
PROFILER User Manual

Profiler
Beta Version

Contents

| | |
|--|-----------|
| CONTENTS..... | II |
| 1 INTRODUCTION | 3 |
| 2 INSTALLATION..... | 3 |
| 2.1 GENERAL REQUIREMENTS..... | 3 |
| 2.2 INSTALLING PROFILER AS A CUCKOO ADD-ON..... | 7 |
| 2.3 INSTALLING PROFILER EXTERNALLY..... | 9 |
| 3 USAGE..... | 10 |
| 3.1 USING PROFILER AS A CUCKOO REPORTING MODULE..... | 10 |
| 3.2 USING PROFILER EXTERNALLY..... | 11 |

1 Introduction

Profiler was designed and developed as part of a forensics methodology which involves executing the same malware specimen multiple times in differently configured systems, in order to observe and record possible diverse behavioral patterns that most modern contextually dependent malware instances exhibit. Profiler's current implementation is meant to extend the functionality of the Cuckoo sandbox malware analysis tool in order to automate the process of correlating and investigating multiple analyses results of the same malware sample. Profiler is able to automatically detect differences and similarities in the malware's activities among its multiple executions, thus identifying any contingent behavioral changes. Profiler's automated processing and reporting features allows analysts to quickly assess malware conduct and provides insights on how the malware operated within an explicit organizational infrastructure, minimizing manual effort and time consuming analysis procedures.

2 Installation

Profiler is designed to run on Linux hosts. Its current implementation was developed and tested on a dedicated physical machine carrying Ubuntu 12.04 LTS Linux distribution as the underlying host operating system, with a running installation of Cuckoo version 0.4. Cuckoo sandbox serves as the main automated malware analysis tool to produce the necessary primary data for analysis.

2.1 General Requirements

Before proceeding on installing and using Profiler, some required software and libraries need first to be installed.

a) Profiler's current implementation utilizes the MongoDB feature that was introduced by Cuckoo in version 0.4, in order to take advantage of its storage, querying and pro-

cessing capabilities. The collection of all the necessary primary analysis results that Profiler processes is performed through the MongoDB database collections. To this end, a working installation of Cuckoo sandbox version 0.4 along with all its required software and libraries needs to be present.

Even though Profiler has been developed and tested using Cuckoo v0.4 it is also expected to work with newer versions of Cuckoo as long as the MongoDB feature is available and the structure of cuckoo's Global Container has not changed.

For information on how to install and use Cuckoo sandbox please refer to Cuckoo's documentation (<http://www.cuckoosandbox.org>).

b) The following Python libraries are required and need to be installed (if they are not already installed) in order for Profiler to generate the HTML reports and to interact (collect and store results) with the MongoDB database.

- **Mako** : for rendering the HTML reports.
- **Pymongo** : for communicating and interacting with the MongoDB database.

They are usually packaged in GNU/Linux Ubuntu and can be installed using the following command:

```
$ sudo apt-get install python-mako python-pymongo
```

c) Since Profiler processes the analysis results stored in the MongoDB collections, cuckoo needs to be configured in order to also store its results in the MongoDB database. To this end cuckoo's MongoDB reporting module needs to be enabled.

Edit cuckoo's "*reporting.conf*" configuration file (it is usually located within cuckoo's installation folder in: *cuckoo/conf/reporting.conf*) and add/change the entry for the [mongodb] reporting module setting the flag "*enabled*" to "*on*" as shown in the highlighted section in Figure 1.

```
# Enable or disable the available reporting modules [on/off].
# If you add a custom reporting module to your Cuckoo setup,
you have to add
# a dedicated entry in this file, or it won't be executed.
# You can also add additional options under the section of
your module and
# they will be available in your Python class.

[jsondump]
enabled = on

[reporhtml]
enabled = on

[pickled]
enabled = on

[metadata]
enabled = on

[maec11]
enabled = on

[mongodb]
enabled = on
```

Figure 1: Editing the file reporting.conf to enable cuckoo's mongodb reporting module.

NOTE: this configuration is imperative and a prerequisite for Profiler to run. If the mongodb module is not enabled Profiler will not find any results to process.

d) The following python scientific computing packages are required and need to be installed in order to enable the automated charts generation feature that Profiler provides.

- [Numpy](http://www.scipy.org) and [Scipy](http://www.scipy.org) scientific computing tools (<http://www.scipy.org>).
- [Matplotlib](http://matplotlib.org) plotting library (<http://matplotlib.org>).

On Ubuntu Linux the necessary packages can be installed from repositories using the following command:

```
$ sudo apt-get install python-numpy python-scipy python-matplotlib
```

For more information on how to install these packages or any additional software that they might require please refer to their official websites.

e) One of Profiler's features is its ability to identify and trace a specific behavior back to the original analysis of the sample and the related raw and detailed cuckoo analysis results. However cuckoo currently does not relate the MongoDB's records with the respective SQL ID of each analysis it performs. Thus it is not possible to connect the mongodb's data with the Cuckoo's detailed analysis results that are stored in separate directories for each different analysis request. To enable this feature, the "info.py" processing module of cuckoo needs to be customized so as to additionally store the specific analysis path of the produced results for each analysis request.

Edit the file "info.py" (it is usually located within cuckoo's installation folder in: *cuckoo/modules/processing/info.py*) and add the following python commands:

```
analysis_path = self.analysis_path
```

```
"analysis_path": analysis_path
```

in lines 23 and 30 respectively of the original "info.py" file as shown in the highlighted sections in Figure 2.

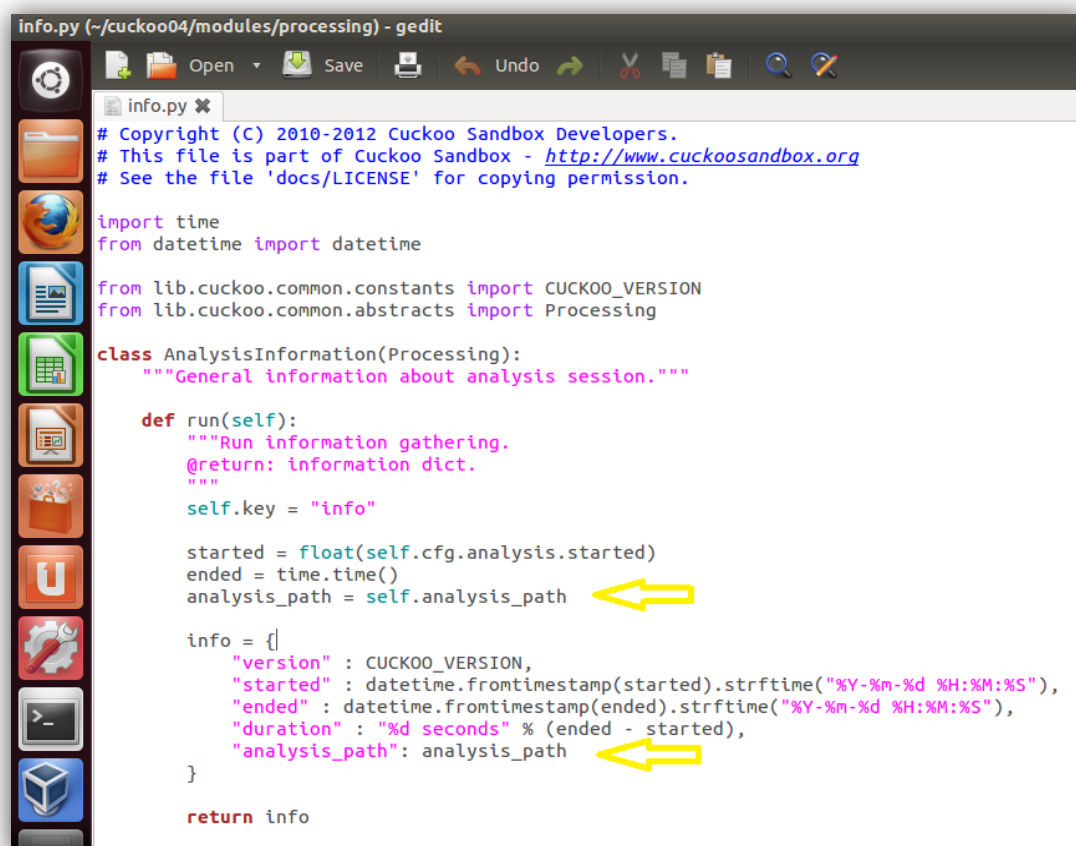


Figure 2: Customizing cuckoo's "info.py" file.

NOTE: If you have Cuckoo running during the above modifications, you will need to shut it down and then restart it in order for the changes to be incorporated within cuckoo.

2.2 Installing Profiler as a Cuckoo Add-on

After the proper installation of all the required components from section 2.1, Profiler can be now installed within cuckoo sandbox to be used as an additional reporting module.

Perform the following steps:

1. Extract the contents of the profiler-beta.zip file to your desired location inside your Linux host.
2. Copy the directory “*data-profiler*” inside cuckoo’s root folder (Figure 3a).
3. Copy the file “*profiler.py*” inside cuckoo’s reporting modules directory (it is usually located in: *cuckoo/modules/reporting/*). The “*reporting*” directory of cuckoo will look something like the one shown in Figure 3b.

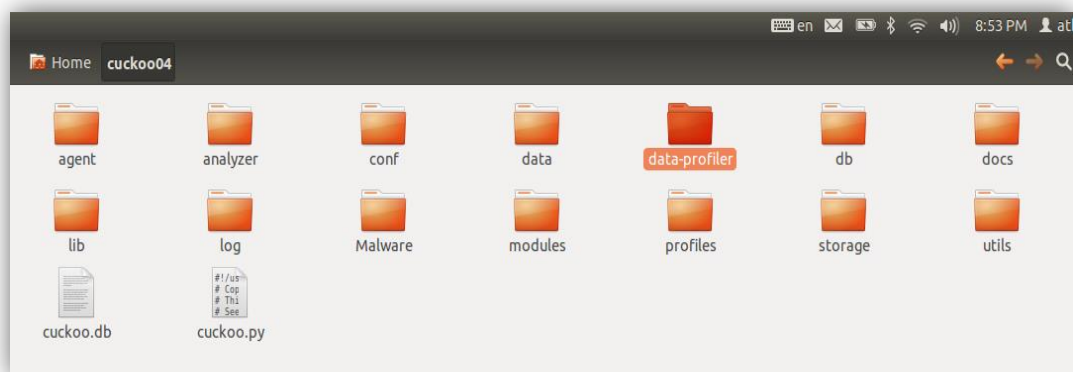


Figure 3a: Copying the directory “*data-profiler*” inside cuckoo’s root folder.

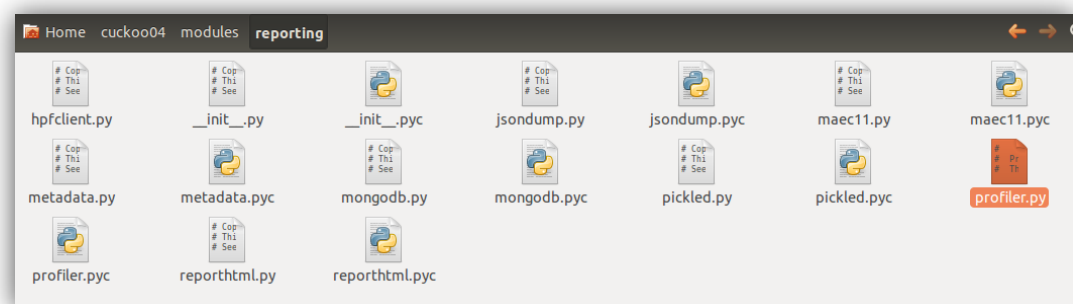


Figure 3b: Copying the file “*profiler.py*” inside cuckoo’s reporting directory.

4. Edit cuckoo's "*reporting.conf*" configuration file (it is usually located within cuckoo's installation folder in: *cuckoo/conf/reporting.conf*) and add the entry for the [profiler] reporting module setting the flag "*enabled*" to "*on*" as shown in the highlighted section in Figure 4.

```
# a dedicated entry in this file, or it won't be executed.
# You can also add additional options under the section of
# your module and
# they will be available in your Python class.

[jsondump]
enabled = on

[reporthtml]
enabled = on

[pickled]
enabled = on

[metadata]
enabled = on

[maec11]
enabled = on

[mongodb]
enabled = on

[profiler]
enabled = on
```

Figure 4: Editing the file *reporting.conf* to add Profiler as a reporting module.

NOTE: The Profiler module must be added after the [mongodb] entry. The order of the entries in this file is important. Cuckoo must first execute the mongodb reporting module in order to store the results in the MongoDB database and then execute the profiler reporting module, which in turn will retrieve the results from the MongoDB for further processing.

5. If you have Cuckoo running during the above modifications, you will need to shut it down and then restart it in order for the changes to be incorporated within cuckoo.

At this point Profiler is ready to be used. The next time Cuckoo performs an analysis on a malware sample, Profiler will automatically be executed by Cuckoo.

2.3 Installing Profiler Externally

Profiler can be installed and used as an independent processing tool, without implicating itself with cuckoo's analysis procedures. After the proper installation of all the required components from section 2.1, perform the following steps:

1. Extract the contents of the profiler-beta.zip file to your desired location inside your Linux host.
2. If you had previously used Profiler as a Cuckoo reporting module, you need to edit the `reporting.conf` file of cuckoo in order to disable the `[profiler]` entry (set the `enabled` flag to `off`). This is because we don't want Profiler to be automatically executed by Cuckoo. Please see section 2.2 step 4 for more details. Use Cuckoo to analyze malware samples as usual. You can analyze as many malware samples as you want.
3. Copy Cuckoo's SQL database file (usually located in: `cuckoo/db/cuckoo.db`) inside Profiler's running directory (for example: `Profiler-beta/`) as shown in Figure 5. Profiler uses the SQL database in order to retrieve a list of unique md5 hashes that belong to the malware samples that have been analyzed by cuckoo.

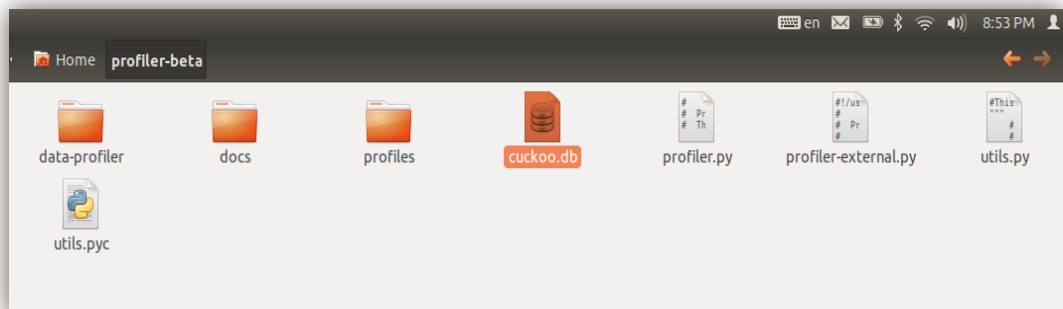


Figure 5: Copying cuckoo's SQL DB inside profiler's directory.

NOTE: You don't need to perform this action each time you run Profiler. A fresh copy of the SQL database needs to be made only if you use cuckoo to analyze new malware samples (because their md5 hashes will not be present in the old copy of the database and Profiler will not search the mongodb for their respective analysis results).

4. In a terminal window use the following command:

```
$ python profiler-external.py
```

Profiler will be executed and will process all the analysis results of all the malware samples that cuckoo has analyzed up to this moment.

3 Usage

Profiler's functionality is twofold. It can be used either as a Cuckoo add-on incorporated as a reporting module within cuckoo's installation, or as an autonomous processing tool independent of cuckoo's analysis procedures.

3.1 Using Profiler as a Cuckoo Reporting Module

Each time Cuckoo performs an analysis on a given sample, a predefined set of reporting modules are called upon, to produce various types of reports. By including profiler as a reporting module, Cuckoo automatically executes Profiler in every malware analysis request.

Each time Cuckoo performs an analysis on a malware sample, Profiler is automatically executed. Upon execution Profiler connects to the MongoDB database and retrieves all the stored analysis results of the malware sample under analysis. Profiler will process and correlate those results to produce the respective reports (.txt, .html, .json) and charts which are stored in a "*profiles*" folder inside cuckoo's running directory. All of the results are stored in separate subfolders named after each malware's md5 hash value (Figure 6). The malware's profiles are automatically created or updated, upon each malware's execution.

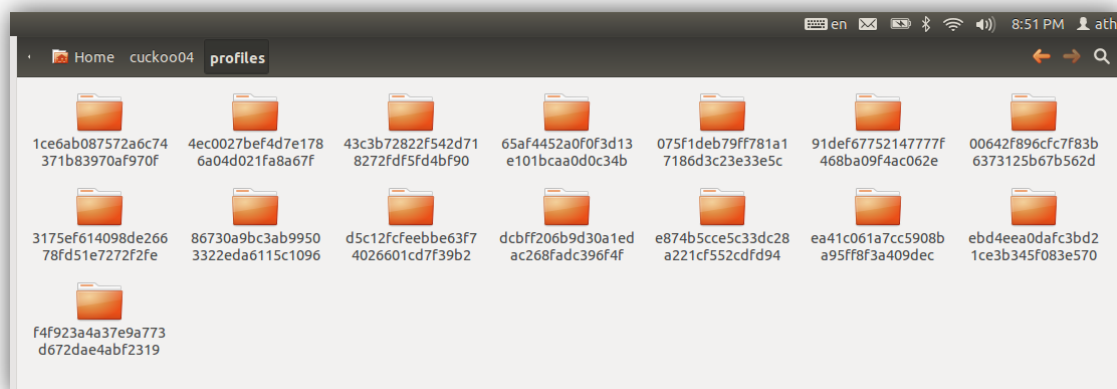


Figure 6: Contents of the “profiles” directory.

In order to understand Profiler’s functionality and usage and take advantage of its features, it would be preferred to analyze each malware sample more than once. If we only execute a sample one time, Profiler’s results will be much similar to cuckoo’s reports since it will have no additional analysis results per sample to correlate and process.

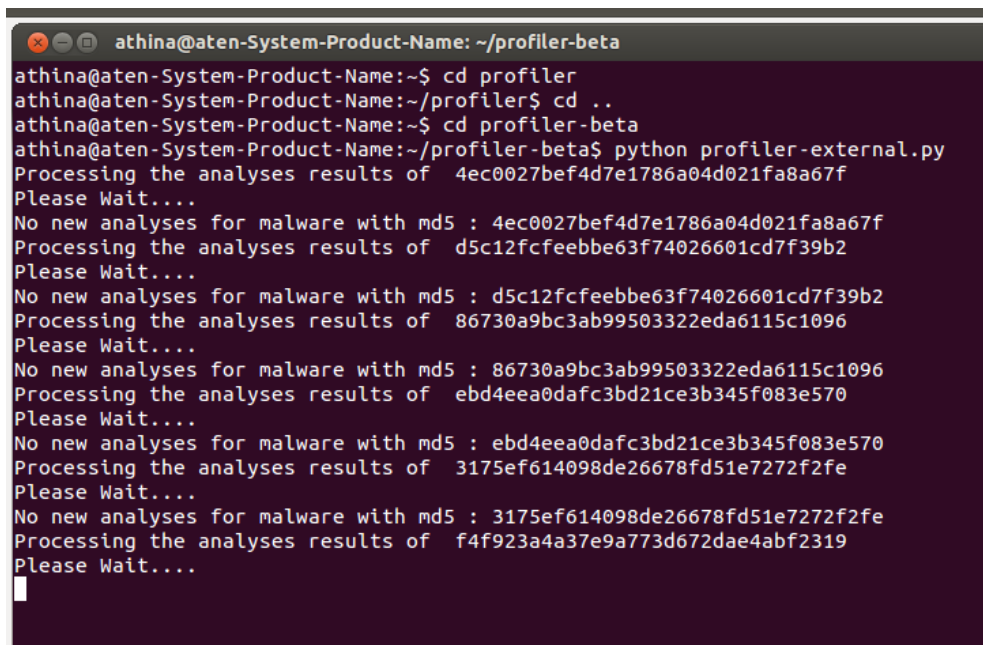
3.2 Using Profiler Externally

Use Cuckoo sandbox to analyze malware samples as usual. Analyze as many malware instances as you desire. For the completeness of Profiler’s reports it would be a good idea to analyze some malware samples multiple times (for example inside different VM’s with different operating systems, patches, applications etc.).

You can run Profiler at any point in between Cuckoo’s analysis processes. In a terminal window move to Profiler’s installation directory and type the following command:

```
$ python profiler-external.py
```

Upon execution Profiler will first connect to Cuckoo's SQL database and retrieve a list of unique md5 hashes that belong to the malware samples that cuckoo has analyzed. It will then connect to the MongoDB database and process all the stored analysis results per malware sample (Figure 7).

A terminal window with a dark background and light text. The window title is 'athina@aten-System-Product-Name: ~/profiler-beta'. The user 'athina' is at the prompt. The commands and output are as follows:

```
athina@aten-System-Product-Name:~$ cd profiler
athina@aten-System-Product-Name:~/profiler$ cd ..
athina@aten-System-Product-Name:~$ cd profiler-beta
athina@aten-System-Product-Name:~/profiler-beta$ python profiler-external.py
Processing the analyses results of 4ec0027bef4d7e1786a04d021fa8a67f
Please Wait....
No new analyses for malware with md5 : 4ec0027bef4d7e1786a04d021fa8a67f
Processing the analyses results of d5c12fcfeebbe63f74026601cd7f39b2
Please Wait....
No new analyses for malware with md5 : d5c12fcfeebbe63f74026601cd7f39b2
Processing the analyses results of 86730a9bc3ab99503322eda6115c1096
Please Wait....
No new analyses for malware with md5 : 86730a9bc3ab99503322eda6115c1096
Processing the analyses results of ebd4eea0dafc3bd21ce3b345f083e570
Please Wait....
No new analyses for malware with md5 : ebd4eea0dafc3bd21ce3b345f083e570
Processing the analyses results of 3175ef614098de26678fd51e7272f2fe
Please Wait....
No new analyses for malware with md5 : 3175ef614098de26678fd51e7272f2fe
Processing the analyses results of f4f923a4a37e9a773d672dae4abf2319
Please Wait....
```

Figure 7: Executing “*profiler-external.py*”

Profiler's total execution time depends on the volume of analysis results that it has to process. If you have a large database of malware samples and you want to analyze each sample multiple times it would be wise to run Profiler more often in between Cuckoo's analyses procedures.

Upon completion Profiler will create a “*profiles*” folder, located inside its running directory, which contains all of Profiler's results for all the different malware samples that it processed. The results are stored in separate subfolders named after each malware's md5 hash value. The “*profiles*” directory is used in order to store the .txt, .html, .json reports that Profiler creates along with the generated charts that represent the dropped files and the API calls that each malware invoked during its executions. Profiler's results are additionally stored in a “*profiles*” collection inside the MongoDB database.

The profiles of the malware samples can be created and updated at any point, irrelevant of how often Profiler is executed in between the analysis processes of Cuckoo. Every time Profiler is executed, old malware profiles are updated, upon the recognition of ad-dable analysis results, and new profiles are created for each additional malware sample.