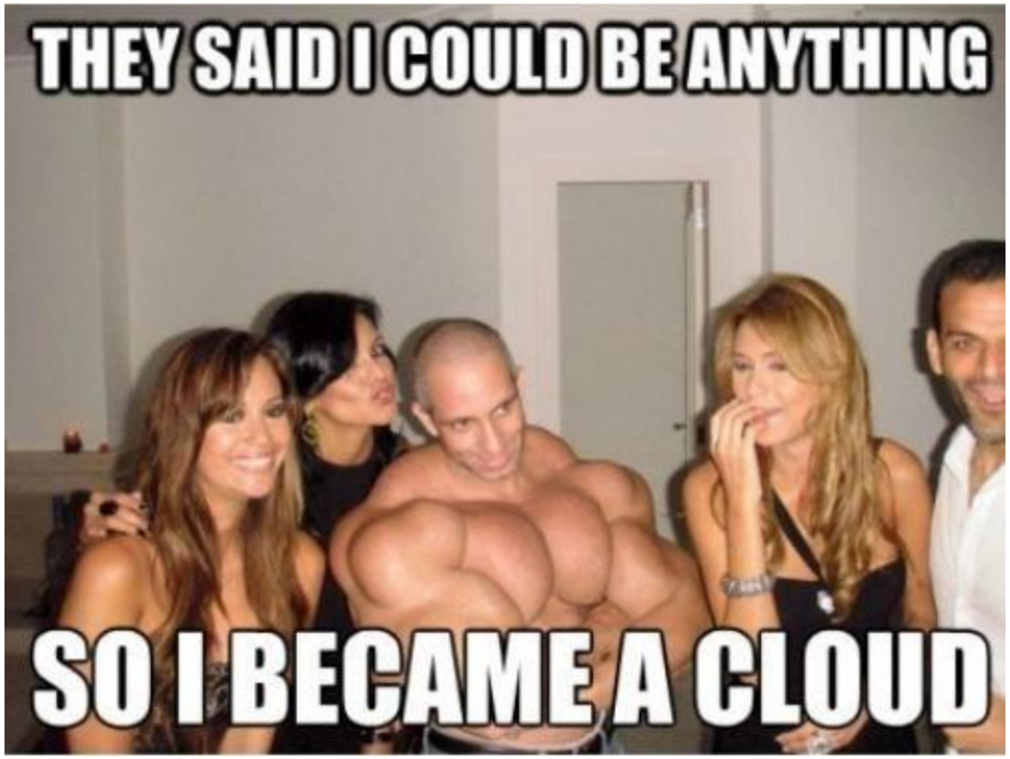
Mathieu Deous – Julien Reveret

Forensic analysis of a Linux web server

Agenda

Who are we ?
Performing forensic analysis on a compromised web server
What to search, where, how ? Logs but also dynamic analysis
What about privilege escalation ?
How has rootkit detection evolved ?

Who are we ?



- Security guys doing both offensive and defensive stuff
- Our company is hosting 3000+ sites in a private cloud

Agenda

Who are we ?
Performing forensic analysis on a compromised web server
What to search, where, how ? Logs but also dynamic analysis
What about privilege escalation ?
How has rootkit detection evolved ?

Performing forensic analysis on a compromised web server

Some people think finding out what attackers did on a server is an easy task since their activities are « obviously malicious » :

- Sending spam
- Scanning /DoSing other servers
- Hosting phishing sites

But sometimes there can be days, weeks, months between a compromise and illegal activities…

Performing forensic analysis on a compromised web server

You end up trying to find files which don't exist anymore

Sometimes access to webshells are sold on underground markets:

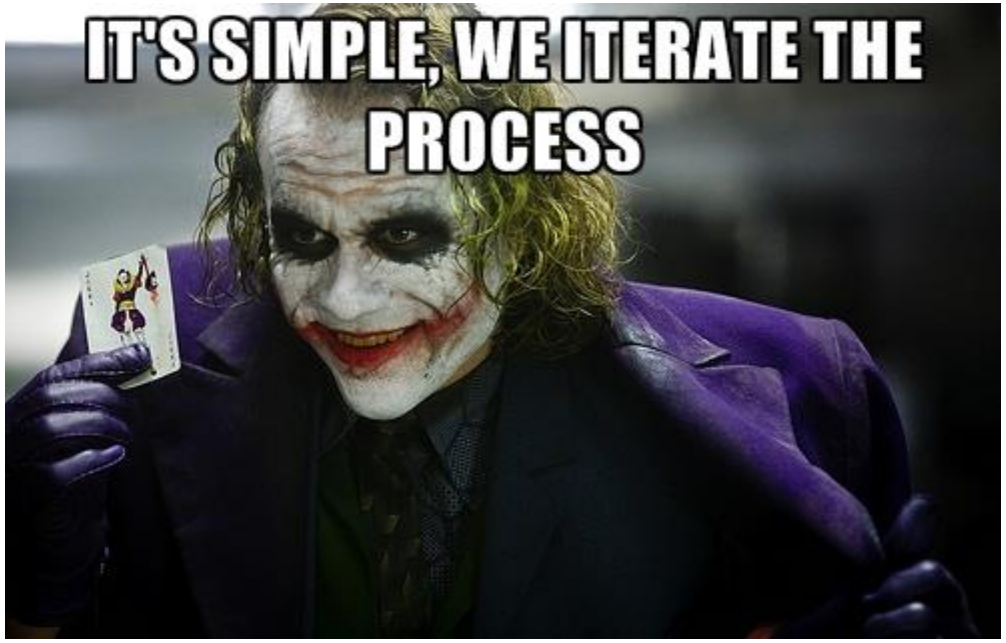
→ Tons of different IP addresses

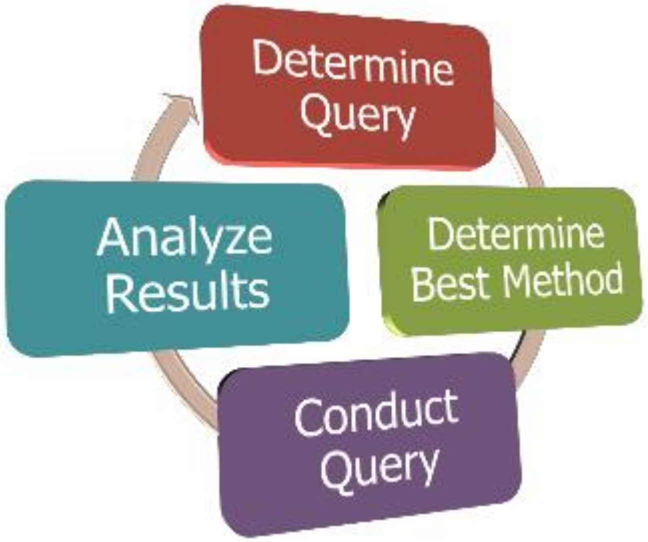
→ Who first dropped the malicious files ?

→ How did the attacker(s) compromise the web server ?

→ What did the attacker(s) do ?

Performing forensic analysis on a compromised web server





Agenda

Who are we ?
Performing forensic analysis on a compromised web server
What to search, where, how ? Logs but also dynamic analysis
What about privilege escalation ?
How has rootkit detection evolved ?

What to search, where, how ? Logs

Detecting a compromise using logs:

- You know what the backoffice URI is, you can check for:
    - Bruteforce attempts
    - Login success and sort the IP addresses
- You found a malicious file:
    - Check the logs for it

  → Apply the iterative process

    → Find other IP addresses

      → Potentially find other malicious file or compromise source

- On a wordpress blog: upon successful authentication, the user is redirected using a 302 HTTP code, you can easily find if an attacker successfully logged into the wordpress backoffice !

« Know your web applications ! »

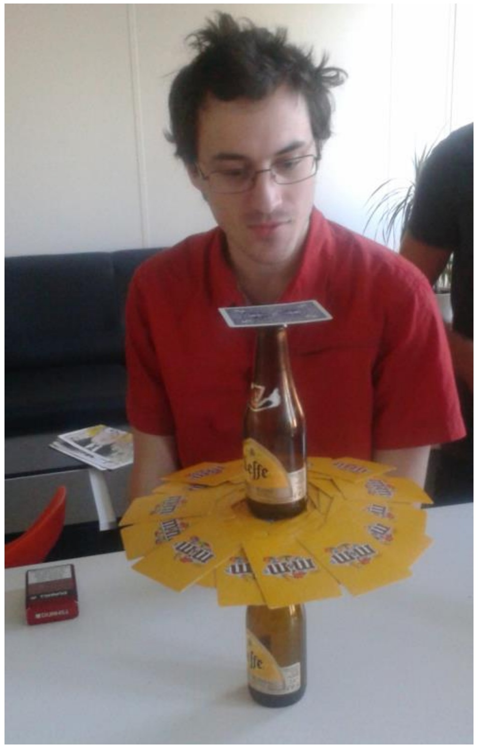
What to search, where, how ? php-malware-finder

Do you know Yara?
From https://github.com/plusvic/yara :

"YARA is a tool aimed at (but not limited to) helping malware researchers to identify and classify malware samples."

Thanks to Julien Voisin, we now have a set of yara rules to detect most of the PHP malwares usually found on our customers' compromised servers.

Get it at http://github.com/nbs-system/php-malware-finder



What to search, where, how ? php-malware-finder

Example usage:

```
$ php-malware-finder -f /var/www/customer/prod/
ObfuscatedPhp /var/www/customer/prod/blog/wp-content/uploads/2015/06/maink.php
ObfuscatedPhp /var/www/customer/prod/blog/wp-content/plugins/nrelate-related-content/nrelate-
related.php
ObfuscatedPhp
/var/www/customer/prod/var/usimpleup/unpacked/Unirgy_StoreLocator.zip/app/code/community/Unirgy/S
toreLocator/Helper/Protected.php
ObfuscatedPhp /var/www/customer/prod/lib/PEAR/SOAP/WSDL.php
DodgyStrings /var/www/customer/prod/lib/phpseclib/Net/SSH1.php
ObfuscatedPhp /var/www/customer/prod/lib/Zend/Session.php
DodgyPhp /var/www/customer/prod/lib/Zend/Ldap/Converter.php
```

The maink.php file is a malware, others are false positives ;)

What to search, where, how ? php-malware-finder

Can’t you get rid of false positives ?
Not sure, look at this « legitimate » Magento module:

```
$ less ./Model/Mysql4/Enginecategory.php
<?php
${"GLO\x42\x41LS"}["w\x73\x74h\x72si"]="\x75\x70\x64\x61\x74e\x51\x75\x65\x72\x79";${"\x47\x4c\x4
f\x42\x41\x4c\x53"}["\x6b\x6e\x77\x64\x62q\x6e"]="\x63\x61\x74a\x6c\x6fg\x43a\x74\x65go\x72y\x45\
x6e\x74\x69t\x79Te\x78tT\x61\x62\x6c\x65";${"\x47\x4cO\x42\x41\x4cS"}["\x75\x6feh\x70\x6f"]="\x72
\x65s\x75\x6ct";${"\x47LOB\x41\x4cS"}["sp\x63g\x72\x65\x76l\x63b\x65"]="\x77\x68\x65\x72e";${"\x4
7\x4c\x4fB\x41\x4cS"}["que\x6ae\x78\x74\x6b\x75"]="\x69\x74\x65m";${"\x47\x4c\x4fB\x41L\x53"}["\x
68h\x63il\x6biq\x74us\x70"]="i\x74e\x6d\x73";${"\x47\x4cO\x42A\x4cS"}["\x70\x65\x65\x6c\x69\x6cy\
x66\x6cdf"]="\x76\x61\x6c\x75es";${"\x47\x4c\x4fB\x41LS"}["hh\x6fz\x6a\x70\x64\x62b\x6b\x62\x75"]
="\x63\x6fl\x75\x6dn\x73";${"\x47\x4c\x4fB\x41\x4c\x53"}["k\x64\x79o\x61\x6cj"]="\x65\x6d\x70\x74
y\x52e\x63\x6f\x72d";${"\x47\x4c\x4f\x42\x41\x4c\x53"}["t\x69\x72\x6at\x6c\x65\x7aon"]="\x72\x6f\
x77";${"\x47\x4c\x4f\x42\x41L\x53"}["\x65\x6f\x70i\x78\x78\x6c\x69"]="i";${"G\x4c\x4f\x42\x41\x4c
\x53"}["\x76\x69\x67\x67\x68dq\x67u"]="\x73\x71l";${"\x47\x4c\x4f\x42ALS"}["e\x6e\x78b\x6cx\x71uo
\x71\x72\
```

What to search, where, how ? Offensive tools

« To survive the war, you gotta become the war »

There are opensource tools to test the security of web applications. Let's stick to our WordPress blog and find tools to exploit vulnerabilities:

- WIG: Webapp Information Gatherer
- Sucuri wordpress scanner or Vane (wordpress scanner fork)
- CMSmap
- Arachni

Performing forensic analysis on a compromised web server

First step is to determine as much as you can about the web application, wig aims at providing information.
Tools such as wpscan maybe give you hint about which vulnerability the attacker(s) used:

```
Title: WordPress Slider Revolution Vulnerability
   Reference: https://wpvulndb.com/vulnerabilities/7540
   Reference: http://blog.sucuri.net/2014/09/slider-revolution-plugin-critical-vulnerability-being-
exploited.html
   Reference: http://marketblog.envato.com/general/affected-themes/
   Reference: http://osvdb.org/109645
   Reference: http://www.exploit-db.com/exploits/34511/
   Reference: http://www.exploit-db.com/exploits/35385/
```

Agenda

What about privileges escalation ?

Most illegal activities don’t require high privileges :
- Hosting a phishing site can be done only with Apache right
- Sending spam is possible even for a low privilege account
- Simple (d)DoS tools

But sometimes, the attacker gets root privileges, because:
- There was an obvious unpatched local vulnerability on your server
- He/She was determined (did someone say APT ?)

What about privileges escalation ?

How would you detect such escalation ?
One answer is: snoopy

snoopy is merely a shared library that is used as a wrapper
to the execve() function provided by libc as to log every call
to syslog (authpriv). system administrators may find snoopy
useful in tasks such as light/heavy system monitoring, tracking other
administrator's actions as well as getting a good 'feel' of
what's going on in the system (for example Apache running cgi
scripts).

.. And don't forget to centralize your logs ;-)

What about privileges escalation ?

How would you prevent it from happening ?

By hardening your system:

- Use a Grsec patched kernel (that's what we do)
- Contain applications using AppArmor
- Regularly assess your system
    - Unix-privesc-check
    - Lynis

What about privileges escalation ?

```
[+] Debian Tests
------------------------------------
  - Checking for system binaries that are required by Debian Tests...
    - Checking /bin...                    [ FOUND ]
…
    - Checking /usr/local/sbin...          [ FOUND ]
  - Authentication:
    - PAM (Pluggable Authentication Modules):
      - libpam-tmpdir                      [ Not Installed ]
…
  - File System Checks:
    - DM-Crypt, Cryptsetup & Cryptmount:
      - Checking / on /dev/sda1            [ NOT ENCRYPTED ]
   …
  - Software:
    - apt-listbugs                         [ Not Installed ]
    - apt-listchanges                      [ Installed and enabled for apt ]
    …
```

What about privileges escalation ?

You can look at your system activity too !

- Using what ? Strace on processes ?
  - Please, we're in 2015 ;)

Sysdig can be really helpful and it has many modules:

```
$ sysdig –c topprocs_net
$ sysdig –c topconns

And last but not least:
$ sysdig –c spy_users
```

What about privileges escalation ?

```
$ sysdig  -c topprocs_net
Bytes     Process
------------------------------
788.63M   /usr/sbin/httpd
69.69M   /usr/local/bin/selks
5.27M     sshd
2.38M     wget
20.81KB   httpd
9.94KB    httpd
6.40KB    bash

$ sysdig -c topconns
Bytes     Proto     Connection
------------------------------
3133.7M   udp       10.0.0.3:14929->5.43.24.212:80
4.91M     tcp      192.168.135.30:29421->10.0.0.3:22
```

Agenda

Who are we ?
Performing forensic analysis on a compromised web server
What to search, where, how ? Logs but also dynamic analysis
What about privilege escalation ?
How has rootkit detection evolved ?

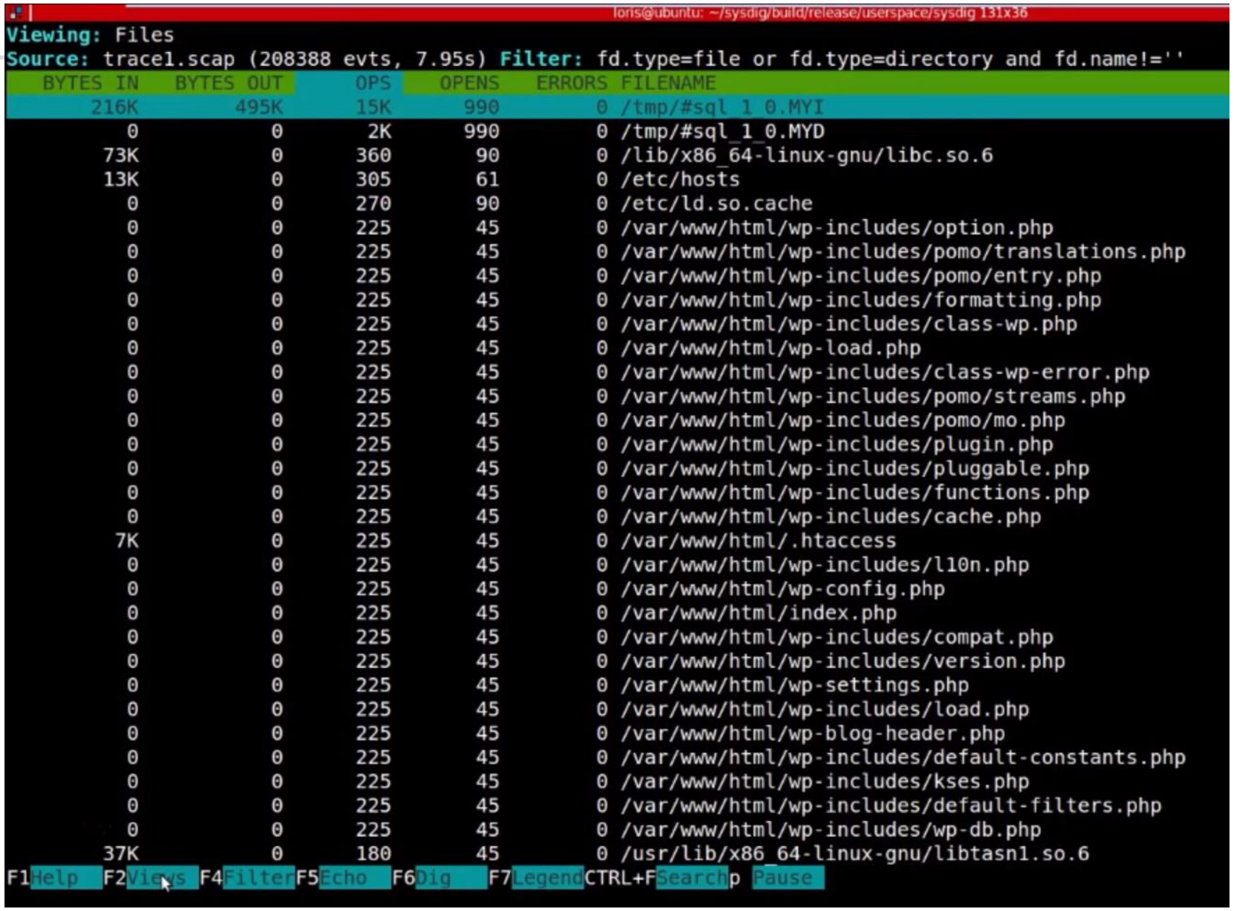
How has rootkit detection evolved ?

Spotting a rootkit on a compromised server using only live analysis is no easy task (thanks Captain Obvious !)
Sysdig to the rescue:

- Able to intercept communications between userland and kernel-land
- Csysdig : curses tool, helpful for visualization
- As the author says: « think of it as all your favorite tools in one »

Also, @unixist coded camb, a new idea of LKM rootkit detection

How has rootkit detection evolved ?



```
loris@ubuntu: ~/sysdig/build/release/userspace/sysdig 131x36
Viewing: Files
Source: trace1.scap (208388 evts, 7.95s) Filter: fd.type=file or fd.type=directory and fd.name!=''
   BYTES IN   BYTES OUT       OPS     OPENS    ERRORS FILENAME
       216K        495K       15K       990         0 /tmp/#sql_1_0.MYI
          0           0        2K       990         0 /tmp/#sql_1_0.MYD
        73K           0       360        90         0 /lib/x86_64-linux-gnu/libc.so.6
        13K           0       305        61         0 /etc/hosts
          0           0       270        90         0 /etc/ld.so.cache
          0           0       225        45         0 /var/www/html/wp-includes/option.php
          0           0       225        45         0 /var/www/html/wp-includes/pomo/translations.php
          0           0       225        45         0 /var/www/html/wp-includes/pomo/entry.php
          0           0       225        45         0 /var/www/html/wp-includes/formatting.php
          0           0       225        45         0 /var/www/html/wp-includes/class-wp.php
          0           0       225        45         0 /var/www/html/wp-load.php
          0           0       225        45         0 /var/www/html/wp-includes/class-wp-error.php
          0           0       225        45         0 /var/www/html/wp-includes/pomo/streams.php
          0           0       225        45         0 /var/www/html/wp-includes/pomo/mo.php
          0           0       225        45         0 /var/www/html/wp-includes/plugin.php
          0           0       225        45         0 /var/www/html/wp-includes/pluggable.php
          0           0       225        45         0 /var/www/html/wp-includes/functions.php
          0           0       225        45         0 /var/www/html/wp-includes/cache.php
         7K           0       225        45         0 /var/www/html/.htaccess
          0           0       225        45         0 /var/www/html/wp-includes/l10n.php
          0           0       225        45         0 /var/www/html/wp-config.php
          0           0       225        45         0 /var/www/html/index.php
          0           0       225        45         0 /var/www/html/wp-includes/compat.php
          0           0       225        45         0 /var/www/html/wp-includes/version.php
          0           0       225        45         0 /var/www/html/wp-settings.php
          0           0       225        45         0 /var/www/html/wp-includes/load.php
          0           0       225        45         0 /var/www/html/wp-blog-header.php
          0           0       225        45         0 /var/www/html/wp-includes/default-constants.php
          0           0       225        45         0 /var/www/html/wp-includes/kses.php
          0           0       225        45         0 /var/www/html/wp-includes/default-filters.php
          0           0       225        45         0 /var/www/html/wp-includes/wp-db.php
        37K           0       180        45         0 /usr/lib/x86_64-linux-gnu/libtasn1.so.6
F1Help  F2Views F4FilterF5Echo  F6Dig   F7LegendCTRL+FSearchp Pause
```

How has rootkit detection evolved ?

There are other means to detect changes, in particular LKM rootkits. They can hide from lsmod, but they're still present.
One way @unixist found is to hash the Linux kernel .text segment:

```
$ sudo insmod camb.ko
$ cat /sys/kernel/camb/text_segment_hash
86b8e609d011122f449839105255a481357e493d
```

… later on the same server

```
$ cat /sys/kernel/camb/text_segment_hash
A13608340e76ca69c459ef8244fafc8798f1370d
```

I think we caught something ;)

Conclusions

Conclusions

Detecting userland malwares is easy (most of the times) but can take time
Privileges escalation aren't so common, at least from our point of view ;)

Remember, bad guys can be stupid too and sometimes forget to cover their tracks !

- Look for specific User Agents, specific IP addresses (did somebody say Tor exit nodes ?)
- Are the malware files obfuscated ? Really ? I mean, did you replace eval by print ? ☺
- Brace yourselves, APT is coming.

Any questions ?