VULNERABILITY ASSESSMENT & REVERSE ENGINEERING (CT-371)

MALWARE ANALYSIS USING CUCKOO SANDBOX

GROUP MEMBERS	ROLL NO
MUHAMMAD MUSAB SALEEM	(CR-22012)
YASRA KHAN	(CR-22005)
BARIRA TARIQ	(CR_22017)

Malware Analysis Using Cuckoo Sandbox

Introduction

Cuckoo Sandbox is an advanced, open-source malware analysis tool designed to execute suspicious files in a controlled environment, monitor their behavior, and generate comprehensive reports. This project involved setting up and enhancing the Cuckoo Sandbox platform with AI integrations, memory forensics, and YARA rule detection to analyze malware in virtual environments without risking the host system. We deployed it on Ubuntu 18.04, integrated supporting tools like Volatility and tcpdump, and utilized a Windows 7 VM for analysis.

Cuckoo provides a web-based interface on the front end, while the backend is powered by MongoDB. One of its key strengths is the ability to export large volumes of data, all accessible through the web interface. It monitors all activity during a malware's execution — including file creation, deletion, downloads, and more — and relays this information back to the sandbox.

Users can extract memory dumps of the malware or even the entire virtual machine for deeper inspection. Cuckoo supports analysis of various file types, including Windows executables, DLLs, PDFs, Office documents, URLs, HTML, PHP scripts, CPL files, macro-enabled documents, VB scripts, ZIP archives, JAR files, Python scripts, and more.

AI-Powered Analysis:

While Cuckoo Sandbox itself doesn't use AI for its core analysis, it can be used to generate data that AI models can then analyze to detect or classify malware.

Methodology and Tools Used

Operating System

- **Host**: Ubuntu 18.04 LTS
- **Guest**: Windows 7 x64 VM

Main Tools & Technologies

Tool	Purpose
Cuckoo Sandbox	Dynamic malware analysis and behavior observation
VirtualBox	Hosting isolated guest VMs
Python 2.7	Cuckoo core components
Volatility	Memory forensics and RAM analysis
tcpdump	Network traffic monitoring
MongoDB/PostgreSQL	Backend databases for storing logs
vmcloak	Automates the Windows VM creation for sandboxing
YARA	Signature-based malware detection
Supervisord	Background service management for automation

Cuckoo Sandbox Installation & Configuration Guide

Prerequisites

This is the process we followed to install Cuckoo Sandbox on my machine. To begin with, several prerequisite software packages and libraries need to be installed before setting up and configuring Cuckoo itself.

Since Cuckoo's core components are entirely developed in Python, having the correct version of Python installed on your system is essential.

The following packages are required to let Cuckoo get installed:

- Install Python 2.7 and essential development libraries
  - *sudo apt-get install python python-pip python-dev libffi-dev libssl-dev*
- Install tools for creating isolated Python environments
  - *sudo apt-get install python-virtualenv python-setuptools*
- Install additional libraries needed for image processing and compression
  - *sudo apt-get install libjpeg-dev zlib1g-dev swig*

ADDITIONAL PACKAGE REQUIREMENTS FOR CUCKOO SANDBOX

- To fully utilize all the features of Cuckoo Sandbox, including its web interface and support for various virtualization backends, you'll need to install several additional packages:
- Install MongoDB - required for the Django-based web interface
  - *sudo apt-get install mongodb*
- Install PostgreSQL - used as the main database engine
  - *sudo apt-get install postgresql libpq-dev*
- Install KVM and related tools if using KVM as the virtualization backend
  - *sudo apt-get install qemu-kvm libvirt-bin ubuntu-vm-builder bridge-utils python-libvirt*
- Install XenAPI Python module if using XenServer

- *sudo pip install XenAPI*

Installing tcpdump and volatility for cuckoo sandbox

Installing tcpdump

tcpdump is a powerful command-line packet analyzer used to capture or filter TCP/IP traffic over network interfaces. Cuckoo Sandbox uses tcpdump by default to monitor and record network activity during malware execution.
To install tcpdump along with AppArmor utilities:

```
sudo apt-get install tcpdump apparmor-utils

sudo aa-disable /usr/sbin/tcpdump
```

If apparmor is not enabled on your system, you can simply install tcpdump with:

```
sudo apt-get install tcpdump
```

Since tcpdump requires root privileges, and cuckoo should not run as root, you need to configure the correct permissions:

```
sudo groupadd pcap

sudo usermod -a -G pcap cuckoo

sudo chgrp pcap /usr/sbin/tcpdump

sudo setcap cap_net_raw,cap_net_admin=eip /usr/sbin/tcpdump
```

If the setcap tool is not available on your system, install it using:

```
sudo apt-get install libcap2-bin
```

Installing volatility

Volatility is an optional yet highly recommended memory forensics framework. It allows in-depth analysis of memory dumps created during malware execution, providing insights into advanced threats such as rootkits that evade regular monitoring.

TO INSTALL VOLATILITY ON UBUNTU:

Step 1: Update package list

```
sudo apt-get update -y
```

Step 2: Install Volatility and its dependencies

```
sudo apt-get install -y volatility
```

Volatility works out of the box and requires no additional configuration. You can use it independently or integrate it into Cuckoo for enhanced memory analysis capabilities.

Installing M2Crypto

M2Crypto is a Python wrapper for OpenSSL, offering support for SSL, RSA, DSA, and more. It is required by some components of Cuckoo Sandbox for secure communications and cryptographic functions.

Before installing M2Crypto, make sure that SWIG (Simplified Wrapper and Interface Generator) is installed, as it is a prerequisite for compiling M2Crypto from source.

RUN THE FOLLOWING COMMANDS:
Install SWIG - required to build M2Crypto

```
sudo apt-get install swig
```

Install the specific compatible version of M2Crypto

```
sudo pip install m2crypto==0.24.0
```

If you're having trouble installing M2Crypto (especially the older 0.24.0 version), or you'd prefer an alternative that avoids SWIG dependencies, you can consider using pyOpenSSL instead, if the component requiring M2Crypto is not strictly dependent on it.
Install pip if not already present

```
sudo apt-get install python-pip
```

Install pyOpenSSL (a widely used alternative to M2Crypto)

```
sudo pip install pyOpenSSL==19.0.0
```

Note: While pyOpenSSL provides many of the same features, it's not a drop-in replacement for M2Crypto in all tools. If Cuckoo or any plugin specifically requires M2Crypto, you'll still need to install it using SWIG.
If you're set on M2Crypto, here's a more stable approach:
Ensure dependencies are in place

```
sudo apt-get install swig libssl-dev python-dev
```

Install a compatible version

```
sudo pip install m2crypto==0.24.0
```

Installing cuckoo sandbox

Step 1: Create a New User for Cuckoo

It's a best practice to run Cuckoo under a dedicated non-root user for security reasons.

```
sudo adduser cuckoo
```

Step 2: Install Required Python Tools

Ensure you have the latest versions of pip and setuptools.

```
sudo pip install -U pip setuptools
```

Step 3: Install Cuckoo

You can install the latest version of Cuckoo directly from PyPI:

```
sudo pip install -U cuckoo
```

Step 4: (Recommended) Use a Virtual Environment

It is highly recommended to isolate Cuckoo and its dependencies in a Python virtual environment:

Create a virtual environment

```
virtualenv venv
```

Activate the virtual environment

```
$. venv/bin/activate
```

Upgrade pip and setuptools inside the virtual environment

```
pip install -U pip setuptools
```

Install Cuckoo in the virtual environment *pip install -U cuckoo*

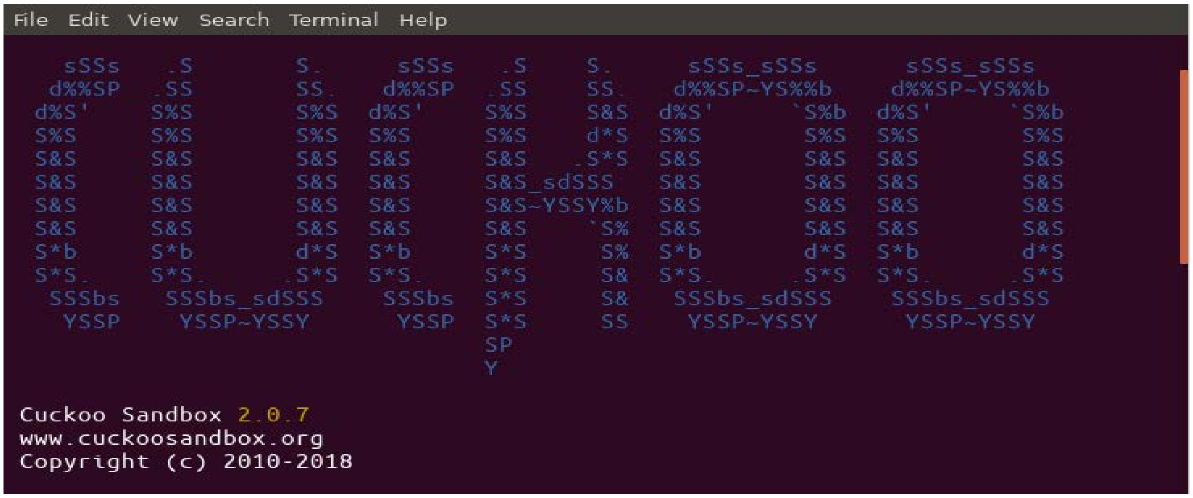
Cuckoo Working Directory

When you first run Cuckoo, a working directory (~/.cuckoo) will be automatically created. This directory contains:

- Configuration files
- Custom and default signatures
- Cuckoo Analyzer and Agent
- YARA rules
- Storage for analysis results
- And other runtime data
- Initial Configuration

To initialize Cuckoo and generate the required working directory structure:

cuckoo -d



```
  sSSs   .S       S.    sSSs   .S    S.     sSSs_sSSs      sSSs_sSSs
 d%%SP  .SS       SS.  d%%SP  .SS    SS.   d%%SP~YS%%b    d%%SP~YS%%b
d%S'    S%S       S%S d%S'    S%S    S&S  d%S'     `S%b  d%S'     `S%b
S%S     S%S       S%S S%S     S%S    d*S  S%S       S%S  S%S       S%S
S&S     S&S       S&S S&S     S&S   .S*S  S&S       S&S  S&S       S&S
S&S     S&S       S&S S&S     S&S_sdSSS   S&S       S&S  S&S       S&S
S&S     S&S       S&S S&S     S&S~YSSY%b  S&S       S&S  S&S       S&S
S&S     S&S       S&S S&S     S&S    `S%  S&S       S&S  S&S       S&S
S*b     S*b       d*S S*b     S*S     S%  S*b       d*S  S*b       d*S
S*S.    S*S.     .S*S S*S.    S*S     S&  S*S.     .S*S  S*S.     .S*S
 SSSbs   SSSbs_sdSSS   SSSbs  S*S     S&   SSSbs_sdSSS    SSSbs_sdSSS
  YSSP    YSSP~YSSY     YSSP  S*S     SS    YSSP~YSSY      YSSP~YSSY
                              SP
                              Y

Cuckoo Sandbox 2.0.7
www.cuckoosandbox.org
Copyright (c) 2010-2018
```

The -d flag runs Cuckoo in debug mode, which is helpful for troubleshooting during setup.

Configuring Multiple Cuckoo Instances with Alternative CWD Paths

To run multiple instances of Cuckoo with different configurations using the same setup, you can configure alternative CWD (Current Working Directory) paths. This allows each instance to operate independently with its own set of configurations and results.

Step 1: Create a Custom CWD Directory

First, create a directory where your alternate Cuckoo working directories will reside:

sudo mkdir /opt/cuckoo

Step 2: Change Ownership to the Cuckoo User

Ensure that the Cuckoo user has ownership of the new directory:

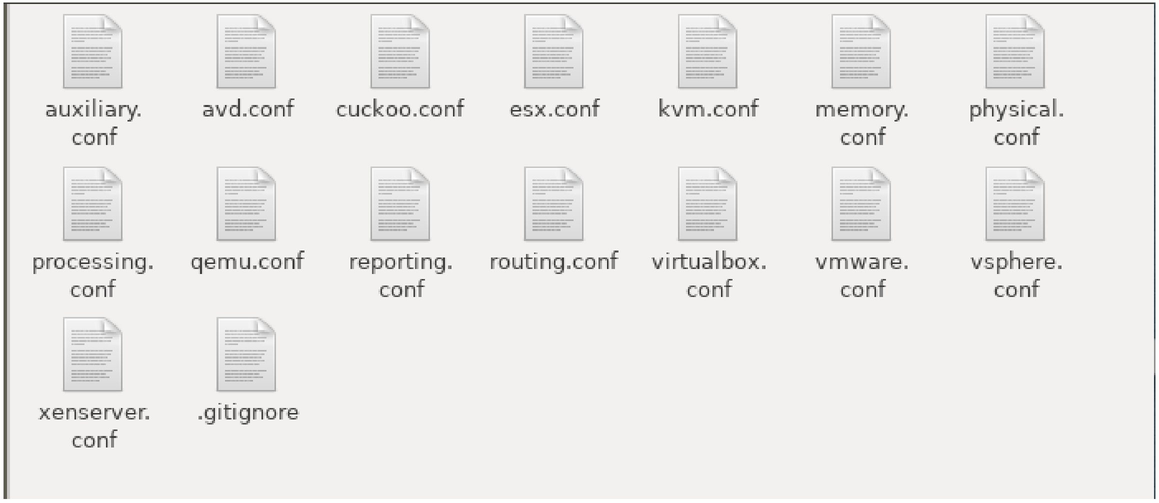
```
sudo chown cuckoo:cuckoo /opt/cuckoo
```

Step 3: Run Cuckoo with the Custom CWD

When launching a Cuckoo instance, specify the new CWD path using the --cwd option:

cuckoo --cwd /opt/cuckoo

This tells Cuckoo to use /opt/cuckoo as its working directory, where all configurations, data, and results will be stored for this particular instance.



Cuckoo Configuration files

- cuckoo.conf: for configuring general behavior and analysis options.
- auxiliary.conf: for enabling and configuring auxiliary modules.
- <machinery>.conf: for defining the options for your virtualization software (the file has the same name of the machinery module you choose in cuckoo.conf).
- memory.conf: Volatility configuration.
- processing.conf: for enabling and configuring processing modules.
- reporting.conf: for enabling or disabling report formats.

Creating a Windows 7 Virtual Machine for Cuckoo Sandbox

To set up a Windows 7 VM for use with Cuckoo Sandbox, follow these steps to ensure the virtual machine is properly configured and ready for malware analysis.

Step 1: Prepare the Environment

Before creating the VM, ensure you have the Windows 7 .iso image in your current directory. This image will be used for the installation.

Create a Mount Directory:
Set up a directory where the Windows 7 installation ISO will be mounted:

```
sudo mkdir /mnt/win7
sudo chown cuckoo:cuckoo /mnt/win7
```

Check User Group Policy:
Ensure that the cuckoo user is a member of the vboxusers group for VirtualBox access:

```
sudo usermod -a -G vboxusers cuckoo
```

Step 2: Mount the ISO File

Next, mount the Windows 7 ISO image to make it accessible for the installation:

Mount the ISO:

```
sudo mount -o ro,loop win7ultimate.iso /mnt/win7
```

Step 3: Install Dependencies

To ensure your system is fully prepared for the VM setup, install the necessary dependencies:

Install Required Packages:

```
sudo apt-get -y install build-essential libssl-dev libffi-dev python-dev genisoimage

sudo apt-get -y install zlib1g-dev libjpeg-dev

sudo apt-get -y install python-pip python-virtualenv python-setuptools swig
```

Step 4: Install and Configure vmcloak

vmcloak is a tool that automates the creation and configuration of virtual machines for use in sandboxing environments like Cuckoo. This tool simplifies the setup process by handling most of the configurations for you.

Install vmcloak:

```
pip install -U vmcloak
```

Create a VirtualBox Network:
Remove any existing vboxnet networks, and then create a new one using vmcloak:

```
vmcloak-vboxnet0
```

Step 5: Configure the Virtual Machine

Once the dependencies are installed, you can create and configure the Windows 7 virtual machine:

Initialize the VM:
Set up the VM configuration, specifying 2 CPUs and 2GB RAM (ideal for malware analysis):

```
vmcloak init --verbose --win7x64 win7x64base --cpus 2 --ramsize 2048
```

Clone the New VM:
Clone the newly created VM for further use. This will save you time by eliminating the need to configure a VM from scratch in the future:

```
vmcloak clone win7x64base win7x64cuckoo
```

List Required Packages for the VM:
You can view the list of packages needed for the VM configuration:

```
vmcloak list deps
```

Step 6: Final Checks

Once everything is configured, your Windows 7 VM should be ready to integrate with Cuckoo Sandbox. Ensure the VM is functioning as expected and is properly isolated from the host system.

Web Interface for Cuckoo Sandbox

Cuckoo provides a comprehensive web interface built with Django that allows you to interact with the system. You can easily submit files, browse through reports, and search through all the analysis results using this interface.

Starting the Web Interface

To start the Cuckoo Web Interface, you can use one of the following commands in your terminal:

Start Web Interface:

```
$ cuckoo web runserver
```

Start Web Interface with Host Binding:
Alternatively, if you want to bind the interface to all available network interfaces, use the following command:

```
$ cuckoo web -H 0
```

Starting Cuckoo Sandbox

To start the Cuckoo Sandbox process, simply use the command below:

```
$ cuckoo
```

Note: Make sure to activate your virtual environment before starting Cuckoo:

```
$. venv/bin/activate
```

Cuckoo provides several command-line options, which you can check using:

```
$ cuckoo — help
```

Cuckoo Processing Instances

When working with more than 4 VMs, it's recommended to use Cuckoo processing instances. To set this up, first configure Postgres and disable the processing of results in the main Cuckoo process.

Edit Configuration:
Open the cuckoo.conf file and change the following setting:

```
process_results = yes  # Change to
process_results = no
```

Start Processing Instances:
Once the configuration is updated, you can start one or more processing instances using the following command:

```
cuckoo process <instance_name>
```

Running Cuckoo in the Background with Supervisord

If you'd like to run Cuckoo and its supporting processes in the background, supervisord is the recommended solution. Cuckoo generates a supervisord.conf file in the CWD (current working directory) to make this setup easy.

Install supervisord:

sudo apt-get install supervisord

Start Cuckoo in the Background:
Once supervisord is installed, you can start and stop Cuckoo in the background with the following commands:

Start Cuckoo:

supervisorctl start cuckoo

Stop Cuckoo:

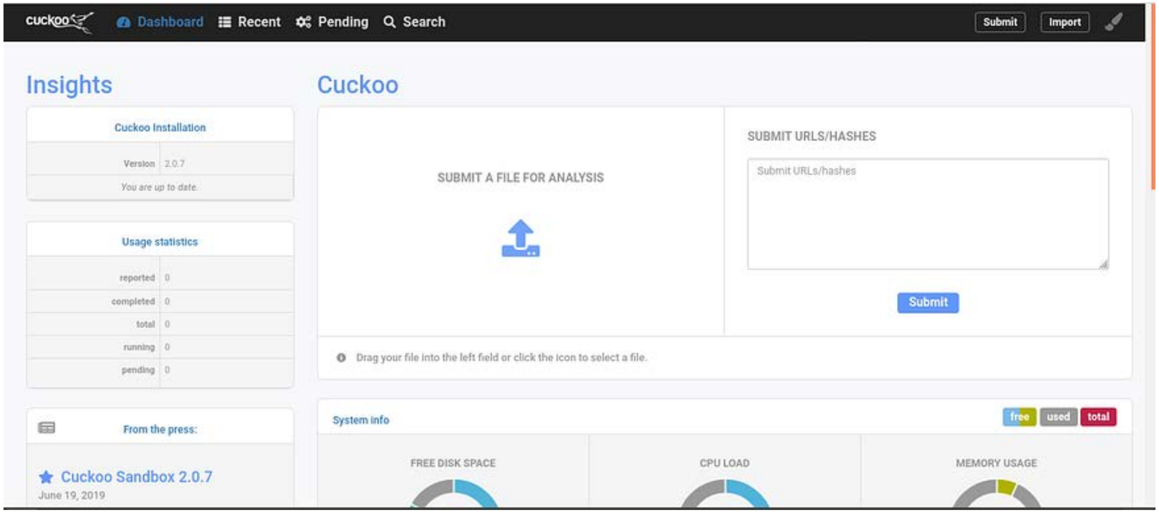
supervisorctl stop cuckoo

By following these steps, you can easily set up the Cuckoo web interface and run it as a background process using supervisord for efficient and uninterrupted malware analysis.

Uploading Malicious file:

Now, go to the web interface as shown in figure and select SUBMIT A FILE FOR ANALYSIS and upload the file you want to analyse.

You can upload a file from your machine or from the web.



Cuckoo Installation

Version	2.0.7

You are up to date.

Usage statistics

reported	0
completed	0
total	0
running	0
pending	0

Drag your file into the left field or click the icon to select a file.

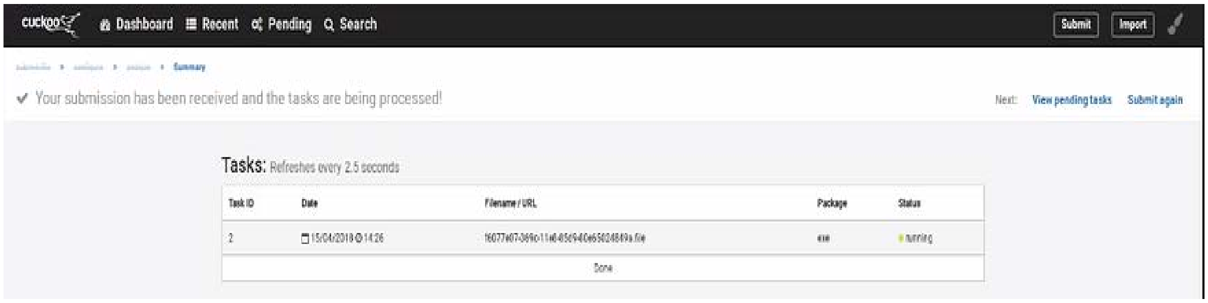
System info

From the press:

Cuckoo Sandbox 2.0.7
June 19, 2019

Web Interface of Cuckoo Sandbox

After uploading the files, recheck and submit them.



✓ Your submission has been received and the tasks are being processed!

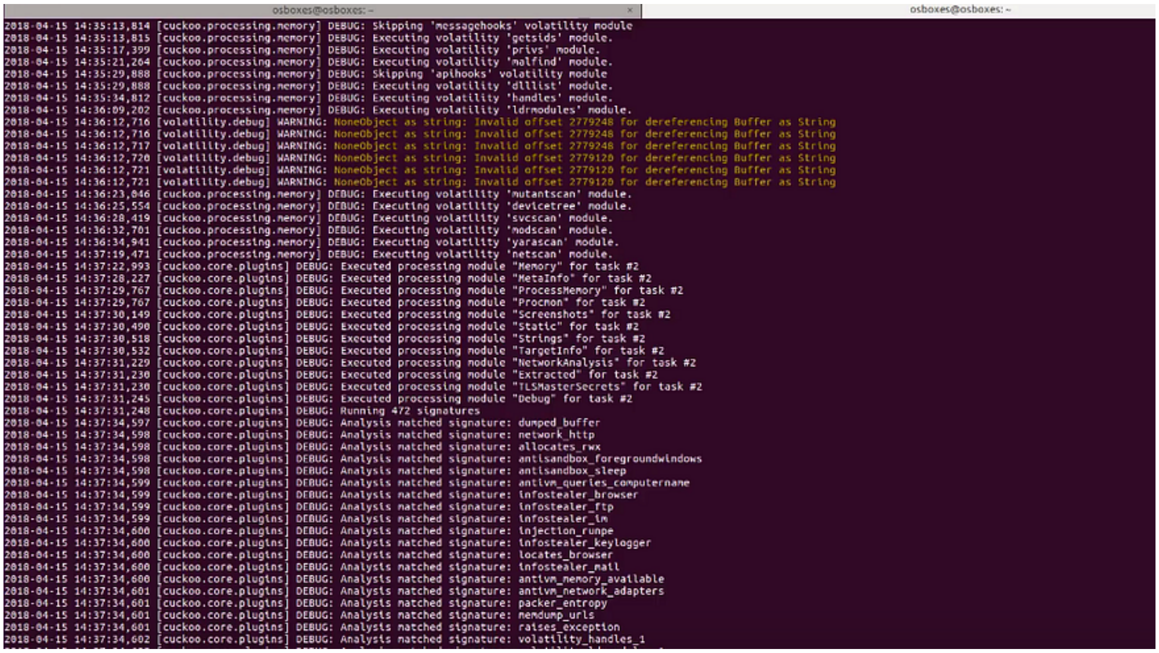
Tasks: Refreshes every 2.5 seconds

Task ID	Date	Filename / URL	Package	Status
2	15/04/2018 14:26	f6077e07-369c-11e8-85d9-40e65024849a.file	exe	running

Done

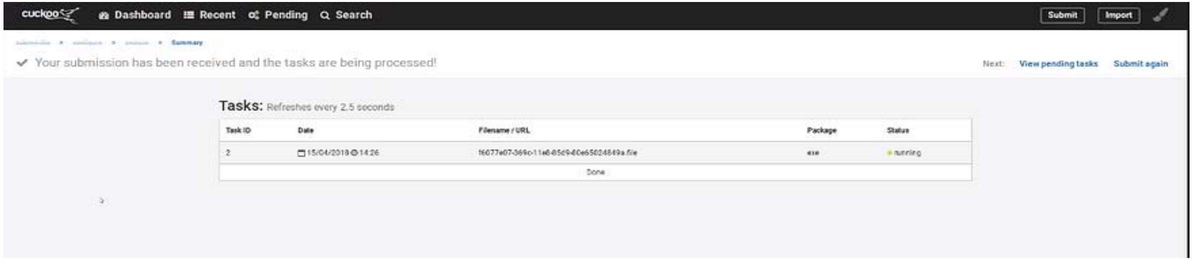
Verification after submitting a file

After submitting, the Sandbox starts analysing the file as shown in figure



```
2018-04-15 14:35:13,814 [cuckoo.processing.memory] DEBUG: Skipping 'messagehooks' volatility module
2018-04-15 14:35:13,815 [cuckoo.processing.memory] DEBUG: Executing volatility 'getsids' module.
2018-04-15 14:35:17,399 [cuckoo.processing.memory] DEBUG: Executing volatility 'privs' module.
2018-04-15 14:35:21,264 [cuckoo.processing.memory] DEBUG: Executing volatility 'malfind' module.
2018-04-15 14:35:29,888 [cuckoo.processing.memory] DEBUG: Skipping 'apihooks' volatility module
2018-04-15 14:35:29,888 [cuckoo.processing.memory] DEBUG: Executing volatility 'dlllist' module.
2018-04-15 14:35:34,812 [cuckoo.processing.memory] DEBUG: Executing volatility 'handles' module.
2018-04-15 14:36:09,202 [cuckoo.processing.memory] DEBUG: Executing volatility 'ldrmodules' module.
2018-04-15 14:36:12,716 [volatility.debug] WARNING: NoneObject as string: Invalid offset 2779248 for dereferencing Buffer as String
2018-04-15 14:36:12,716 [volatility.debug] WARNING: NoneObject as string: Invalid offset 2779248 for dereferencing Buffer as String
2018-04-15 14:36:12,717 [volatility.debug] WARNING: NoneObject as string: Invalid offset 2779248 for dereferencing Buffer as String
2018-04-15 14:36:12,720 [volatility.debug] WARNING: NoneObject as string: Invalid offset 2779120 for dereferencing Buffer as String
2018-04-15 14:36:12,721 [volatility.debug] WARNING: NoneObject as string: Invalid offset 2779120 for dereferencing Buffer as String
2018-04-15 14:36:12,721 [volatility.debug] WARNING: NoneObject as string: Invalid offset 2779120 for dereferencing Buffer as String
2018-04-15 14:36:23,046 [cuckoo.processing.memory] DEBUG: Executing volatility 'mutantscan' module.
2018-04-15 14:36:25,554 [cuckoo.processing.memory] DEBUG: Executing volatility 'devicetree' module.
2018-04-15 14:36:28,419 [cuckoo.processing.memory] DEBUG: Executing volatility 'svcscan' module.
2018-04-15 14:36:32,701 [cuckoo.processing.memory] DEBUG: Executing volatility 'modscan' module.
2018-04-15 14:36:34,941 [cuckoo.processing.memory] DEBUG: Executing volatility 'yarascan' module.
2018-04-15 14:37:19,471 [cuckoo.processing.memory] DEBUG: Executing volatility 'netscan' module.
2018-04-15 14:37:22,993 [cuckoo.core.plugins] DEBUG: Executed processing module "Memory" for task #2
2018-04-15 14:37:28,227 [cuckoo.core.plugins] DEBUG: Executed processing module "MetaInfo" for task #2
2018-04-15 14:37:29,767 [cuckoo.core.plugins] DEBUG: Executed processing module "ProcessMemory" for task #2
2018-04-15 14:37:29,767 [cuckoo.core.plugins] DEBUG: Executed processing module "Procmon" for task #2
2018-04-15 14:37:30,149 [cuckoo.core.plugins] DEBUG: Executed processing module "Screenshots" for task #2
2018-04-15 14:37:30,490 [cuckoo.core.plugins] DEBUG: Executed processing module "Static" for task #2
2018-04-15 14:37:30,518 [cuckoo.core.plugins] DEBUG: Executed processing module "Strings" for task #2
2018-04-15 14:37:30,532 [cuckoo.core.plugins] DEBUG: Executed processing module "TargetInfo" for task #2
2018-04-15 14:37:31,229 [cuckoo.core.plugins] DEBUG: Executed processing module "NetworkAnalysis" for task #2
2018-04-15 14:37:31,230 [cuckoo.core.plugins] DEBUG: Executed processing module "Extracted" for task #2
2018-04-15 14:37:31,230 [cuckoo.core.plugins] DEBUG: Executed processing module "TLSMasterSecrets" for task #2
2018-04-15 14:37:31,245 [cuckoo.core.plugins] DEBUG: Executed processing module "Debug" for task #2
2018-04-15 14:37:31,248 [cuckoo.core.plugins] DEBUG: Running 472 signatures
2018-04-15 14:37:34,597 [cuckoo.core.plugins] DEBUG: Analysis matched signature: dumped_buffer
2018-04-15 14:37:34,598 [cuckoo.core.plugins] DEBUG: Analysis matched signature: network_http
2018-04-15 14:37:34,598 [cuckoo.core.plugins] DEBUG: Analysis matched signature: allocates_rwx
2018-04-15 14:37:34,598 [cuckoo.core.plugins] DEBUG: Analysis matched signature: antisandbox_foregroundwindows
2018-04-15 14:37:34,598 [cuckoo.core.plugins] DEBUG: Analysis matched signature: antisandbox_sleep
2018-04-15 14:37:34,599 [cuckoo.core.plugins] DEBUG: Analysis matched signature: antivm_queries_computername
2018-04-15 14:37:34,599 [cuckoo.core.plugins] DEBUG: Analysis matched signature: infostealer_browser
2018-04-15 14:37:34,599 [cuckoo.core.plugins] DEBUG: Analysis matched signature: infostealer_ftp
2018-04-15 14:37:34,599 [cuckoo.core.plugins] DEBUG: Analysis matched signature: infostealer_im
2018-04-15 14:37:34,600 [cuckoo.core.plugins] DEBUG: Analysis matched signature: injection_runpe
2018-04-15 14:37:34,600 [cuckoo.core.plugins] DEBUG: Analysis matched signature: infostealer_keylogger
2018-04-15 14:37:34,600 [cuckoo.core.plugins] DEBUG: Analysis matched signature: locates_browser
2018-04-15 14:37:34,600 [cuckoo.core.plugins] DEBUG: Analysis matched signature: infostealer_mail
2018-04-15 14:37:34,600 [cuckoo.core.plugins] DEBUG: Analysis matched signature: antivm_memory_available
2018-04-15 14:37:34,601 [cuckoo.core.plugins] DEBUG: Analysis matched signature: antivm_network_adapters
2018-04-15 14:37:34,601 [cuckoo.core.plugins] DEBUG: Analysis matched signature: packer_entropy
2018-04-15 14:37:34,601 [cuckoo.core.plugins] DEBUG: Analysis matched signature: memdump_urls
2018-04-15 14:37:34,601 [cuckoo.core.plugins] DEBUG: Analysis matched signature: raises_exception
2018-04-15 14:37:34,602 [cuckoo.core.plugins] DEBUG: Analysis matched signature: volatility_handles_1
```

Terminal during analysis of the uploaded file.



✓ Your submission has been received and the tasks are being processed!

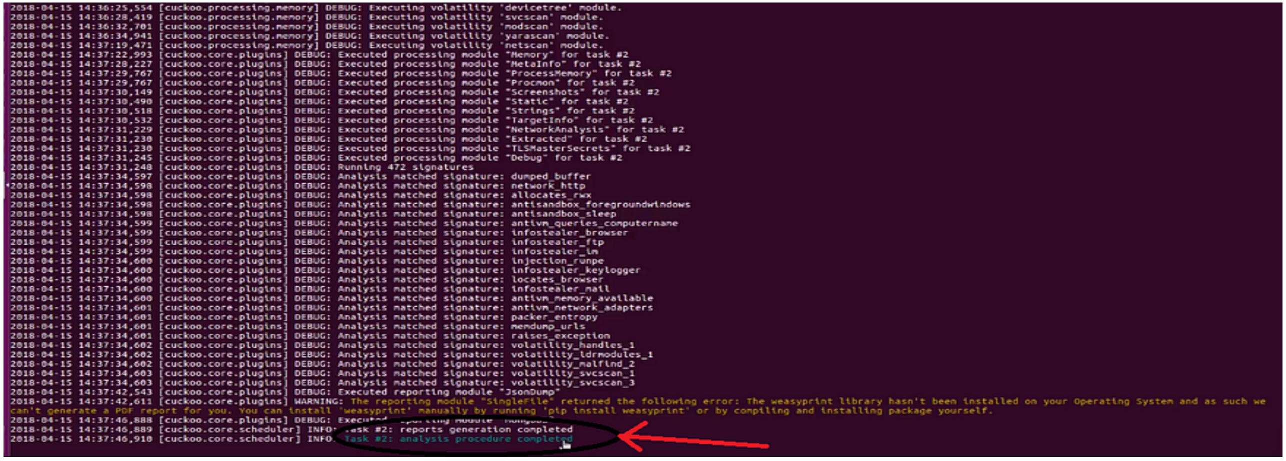
Next: View pending tasks Submit again

Tasks: Refreshes every 2.5 seconds

Task ID	Date	Filename / URL	Package	Status
2	15/04/2018 14:26	f6077e07-369c-11e8-85d9-80e65024849a.file	exe	● running
		Done		

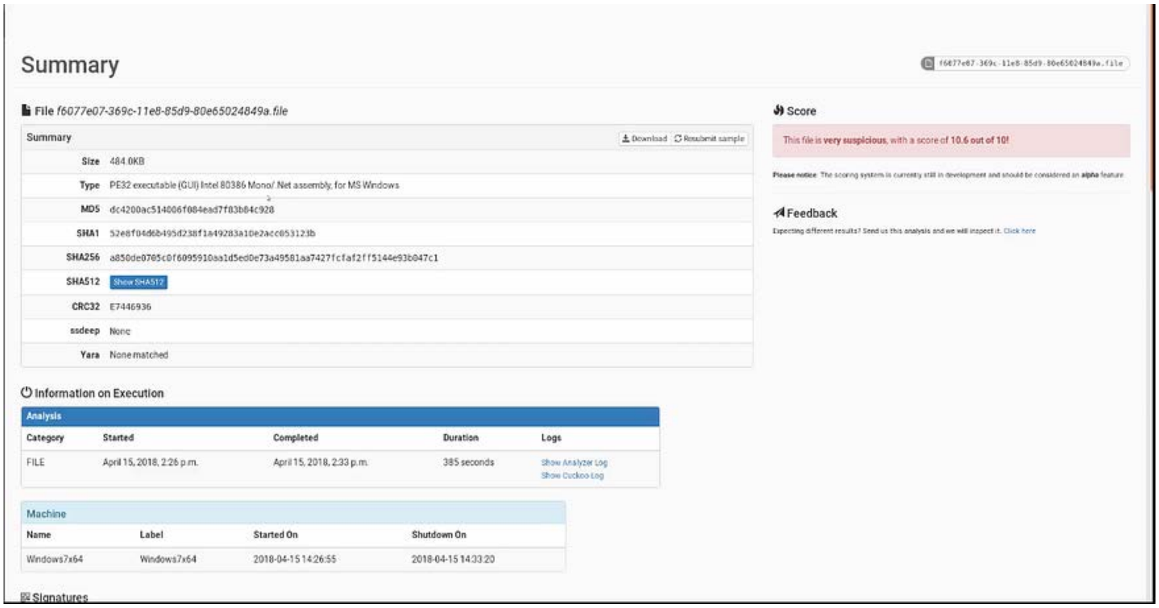
Web Interface during the analysis of a file.

This process may take some time depending on the size and type of the file uploaded and the internet connection, after finishing the analysis the terminal looks as shown in figure



```
2018-04-15 14:36:25,554 [cuckoo.processing.memory] DEBUG: Executing volatility 'devicetree' module.
2018-04-15 14:36:28,419 [cuckoo.processing.memory] DEBUG: Executing volatility 'svcscan' module.
2018-04-15 14:36:32,701 [cuckoo.processing.memory] DEBUG: Executing volatility 'modscan' module.
2018-04-15 14:36:34,941 [cuckoo.processing.memory] DEBUG: Executing volatility 'yarascan' module.
2018-04-15 14:37:19,471 [cuckoo.processing.memory] DEBUG: Executing volatility 'netscan' module.
2018-04-15 14:37:22,993 [cuckoo.core.plugins] DEBUG: Executed processing module "Memory" for task #2
2018-04-15 14:37:28,227 [cuckoo.core.plugins] DEBUG: Executed processing module "MetaInfo" for task #2
2018-04-15 14:37:29,767 [cuckoo.core.plugins] DEBUG: Executed processing module "ProcessMemory" for task #2
2018-04-15 14:37:29,767 [cuckoo.core.plugins] DEBUG: Executed processing module "Procmon" for task #2
2018-04-15 14:37:30,149 [cuckoo.core.plugins] DEBUG: Executed processing module "Screenshots" for task #2
2018-04-15 14:37:30,490 [cuckoo.core.plugins] DEBUG: Executed processing module "Static" for task #2
2018-04-15 14:37:30,518 [cuckoo.core.plugins] DEBUG: Executed processing module "Strings" for task #2
2018-04-15 14:37:30,532 [cuckoo.core.plugins] DEBUG: Executed processing module "TargetInfo" for task #2
2018-04-15 14:37:31,229 [cuckoo.core.plugins] DEBUG: Executed processing module "NetworkAnalysis" for task #2
2018-04-15 14:37:31,230 [cuckoo.core.plugins] DEBUG: Executed processing module "Extracted" for task #2
2018-04-15 14:37:31,230 [cuckoo.core.plugins] DEBUG: Executed processing module "TLSMasterSecrets" for task #2
2018-04-15 14:37:31,245 [cuckoo.core.plugins] DEBUG: Executed processing module "Debug" for task #2
2018-04-15 14:37:31,248 [cuckoo.core.plugins] DEBUG: Running 472 signatures
2018-04-15 14:37:34,597 [cuckoo.core.plugins] DEBUG: Analysis matched signature: dumped_buffer
2018-04-15 14:37:34,598 [cuckoo.core.plugins] DEBUG: Analysis matched signature: network_http
2018-04-15 14:37:34,598 [cuckoo.core.plugins] DEBUG: Analysis matched signature: allocates_rwx
2018-04-15 14:37:34,598 [cuckoo.core.plugins] DEBUG: Analysis matched signature: antisandbox_foregroundwindows
2018-04-15 14:37:34,598 [cuckoo.core.plugins] DEBUG: Analysis matched signature: antisandbox_sleep
2018-04-15 14:37:34,599 [cuckoo.core.plugins] DEBUG: Analysis matched signature: antivm_queries_computername
2018-04-15 14:37:34,599 [cuckoo.core.plugins] DEBUG: Analysis matched signature: infostealer_browser
2018-04-15 14:37:34,599 [cuckoo.core.plugins] DEBUG: Analysis matched signature: infostealer_ftp
2018-04-15 14:37:34,599 [cuckoo.core.plugins] DEBUG: Analysis matched signature: infostealer_im
2018-04-15 14:37:34,600 [cuckoo.core.plugins] DEBUG: Analysis matched signature: injection_runpe
2018-04-15 14:37:34,600 [cuckoo.core.plugins] DEBUG: Analysis matched signature: infostealer_keylogger
2018-04-15 14:37:34,600 [cuckoo.core.plugins] DEBUG: Analysis matched signature: locates_browser
2018-04-15 14:37:34,600 [cuckoo.core.plugins] DEBUG: Analysis matched signature: infostealer_mail
2018-04-15 14:37:34,600 [cuckoo.core.plugins] DEBUG: Analysis matched signature: antivm_memory_available
2018-04-15 14:37:34,601 [cuckoo.core.plugins] DEBUG: Analysis matched signature: antivm_network_adapters
2018-04-15 14:37:34,601 [cuckoo.core.plugins] DEBUG: Analysis matched signature: packer_entropy
2018-04-15 14:37:34,601 [cuckoo.core.plugins] DEBUG: Analysis matched signature: memdump_urls
2018-04-15 14:37:34,601 [cuckoo.core.plugins] DEBUG: Analysis matched signature: raises_exception
2018-04-15 14:37:34,602 [cuckoo.core.plugins] DEBUG: Analysis matched signature: volatility_handles_1
2018-04-15 14:37:34,602 [cuckoo.core.plugins] DEBUG: Analysis matched signature: volatility_ldrmodules_1
2018-04-15 14:37:34,602 [cuckoo.core.plugins] DEBUG: Analysis matched signature: volatility_malfind_2
2018-04-15 14:37:34,603 [cuckoo.core.plugins] DEBUG: Analysis matched signature: volatility_svcscan_1
2018-04-15 14:37:34,603 [cuckoo.core.plugins] DEBUG: Analysis matched signature: volatility_svcscan_3
2018-04-15 14:37:42,543 [cuckoo.core.plugins] DEBUG: Executed reporting module "JsonDump"
2018-04-15 14:37:42,611 [cuckoo.core.plugins] WARNING: The reporting module "SingleFile" returned the following error: The weasyprint library hasn't been installed on your Operating System and as such we
can't generate a PDF report for you. You can install 'weasyprint' manually by running 'pip install weasyprint' or by compiling and installing package yourself.
2018-04-15 14:37:46,888 [cuckoo.core.plugins] DEBUG: Executed reporting module "MongoDB"
2018-04-15 14:37:46,889 [cuckoo.core.scheduler] INFO: Task #2: reports generation completed
2018-04-15 14:37:46,910 [cuckoo.core.scheduler] INFO: Task #2: analysis procedure completed
```

Terminal after finishing file analysis.



Summary

f6077e07-369c-11e8-85d9-80e65024849a.file

File *f6077e07-369c-11e8-85d9-80e65024849a.file*

Summary

Size	484.0KB
Type	PE32 executable (GUI) Intel 80386 Mono/.Net assembly, for MS Windows
MD5	dc4200ac514006f084ead7f03b84c928
SHA1	52e8f04d6b495d238f1a49283a10e2acc053123b
SHA256	a850de0705c0f6095910aa1d5ed0e73a49581aa7427fcfaf2ff5144e93b047c1
SHA512	Show SHA512
CRC32	E7446936
ssdeep	None
Yara	None matched

Information on Execution

Analysis

Category	Started	Completed	Duration	Logs
FILE	April 15, 2018, 2:26 p.m.	April 15, 2018, 2:33 p.m.	385 seconds	Show Analyzer Log Show Cuckoo Log

Machine

Name	Label	Started On	Shutdown On
Windows7x64	Windows7x64	2018-04-15 14:26:55	2018-04-15 14:33:20

Signatures

Score

This file is **very suspicious**, with a score of **10.6 out of 10!**

Please notice: The scoring system is currently still in development and should be considered an **alpha** feature.

Feedback

Expecting different results? Send us this analysis and we will inspect it. Click here

Summary of the analysis.

Conclusion and Recommendations

Findings

- Cuckoo Sandbox successfully executed and analyzed malware in a secure, controlled VM.
- Volatility enabled deep RAM inspection.
- tcpdump effectively captured malicious traffic.
- The sandbox provided detailed logs including system changes, API interactions, and memory snapshots.

Recommendations

- Integrate lightweight AI models to classify malware dynamically using behavioral features.
- Use private isolated networks with internet simulation for better threat replication.
- Set up automated YARA rule updates and regular malware signature feeds.

Challenges Faced

- Faced multiple broken repository and 404 errors while installing tools; resolved using manual .deb downloads and alternate mirrors.
- Required nested virtualization setup on host system which was unsupported by default BIOS settings.
- Encountered compatibility issues with M2Crypto; mitigated by switching to pyOpenSSL where possible.