ) at 2016-08-09 09:28 EDT
Nmap scan report for localhost (127.0.0.1)
Host is up (0.000077s latency).
PORT     STATE SERVICE
3333/tcp open  dec-notes
|_tcp3333-hello-check1: Hello world!

Nmap done: 1 IP address (1 host up) scanned in 0.29 seconds
root@kali:~/Desktop#
```

```
root@kali: ~
File  Edit  View  Search  Terminal  Help
root@kali:~# ncat -l 3333 -k
```

NMAP SCRIPTING ENGINE

NSE SCRIPT WITH BANNER GRABBING

```
author = "lazy hacker"
license = "GPL"
categories = {"safe"}
local shortport = require "shortport"
local nmap = require “nmap"
portrule = shortport.portnumber(3333)

action = function(host, port)
    local out=grab_banner(host, port)
    return out
    end

function grab_banner(host,port)
     local st,buff,banner
     local socket=nmap.new_socket()
     socket:set_timeout(5000)
     banner=""
     st=socket:connect(host, port, "tcp")
     st, buff=socket:receive()
     banner=buff
     return banner
end
```

NMAP SCRIPTING ENGINE

CUSTOM NSE SCRIPT ON ACTION

```
root@kali:~/Desktop# nmap --script tcp3333-hello-check3 -p3333 127.0.0.1

Starting Nmap 7.12 ( https://nmap.org ) at 2016-08-09 09:57 EDT
Nmap scan report for localhost (127.0.0.1)
Host is up (0.000041s latency).
PORT     STATE SERVICE
3333/tcp open  dec-notes
|_tcp3333-hello-check3: Linux

Nmap done: 1 IP address (1 host up) scanned in 0.33 seconds
root@kali:~/Desktop#
```

```
root@kali: ~
File  Edit  View  Search  Terminal  Help
root@kali:~# ncat -l 3333 -k -e /bin/uname
```


WAR10CK
F4RR3LL: how bad you think?
WAR10CK: very
F4RR3LL:
Nmap v 6.22
nmap -v -T4 -A insecure.org scanme.nmap.org 0/24
31.3.3.7 sectools.org
Starting Nmap 4.20ALPHA8 (http://Insecure.Org) at
2006-10-05 13:29 PDT
Initiating Parallel DNS resolution of 2 hosts. at 13:29
Completed Parallel DNS resolution of 2 hosts. at 13:29,
0.00s elapsed
Initiating ARP Ping Scan at 13:29
Scanning 0Bzz [1 port]
Completed ARP Ping Scan at 13:29, 0.01s elapsed
(1 total hosts)
Initiating Parallel DNS resolution of 1 host. at 13:29
Completed Parallel DNS resolution of 1 host. at 13:29,
0.00s elapsed
Initiating SYN Stealth Scan at 13:29
Scanning 2 hosts [1680 ports/host]
Discovered open port 22/tcp on 205.217.153.53
Discovered open port 22/tcp on 205.217.153.62
Discovered open port 80/tcp on 205.217.153.53
Discovered open port 80/tcp on 205.217.153.62
Discovered open port 53/tcp on 205.217.153.53
Discovered open port 53/tcp on 205.217.153.62
Completed SYN Stealth Scan against 205.217.153.53 in
25.94s (1 host left)
Completed SYN Stealth Scan at 13:30, 25.94s elapsed
(1680 total ports)
Initiating Service scan at 13:30
Scanning 6 services on 2 hosts
Completed Service scan at 13:30, 12.02s elapsed
(6 services on 2 hosts)
Initiating OS detection (try #1) against 2 hosts
Retrying OS detection (try #2) against 2 hosts
Initiating Nmap OS Detection against 205.217.153.53 at
41.87Ss
For OSScan assuming port 22 is open, 70 is closed, and

NDIFF

NMAP SUITES: NDIFF

INTRODUCTION

- ndiff - Utility to compare the results of Nmap scans
- Ndiff is a tool to aid in the comparison of Nmap scans. It takes two Nmap XML output files and prints the differences between them.
- The differences observed are:
  - Host states (e.g. up to down)
  - Port states (e.g. open to closed)
  - Service versions (from -sV)
  - OS matches (from -O)
  - Script output

NMAP SUITES: NDIFF

OPTIONS

- -h, --help: Show a help message and exit.
- -v, --verbose: Include all hosts and ports in the output, not only those that have changed.
- --text: Write output in human-readable text format.
- --xml: Write output in machine-readable XML format. The document structure is defined in the file ndiff.dtd included in the distribution.

NMAP SUITES: NDIFF

USAGE

```
root@kali:~/Desktop/ndiff# ls
103dua.xml  103satu.xml
root@kali:~/Desktop/ndiff# ndiff 103satu.xml 103
103dua.xml  103satu.xml
root@kali:~/Desktop/ndiff# ndiff 103satu.xml 103dua.xml
-Nmap 7.12 scan initiated Mon Aug 08 23:11:33 2016 as: nmap -A -T4 -oX 103satu.xml 192.168.2.103
+Nmap 7.12 scan initiated Mon Aug 08 23:46:53 2016 as: nmap -A -T4 -oX 103dua.xml 192.168.2.103

 192.168.2.103, 00:0C:29:BA:E6:DD:
-Not shown: 977 closed ports
+Not shown: 979 closed ports
 PORT     STATE SERVICE     VERSION
 22/tcp   open  ssh         OpenSSH 4.7p1 Debian 8ubuntu1 (protocol 2.0)
-|  ssh-hostkey:
-|    1024 60:0f:cf:e1:c0:5f:6a:74:d6:90:24:fa:c4:d5:6c:cd (DSA)
-|_   2048 56:56:24:0f:21:1d:de:a7:2b:ae:61:b1:24:3d:e8:f3 (RSA)
 25/tcp   open  smtp        Postfix smtpd
-|_ ssl-date: 2016-07-28T10:10:46+00:00; -11d17h33m59s from scanner time.
+|_ ssl-date: 2016-07-28T10:13:24+00:00; -11d17h34m01s from scanner time.
 2121/tcp open  ftp         ProFTPD 1.3.1
 3306/tcp open  mysql       MySQL 5.0.51a-3ubuntu5
 |  mysql-info:
 |    Protocol: 53
 |    Version: .0.51a-3ubuntu5
-|    Thread ID: 42
+|    hread ID: 46
 |    Capabilities flags: 43564
-|    Some Capabilities: SwitchToSSLAfterHandshake, Support41Auth, SupportsTransactions, Speaks41Prot
ocolNew, SupportsCompression, ConnectWithDatabase, LongColumnFlag
+|    Some Capabilities: Support41Auth, SupportsCompression, LongColumnFlag, SwitchToSSLAfterHandshak
e, SupportsTransactions, Speaks41ProtocolNew, ConnectWithDatabase
 |    Status: Autocommit
-|_   Salt: URz2)r(l:Kw:Y0#^2{/>
+|_   Salt: ( kx9Svy:)04L12GM-9t
```

NMAP SUITES: NDIFF

USAGE

```
root@kali:~/Desktop/ndiff# ndiff 103satu.xml 103dua.xml --verbose
-Nmap 7.12 scan initiated Mon Aug 08 23:41:03 2016 as: nmap -A -T4 -oX 103satu.xml 192.168.2.103
+Nmap 7.12 scan initiated Mon Aug 08 23:46:53 2016 as: nmap -A -T4 -oX 103dua.xml 192.168.2.103

 192.168.2.103, 00:0C:29:BA:E6:DD:
 Host is up.
-Not shown: 977 closed ports
+Not shown: 979 closed ports
 PORT     STATE SERVICE     VERSION
 21/tcp   open  ftp         vsftpd 2.3.4
 |_ ftp-anon: Anonymous FTP login allowed (FTP code 230)
-22/tcp   open  ssh         OpenSSH 4.7p1 Debian 8ubuntu1 (protocol 2.0)
-|  ssh-hostkey:
-|    1024 60:0f:cf:e1:c0:5f:6a:74:d6:90:24:fa:c4:d5:6c:cd (DSA)
-|_   2048 56:56:24:0f:21:1d:de:a7:2b:ae:61:b1:24:3d:e8:f3 (RSA)
 23/tcp   open  telnet      Linux telnetd
 25/tcp   open  smtp        Postfix smtpd
 |_ smtp-commands: metasploitable.localdomain, PIPELINING, SIZE 10240000, VRFY, ETRN, STARTTLS, ENHAN
CEDSTATUSCODES, 8BITMIME, DSN,
 |  ssl-cert: Subject: commonName=ubuntu804-base.localdomain/organizationName=OCOSA/stateOrProvinceNa
me=There is no such thing outside US/countryName=XX
 |  Not valid before: 2010-03-17T14:07:45
 |_ Not valid after:  2010-04-16T14:07:45
-|_ ssl-date: 2016-07-28T10:10:16+00:00; -11d17h33m59s from scanner time.
+|_ ssl-date: 2016-07-28T10:13:24+00:00; -11d17h34m01s from scanner time.
 |  sslv2:
 |    SSLv2 supported
 |    ciphers:
 |      SSL2_DES_192_EDE3_CBC_WITH_MD5
 |      SSL2_RC2_128_CBC_WITH_MD5
 |      SSL2_RC4_128_WITH_MD5
 |      SSL2_DES_64_CBC_WITH_MD5
 |      SSL2_RC2_128_CBC_EXPORT40_WITH_MD5
 |_     SSL2_RC4_128_EXPORT40_WITH_MD5
 53/tcp   open  domain      ISC BIND 9.4.2
 |  dns-nsid:
 |_   bind.version: 9.4.2
 80/tcp   open  http        Apache httpd 2.2.8 ((Ubuntu) DAV/2)
```

NMAP SUITES: NDIFF

USAGE

```
root@kali:~/Desktop/ndiff# -diff 103satu.xml 103dua.xml --text > 103ndifftext.txt
root@kali:~/Desktop/ndiff#
```

Open | 103ndifftext.txt ~/Desktop/ndiff | Save

```
-Nmap 7.12 scan initiated Mon Aug 08 23:44:03 2016 as: nmap -A -T4 -oX 103satu.xml 192.168.2.103
+Nmap 7.12 scan initiated Mon Aug 08 23:46:53 2016 as: nmap -A -T4 -oX 103dua.xml 192.168.2.103

 192.168.2.103, 00:0C:29:DA:E6:DD:
-Not shown: 977 closed ports
+Not shown: 979 closed ports
 PORT     STATE SERVICE     VERSION
-22/tcp   open  ssh         OpenSSH 4.7p1 Debian 8ubuntu1 (protocol 2.0)
-|  ssh-hostkey:
-|    1024 60:0f:cf:e1:c0:5f:6a:74:d6:90:24:fa:c4:d5:6c:cd (DSA)
-|    2048 56:56:24:0f:21:1d:de:a7:2b:ae:61:b1:24:3d:e8:f3 (RSA)
 25/tcp   open  smtp        Postfix smtpd
-|_ ssl-date: 2016-07-28T10:10:46+00:00; -11d17h33m59s from scanner time.
+|  ssl-date: 2016-07-28T10:13:21+00:00; -11d17h37m01s from scanner time.
-21/tcp open  ftp       | ProFTPD 1.3.1
 3306/tcp open  mysql       MySQL 5.0.51a-3ubuntu5
  |  mysql-info:
  |    Protocol: 53
  |    Version: .0.51a-3ubuntu5
-|    Thread ID: 11
+|    Thread ID: 46
  |    Capabilities flags: 43564
-|    Some Capabilities: SwitchToSSLAfterHandshake, Support41Auth, SupportsTransactions,
Speaks41ProtocolNew, SupportsCompression, ConnectWithDatabase, LongColumnFlag
+|    Some Capabilities: Support41Auth, SupportsCompression, LongColumnFlag,
SwitchToSSLAfterHandshake, SupportsTransactions, Speaks41ProtocolNew, ConnectWithDatabase
  |    Status: Autocommit
-|_   Salt: URz2)r(l:Kw:Y0#^2:7>
```

NMAP SUITES: NDIFF

USAGE

```
root@kali:~/Desktop/ndiff# ndiff 103satu.xml 103dua.xml --xml > 103ndiff.xml
root@kali:~/Desktop/ndiff# 
```

103satu.xml
~/Desktop/ndiff


```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE nmaprun>
<?xml-stylesheet href="file:///usr/bin/../share/nmap/nmap.xsl" type="text/xsl"?>
<!-- Nmap 7.12 scan initiated Mon Aug  8 23:44:03 2016 as: nmap -A -T4 -oX 103satu.xml 192.168.2.103
-->
<nmaprun scanner="nmap" args="nmap -A -T4 -oX 103satu.xml 192.168.2.103" start="1470714243"
startstr="Mon Aug  8 23:44:03 2016" version="7.12" xmloutputversion="1.04">
<scaninfo type="syn" protocol="tcp" numservices="1000"
services="1,3-4,6-7,9,13,17,19-26,30,32-33,37,42-43,49,53,70,79-85,88-90,99-100,106,109-111,113,119,12
>
<verbose level="0"/>
<debugging level="0"/>
<host starttime="1470714248" endtime="1470714290"><status state="up" reason="arp-response"
reason_ttl="0"/>
<address addr="192.168.2.103" addrtype="ipv4"/>
<address addr="00:0C:29:BA:E6:DD" addrtype="mac" vendor="VMware"/>
<hostnames>
</hostnames>
<ports><extraports state="closed" count="977">
<extrareasons reason="resets" count="977"/>
</extraports>
<port protocol="tcp" portid="21"><state state="open" reason="syn-ack" reason_ttl="64"/><service
name="ftp" product="vsftpd" version="2.3.4" ostype="Unix" method="probed" conf="10"><cpe>cpe:/
a:vsftpd:vsftpd:2.3.4</cpe></service><script id="ftp-anon" output="Anonymous FTP login allowed (FTP
code 230)"/></port>
<port protocol="tcp" portid="22"><state state="open" reason="syn-ack" reason_ttl="64"/><service
```


WAR10CK
F4RR3LL: how bad you think?
WAR10CK: very
F4RR3LL:
Nmap v 6.22
nmap -v -T4 -A insecure.org scanme.nmap.org 0/24
31.3.3.7 sectools.org
Starting Nmap 4.20ALPHA8 (http://Insecure.Org) at
2006-10-05 13:29 PDT
Initiating Parallel DNS resolution of 2 hosts. at 13:29
Completed Parallel DNS resolution of 2 hosts. at 13:29,
0.00s elapsed
Initiating ARP Ping Scan at 13:29
Scanning 0Bzz [1 port]
Completed ARP Ping Scan at 13:29, 0.01s elapsed
(1 total hosts)
Initiating Parallel DNS resolution of 1 host. at 13:29
Completed Parallel DNS resolution of 1 host. at 13:29,
0.00s elapsed
Initiating SYN Stealth Scan at 13:29
Scanning 2 hosts [1680 ports/host]
Discovered open port 22/tcp on 205.217.153.53
Discovered open port 22/tcp on 205.217.153.62
Discovered open port 80/tcp on 205.217.153.53
Discovered open port 80/tcp on 205.217.153.62
Discovered open port 53/tcp on 205.217.153.53
Discovered open port 53/tcp on 205.217.153.62
Completed SYN Stealth Scan against 205.217.153.53 in
25.94s (1 host left)
Completed SYN Stealth Scan at 13:30, 25.94s elapsed
(1360 total ports)
Initiating Service scan at 13:30
Scanning 6 services on 2 hosts
Completed Service scan at 13:30, 12.02s elapsed
(6 services on 2 hosts)
Initiating OS detection (try #1) against 2 hosts
Retrying OS detection (try #2) against 2 hosts
Initiating Parallel OS Detection against 205.217.153.53 at
41.87Ss
For OSScan assuming port 22 is open, 70 is closed, and

NPING

NMAP SUITES: NPING

INTRODUCTION

- nping : Network packet generation tool / ping utility
- Nping is an open-source tool for network packet generation, response analysis and response time measurement.
- Nping allows users to generate network packets of a wide range of protocols, letting them tune virtually any field of the protocol headers.
- Nping can be used as a simple ping utility to detect active hosts, it can also be used as a raw packet generator for network stack stress tests, ARP poisoning, Denial of Service attacks, route tracing, and other purposes.

NMAP SUITES: NPING

INTRODUCTION

- nping : Network packet generation tool / ping utility
- Nping is an open-source tool for network packet generation, response analysis and response time measurement.
- Nping allows users to generate network packets of a wide range of protocols, letting them tune virtually any field of the protocol headers.
- Nping can be used as a simple ping utility to detect active hosts, it can also be used as a raw packet generator for network stack stress tests, ARP poisoning, Denial of Service attacks, route tracing, and other purposes.

NMAP SUITES: NPING

USAGE

```
root@kali:/# nping 192.168.2.103

Starting Nping 0.7.12 ( https://nmap.org/nping ) at 2016-08-09 05:36 EDT
SENT (0.0399s) ICMP [192.168.2.106 > 192.168.2.103 Echo request (type=8/code=0)
id=4537 seq=1] IP [ttl=64 id=22536 iplen=28 ]
SENT (1.0410s) ICMP [192.168.2.106 > 192.168.2.103 Echo request (type=8/code=0)
id=4537 seq=2] IP [ttl=64 id=22536 iplen=28 ]
SENT (2.0436s) ICMP [192.168.2.106 > 192.168.2.103 Echo request (type=8/code=0)
id=4537 seq=3] IP [ttl=64 id=22536 iplen=28 ]
SENT (3.0459s) ICMP [192.168.2.106 > 192.168.2.103 Echo request (type=8/code=0)
id=4537 seq=4] IP [ttl=64 id=22536 iplen=28 ]
SENT (4.0478s) ICMP [192.168.2.106 > 192.168.2.103 Echo request (type=8/code=0)
id=4537 seq=5] IP [ttl=64 id=22536 iplen=28 ]

Max rtt: N/A | Min rtt: N/A | Avg rtt: N/A
Raw packets sent: 5 (140B) | Rcvd: 0 (0B) | Lost: 5 (100.00%)
Nping done: 1 IP address pinged in 5.05 seconds
root@kali:/#
```

NMAP SUITES: NPING

USAGE

```
root@kali:/# nping --tcp -p 80 --flags rst --ttl 2 192.168.2.103

Starting Nping 0.7.12 ( https://nmap.org/nping ) at 2016-08-09 05:43 EDT
SENT (0.0029s) TCP 192.168.2.106:36484 > 192.168.2.103:80 R ttl=2 id=35931 iplen
=40  seq=374298456 win=1480
SENT (1.0039s) TCP 192.168.2.106:36484 > 192.168.2.103:80 R ttl=2 id=35931 iplen
=40  seq=374298456 win=1480
SENT (2.0059s) TCP 192.168.2.106:36484 > 192.168.2.103:80 R ttl=2 id=35931 iplen
=40  seq=374298456 win=1480
SENT (3.0068s) TCP 192.168.2.106:36484 > 192.168.2.103:80 R ttl=2 id=35931 iplen
=40  seq=374298456 win=1480
SENT (4.0086s) TCP 192.168.2.106:36484 > 192.168.2.103:80 R ttl=2 id=35931 iplen
=40  seq=374298456 win=1480

Max rtt: N/A | Min rtt: N/A | Avg rtt: N/A
Raw packets sent: 5 (200B) | Rcvd: 0 (0B) | Lost: 5 (100.00%)
Nping done: 1 IP address pinged in 5.01 seconds
root@kali:/#
```


WAR10CK
F4RR3LL: how bad you think?
WAR10CK: very
F4RR3LL:
Nmap v 6.22
nmap -v -T4 -A insecure.org scanme.nmap.org 0/24
31.3.3.7 sectools.org
Starting Nmap 4.20ALPHA8 (http://Insecure.Org) at
2006-10-05 13:29 PDT
Initiating Parallel DNS resolution of 2 hosts. at 13:29
Completed Parallel DNS resolution of 2 hosts. at 13:29,
0.00s elapsed
Initiating ARP Ping Scan at 13:29
Scanning 0Bzz [1 port]
Completed ARP Ping Scan at 13:29, 0.01s elapsed
(1 total hosts)
Initiating Parallel DNS resolution of 1 host. at 13:29
Completed Parallel DNS resolution of 1 host. at 13:29,
0.00s elapsed
Initiating SYN Stealth Scan at 13:29
Scanning 2 hosts [1680 ports/host]
Discovered open port 22/tcp on 205.217.153.53
Discovered open port 22/tcp on 205.217.153.62
Discovered open port 80/tcp on 205.217.153.53
Discovered open port 80/tcp on 205.217.153.62
Discovered open port 53/tcp on 205.217.153.53
Discovered open port 53/tcp on 205.217.153.62
Completed SYN Stealth Scan against 205.217.153.53 in
25.94s (1 host left)
Completed SYN Stealth Scan at 13:30, 25.94s elapsed
(1680 total ports)
Initiating Service scan at 13:30
Scanning 6 services on 2 hosts
Completed Service scan at 13:30, 12.02s elapsed
(6 services on 2 hosts)
Initiating OS detection (try #1) against 2 hosts
Retrying OS detection (try #2) against 2 hosts
Initiating Nmap OS Detection against 205.217.153.53 at
41.87Ss
For OSScan assuming port 22 is open, 70 is closed, and

ZENMAP

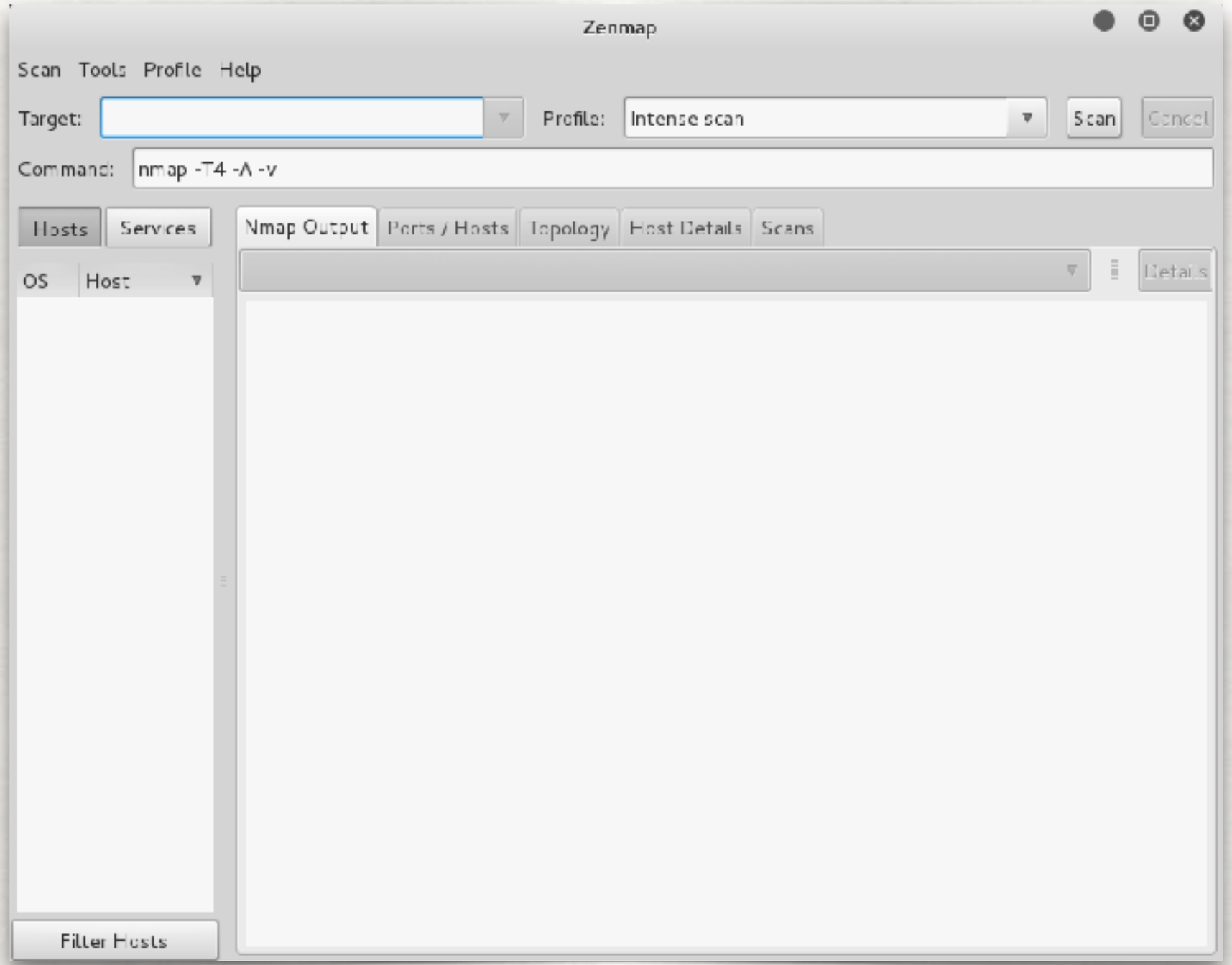
NMAP SUITES: ZENMAP

INTRODUCTION

- Zenmap is the official graphical user interface (GUI) for the Nmap Security Scanner.
- It is a multi-platform, free and open-source application designed to make Nmap easy for beginners to use while providing advanced features for experienced Nmap users.
- Scan results can be saved and viewed later. Saved scans can be compared with one another to see how they differ.
- The results of recent scans are stored in a searchable database.

NMAP SUITES: ZENMAP

USAGE



NMAP SUITES: ZENMAP

WHY GUI?

- Interactive and graphical results viewing
- Comparison
- Convenience
- Repeatability
- Discoverability

NMAP SUITES: ZENMAP

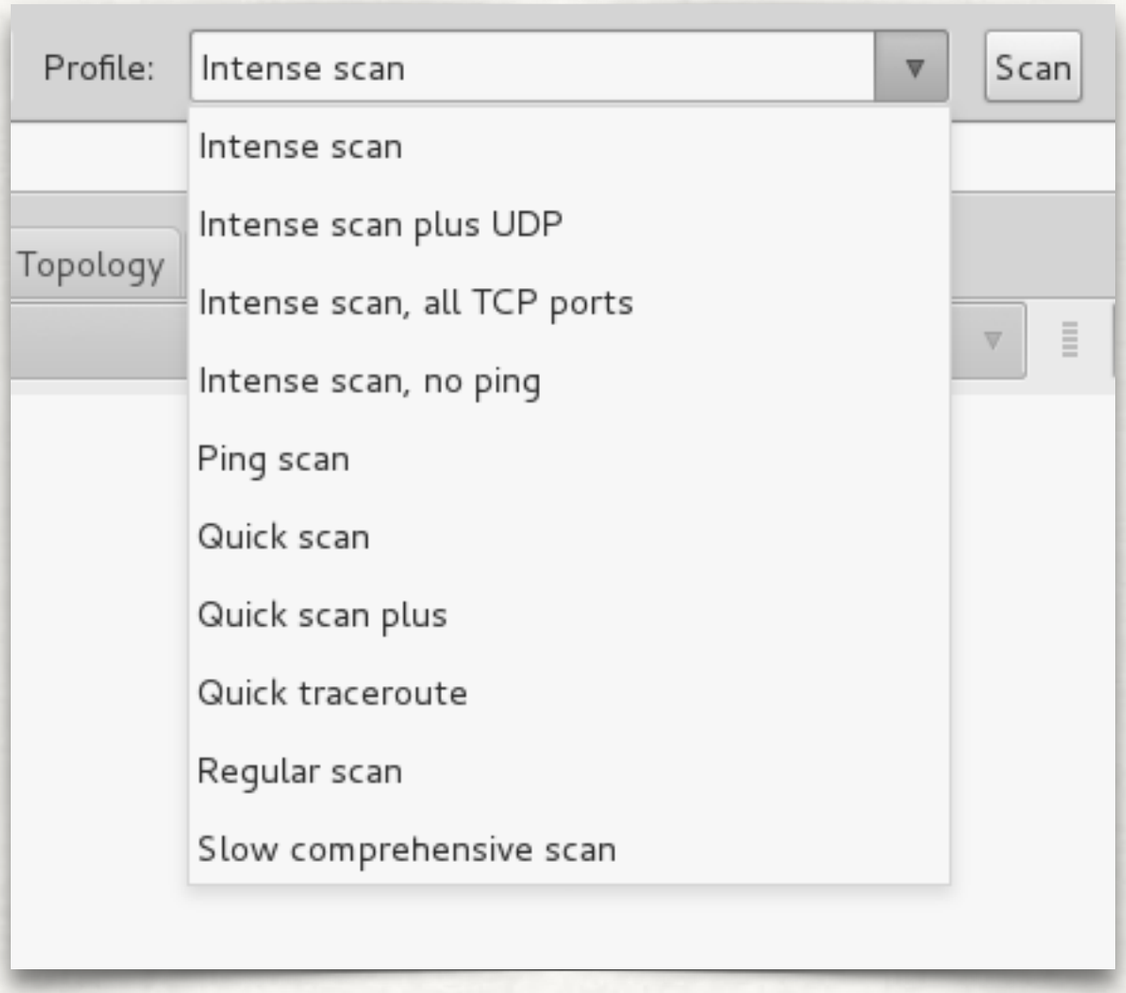
PROFILES

- The “Intense scan” is just one of several scan profiles that come with Zenmap. Choose a profile by selecting it from the “Profile” combo box. Profiles exist for several common scans. After selecting a profile the Nmap command line associated with it is displayed on the screen.

- It is also possible to type in an Nmap command and have it executed without using a profile. Just type in the command and press return or click “Scan”.

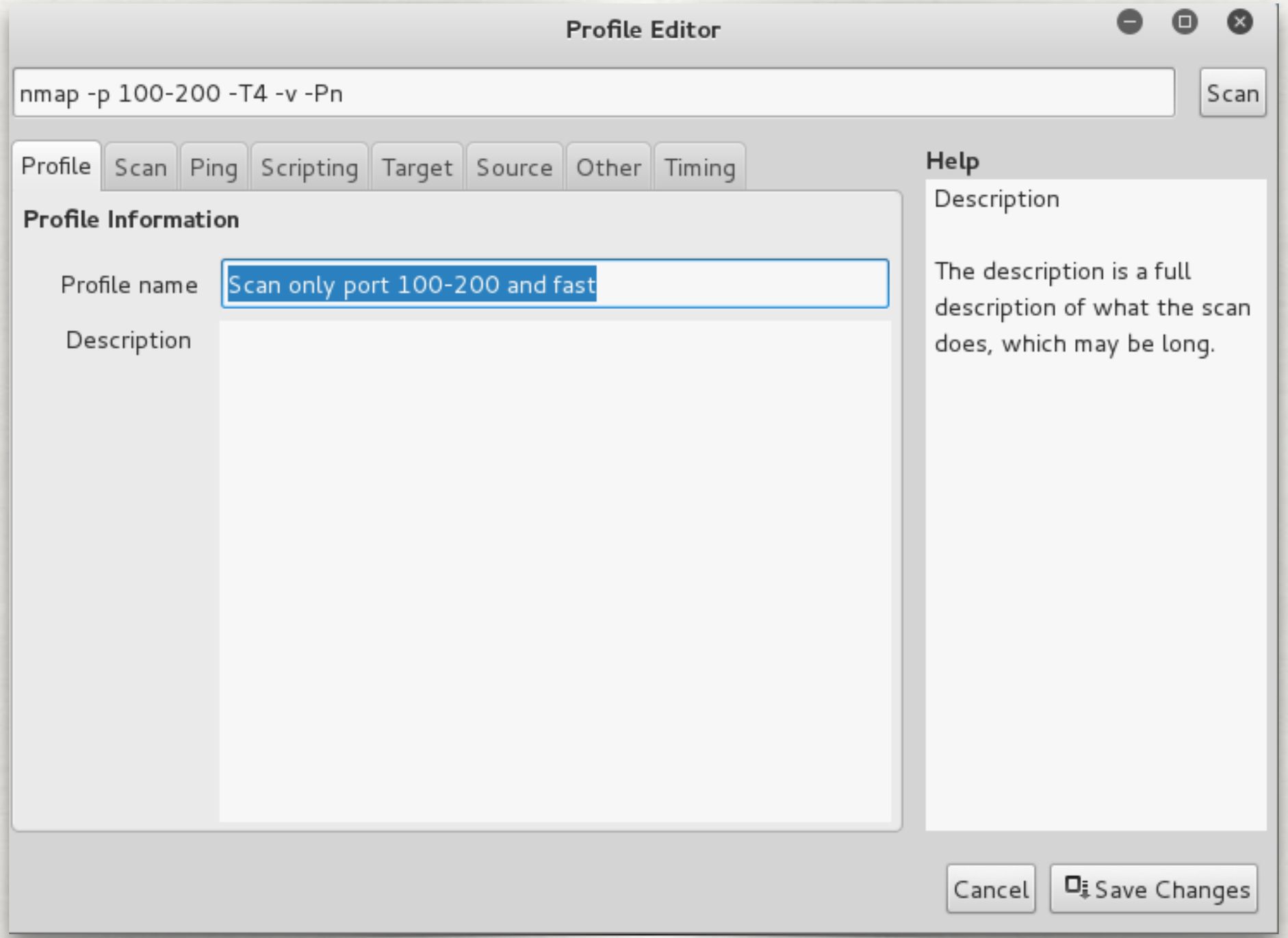
NMAP SUITES: ZENMAP

PROFILE



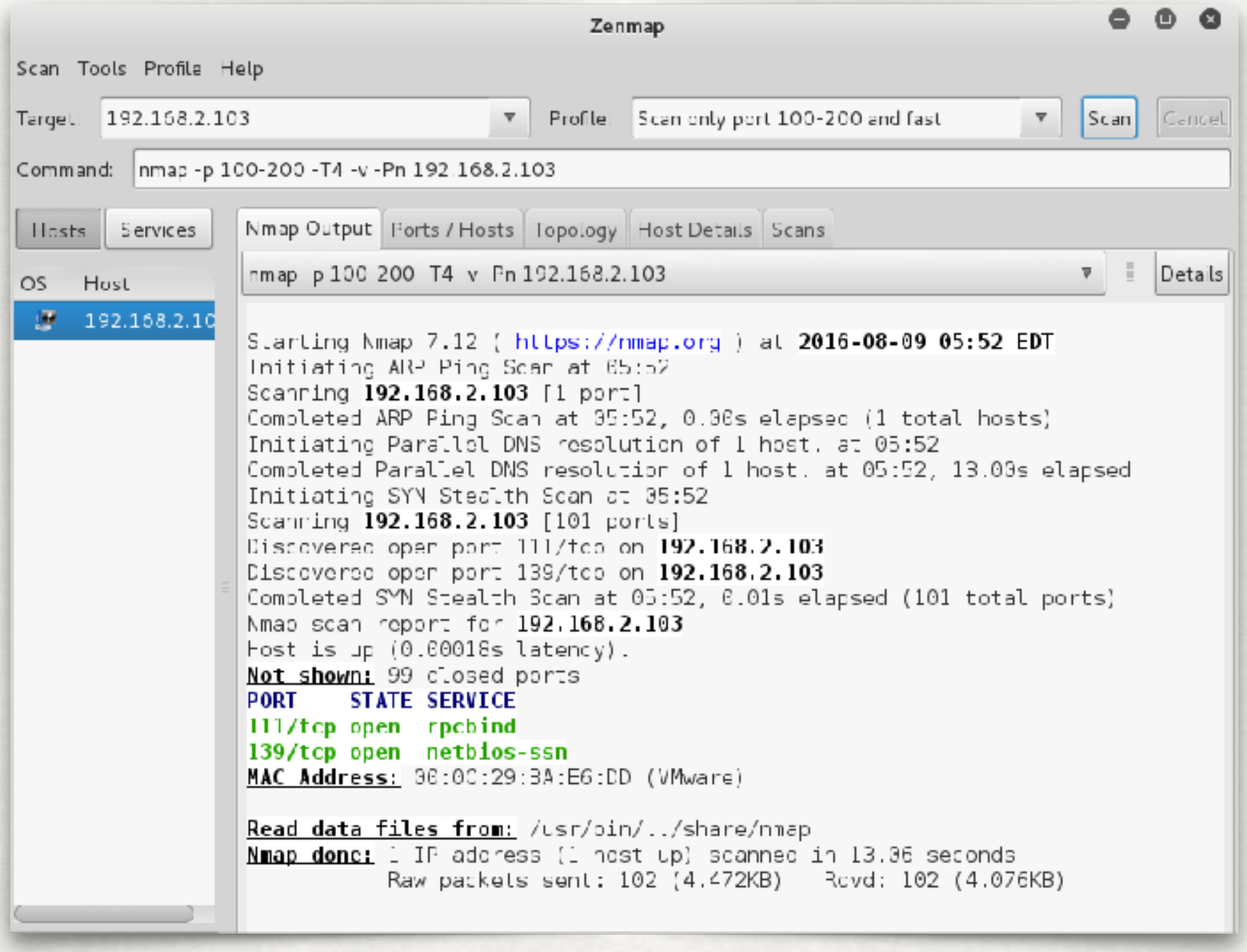
NMAP SUITES: ZENMAP

PROFILE



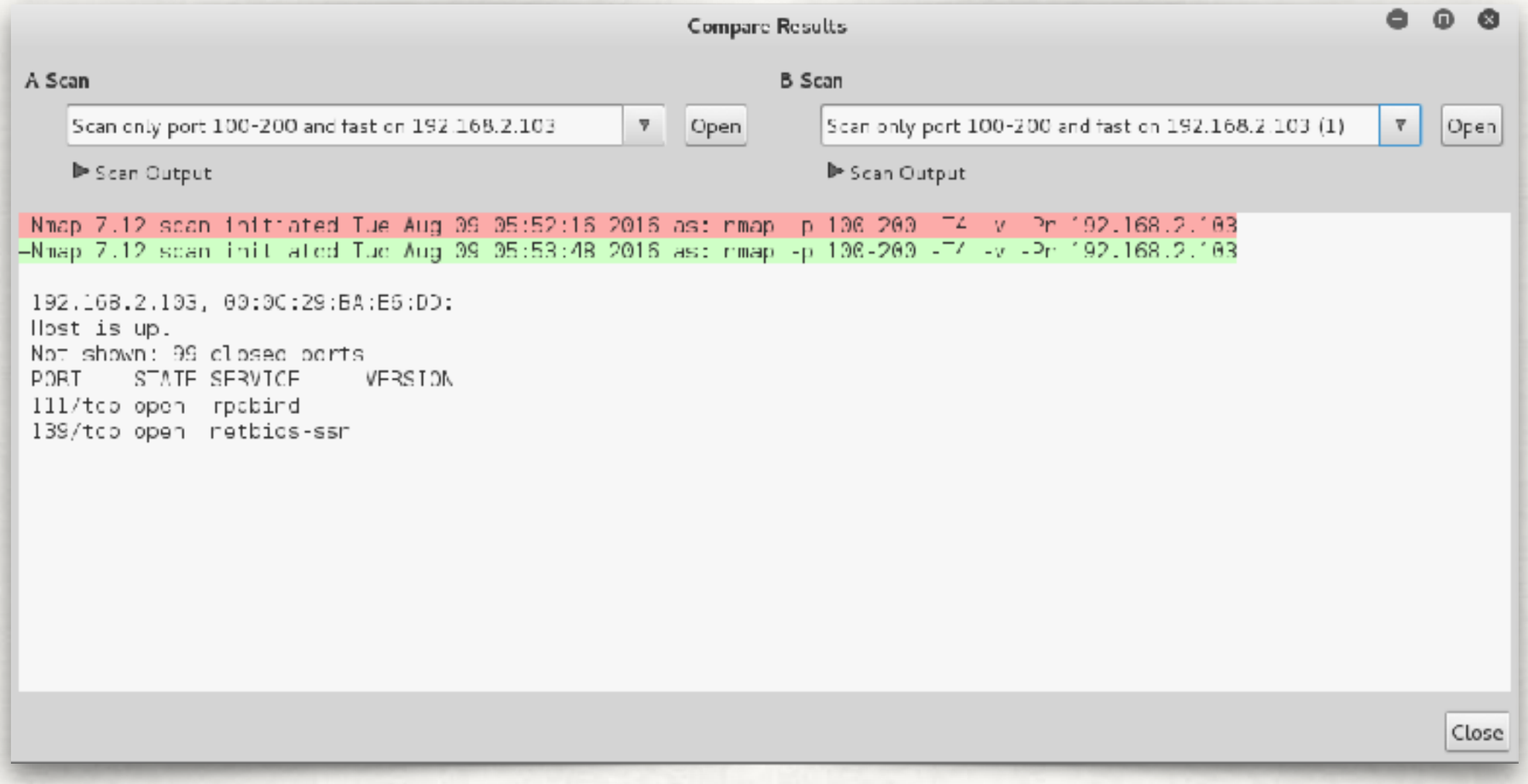
NMAP SUITES: ZENMAP

PROFILE



NMAP SUITES: ZENMAP

COMPARE



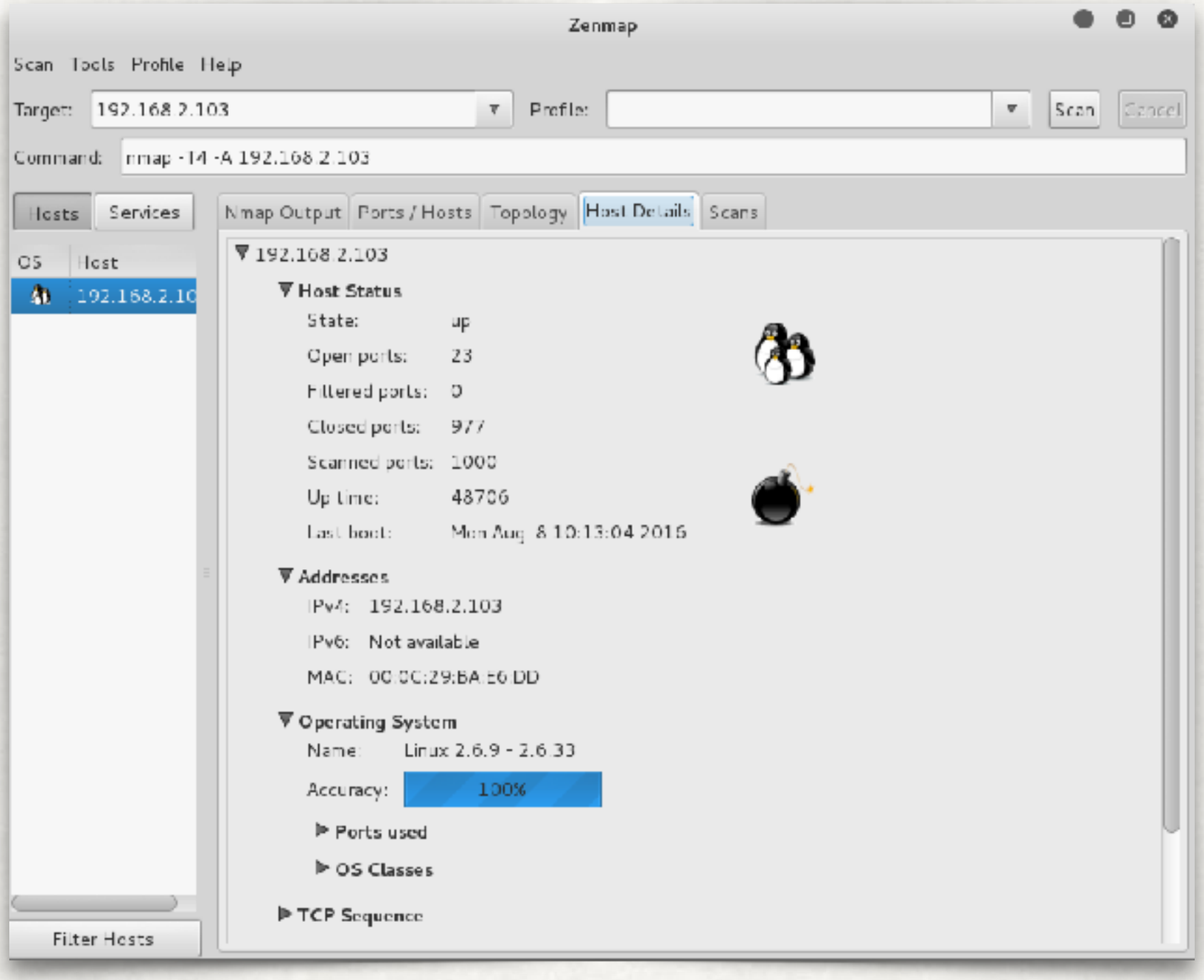
NMAP SUITES: ZENMAP

OTHERS

- Show topology
- Host details
- Add Nmap scan result.

NMAP SUITES: ZENMAP

HOSTS DETAIL



REFERENCE

- NMAP Manual
- NMAP Online book (free content) - https://nmap.org/book/