



Welcome to

Hackerworkshop PROSA

Henrik Kramselund he/him han/ham hlk@zencurity.com @kramse

Slides are available as PDF, kramse@Github
hackerworkshop-prosa.tex in the repo security-courses

Goals



Don't Panic!

Introduce the term penetration testing and basic pentest methods

Introduce some of the basic tools in this genre of hacker tools

Show a hacker lab and run some tools

Practice real-life scenarios - enabling you to do quality port scans!

Point you towards resources, so you can get started with the fun of pentesting tools

Plan



Subjects

- What is penetration testing – learn by using tools
- Get started with hacking, attacks etc.
- Nmap how to get started, and a small test plan, how to scan a network using Nmap
- Wi-Fi hacking introduction – you can borrow a wi-fi card

Demos and recommendations for exercises

- Running various tools
- Note: exercise booklets on Github contain much more than we can go through, continue on your own

Advanced topics:

If time permits, or you want to play around: Get started blasting packets, from single packets with Nping and Scapy, to Hping3 and PenguinPing

Agreements for testing networks



Danish Criminal Code

Straffelovens paragraf 263 Stk. 2. Med bøde eller fængsel indtil 1 år og 6 måneder straffes den, der uberettiget skaffer sig adgang til en andens oplysninger eller programmer, der er bestemt til at bruges i et informationssystem.

Hacking can result in:

- Getting your devices confiscated by the police
- Paying damages to persons or businesses
- If older getting a fine and a record – even jail perhaps
- Getting a criminal record, making it hard to travel to some countries and working in security
- Fear of terror has increased the focus – so dont step over bounds!

Asking for permission and getting an OK before doing invasive tests, always!

Tools and Materials – where to start



- The kickstart document – short and basic information
- This presentation – slides for today, start here
<https://github.com/kramse/security-courses/tree/master/courses/pentest/hackerworkshop>
- Install the Nmap package to join the basic exercises <https://nmap.org/download.html>
- Setup instructions for creating a Kali virtual machine – not needed:
<https://github.com/kramse/kramse-labs>
- Start a download of Kali today, if you want to play with the tools tomorrow
<https://www.kali.org/get-kali/>

Materials – other stuff you might want to look at



- Nmap Workshop exercises

<https://github.com/kramse/security-courses/blob/master/courses/pentest/nmap-workshop/nmap-workshop-exercises.pdf>

- KEA Pentest course exercises

<https://github.com/kramse/security-courses/blob/master/courses/pentest/kea-pentest/kea-pentest-exercises.pdf>

- Also the Simulated DDoS Workshop is available – among others:

<https://github.com/kramse/security-courses/tree/master/presentations/pentest/simulated-ddos-workshop>

We cannot go through all of them, but feel free to ask questions later

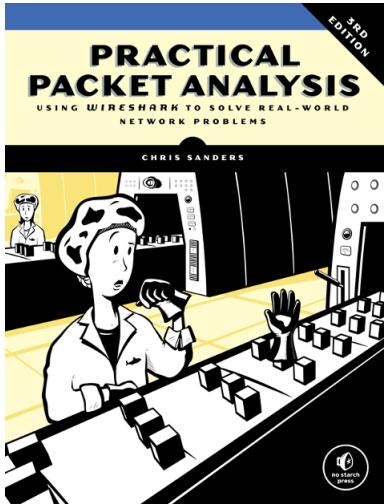
Books and Educational Materials



- *The Linux Command Line: A Complete Introduction*, 2nd Edition
by William Shotts, internet edition <https://sourceforge.net/projects/linuxcommand>
- *Linux Basics for Hackers Getting Started with Networking, Scripting, and Security in Kali*. OccupyTheWeb, December 2018, 248 pp. ISBN-13: 978-1-59327-855-7
- *Gray Hat Hacking: The Ethical Hacker's Handbook*, 6. ed. Allen Harper and others ISBN: 9781264268948
- *Web Application Security*, Andrew Hoffman, 2020, ISBN: 9781492053118
- *Practical Packet Analysis, Using Wireshark to Solve Real-World Network Problems* by Chris Sanders, 3rd ed, ISBN: 978-1-59327-802-1
- *Hacking, 2nd Edition: The Art of Exploitation*, Jon Erickson, February 2008, ISBN-13: 9781593271442
- *Kali Linux Revealed Mastering the Penetration Testing Distribution*
<https://www.kali.org/>

We teach using these books at Copenhagen School of Design and Technology (KEA)

Book: Practical Packet Analysis (PPA)



Practical Packet Analysis, Using Wireshark to Solve Real-World Network Problems by Chris Sanders, 3rd Edition
April 2017, 368 pp. ISBN-13: 978-1-59327-802-1 <https://nostarch.com/packetanalysis3>

Anything you know about technology will help you hack it, so learn basic functionality first

Hacker tools



We realize that SATAN is a two-edged sword – like many tools, it can be used for good and for evil purposes. We also realize that intruders (including wannabees) have much more capable (read intrusive) tools than offered with SATAN.

Source: <http://www.fish2.com/security/admin-guide-to-cracking.html>

- I got into hacking due to the security papers from Spaff and others, like this: *Improving the Security of Your Site by Breaking Into it*, Dan Farmer and Wietse Venema, 1993
- Later in 1995 released the software SATAN
Security Administrator Tool for Analyzing Networks
- Caused some commotion, panic and discussions, every script kiddie can hack, the Internet will melt down!

Use hacker tools!



Port scan can reveal holes in your defense

Web testing tools can crawl through your site and find problems

Pentesting is a verification and proactively finding problems

Its not a silverbullet and mostly find known problems in existing systems

Combined with honeypots they may allow better security

Hacker – cracker



Short answer – dont discuss this

Yes, originally there was another meaning to hacker, but the media has perverted it and today, and since early 1990s it has meant breaking into stuff for the public

Today a hacker breaks into systems!

Reference. Spafford, Cheswick, Garfinkel, Stoll, ...- wrote about this and it was lost

Story is interesting and the old meaning is ALSO used in smaller communities, like hacker spaces full of hackers - doing fun and interesting stuff

- *Cuckoo's Egg: Tracking a Spy Through the Maze of Computer Espionage*, Clifford Stoll
- *Hackers: Heroes of the Computer Revolution*, Steven Levy
- *Practical Unix and Internet Security*, Simson Garfinkel, Gene Spafford, Alan Schwartz

Why even do security testing?



Lots of security problems – finding vulnerable systems, discover hidden servers, forgotten servers
Pentesting may be a requirement from external partners – example VISA PCI standard

- Boss asking: should we do a security test?
- CIO: hmm, okay
- IT Admins: *sigh* – I know the security sucks in places!
- Its not your systems – dont take the criticism personal, but as an opportunity to get things improved
- Pentest tools are great resources for doing discovery of assets, evaluating the security of large installations quickly – in short using pentest tools makes you more efficient!

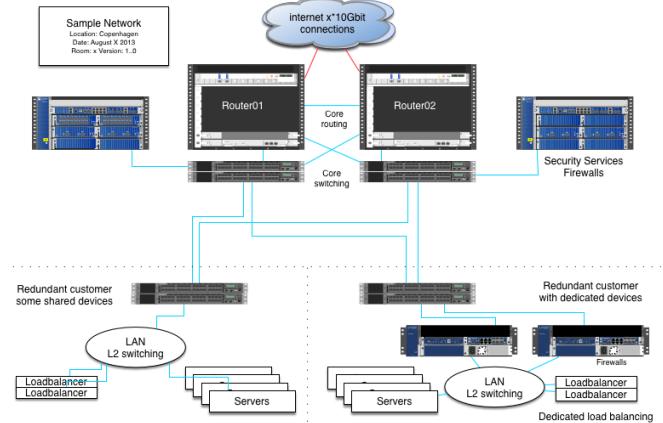
Many see the benefits after doing a pentest, so try it!

Planning a pentest



- Scope – what is being tested
- When is the testing done – time frame, wall clock time
- Where are the attacks coming from – log files will contain the attacks
but other attacks from other sources are likely during the attacks, which must be blocked
- We sometimes go broader than the scope – perhaps checking the router in front with SNMP or doing a small port 80/tcp scan
- Agree if Denial of Service is to be tested
- TL;DR Rules of engagement for the project

Selecting systems for testing



- Routers on the way to critical systems and networks – especially availability
- Firewall – is the environment protected sufficiently, discarding probes
- Mail servers – relay testing and also critical data
- Web servers – holds data, typically has a lot of functionality
- Cloud systems, storage systems, anywhere data is saved

Testing Agreement and Scope Example



Usually the scope could include targets like these:

- 10.0.45.1 – firewall/router
- 10.0.45.0/25 – servers
- Test to be done from monday 1st until friday 5th
- Testing done from 192.0.2.0/28

When testing web servers and sites, especially API – please include hostname, URLs, documentation. If not included some sites and functionality will NOT be tested!

Reporting – results



What is in a pentest report:

- Title, Table of contents, formal report thanks
- Confidentiality agreement – Write "Confidential" on each page
- Executive summary – big companies always want this
- Information about the scan done, what was it
- Scope and targets
- Review of all targets – detailed information and recommendations
- Conclusion – may be more technical
- Appendices – various information, Whois info about subnets and prefixes

When delivering a report, it is up to the organisation to decide which recommendations to implement

Sample report available at: <https://github.com/kramse/pentest-report>

Rules of engagement – rules and ethics for security testing



- NB: big difference between Denmark and other places!
- Security consultant must not be the cause of new vulnerabilities due to the testing
- Security consultant must not install new software on systems without previous agreement
- Security consultant is not to leave insecure system administrator accounts or settings after testing
- Security consultant always contact the customer in case of high-risk vulnerabilities
- It is allowed to peek around in the network – checking if there might be an insecure development or testing server near by
- If you meet other security problems outside of the scope we still report them, but perhaps in an appendix

In general be careful of other people networks and systems

Hacker tools



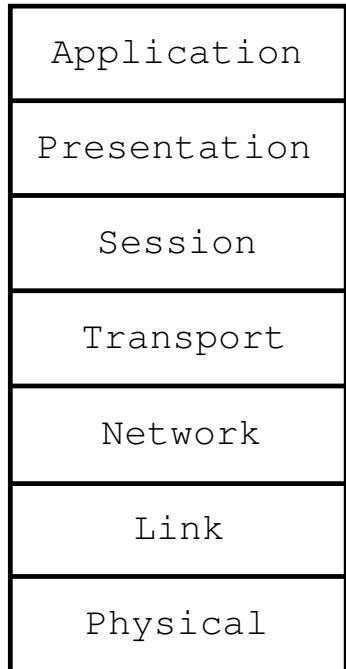
- Everyone use similar tools, see also <http://www.sectools.org/>
- Portscanning Nmap, Nping – test ports and services, Nping is great for firewall admins <https://nmap.org>
- Metasploit Framework – service scanning, exploit development and execution <https://www.metasploit.com/>
- Dedicated niche scanners – wifi Aircrack-ng, web Burp suite, Nikto, Skipfish <http://portswigger.net/burp/>
- Wireshark advanced network sniffing tool – <https://www.wireshark.org/>
- and scripting, PowerShell, Unix shell, Perl, Python, Ruby, ...

Picture: Acid Burn / Angelina Jolie Hackers 1995

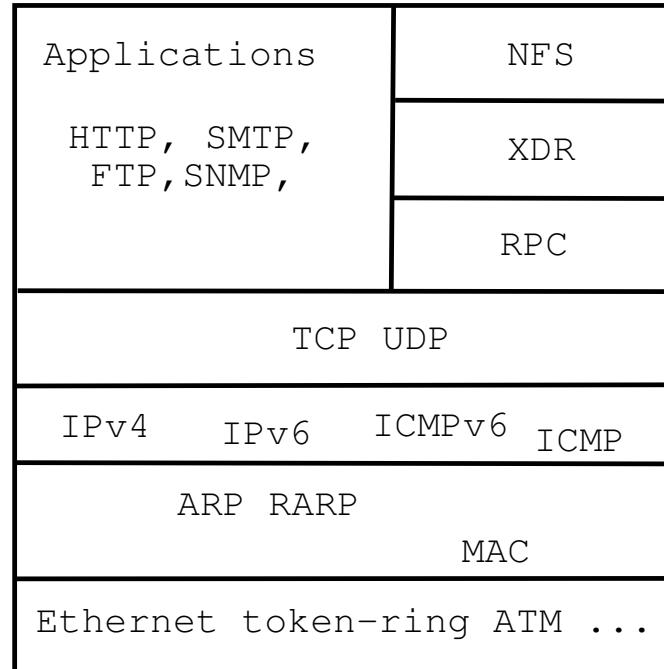
OSI Model and Internet Protocols



OSI Reference Model



Internet protocol suite





What happens now?

Think like a hacker

Reconnaissance

- ping sweep, port scan
- OS detection – TCP/IP or banner grabbing
- Service scan – rpcinfo, netbios, ...
- telnet/netcat interact with services

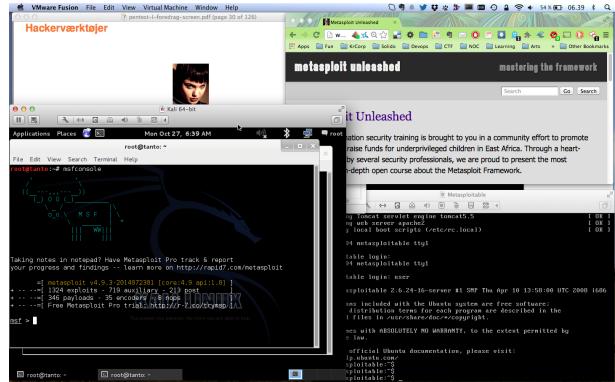
Exploit/test: Metasploit, Nikto, exploit programs

Cleanup/hardening not shown today, but:

- Make a report or document findings
- Change, improve and harden systems
- Go through report with stakeholders, track progress
- Update programs, settings, configurations, architecture

You also need to show others that you are in control of security

Hacker lab setup



- Hardware: any modern laptop with CPU and virtualisation
Don't forget to enable it in the BIOS
- Software: your favourite operating system Windows, Mac, Linux, ...
- Virtualisation software: VMware, Virtual box, pick your poison
- Hacker software: Kali as a Virtual Machine <https://www.kali.org/>
- Soft targets: Metasploitable, Linux, Microsoft Windows, Microsoft Exchange, Windows server, ...

Hackers don't give a shit

Your system is only for testing, development, ...

Your network is a research network, under construction, being phased out, ...

Try something new, go to your management

Bring all the exceptions, all of them, update the risk analysis figures - if this happens it is about 1mill DKK

Ask for permission to go full monty on your security

Think like attackers - don't hold back



The poster features a black and white illustration of a woman in a dark dress, standing and holding a chainsaw in one hand and a laptop in the other. She has a determined or slightly mischievous expression. The background is dark.

Hackers don't give a shit:

- About your project's scope
- It's managed by a third party
- It's a legacy system
- It's "too critical to patch"
- About your outage windows
- About your budget
- You've always done it that way
- About your Go-Live Date
- It's only a pilot/proof of concept
- About Non-Disclosure Agreements
- It wasn't a requirement in the contract
- It's an internal system
- It's really hard to change
- It's due for replacement
- You're not sure how to fix it
- It's handled in the Cloud
- About your Risk Register entry
- The vendor doesn't support that configuration
- It's an interim solution
- It's [insert standard here] compliant
- It's encrypted on disk
- The cost benefit doesn't stack up
- "Nobody else could figure that out"
- You can't explain the risk to "The Business"
- You've got other priorities
- About your faith in the competence of your internal users
- You don't have a business justification
- You can't show Return on Investment
- You contracted out that risk

KIWICON III
28TH & 29TH NOVEMBER 2009

New Zealand's Hacker con - Wellington

Hacking is magic



Hacking looks like magic

Hacking is not magic



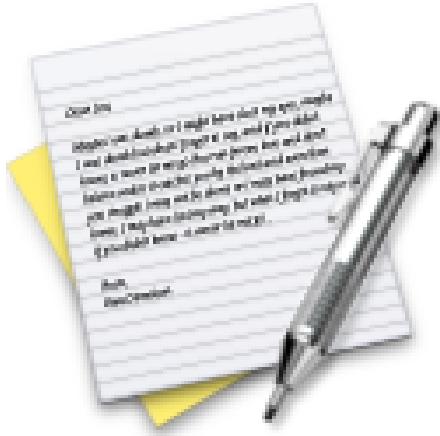
Hacking only demands ninja training and knowledge others don't have



Photo by Kelly Sikkema on Unsplash

Whats the goal, where are the strawberries!

Demo: airodump og aircrack



- Short demo
- Later try yourself, find exercises about Wardriving and Aircrack-ng in exercise booklet
- Getting started Aircrack-ng https://aircrack-ng.org/doku.php?id=getting_started
- You are welcome to *hack our wireless*, you can sniff and crack the password!
- **Warning to all: sniffing might occur – be nice**

Exercise



Now lets do the exercise

⚠ Wardriving Up to 15min

which is number **2** in the exercise PDF.

Hacking example – it is not magic



MAC filtering in IEEE 802.11 wireless networks

- Yes, network card chips have a globally unique MAC address – from production
- Access points allow filtering of frames based on MAC
- Only those matching an allowed list are forwarded – has access to network
- The method doesn't work for security though 😊
- First, most network cards and drivers allow you to change this MAC easily
- Second, you can read the allowed ones, as the active systems on the network
- Further there has been implementation problems in multiple access points

Myths about MAC filtering



The example with MAC filtering is a problematic myth

Why does it happen?

- Marketing – vendors would like to put as many "security features" on the labels and packages
- Customer knowledge – consumers know nothing about the technologies
Don't know what a MAC address is, and why should they
- We are quite few that can understand it, we know what a MAC address is (at least now)

Solutions

- We must spread information about insecure methods for securing data and systems
- We must spread information about secure methods for securing data and systems
- And update our own understanding of those methods, in both groups

MAC filtrering



Kali Linux the pentest toolbox



The most advanced penetration testing distribution, ever.

From the creators of BackTrack comes Kali Linux, the most advanced and versatile penetration testing distribution ever created. BackTrack has grown far beyond its humble roots as a live CD and has now become a full-fledged operating system. With all this buzz, you might be asking yourself: - What's new ?

KALI LINUX
"the quieter you become, the more you are able to hear"

**PENETRATION TESTING,
REDEFINED.**

A Project By Offensive Security

Kali <http://www.kali.org/>

100.000s of videos on youtube alone, searching for kali and \$TOOL

Also versions for Raspberry Pi, mobile and other small computers

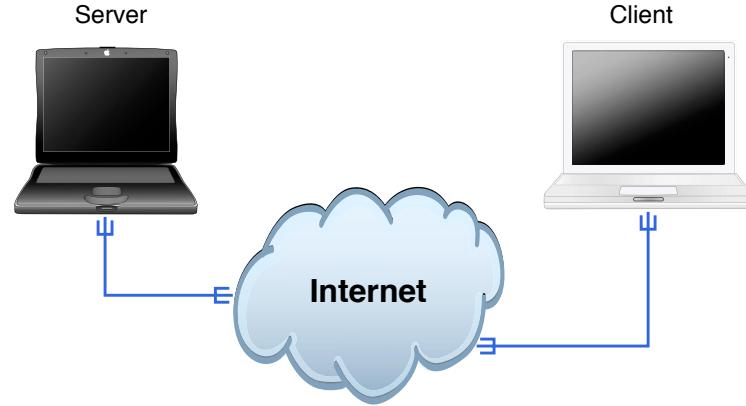
Technically what is hacking



```
main(int argc, char **argv)
{
    char buf[200];
    strcpy(buf, argv[1]);
    printf("%s\n", buf);
}
```



Internet today



Clients and servers

Rooted in academia

Protocols that are from 1983 and some older

Originally very little encryption, now mostly on https/TLS

Trinity breaking in



```
80/tcp      open      http
81/tcp      open      hosts2_ns
10 [mobile]
11 $ nmap -v -sS -O 10.2.2.2
11
13 Starting nmap 0.2.54BETA25
13 Insufficient responses for TCP sequencing (3), OS detection is
13 inaccurate
14 Interesting ports on 10.2.2.2:
14 (The 1539 ports scanned but not shown below are in state: cl
51 Port      State      Service
51 22/tcp    open       ssh
58
68 No exact OS matches for host
68
24 Nmap run completed -- 1 IP address (1 host up) scanned
50 $ sshnuke 10.2.2.2 -rootpw="Z10H0101"
Connecting to 10.2.2.2:ssh ... successful.
ReAttempting to exploit SSHv1 CRC32 ... successful.
IP Resetting root password to "Z10H0101".
System open: Access Level <9>
Hn $ ssh 10.2.2.2 -l root
root@10.2.2.2's password: ■
```

RTF CONTROL
ACCESS GRANTED

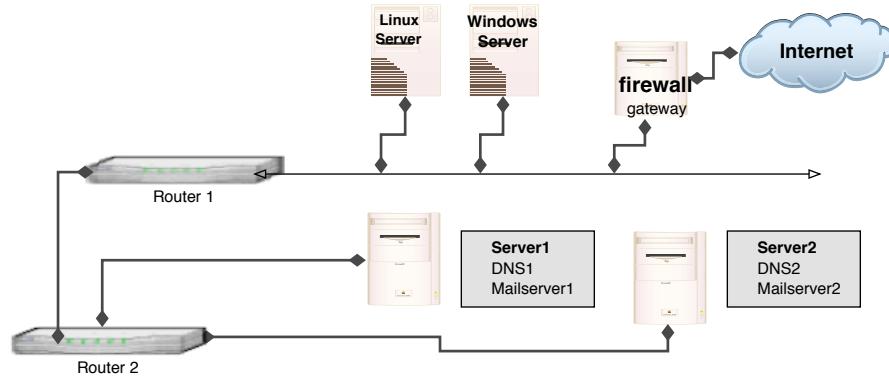
A terminal window showing the results of a nmap scan on host 10.2.2.2. The output shows port 80/tcp is open and serves an http service, while port 22/tcp is open and serves an ssh service. The user then runs sshnuke to exploit the host, successfully connecting via ssh and changing the root password to "Z10H0101". A separate window titled "RTF CONTROL" displays the message "ACCESS GRANTED".

Very realistic – comparable to hacking:

<https://nmap.org/movies/>

https://youtu.be/51IGCTgqE_w

Network mapping



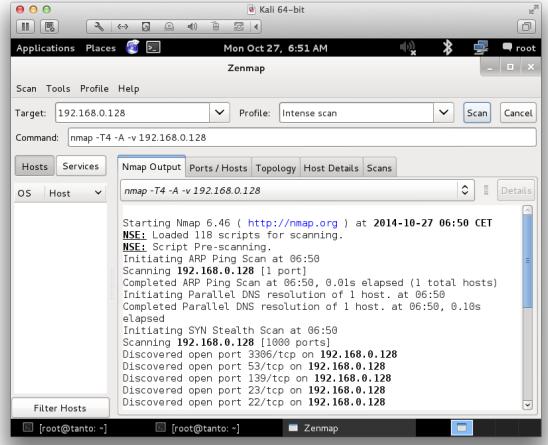
- Using traceroute, Nping and similar programs you can often discover topology information about a network
- Time to Live (TTL) for a packet is decremented for each router it crosses, if set low enough it will time out – and return ICMP message sent
- BTW Default Unix traceroute sends UDP packets, Windows tracert send ICMP packets
Use tools on Kali to try both protocols, or even others



traceroute – with UDP

```
# tcpdump -i en0 host 10.20.20.129 or host 10.0.0.11
tcpdump: listening on en0
23:23:30.426342 10.0.0.200.33849 > router.33435: udp 12 [ttl 1]
23:23:30.426742 safri > 10.0.0.200: icmp: time exceeded in-transit
23:23:30.436069 10.0.0.200.33849 > router.33436: udp 12 [ttl 1]
23:23:30.436357 safri > 10.0.0.200: icmp: time exceeded in-transit
23:23:30.437117 10.0.0.200.33849 > router.33437: udp 12 [ttl 1]
23:23:30.437383 safri > 10.0.0.200: icmp: time exceeded in-transit
23:23:30.437574 10.0.0.200.33849 > router.33438: udp 12
23:23:30.438946 router > 10.0.0.200: icmp: router udp port 33438 unreachable
23:23:30.451319 10.0.0.200.33849 > router.33439: udp 12
23:23:30.452569 router > 10.0.0.200: icmp: router udp port 33439 unreachable
23:23:30.452813 10.0.0.200.33849 > router.33440: udp 12
23:23:30.454023 router > 10.0.0.200: icmp: router udp port 33440 unreachable
23:23:31.379102 10.0.0.200.49214 > safri.domain: 6646+ PTR?
200.0.0.10.in-addr.arpa. (41)
23:23:31.380410 safri.domain > 10.0.0.200.49214: 6646 NXDomain* 0/1/0 (93)
14 packets received by filter
0 packets dropped by kernel
```

Really do Nmap your world



- Nmap is a port scanner, but does more
- Finding your own infrastructure available from the guest network?
- See your printers having all the protocols enabled AND a wireless?

Basic Portscan



What is port scanning

Testing all ports from 0/1 up to 65535

Goal is to identify open ports – vulnerable services

Typically TCP and UDP scans

TCP scanning is more reliable than UDP scanning

TCP handshake is easy to see, due to session setup – services must respond to SYN with SYN-ACK. Otherwise client programs like browsers will not work

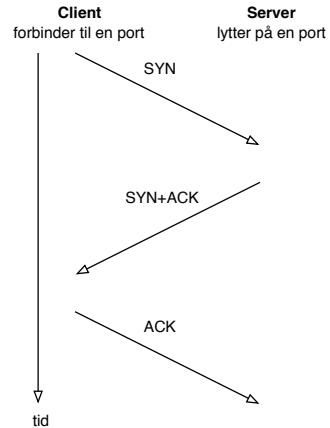
UDP applications respond differently – if at all

They might respond to queries and probes in the correct format,

If no firewall the operating systems will respond with ICMP on closed ports

Use Zenmap while learning Nmap

TCP three-way handshake



- **TCP SYN half-open** scans
- In the old days systems would only log a full TCP connection – so a port scanner sending only SYN would be doing a *stealth*-scans. Today we have Intrusion Detection Systems, so a lot of SYN without ever completing the connection is MORE suspicious
- Note: sending many SYN packets can fill the session table on firewalls, and on servers – preventing new connections – also called **SYN-flooding**

Ping and port sweep



Scanning across a network is called sweeping

Scans using ICMP ping will be a ping-sweep – active IPs

Scans using specific ports are port-sweeps

Easy to detect using modern intrusion detection systems (IDS)

Pro tip: If you are looking for an IDS, look at Suricata <https://suricata.io> and Zeek <https://zeek.org/> – together



Nmap port sweep for web services

```
root@cornerstone:~# nmap -p80,443 172.29.0.0/24
```

```
Starting Nmap 6.47 ( http://nmap.org ) at 2015-02-05 07:31 CET
Nmap scan report for 172.29.0.1
Host is up (0.00016s latency).
PORT      STATE      SERVICE
80/tcp    open       http
443/tcp   filtered  https
MAC Address: 00:50:56:C0:00:08 (VMware)
```

```
Nmap scan report for 172.29.0.138
Host is up (0.00012s latency).
PORT      STATE      SERVICE
80/tcp    open       http
443/tcp   closed    https
MAC Address: 00:0C:29:46:22:FB (VMware)
```

Nmap port sweep after SNMP port 161/UDP



```
root@cornerstone:~# nmap -sU -p 161 172.29.0.0/24
Starting Nmap 6.47 ( http://nmap.org ) at 2015-02-05 07:30 CET
Nmap scan report for 172.29.0.1
Host is up (0.00015s latency).
PORT      STATE      SERVICE
161/udp  open|filtered  snmp
MAC Address: 00:50:56:C0:00:08 (VMware)

Nmap scan report for 172.29.0.138
Host is up (0.00011s latency).
PORT      STATE      SERVICE
161/udp  closed  snmp
MAC Address: 00:0C:29:46:22:FB (VMware)
...
Nmap done: 256 IP addresses (5 hosts up) scanned in 2.18 seconds
```

Often possible on the inside LAN, where less firewalls are enabled



Nmap Advanced OS detection

```
root@cornerstone:~# nmap -A -p80,443 172.29.0.0/24
Starting Nmap 6.47 ( http://nmap.org ) at 2015-02-05 07:37 CET
Nmap scan report for 172.29.0.1
Host is up (0.00027s latency).

PORT      STATE      SERVICE VERSION
80/tcp    open       http      Apache httpd 2.2.26 ((Unix) DAV/2 mod_ssl/2.2.26 OpenSSL/0.9.8zc)
|_http-title: Site doesn't have a title (text/html).
443/tcp   filtered https
MAC Address: 00:50:56:C0:00:08 (VMware)
Device type: media device|general purpose|phone
Running: Apple iOS 6.X|4.X|5.X, Apple Mac OS X 10.7.X|10.9.X|10.8.X
OS details: Apple iOS 6.1.3, Apple Mac OS X 10.7.0 (Lion) - 10.9.2 (Mavericks)
or iOS 4.1 - 7.1 (Darwin 10.0.0 - 14.0.0), Apple Mac OS X 10.8 - 10.8.3 (Mountain Lion)
or iOS 5.1.1 - 6.1.5 (Darwin 12.0.0 - 13.0.0)
OS and Service detection performed.
Please report any incorrect results at http://nmap.org/submit/
```

- Low level operating system identification, often I use nmap -A
- Send packets, observe responses, match with tables of known operating system fingerprints
- An early reference for this was: *ICMP Usage In Scanning* Version 3.0, Ofir Arkin, 2001



Scan for Heartbleed and SSLv2/SSLv3

Nmap includes Nmap scripting engine (NSE)

Example Usage

```
nmap -sV -sC <target>
```

Script Output

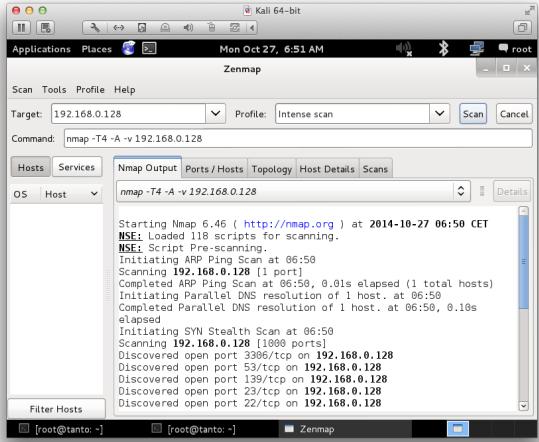
```
443/tcp open  https  syn-ack
| sslv2:
|   SSLv2 supported
|   ciphers:
|     SSL2_DES_192_EDE3_CBC_WITH_MD5
|     SSL2_IDEA_128_CBC_WITH_MD5
|     SSL2_RC2_CBC_128_CBC_WITH_MD5
|     SSL2_RC4_128_WITH_MD5
|     SSL2_DES_64_CBC_WITH_MD5
|     SSL2_RC2_CBC_128_CBC_WITH_MD5
|_    SSL2_RC4_128_EXPORT40_WITH_MD5
```

```
nmap -p 443 --script ssl-heartbleed <target>
```

<https://nmap.org/nsedoc/scripts/ssl-heartbleed.html>

Almost every new popular vulnerability will have Nmap recipe

Demo: Nmap and Zenmap



- Short demo, Nmap, Zenmap – and don't forget Nping
- After the demo we will let you scan!

This is a main part of the workshop, using Nmap to discover systems, identify services, extract information

Exercise



Now lets do the exercise

⚠ Execute nmap TCP and UDP port scan 15 min

which is number 4 in the exercise PDF.

Exercise

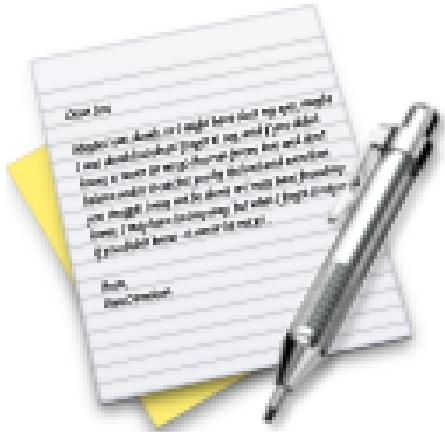


Now lets do the exercise

⚠ Discover active systems ping and port sweep 15 min

which is number **5** in the exercise PDF.

Exercise



Now lets do the exercise

⚠ Perform nmap OS detection 10 min

which is number **6** in the exercise PDF.

Exercise



Now lets do the exercise

⚠ Perform nmap service scan 30 min

which is number **7** in the exercise PDF.

Exercise



Now lets do the exercise

⚠ Nmap full scan – can take hours

which is number **8** in the exercise PDF.

Exercise

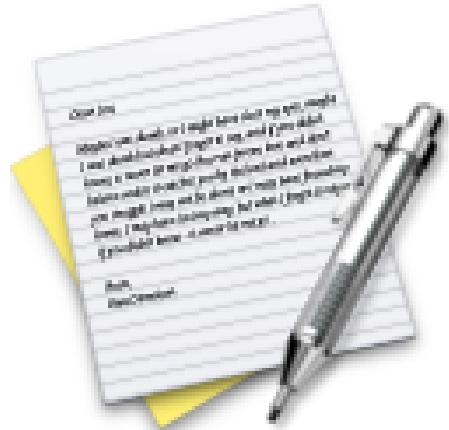


Now lets do the exercise

i Reporting HTML 10 min

which is number **9** in the exercise PDF.

Exercise



Now lets do the exercise

i Nping check ports 10min

which is number **10** in the exercise PDF.

Exercise



Now lets do the exercise

❶ Nmap Scripting Engine NSE scripts 30min

which is number **11** in the exercise PDF.



Nping from the Nmap package

```
root@KaliVM:~# nping --tcp -p 80 www.zecurity.com
```

```
Starting Nping 0.7.70 ( https://nmap.org/nping ) at 2018-09-07 19:06 CEST
SENT (0.0300s) TCP 10.137.0.24:3805 > 185.129.63.130:80 S ttl=64 id=18933 iplen=40 seq=2984847972 win=1480
RCVD (0.0353s) TCP 185.129.63.130:80 > 10.137.0.24:3805 SA ttl=56 id=49674 iplen=44 seq=3654597698 win=16384 <mss 1460>
SENT (1.0305s) TCP 10.137.0.24:3805 > 185.129.63.130:80 S ttl=64 id=18933 iplen=40 seq=2984847972 win=1480
RCVD (1.0391s) TCP 185.129.63.130:80 > 10.137.0.24:3805 SA ttl=56 id=50237 iplen=44 seq=2347926491 win=16384 <mss 1460>
SENT (2.0325s) TCP 10.137.0.24:3805 > 185.129.63.130:80 S ttl=64 id=18933 iplen=40 seq=2984847972 win=1480
RCVD (2.0724s) TCP 185.129.63.130:80 > 10.137.0.24:3805 SA ttl=56 id=9842 iplen=44 seq=2355974413 win=16384 <mss 1460>
SENT (3.0340s) TCP 10.137.0.24:3805 > 185.129.63.130:80 S ttl=64 id=18933 iplen=40 seq=2984847972 win=1480
RCVD (3.0387s) TCP 185.129.63.130:80 > 10.137.0.24:3805 SA ttl=56 id=1836 iplen=44 seq=3230085295 win=16384 <mss 1460>
SENT (4.0362s) TCP 10.137.0.24:3805 > 185.129.63.130:80 S ttl=64 id=18933 iplen=40 seq=2984847972 win=1480
RCVD (4.0549s) TCP 185.129.63.130:80 > 10.137.0.24:3805 SA ttl=56 id=62226 iplen=44 seq=3033492220 win=16384 <mss 1460>
```

```
Max rtt: 40.044ms | Min rtt: 4.677ms | Avg rtt: 15.398ms
Raw packets sent: 5 (200B) | Rcvd: 5 (220B) | Lost: 0 (0.00%)
Nping done: 1 IP address pinged in 4.07 seconds
```

Awesome tool for testing firewall rules, sending probes with specific source port etc.

Buffer overflows a C problem

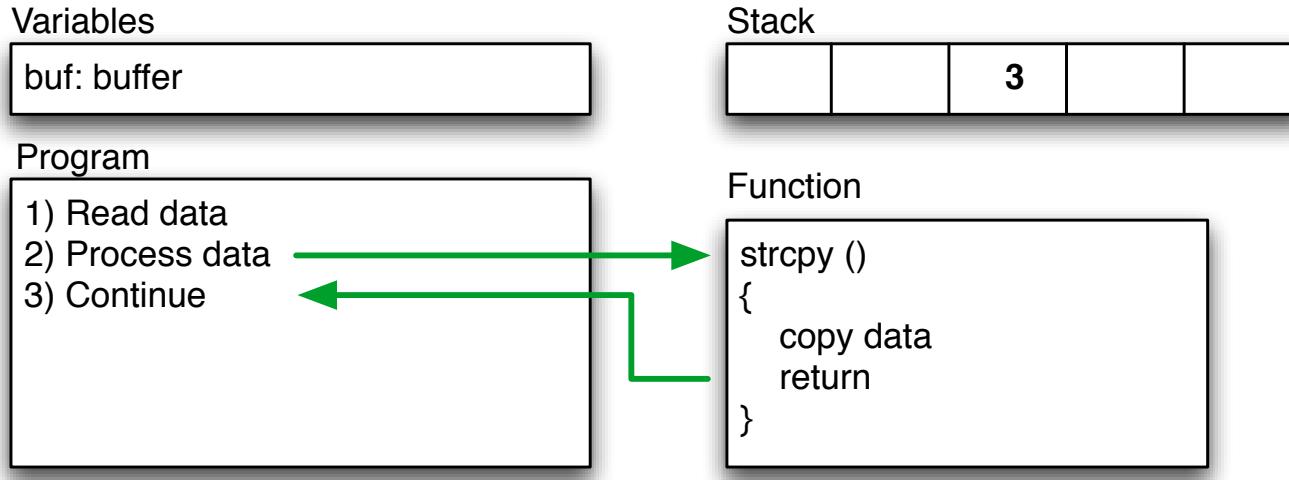


A **buffer overflow** is what happens when writing more data than allocated in some area of memory. Typically the program will crash, but under certain circumstances an attacker can write structures allowing take over of return addresses, parameters for system calls or program execution.

Stack protection is today used as a generic term for multiple technologies used in operating systems, libraries, compilers etc. that protect the stack and other structures from being overwritten or changed through buffer overflows. StackGuard and Propolice are examples of this.

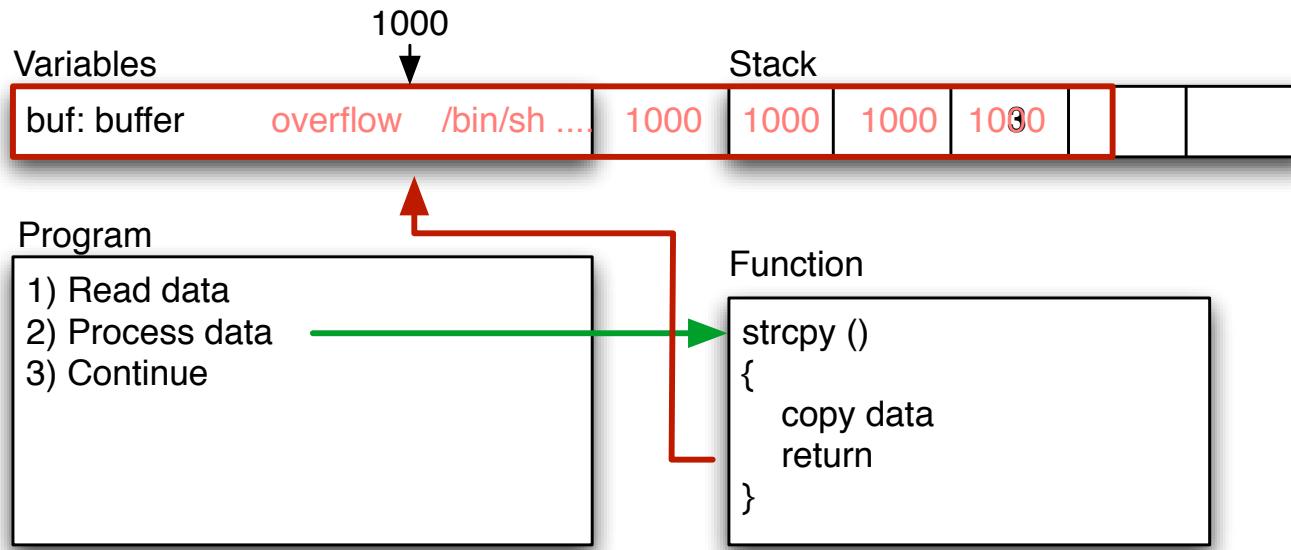
Today we will not go more into detail about this, suffice it to say modern operating systems really employ a lot of methods for making buffer overflows harder and less likely to succeed. OpenBSD even relink the kernel on installation to randomize addresses.

Buffers and stacks, simplified



```
main(int argc, char **argv)  
{    char buf[200];  
    strcpy(buf, argv[1]);  
    printf("%s\n", buf);  
}
```

Overflow – segmentation fault



- Bad function overwrites return value!
- Control return address
- Run shellcode from buffer, or from other place

Exploits – abusing a vulnerability



```
$buffer = "";
>null = "\x00";
$nop = "\x90";
$nopsiz = 1;
$len = 201; // what is needed to overflow, maybe 201, maybe more!
$the_shell_pointer = 0x01101d48; // address where shellcode is
# Fill buffer
for ($i = 1; $i < $len;$i += $nopsiz) {
    $buffer .= $nop;
}
$address = pack('l', $the_shell_pointer);
$buffer .= $address;
exec "$program", "$buffer";
```

- Exploit/exploit program are designed to exploit a specific vulnerability, often a specific version on a specific release on a specific CPU architecture
- Might be a 5 line program written in Perl, Python or a C program
- Today we often see them as modules written for Metasploit allowing it to be combined with different payloads

Kali virtual machine – working with buffer overflow



The screenshot shows a Kali Linux desktop environment with two terminal windows open. The top terminal window is titled 'Terminal' and shows the command 'compile.sh' being run in a directory. The bottom terminal window is titled 'root@kali:' and shows the source code for 'overflow.c', which contains a buffer overflow vulnerability. It also shows the user running GDB on the program and observing a segmentation fault at address 0x7004343.

```
compile.sh --- /GHHv5/ch11 --- Atom
File Edit View Selection Find Packages Help
compile.sh overflow.c
1 gcc -o overflow32 -m32 overflow.c -ggdb -mpreferred-stack-boundary=4 -fno-stack-protector -z execstack -no-pie
2 gcc -o overflow64 overflow.c -ggdb -mpreferred-stack-boundary=4 -fno-stack-protector -z execstack -no-pie
3

//overflow.c
#include <string.h>
main(){
    char str1[10]; //declare a 10 byte string
    //next, copy 35 bytes of "A" to str1
    strcpy (str1, "AAAAAAAAAAABCC");
}
root@kali:~/GHHv5/ch11# gdb -q overflow64
Reading symbols from overflow64...
(gdb) run
Starting program: /root/GHHv5/ch11/overflow64
[Inferior 1 (process 61914) exited normally]
(gdb) quit
root@kali:~/GHHv5/ch11# gdb -q overflow32
Reading symbols from overflow32...
(gdb) run
Starting program: /root/GHHv5/ch11/overflow32
Program received signal SIGSEGV, Segmentation fault.
0x7004343 in ?? ()
(gdb) 
```

- Example screenshot from a session with a small vulnerable program
- Notice the hex return address ends in 4343 which are the characters/bytes sent as "CC"

How to find these buffer overflows



Black box testing

Closed source reverse engineering

White box testing

Open source read and analyze the code – tools exist

Trial and error – fuzzing inputs to a program, save crashes, analyze them

Reverse engineer specific updates, so this part was changed, nice – this is where the bug is

Terms used



- **Privilege escalation** is when a privileged program is vulnerable and can be abused to escalate privileges. Example from unauthenticated user to a user account, or from regular user and becoming administrator (root on Unix) or even SYSTEM on Windows. Kernels and drivers are also often susceptible to this
- **Local vs. remote** exploit describe if the attack is done over some network, or locally on a system
- **Remote root exploit** are the worst kind, since they work over the network, and gives complete control aka root on Unix
- **Zero-day exploits** is a term used for those exploits that suddenly pop up, without previous warning. Often found during incident response at some network. We prefer that security researchers that discover a vulnerability uses a **responsible disclosure** process that involves the vendor .
- **Remote Code Execution (RCE)** exploit going over the network executing commands/code

Insecure programming buffer overflows 101



- Small demo program `demo.c`, try on older Linux or Raspberry Pi
- Has built-in shell code
- Compile: `gcc -o demo demo.c`
- Run program `./demo test`
- Goal: Break and insert return address

```
main(int argc, char **argv)
{
    char buf[10];
    strcpy(buf, argv[1]);
    printf("%s\n",buf);
}
the_shell()
{ system("/bin/sh"); }
```

GDB GNU Debugger



GNU compiler and debugger are OK for this, can fit on a slide!

Lots of other debuggers exist

Try `gdb ./demo` and run the program with some input from the *gdb prompt* using `run 1234`

When you realize the input overflows the buffer, crashed program execution you can work towards getting the address from `nm demo` of the function `the_shell` – and into the program

Use: `nm demo | grep shell`

The art is to generate a string long enough to overflow, and having the correct data, so the address ends up in the right place

Perl be used for generating AA...AAA like this, with back ticks, ``perl -e "print 'A'x10``

Debugging af C med GDB



Test with input

- ./demo longstringwithalotofdatyacrashtheprogram
- gdb demo followed by
run AAAAAAAAAAAAAAAAAAAAAAA
- Compile program: gcc -o demo demo.c
- Run program ./demo 123456...7689 until it dies
- Then retry in GDB

GDB output



Pentest I – Introduktion og basale metoder

```
hlk@bigfoot:demo$ gdb demo
GNU gdb 5.3-20030128 (Apple version gdb-330.1) (Fri Jul 16 21:42:28 GMT 2004)
Copyright 2003 Free Software Foundation, Inc.
GDB is free software, covered by the GNU General Public License, and you are
welcome to change it and/or distribute copies of it under certain conditions.
Type "show copying" to see the conditions.
There is absolutely no warranty for GDB. Type "show warranty" for details.
This GDB was configured as "powerpc-apple-darwin".
Reading symbols for shared libraries .. done
(gdb) run AAAAAAAAAAAAAAAAAAAAAAAA
Starting program: /Volumes/userdata/projects/security/exploit/demo/demo AAAAAAAAAAAAAAAAAAAAAAAA
Reading symbols for shared libraries . done
AAAAAAAAAAAAAAAAAAAAAAA

Program received signal EXC_BAD_ACCESS, Could not access memory.
0x41414140 in ?? ()
(gdb)
```

The Exploit Database – dagens buffer overflow



EXPLOIT DATABASE

GET CERTIFIED

Show 15 ▾

Verified Has App

▼ Filters % Reset All

Search:

Date	D	A	V	Title	Type	Platform	Author
2019-02-25				Drupal < 8.6.9 - REST Module Remote Code Execution	WebApps	PHP	leonjza
2019-02-25				Xlight FTP Server 3.9.1 - Buffer Overflow (PoC)	DoS	Windows	Logan Whitmire
2019-02-25				Advance Gift Shop Pro Script 2.0.3 - SQL Injection	WebApps	PHP	Mr Winst0n
2019-02-25				News Website Script 2.0.5 - SQL Injection	WebApps	PHP	Mr Winst0n
2019-02-25				PHP Ecommerce Script 2.0.6 - Cross-Site Scripting / SQL Injection	WebApps	PHP	Mr Winst0n
2019-02-25				zzzphp CMS 1.6.1 - Remote Code Execution	WebApps	PHP	Yang Chenglong
2019-02-25				Jenkins Plugin Script Security 1.49/Declarative 1.3.4/Groovy 2.60 - Remote Code Execution	WebApps	Java	wetw0rk
2019-02-23				Drupal < 8.6.10 / < 8.5.11 - REST Module Remote Code Execution	WebApps	PHP	Charles Fol
2019-02-22				Teracue ENC-400 - Command Injection / Missing Authentication	WebApps	Hardware	Stephen Shkadoon
2019-02-22				Micro Focus Flir 3.4.0.217 - Path Traversal / Local Privilege Escalation	WebApps	Linux	SecureAuth
2019-02-22				Nuuo Central Management - Authenticated SQL Server SQL Injection (Metasploit)	Remote	Windows	Metasploit
2019-02-22				WebKit JSC - reifyStaticProperty Needs to set the PropertyAttribute::CustomAccessor flag for CustomGetterSetter	DoS	Multiple	Google Security Research
2019-02-22				Quest NetVault Backup Server < 11.4.5 - Process Manager Service SQL Injection / Remote Code Execution	WebApps	Multiple	Chris Anastasio
2019-02-21				AirDrop 2.0 - Denial of Service (DoS)	DoS	Android	s4vitar
2019-02-21				MikroTik RouterOS < 6.43.12 (stable) / < 6.42.12 (long-term) - Firewall and NAT Bypass	Remote	Hardware	Jacob Baines

Showing 1 to 15 of 40,914 entries

FIRST PREVIOUS 1 2 3 4 5 ... 2728 NEXT LAST

Check <http://www.exploit-db.com/> and <https://www.exploit-db.com/google-hacking-database> and use Metasploit first!

Metasploit



The world's most used penetration testing framework

Knowledge is power, especially when it's shared. A collaboration between the open source community and Rapid7, Metasploit helps security teams do more than just verify vulnerabilities, manage security assessments, and improve security awareness; it empowers and arms defenders to always stay one step (or two) ahead of the game.

Source: <https://www.metasploit.com/>

- Has database support and integration with Nmap using the command `db_nmap`
- Includes various scanners: <https://www.offsec.com/metasploit-unleashed/scanner-snmp-auxiliary-modules/>
- Metasploit Unleashed - Free Online Ethical Hacking Course
<https://www.offsec.com/metasploit-unleashed/>

Transport Layer Security (TLS)



Originally from Netscape Communications Inc.

Secure Sockets Layer (SSL) was adopted by the IETF newer versions are called Transport Layer Security (TLS)

TLS 1.0 based on generalized version of SSL Version 3.0

RFC-2246 The TLS Protocol Version 1.0 from Januar 1999

RFC-3207 SMTP STARTTLS allows the use of TLS with SMTP mail protocols

Today most sites and servers support TLS Server Name Indication (SNI) name based certificates

TLS Server Name Indication example



Secure Sockets Layer

- ▼ TLSv1.2 Record Layer: Handshake Protocol: Client Hello
 - Content Type: Handshake (22)
 - Version: TLS 1.0 (0x0301)
 - Length: 198
- ▼ Handshake Protocol: Client Hello
 - Handshake Type: Client Hello (1)
 - Length: 194
 - Version: TLS 1.2 (0x0303)
 - Random
 - Session ID Length: 0
 - Cipher Suites Length: 32
 - Cipher Suites (16 suites)
 - Compression Methods Length: 1
 - Compression Methods (1 method)
 - Extensions Length: 121
 - Extension: Unknown 56026
 - Extension: renegotiation_info
 - ▼ Extension: server_name
 - Type: server_name (0x0000)
 - Length: 16
 - ▼ Server Name Indication extension
 - Server Name list length: 14
 - Server Name Type: host_name (0)
 - Server Name length: 11
 - Server Name: twitter.com
 - Extension: Extended Master Secret

0050	a4	1d	52	8f	2c	18	99	91	54	68	0a	77	0d	95	73	64	.R.,...	Th.w..sd
0060	7d	00	00	20	5a	5a	c0	2b	c0	2f	c0	2c	c0	30	cc	a9	}. ZZ.+	./.,.0..
0070	cc	a8	cc	14	cc	13	c0	13	c0	14	00	9c	00	9d	00	2f/
0080	00	35	00	0a	01	00	00	79	da	da	00	00	ff	01	00	01	.5.....y
0090	00	00	00	10	00	0e	00	00	0b	74	77	69	74	74	65twitte	
00a0	72	2e	63	6f	6d	00	17	00	00	00	23	00	00	00	0d	00	r.com#....
00b0	14	00	12	04	03	08	04	04	01	05	03	08	05	05	01	08

Heartbleed CVE-2014-0160



The Heartbleed Bug

The Heartbleed Bug is a serious vulnerability in the popular OpenSSL cryptographic software library. This weakness allows stealing the information protected, under normal conditions, by the SSL/TLS encryption used to secure the Internet. SSL/TLS provides communication security and privacy over the Internet for applications such as web, email, instant messaging (IM) and some virtual private networks (VPNs).

The Heartbleed bug allows anyone on the Internet to read the memory of the systems protected by the vulnerable versions of the OpenSSL software. This compromises the secret keys used to identify the service providers and to encrypt the traffic, the names and passwords of the users and the actual content. This allows attackers to eavesdrop on communications, steal data directly from the services and users and to impersonate services and users.



Nok det mest kendte SSL/TLS exploit

Source: <http://heartbleed.com/>

Heartbleed hacking



```
06b0: 2D 63 61 63 68 65 0D 0A 43 61 63 68 65 2D 43 6F -cache..Cache-Co
06c0: 6E 74 72 6F 6C 3A 20 6E 6F 2D 63 61 63 68 65 0D ntrol: no-cache.
06d0: 0A 0D 0A 61 63 74 69 6F 6E 3D 67 63 5F 69 6E 73 ...action=gc_ins
06e0: 65 72 74 5F 6F 72 64 65 72 26 62 69 6C 6C 6E 6F ert_order&billno
06f0: 3D 50 5A 4B 31 31 30 31 26 70 61 79 6D 65 6E 74 =PZK1101&payment
0700: 5F 69 64 3D 31 26 63 61 72 64 5F 6E 75 6D 62 65 _id=1&card_numbe
0710: XX r=4060xxxx413xxx
0720: 39 36 26 63 61 72 64 5F 65 78 70 5F 6D 6F 6E 74 96&card_exp_mont
0730: 68 3D 30 32 26 63 61 72 64 5F 65 78 70 5F 79 65 h=02&card_exp_ye
0740: 61 72 3D 31 37 26 63 61 72 64 5F 63 76 6E 3D 31 ar=17&card_cvn=1
0750: 30 39 F8 6C 1B E5 72 CA 61 4D 06 4E B3 54 BC DA 09.1...r.aM.N.T..
```

- Obtained using Heartbleed proof of concepts - Gave full credit card details
- "can XXX be exploited" - yes, clearly! PoCs ARE needed without PoCs even Akamai wouldn't have repaired completely!
- The internet was ALMOST fooled into thinking getting private keys from Heartbleed was not possible - scary indeed.



SSL/TLS udgaver af protokoller

Check with your system administrator before changing any of the advanced options below:

IMAP Path Prefix: INBOX

Port: 993 Use SSL

Authentication: Password

Mange protokoller findes i udgaver hvor der benyttes SSL

HTTPS vs HTTP

IMAPS, POP3S, osv.

Bemærk: nogle protokoller benytter to porte IMAP 143/tcp vs IMAPS 993/tcp

Andre benytter den samme port men en kommando som starter:

SMTP STARTTLS RFC-3207

ssllscan



```
root@kali:~# ssllscan --ssl2 web.kramse.dk
Version: 1.10.5-static
OpenSSL 1.0.2e-dev xx XXX xxxx
```

Testing SSL server web.kramse.dk on port 443

...

SSL Certificate:

```
Signature Algorithm: sha256WithRSAEncryption
RSA Key Strength: 2048
```

Subject: *.kramse.dk

Altnames: DNS:*.kramse.dk, DNS:kramse.dk

Issuer: AlphaSSL CA - SHA256 - G2

Source: Originally ssllscan from <http://www.titania.co.uk> but use the version on Kali

SSLLscan can check your own sites, while Qualys SSL Labs only can test from hostname

Exercise



Now lets do the exercise

⚠ SSL/TLS scanners 15 min

which is number **13** in the exercise PDF.

Exercise



Now lets do the exercise

⚠ Internet scanners 15 min

which is number **14** in the exercise PDF.

Passwords are not chosen completely random



The 50 Most Used Passwords

- | | | | | |
|--------------|--------------|----------------|--------------|-------------|
| 1. 123456 | 11. 123123 | 21. mustang | 31. 7777777 | 41. harley |
| 2. password | 12. baseball | 22. 666666 | 32. f*cky*u | 42. zxcvbnm |
| 3. 12345678 | 13. abc123 | 23. qwertyuiop | 33. qazwsx | 43. asdfgh |
| 4. qwerty | 14. football | 24. 123321 | 34. jordan | 44. buster |
| 5. 123456789 | 15. monkey | 25. 1234...890 | 35. jennifer | 45. andrew |
| 6. 12345 | 16. letmein | 26. p*s*y | 36. 123qwe | 46. batman |
| 7. 1234 | 17. shadow | 27. superman | 37. 121212 | 47. soccer |
| 8. 111111 | 18. master | 28. 270 | 38. killer | 48. tigger |
| 9. 1234567 | 19. 696969 | 29. 654321 | 39. trustno1 | 49. charlie |
| 10. dragon | 20. michael | 30. 1qaz2wsx | 40. hunter | 50. robert |

Source: <https://wpengine.com/unmasked/>

Brute force



We call it brute force – when testing all possibilities

Hydra (c) by van Hauser / THC <vh@thc.org>

Syntax: hydra [[[[-l LOGIN|-L FILE] [-p PASS|-P FILE]] | [-C FILE]]
[-o FILE] [-t TASKS] [-g TASKS] [-T SERVERS] [-M FILE] [-w TIME]
[-f] [-e ns] [-s PORT] [-S] [-vV] server service [OPT]

Options:

- S connect via SSL
- s PORT if the service is on a different default port, define it here
- l LOGIN or -L FILE login with LOGIN name, or load several logins from FILE
- p PASS or -P FILE try password PASS, or load several passwords from FILE
- e ns additional checks, "n" for null password, "s" try login as pass
- C FILE colon seperated "login:pass" format, instead of -L/-P option
- M FILE file containing server list (parallizes attacks, see -T)
- o FILE write found login/password pairs to FILE instead of stdout

...

Cracking passwords – JtR and Hashcat



John the Ripper is a fast password cracker, currently available for many flavors of Unix (11 are officially supported, not counting different architectures), Windows, DOS, BeOS, and OpenVMS. Its primary purpose is to detect weak Unix passwords.

- Hashcat is the world's fastest CPU-based password recovery tool.
- oclHashcat-plus is a GPGPU-based multi-hash cracker using a brute-force attack (implemented as mask attack), combinator attack, dictionary attack, hybrid attack, mask attack, and rule-based attack.
- oclHashcat-lite is a GPGPU cracker that is optimized for cracking performance. Therefore, it is limited to only doing single-hash cracking using Markov attack, Brute-Force attack and Mask attack.
- John the Ripper password cracker old skool men stadig nyttig

Source:

<https://hashcat.net/wiki/>

<http://www.openwall.com/john/>

Exercise



Now lets do the exercise

i SNMP walk 15min

which is number **15** in the exercise PDF.

OWASP top ten



The OWASP Top Ten provides a minimum standard for web application security. The OWASP Top Ten represents a broad consensus about what the most critical web application security flaws are.

- The Open Web Application Security Project (OWASP) <http://www.owasp.org>
- Also has Zed Attack Proxy (ZAP)
https://www.owasp.org/index.php/OWASP_Zed_Attack_Proxy_Project

OWASP Juice Shop Project



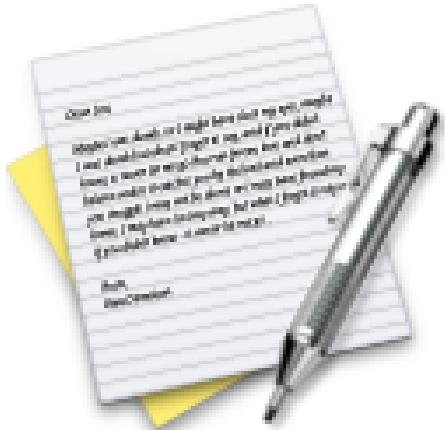
We will also use the OWASP Juice Shop Tool Project as a running example. This is an application which is modern AND designed to have security flaws.

Read more about this project at: https://www.owasp.org/index.php/OWASP_Juice_Shop_Project
<https://github.com/bkimminich/juice-shop>

It is recommended to buy the Pwning OWASP Juice Shop Official companion guide to the OWASP Juice Shop from <https://leanpub.com/juice-shop> - suggested price USD 5.99

I will show a very small part of this web application

Exercise



Now lets do the exercise

ⓘ Run OWASP Juice Shop 45 min

which is number **16** in the exercise PDF.

Scanning Web Applications / Web Servers



Description Nikto is an Open Source (GPL) web server scanner which performs comprehensive tests against web servers for multiple items, including over 6700 potentially dangerous files/programs, checks for outdated versions of over 1250 servers, and version specific problems on over 270 servers. It also checks for server configuration items such as the presence of multiple index files, HTTP server options, and will attempt to identify installed web servers and software. Scan items and plugins are frequently updated and can be automatically updated.

Source: <https://cirt.net/Nikto2>

- Nikto is quite old, some would say mature tool. Always recommended to run this, cheap, quick, finds interesting links
- Whatweb can be used to check up on a site, what framework does it use, how is it configured, where is it hosted
- Then afterwards I recommend Burp Suite professional <https://portswigger.net/burp/pro>
- The exercise booklet also has a few extra exercises about session tokens, and SQL injection

Exercise

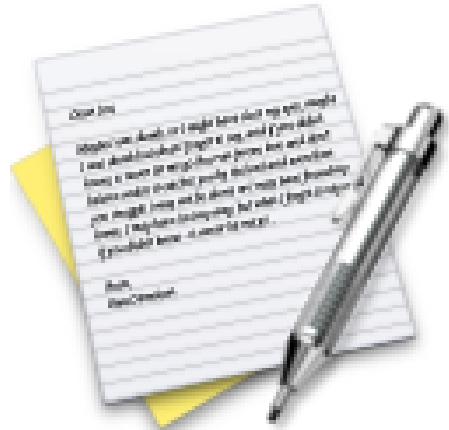


Now lets do the exercise

⚠ Nikto Web Scanner 15 min

which is number **20** in the exercise PDF.

Exercise



Now lets do the exercise

⚠ Whatweb Scanner 15 min

which is number **21** in the exercise PDF.

Questions?



Henrik Kramselund he/him han/ham hlk@zencurity.com @kramse

You are always welcome to send me questions later via email

Mobile: +45 2026 6000

Email: hlk@zencurity.com

Advanced topics: DDoS Testing

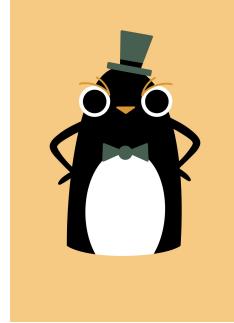


Below are some more information about DDoS testing. This is a scenario you can perform after learning basic port scanning techniques.

The main points would be:

- Port scan the network that is to be tested
- Register the ports open on the link net - routed network in front of infrastructure
- Register the ports open on the firewalls, load balancers, servers etc. in the infrastructure
- Decide on a time for testing, inform all parties
- Perform testing 5-15minutes at a time using a packet blaster
- Note changes in uptime, latency, resources on the devices etc.
- Create a report, improve infrastructure

Packet generators



- Scapy is an example packet generator, allowing you to use Python
- It is very flexible and often *fast enough*, but other times, not fast enough
- Lots of other tools exist, some work as kernel modules even
- I love hping3 and t50 – and use them for stress testing firewalls and devices
- Started building a replacement or penguinping

♡ love internet packets ♡

hping3 packet generator



```
usage: hping3 host [options]
-i --interval  wait (uX for X microseconds, for example -i u1000)
--fast        alias for -i u10000 (10 packets for second)
--faster      alias for -i u1000 (100 packets for second)
--flood       sent packets as fast as possible. Don't show replies.
```

...

hping3 is fully scriptable using the TCL language, and packets can be received and sent via a binary or string representation describing the packets.

- Hping3 packet generator is a very flexible tool to produce simulated DDoS traffic with specific characteristics
- Home page: <http://www.hping.org/hping3.html>
- Source repository <https://github.com/antirez/hping>

My primary DDoS testing tool, easy to get specific rate pps

t50 packet generator



```
root@cornerstone03:~# t50 -?
T50 Experimental Mixed Packet Injector Tool 5.4.1
Originally created by Nelson Brito <nbrito@sekure.org>
Maintained by Fernando Mercês <fernando@mentebinaria.com.br>
```

Usage: T50 <host> [/CIDR] [options]

Common Options:

```
--threshold NUM      Threshold of packets to send      (default 1000)
--flood              This option supersedes the 'threshold'
```

...

6. Running T50 with '--protocol T50' option, sends ALL protocols sequentially.

```
root@cornerstone03:~# t50 -? | wc -l
264
```

- T50 packet generator, another high speed packet generator can easily overload most firewalls by producing a randomized traffic with multiple protocols like IPsec, GRE, MIX
home page: <http://t50.sourceforge.net/resources.html>

Extremely fast and breaks most firewalls when flooding, easy 800k pps/400Mbps



Running full port scan on network

Hint: use a variable to keep the target address, carefully enter it and avoid mystyping it later

```
# export CUST_NET4="192.0.2.0/24"
# export CUST_NET6="2001:DB8:ABCD:1000::/64"
# nmap -p 1-65535 -Pn -A -oA full-scan $CUST_NET4

# export CUST_IP=192.0.2.138
# date;time hping3 -q -c 1000000 -i u60 -S -p 80 $CUST_IP
```

Better yet, script it all – but most likely you will want to repeat specific steps.

Nmap port sweep for TCP services, full TCP scan



Goal is to enumerate the ports that are allowed through the network.

```
# nmap -Pn -A -p 1-65535 -oA full-tcp-customer-ipv4 $CUST_NET4
...
Nmap scan report for 192.0.2.138
Host is up (0.00012s latency).
PORT      STATE    SERVICE
80/tcp    open     http
443/tcp   closed   https
# nmap -Pn -A -p 1-65535 -oA full-tcp-customer-ipv6 $CUST_NET6
# nmap -Pn -A -p 1-65535 -oA full-tcp-linknet-ipv4 $LINK_NET4
# nmap -Pn -A -p 1-65535 -oA full-tcp-linknet-ipv6 $LINK_NET6
```

Note: Pretty harmless, if something dies, then it is *vulnerable to normal traffic* - and should be fixed!

Options:

-Pn -- Scan all IPs, dont use ping or TCP ping to check alive -A advanced -- perform full TCP connection and grab banner
-p 1-65535 -- full portscan all ports -oA filename -- Saves output in "all formats" normal, XML, and grepable formats



Running Attacks with hping3

```
# export CUST_IP=192.0.2.1
# date;time hping3 -q -c 1000000 -i u60 -S -p 80 $CUST_IP
```

Expected output:

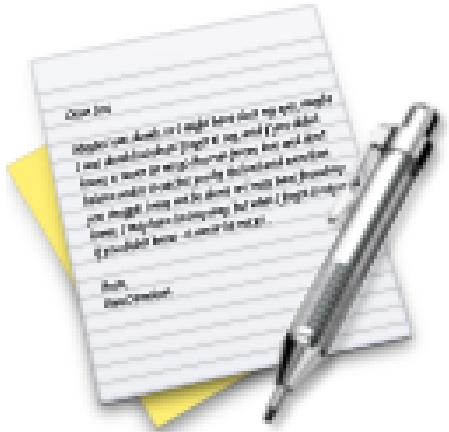
```
# date;time hping3 -q -c 1000000 -i u60 -S -p 80 $CUST_IP
Thu Jan 21 22:37:06 CET 2016
HPING 192.0.2.1 (eth0 192.0.2.1): S set, 40 headers + 0 data bytes

--- 192.0.2.1 hping statistic ---
1000000 packets transmitted, 999996 packets received, 1% packet loss
round-trip min/avg/max = 0.9/7.0/1005.5 ms

real 1m7.438s
user 0m1.200s
sys 0m5.444s
```

Dont forget to do a killall hping3 when done ☺

Exercise

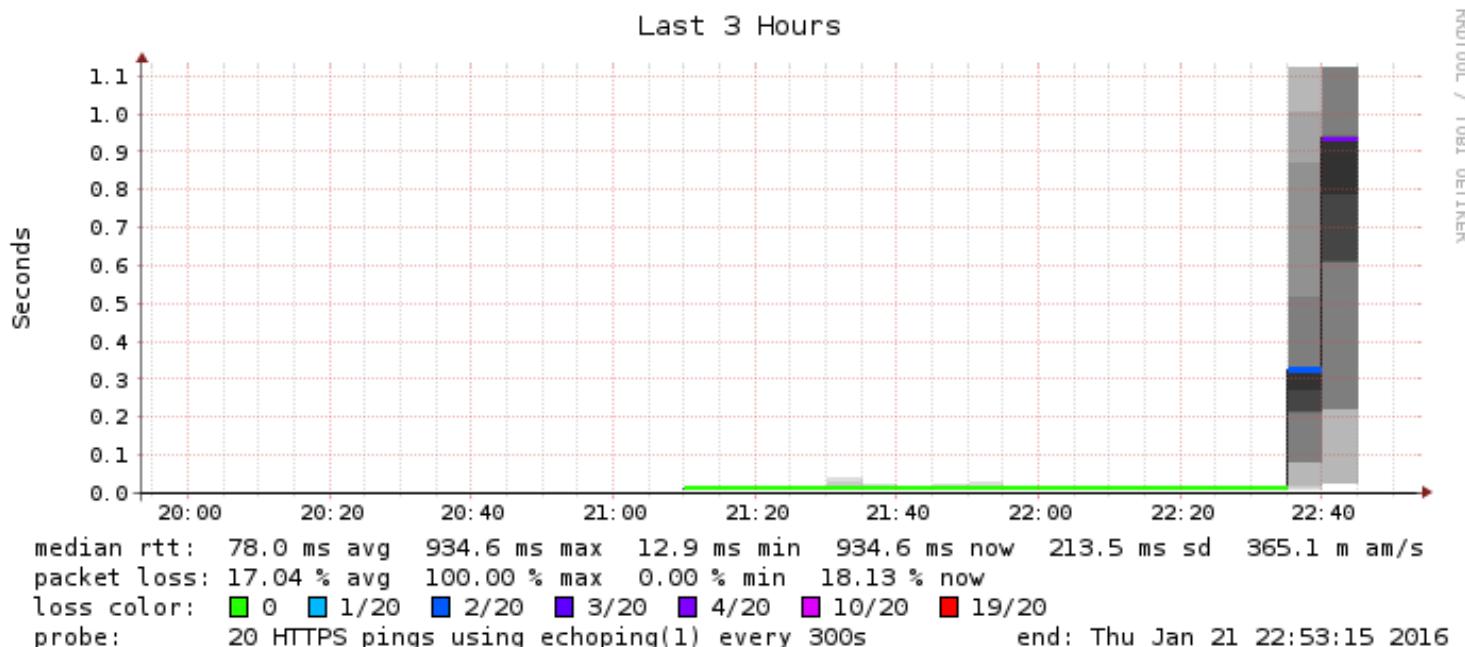


Now lets do the exercise

● TCP SYN flooding 30min

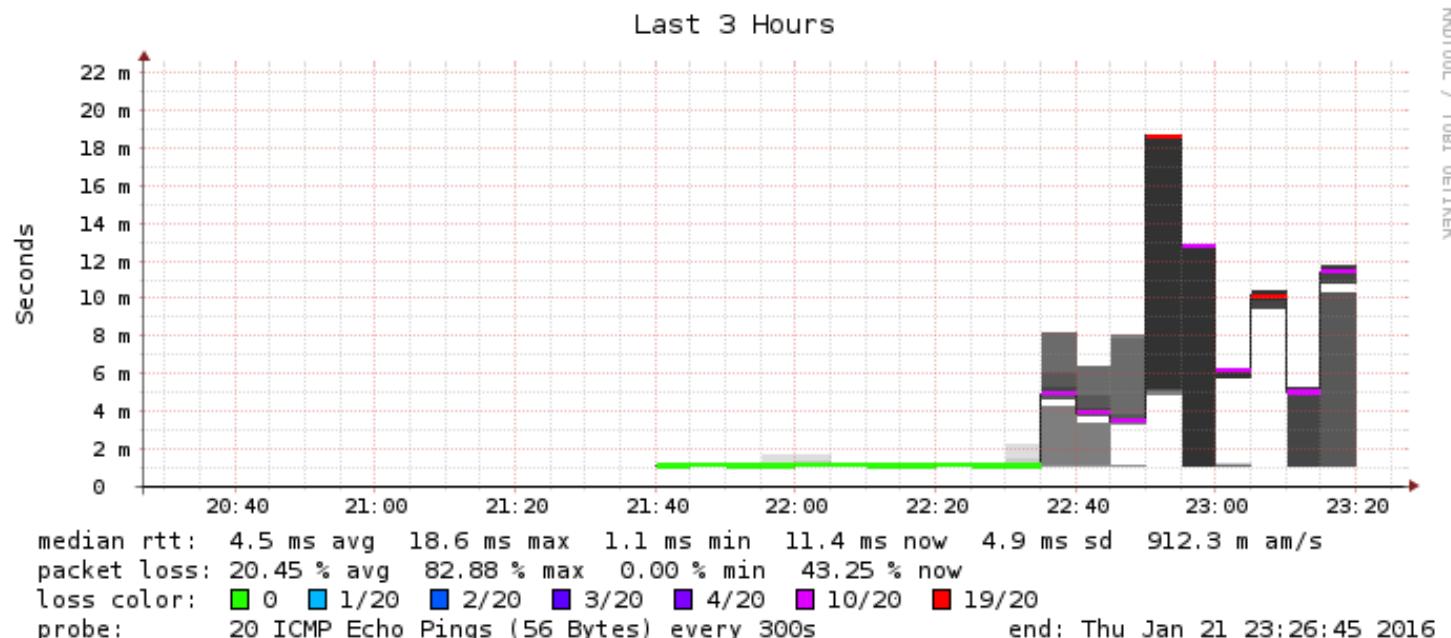
which is number **24** in the exercise PDF.

Rocky Horror Picture Show - 1



Really does it break from 50.000 pps SYN attack?

Rocky Horror Picture Show - 2

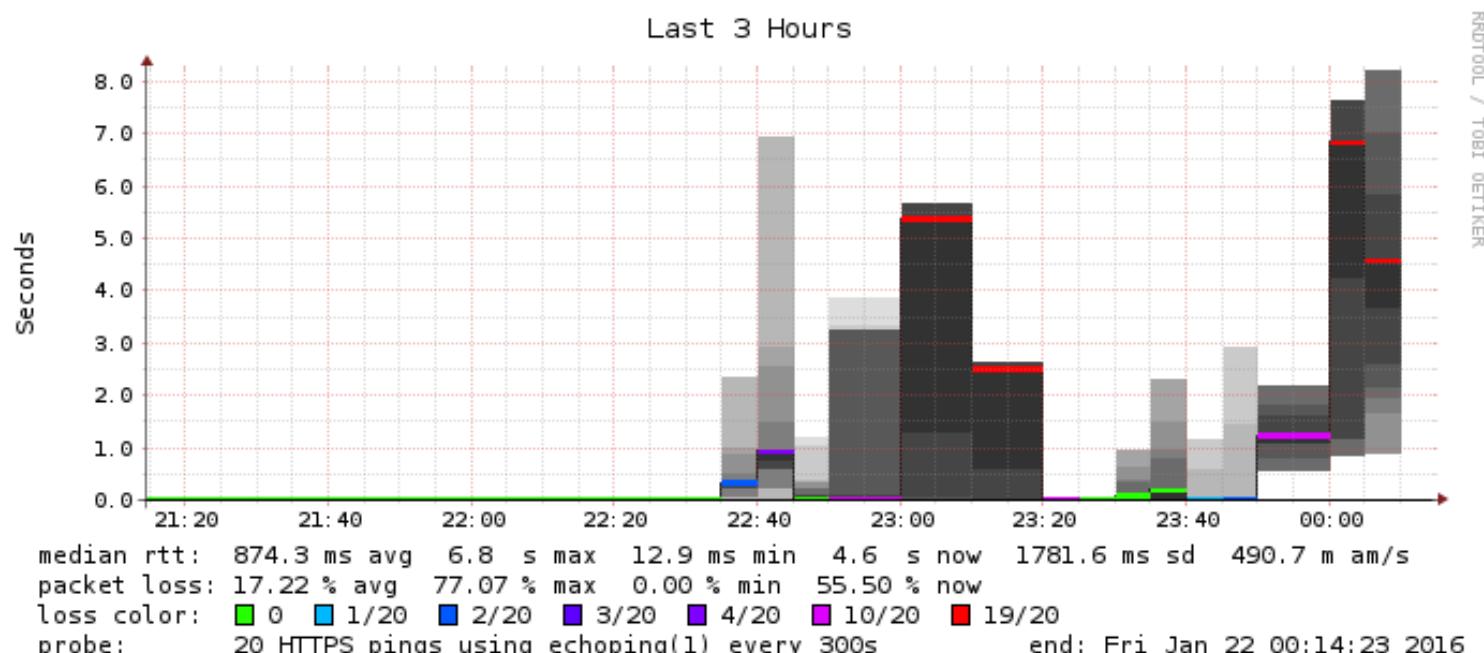


Oh no 500.000 pps UDP attacks work?

Rocky Horror Picture Show - 3



Oh no spoofing attacks work?



Penguiping packet generator



```
[Projects] Terminal - hlk@penguin01: ~
File Edit View Terminal Tabs Help
root@penguin01:/home/hlk/projects/MoonGen# ./build/MoonGen ./examples/penguiping-02.lua 10.0.49.1 -a 10.1.2.3 -r 1000 -S -A -F -U -P -R

EAL: Detected 16 lcore(s)
EAL: No free hugepages reported in hugepages-1048576kB
EAL: Probing VFIO support...
EAL: PCI device 0000:01:00.0 on NUMA socket -1
EAL:   Invalid NUMA socket, default to 0
EAL:     probe driver: 8086:10fb net_ixgbe
EAL: PCI device 0000:01:00.1 on NUMA socket -1
EAL:   Invalid NUMA socket, default to 0
EAL:     probe driver: 8086:10fb net_ixgbe

Device 0: 00:25:90:32:9f:F2 (Intel Corporation 82599ES 10-Gigabit SFI/SFP+ Network Connection)
Device 1: 00:25:90:32:9f:F3 (Intel Corporation 82599ES 10-Gigabit SFI/SFP+ Network Connection)
PMD: ixgbe_dev_link_states_print(): Port 0: Link Up - speed 0 Mbps - half-duplex

TCP mode get TCP packet
ETH 00:25:90:32:9f:f2 > 00:00:00:00:00:00 type 0x0800 [IP4]
IP4 10.1.2.3 > 10.0.49.1 4 ihl 5 tos 0 len 46 id 0 flags 0 frag 0 ttl 64 proto 0x06 (TCP) cksum 0x0000 [-]
TCP 52049 > 80 seq 1 acks 0 offset 0x5 reserved 0x0 flags 0x3f [URG|ACK|PSH|RST|SYN|FIN] win 10 cksum 0x0000 urg 0 []
    0000 0000 0000 0025 0032 0ff2 0000 4500
    002e 0000 0000 4006 0000 0a01 0000 0a00
    3101 cb51 0050 0000 0001 0000 0000 503f
    000a 0000 0000 0000 0000 0000 0000 0000

[Device: id=0] TX: 1.95 Mpps, 1000 Mbit/s (1312 Mbit/s with framing)
[Device: id=0] TX: 1.94 Mpps, 994 Mbit/s (1304 Mbit/s with framing)
[Device: id=0] TX: 1.95 Mpps, 1000 Mbit/s (1312 Mbit/s with framing)
[Device: id=0] TX: 1.95 Mpps, 1000 Mbit/s (1312 Mbit/s with framing)
[Device: id=0] TX: 1.95 Mpps, 1000 Mbit/s (1312 Mbit/s with framing)
[Device: id=0] TX: 1.95 Mpps, 1000 Mbit/s (1312 Mbit/s with framing)
[Device: id=0] TX: 1.95 Mpps, 1000 Mbit/s (1312 Mbit/s with framing)
[Device: id=0] TX: 1.95 Mpps, 1000 Mbit/s (1312 Mbit/s with framing)
[Device: id=0] TX: 1.95 Mpps, 1000 Mbit/s (1312 Mbit/s with framing)
```

- PenguinPing packet generator, my high speed packet generator home page: <https://penguiping.org>
- First versions are only about 230 lines of Lua code and implement basic command line to replace hping3
- Built on top of MoonGen/libmoon <https://github.com/emmericp/MoonGen>

Extremely fast and allows easy customization

Running MoonGen



```
root@penguin01:~/projects/MoonGen# ./build/MoonGen ./examples/I3-tcp-syn-flood.lua 0 -d 192.0.2.138
[INFO] Initializing DPDK. This will take a few seconds...
EAL: Detected 16 lcore(s)
[INFO] Found 1 usable devices:
      Device 0: 00:25:90:32:9F:F3 (Intel Corporation 82599ES 10-Gigabit SFI/SFP+ Network Connection)
PMD: ixgbe_dev_link_status_print(): Port 0: Link Down
[INFO] Device 0 (00:25:90:32:9F:F3) is up: 10000 MBit/s
[INFO] Detected an IPv4 address.

...
[Device: id=0] TX: 14.88 Mpps, 7619 Mbit/s (9999 Mbit/s with framing)
[Device: id=0] TX: 14.48 Mpps, 7414 Mbit/s (9730 Mbit/s with framing)
[Device: id=0] TX: 14.88 Mpps, 7619 Mbit/s (10000 Mbit/s with framing)
```

- Installed Debian Linux – little bit of disable secure boot, RAID/AHCI settings, ...
- After install – tuning and enabling Hugepages
- Clone the repository <https://github.com/emmericp/MoonGen> build and run
- **Note: the full 14.8Mpps is done using a single core!**



Turn up and down as you please with the -r rate option

```
root@penguin01:~/projects/MoonGen# ./build/MoonGen ./examples/l3-tcp-syn-flood.lua 0 -r 5000 -d 192.0.2.138
[INFO] Initializing DPDK. This will take a few seconds...
EAL: Detected 16 lcore(s)
[INFO] Found 1 usable devices:
      Device 0: 00:25:90:32:9F:F3 (Intel Corporation 82599ES 10-Gigabit SFI/SFP+ Network Connection)
PMD: ixgbe_dev_link_status_print(): Port 0: Link Down
[INFO] Device 0 (00:25:90:32:9F:F3) is up: 10000 MBit/s
[INFO] Detected an IPv4 address.

[Device: id=0] TX: 9.77 Mpps, 5000 Mbit/s (6562 Mbit/s with framing)
[Device: id=0] TX: 9.68 Mpps, 4955 Mbit/s (6504 Mbit/s with framing)
[Device: id=0] TX: 9.77 Mpps, 5000 Mbit/s (6562 Mbit/s with framing)
```

IPv6 and UDP, replace tcp with udp in example:

```
./build/MoonGen ./examples/l3-tcp-syn-flood.lua 0 -r 5000 -d 2001:DB8:ABCD:0053::138 -i 2001:DB8:ABCD:0053::1
./build/MoonGen ./examples/l3-udp-flood-hlk.lua 0 -r 5000 -d 2001:DB8:ABCD:0053::138 -i 2001:DB8:ABCD:0053::1
```

PenguinPing – re-implementing hping3 with Lua



```
root@penguin01:~/projects/MoonGen# ./build/MoonGen ./examples/pinguinping-02.lua 10.0.49.1 -a 10.1.2.3 -r 10000 -S -p 80
[INFO] Initializing DPDK. This will take a few seconds...
[INFO] Found 2 usable devices:
      Device 0: 00:25:90:32:9F:F2 (Intel Corporation 82599ES 10-Gigabit SFI/SFP+ Network Connection)
[INFO] Device 0 (00:25:90:32:9F:F2) is up: 10000 MBit/s
TCP mode get TCP packet
IP4 10.1.2.3 > 10.0.49.1 ver 4 ihl 5 tos 0 len 46 id 0 flags 0 frag 0 ttl 64 proto 0x06 (TCP) cksum 0x0000 [-]
TCP 52049 > 80 seq# 1 ack# 0 offset 0x5 reserved 0x00 flags 0x02 [-|-|-|SYN|-] win 10 cksum 0x0000 urg 0 []
  0x0000: 0000 0000 0000 0025 9032 9ff2 0800 4500
  0x0010: 002e 0000 0000 4006 0000 0a01 0203 0a00
  0x0020: 3101 cb51 0050 0000 0001 0000 0000 5002
  0x0030: 000a 0000 0000 0000 0000 0000

[Device: id=0] TX: 14.88 Mpps, 7619 Mbit/s (9999 Mbit/s with framing)
[Device: id=0] TX: 14.78 Mpps, 7568 Mbit/s (9933 Mbit/s with framing)
[Device: id=0] TX: 14.88 Mpps, 7619 Mbit/s (10000 Mbit/s with framing)
```

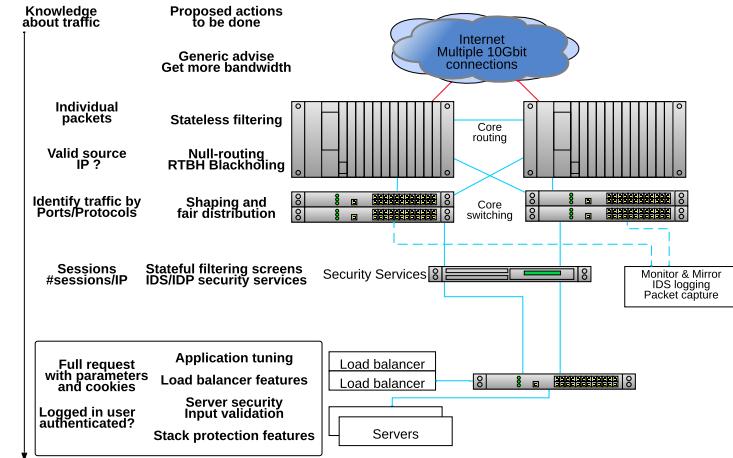
- Using Lua we can implement the same attacks from Hping3 easily
- Only about 230 lines of Lua using MoonGen and libmoon
- Can run at specific rate up to full 10Gbps / 14.8 Million packets per second using a single CPU core

Recommendations During Test



- Run each test for at least 5 minutes, or even 15 minutes
Some attacks require some build-up before resource run out
- Take note of any change in response, higher latency, lost probes
- If you see a change, then re-test using the same parameters, or a little less first
- We want to know the approximate level where it breaks
- If you want to change environment, then wait until all scenarios are tested
- Keep a log handy, write notes and start the session with script `ddos-date-customer.log`
- Check once in a while if you have some process running, using `ps auxw | grep hping3`
- Run multiple instances of the tools. One process might generate 800.000 pps, while two may double it. Though 10 processes might not be 10 times exactly

Conclusion Pentest, DDoS and network attacks



- You really should try testing, and investigate your existing devices all of them
- This is just one small part of your security posture, extra slides has my take on enterprise network security

Advanced topics: Exploit development



After learning a bit about hacking you may want to look more into exploit techniques. Below are some pointers and subjects to research

- Book: Gray Hat Hacking: The Ethical Hacker's Handbook, Sixth Edition Af Allen Harper et al
- Older books: Hacking, 2nd edition Jon Ericksson and The Shellcoder's Handbook – Discovering and Exploiting Security Holes 2e Af C. Anley
- Difference between the oldest, most simple stack based overflows
- The parts of a shell code running system calls, and which system calls exec etc.
- How to avoid having shell code - return into libc, calling functions all the way to modern Return-Oriented Programming (ROP)
- Recommended shell code video:

EXPLORING NEW DEPTHS OF THREAT HUNTING ...OR HOW TO WRITE ARM SHELLCODE IN SIX MINUTES

Speaker: Maria Markstedter, Azeria Labs

<https://www.youtube.com/watch?v=DGJZBDlhIGU>

Return-Oriented Programming (ROP)



Advanced subject Return-Oriented Programming (ROP)

Return-Oriented Programming: Systems, Languages, and Applications Ryan Roemer, Erik Buchanan, Hovav Shacham and Stefan Savage University of California, San Diego
<https://hovav.net/ucsd/dist/rop.pdf>

Then look into how a security oriented operating system has decided to prevent this method:

Removing ROP Gadgets from OpenBSD Todd Mortimer
<https://www.openbsd.org/papers/asiabsdcon2019-rop-paper.pdf>



Advanced topics: Web hacking lab

Practice real-life scenarios - enabling you to do quality port scans!

I brought a web hacker lab based on OWASP Juice Shop

Recommmended for all developers: Try running the OWASP Juice Shop

This is an application which is modern AND designed to have security flaws.

Read more about this project at:

<https://www2.owasp.org/www-project-juice-shop/> and

<https://github.com/bkimminich/juice-shop>

It is recommended to buy the Pwning OWASP Juice Shop Official companion guide to the OWASP Juice Shop from <https://leanpub.com/juice-shop> - suggested price USD 5.99. Alternatively read online at <https://pwning.owasp-juice.shop/>

Sometimes the best method is running the Docker version

Next steps: Concrete advice for enterprise networks



- Portscanning - start using portscans in your networks, verify how far malware and hackers can travel, and identify soft systems needing updates or isolation
- Have separation – anywhere, starting with organisation units, management networks, server networks, customers, guests, LAN, WAN, Mail, web, ...
- Use Web proxies - do not allow HTTP directly except for a short allow list, do not allow traffic to and from any new TLD
- Use only your own DNS servers, create a pair of Unbound servers, point your internal DNS running on Windows to these
Create filtering, logging, restrictions on these Unbound DNS servers
<https://www.nlnetlabs.nl/projects/unbound/about/> and also <https://pi-hole.net/>
- Only allow SMTP via your own mail servers, create a simple forwarder if you must

Allow lists are better than block list, even if it takes some time to do it



Capture data and logs!

- Run DNS query logs – when client1 is infected with malware from domain malwareexample.com, then search for more clients infected
- Run Zeek and gather information about all HTTPS sessions – captures certificates by default, and we can again search for certificate related to malwareexample.com
- Run network logging – session logs in enterprise networks are GREAT (country wide illegal logging is of course NOT)

Make sure to check with employees, inform them!

DROP SOME TRAFFIC NOW



- Drop some traffic on the border of everything
- Seriously do NOT allow Windows RPC across borders
- Border here may be from regional country office back to HQ
- Border may be from internet to internal networks
- Block Windows RPC ports, 135, 137, 139, 445
- Block DNS directly to internet, do not allow clients to use any DNS, fake 8.8.8.8 if you must internally
- Block SMTP directly to internet
- Create allow list for internal networks, client networks should not contact other client networks but only relevant server networks

You DONT need to allow direct DNS towards internet, except from your own recursive DNS servers

If you get hacked by Windows RPC from the Internet, you probably deserve it, sorry for being blunt

Best would be to analyze traffic and create allow lists, some internal networks to not need internet at all



Stateless firewall filter throw stuff away

```
hlk@MX-CPH-02> show configuration firewall filter all | no-more
/* This is a sample, better to use BGP flowspec or BGP based RTBH */
term edgeblocker {
    from {
        source-address {
            84.xx.xxx.173/32;
...
            87.xx.xxx.171/32;
        }
        destination-address {
            192.0.2.16/28;
        }
        protocol [ tcp udp icmp ];
    }
    then {
        count edge-block;
        discard;
    }
}
```

Example how to do it wirespeed – with Junos Hint: can also leave out protocol and then it will match all protocols

Default permit



One of the early implementers of firewalls Marcus J. Ranum summarized in 2005 The Six Dumbest Ideas in Computer Security https://www.ranum.com/security/computer_security/editorials/dumb/ which includes the always appropriate discussion about default permit versus default deny.

#1) Default Permit

This dumb idea crops up in a lot of different forms; it's incredibly persistent and difficult to eradicate. Why? Because it's so attractive. Systems based on "Default Permit" are the computer security equivalent of empty calories: tasty, yet fattening.

The most recognizable form in which the "Default Permit" dumb idea manifests itself is in firewall rules. Back in the very early days of computer security, network managers would set up an internet connection and decide to secure it by turning off incoming telnet, incoming rlogin, and incoming FTP. Everything else was allowed through, hence the name "Default Permit." This put the security practitioner in an endless arms-race with the hackers.

- Allow all current networks today on all ports for all protocols *is* an allow list
Which tomorrow can be split into one for TCP, UDP and remaining, and measured upon
- Measure, improve, repeat

We cannot do X



We cannot block SMTP from internal networks, since we do not know for sure if vendor X equipment needs to send the MOST important email alert at some unspecific time in the future

Cool, then we can do an allow list starting today on our border firewall:

```
table <smtp-exchange> { $exchange1 $exchange2 $exchange3 }
table <smtp-unknown> persist file "/firewall/mail/smtp-internal-unknown.txt"
# Regular use, allowed
pass out on egress inet proto tcp from smtp-exchange to any port 25/tcp
# Unknown, remove when phased out
pass out on egress inet proto tcp from smtp-internal to any port 25/tcp
```

Year 0 the unknown list may be 100% of all internal networks, but new networks added to infrastructure are NOT added, so list will shrink – evaluate the list, and compare to network logs, did networks send ANY SMTP for 1,2,3 years?

Zeek is a framework and platform



The Zeek Network Security Monitor

While focusing on network security monitoring, Zeek provides a comprehensive platform for more general network traffic analysis as well. Well grounded in more than 15 years of research, Zeek has successfully bridged the traditional gap between academia and operations since its inception.

<https://www.Zeek.org/> Does useful things out of the box using more than 10.000 script lines

Suricata IDS/IPS/NSM



Suricata is a high performance Network IDS, IPS and Network Security Monitoring engine.

<http://suricata-ids.org/> <http://openinfosecfoundation.org>

Suricata, Zeek og DNS Capture – it a nice world, use it!

<https://github.com/kramse/security-courses/tree/master/courses/networking/suricatazeek-workshop>



Firewall – Another definition

I am also fond of this longer and technical definition from RFC4949:

\$ firewall

1. (I) **An internetwork gateway that restricts data communication traffic to and from one of the connected networks** (the one said to be "inside" the firewall) and thus protects that network's system resources against threats from the other network (the one that is said to be "outside" the firewall). (See: guard, security gateway.)
2. (O) **A device or system that controls the flow of traffic between networks using differing security postures.** Wack, J. et al (NIST), "Guidelines on Firewalls and Firewall Policy", Special Publication 800-41, January 2002.

Tutorial: A firewall typically protects a smaller, secure network (such as a corporate LAN, or even just one host) from a larger network (such as the Internet). The firewall is installed at the point where the networks connect, and the firewall applies policy rules to control traffic that flows in and out of the protected network.

Firewall – Another definition



\$ firewall, continued

A firewall is not always a single computer. For example, a firewall may consist of a pair of filtering routers and one or more proxy servers running on one or more bastion hosts, all connected to a small, dedicated LAN (see: buffer zone) between the two routers.

The external router blocks attacks that use IP to break security (IP address spoofing, source routing, packet fragments), while proxy servers block attacks that would exploit a vulnerability in a higher-layer protocol or service. The internal router blocks traffic from leaving the protected network except through the proxy servers.

The difficult part is defining criteria by which packets are denied passage through the firewall, because a firewall not only needs to keep unauthorized traffic (i.e., intruders) out, but usually also needs to let authorized traffic pass both in and out.

Routing Security



- Use MD5 passwords or better authentication for routing protocols 
- TTL Security – avoid routed packets
- Max prefix – of course, only allow expected networks
- Prefix filtering – only parts of IPv6 space is used
- TCP Authentication Option [RFC 5925] replaces TCP MD5 [RFC 2385]
- Turn ON RPKI for both IPv4 and IPv6 prefixes, 
<https://nlnetlabs.nl/projects/rpki/about/>
- Drop bogons on IPv4 and IPv6, article with multiple references YMMV
<https://theinternetprotocolblog.wordpress.com/2020/01/15/some-notes-on-ipv6-bogon-filtering/>

Mutually Agreed Norms for Routing Security (MANRS)



Mutually Agreed Norms for Routing Security (MANRS) is a global initiative, supported by the Internet Society, that provides crucial fixes to reduce the most common routing threats.

Source: https://www.manrs.org/wp-content/uploads/2018/09/MANRS_PDF_Sep2016.pdf

- Problems related to incorrect routing information
- Problems related to traffic with spoofed source IP addresses
- Problems related to coordination and collaboration between network operators
- Also BCP38 RFC2827 *Network Ingress Filtering: Defeating Denial of Service Attacks which employ IP Source Address Spoofing*

You should all ask your internet providers if they know about MANRS, and follow it. We should ask our government and institutions to support and follow MANRS and good practices for network on the Internet

uRPF unicast Reverse Path Forwarding



Reverse path forwarding (RPF) is a technique used in modern routers for the purposes of ensuring loop-free forwarding of multicast packets in multicast routing and to help prevent IP address spoofing in unicast routing.

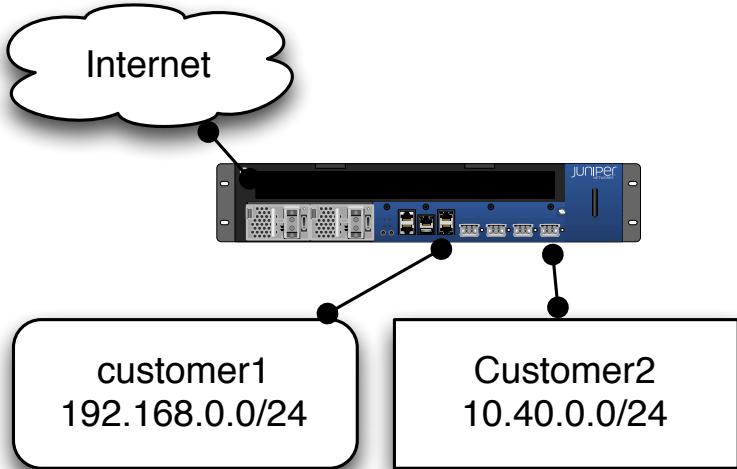
Source: http://en.wikipedia.org/wiki/Reverse_path_forwarding

Configuring Unicast RPF Strict Mode

In strict mode, unicast RPF checks whether the incoming packet has a source address that matches a prefix in the routing table, **and whether the interface expects to receive a packet with this source address prefix.**

This means, random source spoofed packets – not matching a current route entry in the global table, will get discarded early on!

Strict vs loose mode RPF



```
user@router# show interfaces
ge-0/0/0 {
    unit 2 {
        family inet {
            rpf-check fail-filter rpf-special-case-dhcp;
            address 192.168.0.254/24;
        }
    }
}
ge-0/0/1 {
    unit 2 {
        family inet {
            rpf-check fail-filter rpf-special-case-dhcp;
            address 10.40.0.254/24;
        }
    }
}
```