NCL Cuckoo Installation and Usage Guide

7 November 2019

Prepared by:

Daryl Quek and Sean Pea

Contents

1. Introduction

This user guide provides instructions to start and configure a sandbox environment using Cuckoo and Vagrant. Before starting, ensure that Internet access has been granted for the node to be used as the host.

Please follow the instructions here closely as some of them are crucial steps that are not provided in the official Cuckoo documentation.

It is highly recommended that you use Ubuntu1604-64-GUI as the operating system for your node.

An automated installation script *cuckoo_install.sh* is provided that will give you a basic working setup. To use this:

```
user@node:~$ sudo ./cuckoo_install.sh <all|ubuntu|windows7>
```

You can then proceed to *Section 14* to begin using Cuckoo.

2. Installing VirtualBox

VirtualBox is used as the virtualization platform for the Cuckoo analysis machines.

```
user@node:~$ sudo apt-get -y install virtualbox virtualbox-ext-
pack
```

3. Installing Vagrant

Vagrant is used to provision the Cuckoo analysis machines.

Vagrant can be installed by downloading and installing the relevant installation file from the Vagrant website. The Vagrant *disksize* plugin is also needed to configure the disk space allocated to the Vagrant box later.

```
user@node:~$ wget
https://releases.hashicorp.com/vagrant/2.2.6/vagrant_2.2.6_x86_64
.deb
user@node:~$ sudo dpkg -i vagrant_2.2.6_x86_64.deb
user@node:~$ sudo apt-get install -f
user@node:~$ vagrant plugin install vagrant-disksize
```

4. Vagrant Boxes

Next, download the Vagrant box to be used as your analysis machine for Cuckoo. A repository of images can be found at https://app.vagrantup.com/boxes/search. This example below downloads an Ubuntu 18.04 x86_64 box and a Windows 7 x86 box:

```
user@node:~$ vagrant box add <name or address of image>
user@node:~$ vagrant box add ubuntu/bionic64
user@node:~$ vagrant box add datacastle/windows7
```

5. Creating a Vagrant Project

A new Vagrant project can be created by creating a directory and initiating a box. The following example creates a Vagrant project by initializing the Ubuntu 18.04 box we downloaded previously:

```
user@node:~$ cd /users/<user>
user@node:~$ mkdir cuckoo_ubuntu
user@node:~$ cd cuckoo_ubuntu
user@node:~$ vagrant init ubuntu/bionic64
```

The steps are the same for the Windows box:

```
user@node:~$ cd /users/<user>
user@node:~$ mkdir cuckoo_windows7
user@node:~$ cd cuckoo_windows
user@node:~$ vagrant init datacastle/windows7
```

A Vagrantfile with default parameters will be created in each of the project's directory.

6. Configuring Vagrant Boxes

Vagrant boxes can be configured by modifying the *Vagrantfile* in the Vagrant project directory.

The bottom *Vagrantfile* has been configured to use the Ubuntu 18.04 box initialized above. There are two network interfaces on the system: a NAT and a host-only interface with a static IP of *192.168.101.10*. The host-only interface is necessary for Cuckoo to communicate with the Vagrant box. Port 8080 on the host VM is forwarded to port 80 on the guest Vagrant box for public access. 20GB disk space is allocated and RAM allocated to the box is 2048MB (minimal recommended). The associated lines in the configuration file are shown below.

```
Vagrant.configure("2") do |config|
      config.vm.box = "ubuntu/bionic64"
      config.disksize.size = "20GB"
      config.vm.network "forwarded_port", guest: 80, host:8080
      config.vm.network "private_network", ip: "192.168.101.10"
      config.vm.provider "virtualbox" do |vb|
            vb.name = "cuckoo_ubuntu"
            vb.memory = "2048"
      end
end
```

Similarly, the bottom *Vagrantfile* has been configured to use the Windows 7 box initialized above. There are also two network interfaces on the system: a NAT and a host-only interface with a static IP of *192.168.101.20*.

```
Vagrant.configure("2") do |config|
      config.vm.box = "datacastle/windows7"
      config.vm.network "private_network", ip: "192.168.101.20"
      config.vm.provider "virtualbox" do |vb|
            vb.name = "cuckoo_windows7"
            vb.memory = "2048"
      end
end
```

The Vagrant box can also be provisioned by modifying the Vagrantfile. Provisioning uses shell scripts by default, but provisioners such as Puppet, Chef, Ansible, Salt, and Docker can also be used.

7. Using a Vagrant Box

Once a Vagrant box has been configured, it can be booted up and accessed via SSH:

```
user@node:~$ cd /users/<user>/cuckoo_ubuntu
user@node:~$ vagrant up
user@node:~$ vagrant ssh
vagrant@ubuntu-bionic:~$ whoami
vagrant
```

To logoff and shutdown the Vagrant box:

```
vagrant@ubuntu-bionic:~$ logout
user@node:~$ vagrant halt
```

Files can be transferred between the host and guest using the synced directory. This is the Vagrant box directory on the host and */vagrant* on the guest.

The Vagrant box can also be accessed manually using the credentials *vagrant:vagrant*.

```
user@node:~$ cd /users/<user>/cuckoo_ubuntu
user@node:~$ touch example_file
user@node:~$ vagrant ssh
vagrant@ubuntu-bionic:~$ ls /vagrant
example_file  Vagrantfile
```

8. Installing Cuckoo

Some dependencies will need to be installed before installing Cuckoo. Please follow the instructions at https://cuckoo.readthedocs.io/en/latest/installation/host/requirements/.

It is strongly recommended to install Cuckoo in a virtualenv, and the remainder of this guide is tailored for that. To create a virtualenv:

```
user@node:~$ cd /users/<user>
user@node:~$ virtualenv cuckoo env
```

To install Cuckoo, follow the instructions at https://cuckoo.readthedocs.io/en/latest/installation/host/installation/.

9. (Optional) Setting Up PostgreSQL DBMS

This is recommended if you will be using more than a couple of Cuckoo analysis machines and multiple Cuckoo processing instances.

```
user@node:~$ cd /users/<user>
user@node:~$ . cuckoo_env/bin/activate
(cuckoo_env) user@node:~$ pip install psycopg2
(cuckoo_env) user@node:~$ deactivate
user@node:~$ sudo -u postgres psql
psql (9.5.13)
Type "help" for help.

postgres=# CREATE DATABASE cuckoo;
CREATE DATABASE
postgres=# CREATE USER cuckoo WITH ENCRYPTED PASSWORD 'password';
CREATE ROLE
postgres=# GRANT ALL PRIVILEGES ON DATABASE cuckoo TO cuckoo;
GRANT
postgres=# \q
```

10. Cuckoo Configuration

Cuckoo has to be run once to create the Cuckoo project directory at */users/<user>/.cuckoo*:

```
user@node:~$ cd /users/<user>
user@node:~$ . cuckoo_env/bin/activate
(cuckoo_env) user@node:~$ cuckoo -d
(cuckoo_env) user@node:~$ deactivate
user@node:~$ ls .cuckoo
agent     conf          elasticsearch  log      pidfiles
storage  supervisord       web         yara
analyzer  distributed    init   .py    monitor  signatures  stuff
```

The Cuckoo project directory will now be referred to as the *$CWD*.Cuckoo has to be configured to use a chosen virtualization platform. For simplicity, this guide uses VirtualBox. Cuckoo can also be configured to use VMware Workstation. Modify */users/<user>/.cuckoo/cuckoo.conf* such that:

```
...
ignore_vulnerabilities = yes       // If running virtualbox-5.1
...
machinery= virtualbox
...
[resultserver]
ip = 192.168.101.1   // Or the IP of your vboxnet0 interface
...
```

Modifying the *resultserver* IP address above ensures that the Cuckoo host is able to receive analysis results from the Cuckoo analysis machines. *ignore_vulnerabilities* also has to be toggled when running older versions of Virtualbox so that Cuckoo will not terminate.

11. Cuckoo Malware Signatures

Cuckoo malware signatures can be downloaded and updated from the Cuckoo Community:

```
user@node:~$ cd /users/<user>
user@node:~$ . cuckoo_env/bin/users
(cuckoo_env) user@node:~$ cd /users/<user>/.cuckoo
(cuckoo_env) user@node:~$ cuckoo community
(cuckoo_env) user@node:~$ deactivate
```

12. Prepare Linux Vagrant Box for Cuckoo

Some dependencies have to be installed on the Vagrant boxes to enable Cuckoo to use them. Please follow the instructions at:

Linux – https://cuckoo.readthedocs.io/en/latest/installation/guest/linux/
Windows – https://cuckoo.readthedocs.io/en/latest/installation/guest/creation/

The Linux documentation above is lacking some steps which are covered in the Windows documentation. Namely, */users/user/.cuckoo/agent/agent.py* has to be transferred from the Cuckoo host to the guest VM and set up to run on boot (only instructions for the latter is given in the Cuckoo documentation). Ensure that python is installed and that the agent is given executable permissions too (*chmod +x agent.py*).

```
user@node:~$ cd /users/<user>/cuckoo_ubuntu
user@node:~$ cp ../.cuckoo/agent/agent.py .
user@node:~$ vagrant up
user@node:~$ vagrant ssh
vagrant@ubuntu-bionic:~$ cp /vagrant/agent.py .
vagrant@ubuntu-bionic:~$ chmod +x agent.py
vagrant@ubuntu-bionic:~$ sudo crontab -e
```

After configuring the analysis machine, reboot it and verify that *agent.py* is running:

```
vagrant@ubuntu-bionic:~$ logout
user@node:~$ vagrant halt
user@node:~$ vagrant up
user@node:~$ vagrant ssh
vagrant@ubuntu-bionic:~$ ps -ef | grep agent
root      1048   974  0 16:36 ?        00:00:00 /bin/sh -c python
/home/vagrant/agent.py
root      1050  1048  0 16:36 ?        00:00:00 python
/home/vagrant/agent.py
vagrant   1744  1729  0 16:43 pts/0    00:00:00 grep --color=auto
agent
```

If *agent.py* is running fine, logout and take a snapshot of the running analysis machine:

```
vagrant@ubuntu-bionic:~$ logout
user@node:~$ vboxmanage snapshot cuckoo_ubuntu take
cuckoo_snapshot --pause
user@node:~$ vboxmanage controlvm cuckoo_ubuntu poweroff
user@node:~$ vboxmanage snapshot cuckoo_ubuntu restorecurrent
```

13. Adding Cuckoo Analysis Machines

Strangely, Cuckoo does not work well with the default preconfigured analysis machines, even after they have been configured manually. It is best to remove the preconfigured analysis machine and add in the Vagrant box that was initialized previously.

```
user@node:~$ cd /users/<user>
user@node:~$ . cuckoo_env/bin/activate
(cuckoo_env) user@node:~$ cuckoo machine --delete cuckoo1
(cuckoo_env) user@node:~$ cuckoo machine --add cuckoo_ubuntu
192.168.101.10 --platform linux --snapshot cuckoo_snapshot
(cuckoo_env) user@node:~$ cuckoo machine --add cuckoo_windows7
192.168.101.20 --platform windows --snapshot cuckoo_snapshot
```

14. Starting Cuckoo Controller

It is recommended to use Cuckoo in a virtual environment to prevent version conflicts as Cuckoo does not use the latest versions of some packages that it depends on. Both a global and a virtual environment installation are provided on the sample VM.

To start Cuckoo in a virtual environment, start the sample virtual environment */users/user/cuckoo_env*. Other Cuckoo instances can be run by pointing the $CWD to the location of

```
user@node:~$ cd /users/<user>
user@node:~$ . cuckoo_env/bin/activate
(cuckoo_env) user@node:~$ cuckoo --cwd ~/.cuckoo
```

the respective instances.

It is also possible to run Cuckoo in the background using *supervisor*:

```
user@node:~$ cd /users/<user>
user@node:~$ . cuckoo_env/bin/activate
(cuckoo_env) user@node:~$ cd /users/<user>/.cuckoo
(cuckoo_env) user@node:~$ sudo
/users/<user>/cuckoo_env/bin/supervisord -c supervisord.conf
```

15. Starting Cuckoo Web Interface

Cuckoo provides a graphical web interface that is more intuitive to use. This can be started with:

```
user@node:~$ cd /users/<user>
user@node:~$ . cuckoo_env/bin/activate
(cuckoo_env) user@node:~$ cuckoo web runserver 0.0.0.0:<PORT>
```

The web interface can be used through VNC on the node using the NCL website. Malware and URL submissions can be made through the web interface, and the subsequent reports can also be viewed there.

Alternatively, you can use your browser on your local machine by forwarding a port on your local machine with the following command:

ssh -L 10101:n0.MalwareAnalysis.NCLSecurity.ncl.sg:10101 spea5257@users.ncl.sg

16. Stopping Cuckoo

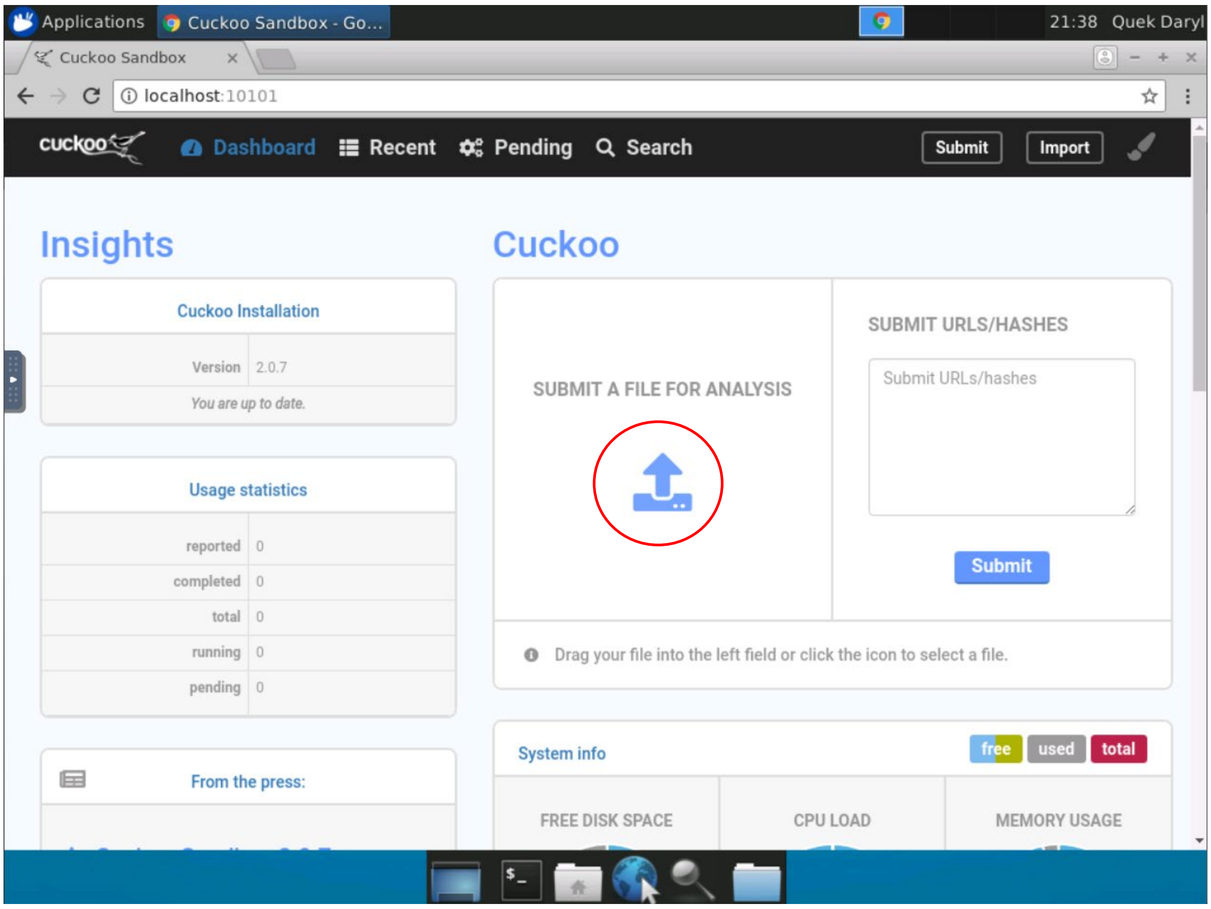
A running Cuckoo instance can be terminated by simply issuing a *CTRL-C* command.

If using *supervisord,* Cuckoo can be stopped by:

```
$ cd /users/<user>
$ . cuckoo_env/bin/activate
(cuckoo_env)$ cd /users/<user>/.cuckoo
(cuckoo_env)$ sudo /users/<user>/cuckoo_env/bin/supervisord stop
cuckoo:
```

17. Submitting Malware for Analysis

Malware can be submitted through the command line or the web interface. For ease of use, the web interface is recommended. To do so, access *localhost:<PORT>* on a browser on your experiment node using the VNC user interface. You will reach the Cuckoo dashboard as seen in the screenshot below.



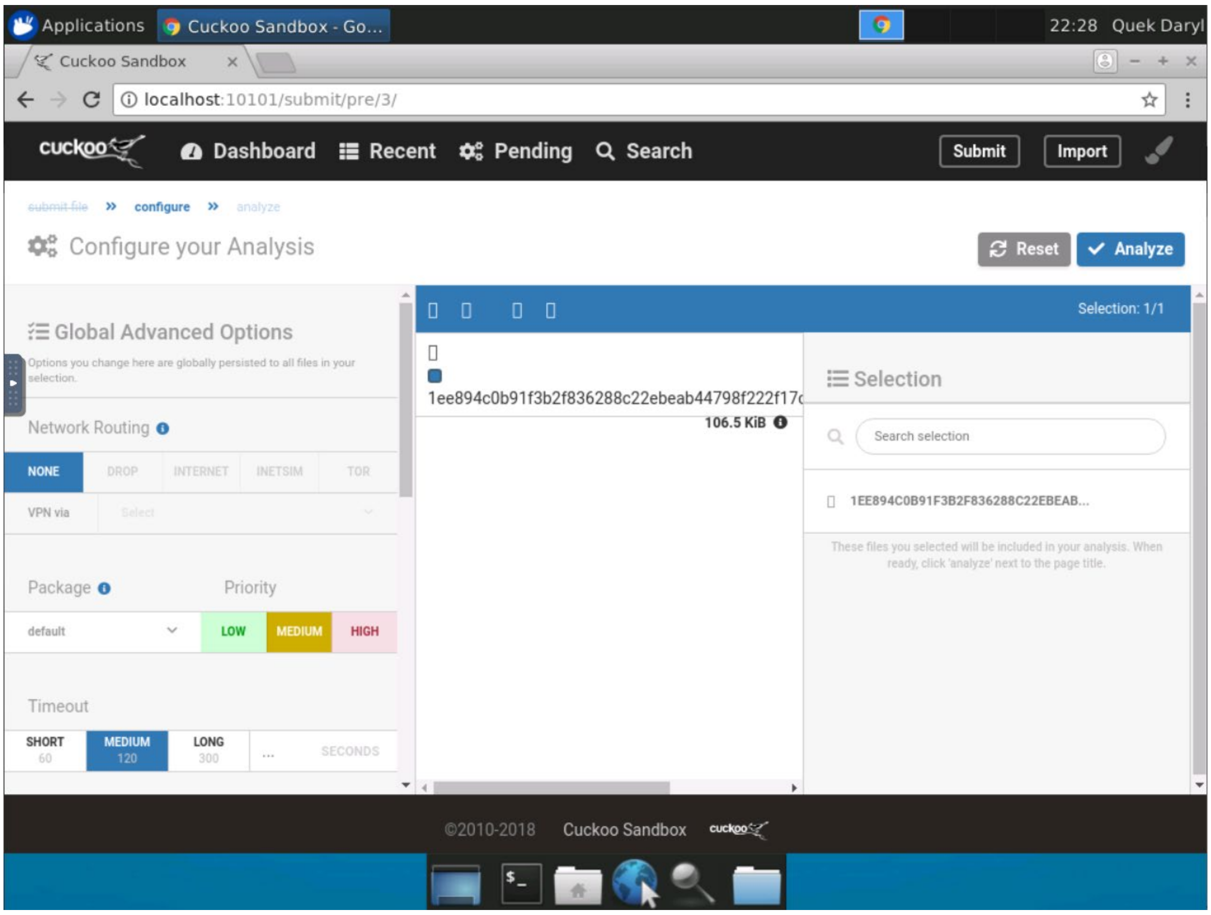
Version	2.0.7

You are up to date.

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.

To submit a file for analysis, click on the button circled in red above. A file browser will pop-up for you to choose your file. Which will bring you to the "Configure your Analysis" page as seen on the next page.

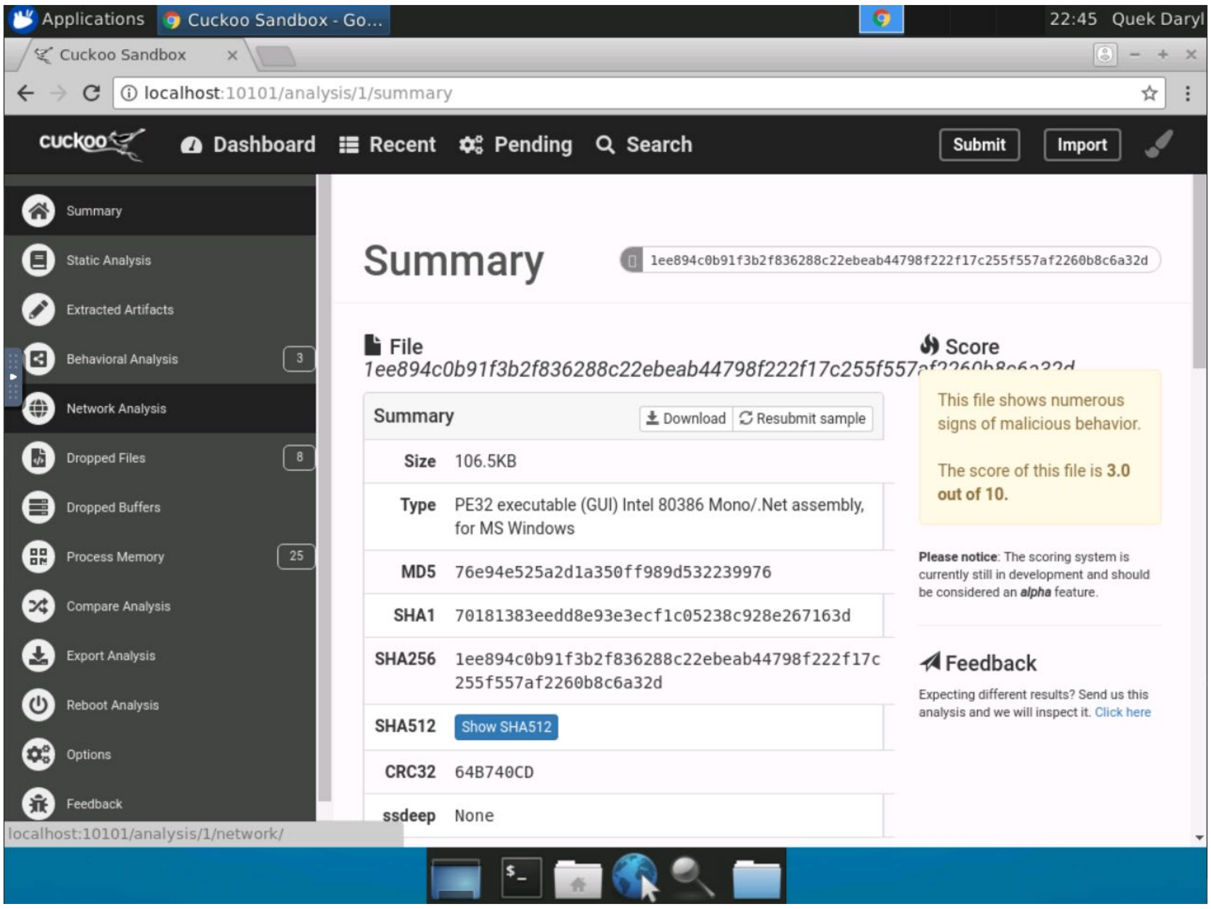

Applications
Cuckoo Sandbox - Go...
22:28 Quek Daryl
Cuckoo Sandbox
localhost:10101/submit/pre/3/
Dashboard
Recent
Pending
Search
Submit
Import
~~submit file~~ » configure » analyze
Configure your Analysis
Reset
Analyze
Global Advanced Options
Options you change here are globally persisted to all files in your selection.
Network Routing
NONE
DROP
INTERNET
INETSIM
TOR
VPN via
Select
Package
Priority
default
LOW
MEDIUM
HIGH
Timeout
SHORT 60
MEDIUM 120
LONG 300
SECONDS
Selection: 1/1
1ee894c0b91f3b2f836288c22ebeab44798f222f17c
106.5 KiB
Selection
Search selection
1EE894C0B91F3B2F836288C22EBEAB...
These files you selected will be included in your analysis. When ready, click 'analyze' next to the page title.
©2010-2018 Cuckoo Sandbox

On this page, configure the different parameters for the analysis. Some key fields are:

- Package: Select the file type of the file to determine it will be executed
- Remote Control: This enables RDP/VNC/SSH to remotely control the VM
- Enabled Injection
- Process Memory Dump
- Full Memory Dump: Dumps the entire VM memory
- Machine: Select the target machine to run the analysis on

These configurations can be made globally on the left pane, or for individual files on the right pane.

Once configuration is complete, click the “Analyze” button and wait a few minutes for analysis to be completed.



Summary 1ee894c0b91f3b2f836288c22ebeab44798f222f17c255f557af2260b8c6a32d

File
1ee894c0b91f3b2f836288c22ebeab44798f222f17c255f557af2260b8c6a32d

Summary — Download — Resubmit sample

Size	106.5KB
Type	PE32 executable (GUI) Intel 80386 Mono/.Net assembly, for MS Windows
MD5	76e94e525a2d1a350ff989d532239976
SHA1	70181383eedd8e93e3ecf1c05238c928e267163d
SHA256	1ee894c0b91f3b2f836288c22ebeab44798f222f17c255f557af2260b8c6a32d
SHA512	Show SHA512
CRC32	64B740CD
ssdeep	None

Score

This file shows numerous signs of malicious behavior.

The score of this file is **3.0 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

One analysis is completed, you can view the generated report. Depending on the file submitted and analysis configuration, various detailed information can be accessed by expanding the left pane.

18. Clear All Previous Records

To clear all the previous analysis results and binaries, run the following:

```
$ cd /users/<user>
$ . cuckoo_env /bin/activate
(cuckoo_env)$ cuckoo clear
```