Creating a Distributed Malware Analysis Toolchain with MASS

Fabian Rump
University of Bonn
rumpf@cs.uni-bonn.de

Timm Behner
Fraunhofer FKIE
timm.behner@fkie.fraunhofer.de

***Abstract*—In this demonstration we present the Malware Analysis and Storage System (MASS), a novel framework for malware analysis. MASS is designed as a distributed and scalable system and aims to empower cooperation between malware researchers. We will show the central aspects of the framework and explain the malware analysis process flow. An attendee can see how to submit and query malware samples from the MASS server and further how to use its REST API and the MASS Python client to incorporate analysis methods into the MASS infrastructure.**

I. Introduction and Motivation

As the everyday use of connected devices such as computers or mobile phones increases, the threat posed by malicious software (malware) is contiually growing [11]. At the same time, malware grows both in quantity and in variety and new malware campaigns are launched on a day by day basis [11].

To detect and oppose those threats, security researchers create analysis tools which are nowadays often precisely tailored to produce or extract exactly one specific piece of information from a malware sample. However, due to the frequent changes and the limited scope of each analysis tool, the interoperability between these tools is often very limited and many tools are not prepared for a large scale analysis scenario.

The Malware Analysis and Storage System (MASS) is an Open-Source framework specifically designed to collect and combine the results of different analysis tools to yield a more complete picture of malware behavior. Additionally, MASS facilitates typical tasks such as analyzing a high number of malware samples and collecting analysis results for statistical analysis and reporting. MASS aims to support transparent and reproducible evaluation of new malware analysis methods and thereby empower the collaboration between security researchers. The complete source code of MASS is available under the terms of the MIT license [1].

II. Malware Analysis with MASS

Detailed foundations of the MASS infrastructure including a performance analysis of the sample submission and analysis processes are covered by our accompanying paper which has been accepted for the 42nd IEEE LCN conference [9]. Thus for this proposal we will just give a short overview of the MASS infrastructure and about how the malware analysis workflow is organized in order to better describe the scope and significance of the demonstration in the following sections.

The MASS infrastructure is outlined in Figure 1. The central element is the MASS server, which is a Python Flask [7] application with a MongoDB [6] database backend. The MASS server offers an interactive web interface for providing information to human users. External applications connect to the MASS server using a REST API. All analysis functionality is decoupled from the MASS server and integrated into dedicated analysis systems, which also use the REST API for exchange of control information and analysis data. All communication with the MASS server is based on HTTP/HTTPS and thus can even work in environments with restrictive firewalls or proxy servers.

The analysis workflow for each sample is based on the concept of assigning tags to the sample. A schematic visualization of this process is given in Figure 2. Upon submission of a sample the MASS server assigns an initial set of tags to each sample, e.g. based on the MIME-type of a file. Thereafter a matching between the sample tags and the tag filters of the available analysis systems is executed. For each matching analysis system, the MASS server schedules the analysis of the sample on one of the available system instances. The analysis system instance executes the analysis and posts the report to the MASS server. The report may contain new tags for the sample, which may lead to a repetition of this cycle until no more suitable analysis systems are discovered.

III. Novelty and Significance

We propose MASS as a framework for supporting scientific malware analysis. In a scientific environment can help to reduce a number of typical problems:

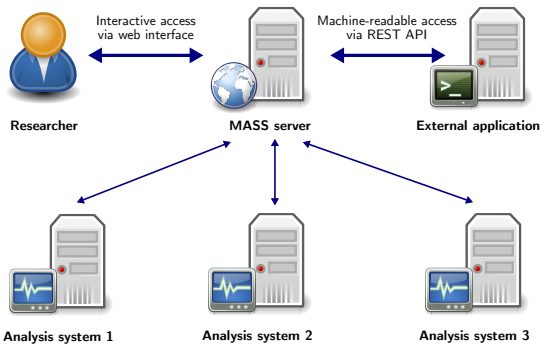


Fig. 1. Conceptual overview of the MASS infrastructure

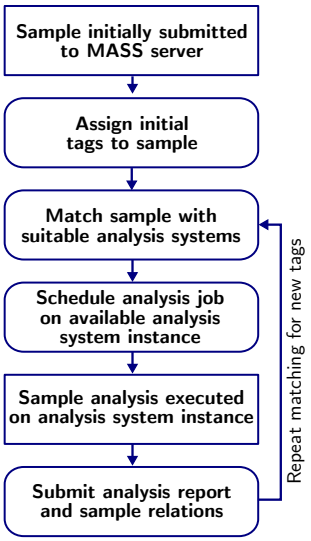


Fig. 2. Overview of the analysis process

First, it is a common academic standard to provide a transparent and reproducible evaluation of the presented work in order to make the results verifiable and comparable to other approaches. In reality this is often not the case when new malware analysis methods are presented. For example the findings of Rossow et al. [8] suggest that many researchers fail to specify the selection of malware samples and the configuration of their analysis systems. With the MASS framework, researchers can easily provide access to all experimental results in order to support their academic work. Other researchers can use the same infrastructure to verify the results and to compare it with own approaches.

Second, the development and testing of a new analysis method introduces some "housekeeping" tasks, e.g. administration of test samples, scheduling and execution of analyses, storing the results of each analysis and providing tools for an automated processing of the results. The needed infrastructure for these tasks is often rewritten for every experiment, yielding a duplication of work and a possible error source. By using MASS researchers can base their analyses on a solid and well-tested foundation and therefore reduce the time effort spent on tasks not related to the actual development of new analysis methods.

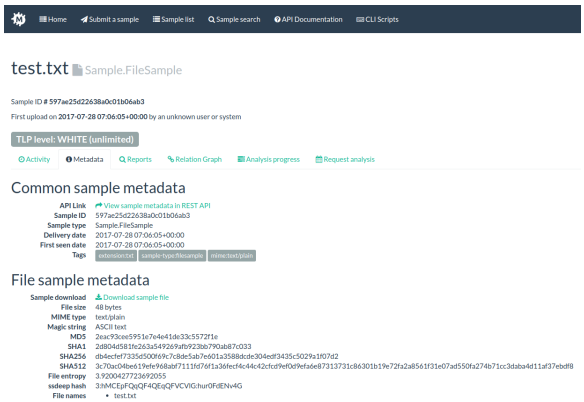
Third, collaboration between different malware researchers is often very limited today. There already exist hosted services such as VirusTotal [10], however these services often enforce certain terms and provide only limited extensibility. In contrast, the MASS infrastructure is self-hosted and leaves the malware researchers in control over their data. In addition, since MASS is an Open-source project, new functionality can be contributed to the framework by the research community as needed.

Finally, due to the distributed system architecture in combination with an access control mechanism based on the Traffic Light Protocol (TLP) [3] MASS can provide a clear benefit for improving the interaction and collaboration between different research organizations in situations where not all the available information can be freely shared between all participating parties due to specific policies. Here MASS can provide a clear advantage compared to other existing malware analysis solutions such as Malice [5], Viper [4] or Polychombr [2].

IV. Scope of the Demonstration

With our demonstration at the 42nd IEEE LCN conference we intend to give an overview of the different elements which form the MASS infrastructure: We will present a running MASS server on one of our laptops. We will explain the typical ways of interaction with the server, namely the interactive web interface and the REST API. Based on the REST API we will show how to remotely interact with the MASS server from the second laptop in a programmatic way, e.g. to query samples or to execute analyses within a Python program. With these examples we will outline how a more complex tool chain of analysis systems can be realized with MASS. Finally we will provide live demonstration of such a toolchain executing analyses on real malware data.

Some examples of our demonstration content are presented in the following. Figure 3a shows the Web interface of the



(a) Web interface

(b) Direct API access

```
from mass_api_client import ConnectionManager
from mass_api_client.resources import FileSample

cm = ConnectionManager()
cm.register_connection('default', 'IjU3YmM2NTZjMjI2MzhhM2QzY2UwNWNhYyI.y6vXgXksA19-
qiLddRCrMou_1Ko', 'http://localhost:5000/api/')

result = FileSample.query(md5sum='2eac93cee5951e7e4e41de33c5572f1e')

for item in result:
  print(item.file_names)
  with item.temporary_file() as f:
    # do some stuff ...
```

(c) Python client

Fig. 3. Demonstration of the different ways to interact with the MASS server

MASS server, in this particular case the detail view of a file sample. This contains metadata of the file, e.g. values of different hash functions. From here a researcher could further see if reports for this file are present or still pending and if the file is in relation with other samples.

The interaction with the MASS server from the command line via curl is shown in Figure 3b. In this case the same metadata seen in Figure 3a is queried and returned as a JSON string.

Figure 3c shows an example for the interaction with the MASS server from a Python program using our Python MASS client. First we register a connection to the MASS server and pass the URL as well as our API key. Then we query if a file is present which has the respective MD5 hash. If such a file exists and thus the query is not empty, we first print all known file names to the standard output and then download the content of the file to a temporary location for further analyses.

To demonstrate a malware analysis toolchain with MASS, we chose a common scenario. Suppose we receive a compressed and password protected ZIP file, containing a set of malware samples, which we would like to analyse. First we submit the archive to MASS. On submission it is recognized as a ZIP archive and thus receives a respective tag. For decompressing the file we connect an analysis system which unpacks archives using a list of well-known passwords. The analysis result of the ZIP archive is one or several files which are again submitted to the MASS server. It is common that malware is obfuscated or packed. Thus the next analysis system could recognize if this is the case for each submitted file. If for example a file was packed with UPX, the analysis system adds a respective tag to the sample. In this case the sample is scheduled to be unpacked by an UPX unpacker with resulting executable binary which is submitted to MASS. From here we could further use a static or dynamic analysis, as well as an external source such as VirusTotal, to gain more information.

V. Technical Requirements

For the demonstration we will prepare and provide two laptops. In addition, we will bring a small switch/router and Ethernet cables to create a local test network for demonstration. All devices need electrical power and have Type C or Type F plugs. If no suitable adapter for this type of power outlet can be provided, we kindly ask the organizers to inform us ahead of the conference so that we can take care of this. In order to present our demonstration in a proper way, two large screens or projectors with HDMI or VGA input are required. A desk to place the equipment is needed as well. Access to the local WiFi is not strictly required, but would be clearly beneficial to show additional MASS project content when needed.

VI. Conclusion and Future Work

In this demonstration proposal we have outlined that malware researchers are in need of a collaborative, interoperable and extendable system in order to transparently develop and evaluate new malware analysis techniques. We proposed MASS as an Open-Source framework for scientific malware analysis and explained the novelty and significance of the framework by defining typical challenges and problems which MASS tries to solve and by comparing MASS to other existing systems for malware analysis. We described the planned contents of the demonstration such as showing the different ways of interaction with the MASS server or demonstrating an exemplary malware analysis toolchain.

With our demonstration we hope to raise awareness about this new system and to encourage other researchers to use MASS and to provide ideas for the future development. In addition we intend to use the contents and gained feedback of this demonstration to improve the online documentation in order to provide exemplary scenarios similar to those shown at the conference venue for other researchers which start using MASS in their research groups.

For future demonstrations we plan to give a detailed insight into some of the more advanced mechanisms of MASS such as the authentication and access control systems or the scheduling service. Additionally, in order to enhance collaboration between different research groups a federation service to automatically synchronize available information of multiple MASS installations is currently in development. We hope to present this federation service in future demonstrations once the implementation is finished.

VII. Acknowledgements

The authors want to thank student assistant Christopher Schmidt, who did a significant part of the MASS Python client implementation and thereby greatly improved the extensibility of the MASS infrastructure.

References

[1] "MASS Open-source project website on Github," https://mass-project.github.io/, accessed: 2017-07-20.
[2] A. Chevalier, S. Le Berre, and T. Pourcelot, "Polichombr Github project," https://github.com/ANSSI-FR/polichombr, accessed: 2017-07-20.
[3] European Union Agency for Network and Information Security (ENISA), "Considerations on the Traffic Light Protocol," https://www.enisa.europa.eu/topics/national-csirt-network/glossary/considerations-on-the-traffic-light-protocol, accessed: 2017-07-20.
[4] C. Guarnieri, "Viper Homepage," http://viper.li/, accessed: 2017-07-20.
[5] J. Maine, "Malice Github project," https://github.com/maliceio/malice, accessed: 2017-07-20.
[6] I. MongoDB, "MongoDB Homepage," https://www.mongodb.com/, accessed: 2017-07-20.
[7] A. Ronacher, "Flask Homepage," http://flask.pocoo.org/, accessed: 2017-07-20.
[8] C. Rossow, C. J. Dietrich, C. Grier, C. Kreibich, V. Paxson, N. Pohlmann, H. Bos, and M. Van Steen, "Prudent practices for designing malware experiments: Status quo and outlook," in *Security and Privacy (SP), 2012 IEEE Symposium on*. IEEE, 2012, pp. 65–79.
[9] F. Rump, T. Behner, and R. Ernst, "Distributed and Collaborative Malware Analysis with MASS," in *2017 IEEE 42nd Conference on Local Computer Networks (LCN) (accepted)*, 2017.
[10] VirusTotal, "Credits and Acknowledgements," https://www.virustotal.com/en/about/credits/, accessed: 2017-07-20.
[11] P. Wood, B. Nahorney, K. Chandrasekar, S. Wallace, and K. Haley, "Symantec Internet Security Threat Report," 2016.