Sharing and replaying attack scenarios with Moirai

Guillaume Brogi*†, Valérie Viet Triem Tong‡
* Akheros and † Conservatoire National des Arts et Métiers, Département IMATH, EA4629, Paris, France
guillaume.brogi@akheros.com
‡EPI CIDRE, CentraleSupelec, Inria, Université de Rennes 1, CNRS, IRISA, UMR 6074, Rennes, France
valerie.viettriemtong@centralesupelec.fr

***Abstract*—Datasets are necessary for evaluating and comparing security solutions. Today, the most well-known public dataset is still the oft-decried IDEVAL dataset. Even if we don't take into account all the inherent shortcomings of this dataset, the fact that it dates back to 1999 means its relevance is all but lost. Without a public dataset, new security solutions cannot be compared to existing ones. In this article, we argue for the need of a public and modern dataset for the evaluation of security solutions. Moreover, we argue that traditional datasets are too restrictive in the approaches they allow. Thus, we present Moirai. Instead of sharing datasets, Moirai shares the scenarios used to create datasets. This allows for the creation of complex scenarios which could, for example, represent an Advanced Persistent Threat (APT). By sharing the scenarios, Moirai allows solutions based on disparate ideas to be compared.**

***Index Terms*—public dataset, simulation, advanced persistent threat, security solution, reproducibility**

I. Introduction

Whatever the research topic, data are paramount in order to validate assumptions and measure performance; information security is no exception. Researcher can generate the data themselves. However, this doesn't scale well, taking valuable time away from the research. The alternative is to use data from a dataset. Datasets can be either private or public and provide the researcher with premade data. In addition, public datasets enable researchers to compare their solutions with prior art using the same dataset. This is particularly important since, often, the results are published but not the solution itself. In this article, we focus on datasets meant for the evaluation of intrusion detection systems (IDS), whether network- or host-based and whether they are meant to protect single hosts or whole networks.

Real-world datasets are difficult to produce because of legal, ethical and technical reasons. First, all sensitive and private information must be either removed or anonymised. Then, every attack must be identified, with as few false positives and negatives as possible. Finally, companies may feel that such datasets showing they have attacks on their systems will tarnish their image. Thus, we focus on generated datasets. The IDEVAL dataset [1], [2] created by the DARPA and the MIT Lincoln Laboratory is probably the most famous public dataset used for evaluating IDS solutions. It is a comprehensive dataset including both host-based and network-based data from a diversity of operating systems. There are both parts of the dataset without attacks and parts containing attacks, with the attacks being clearly labeled. Despite its numerous flaws [3], [4], no other dataset comes close and 18 years later, it is still being used [5], [6].

On top of its original flaws, the advances in computing over the last 20 years mean that the dataset is not representative anymore. In 2006, Qian et al. [7] were already proposing a redesigned dataset based on IDEVAL. A 2014 study by Koch et al. [8] showed that available datasets still do not fulfill all requirements for realistic datasets, the closest one being the PREDICT dataset (recently renamed to IMPACT [9]) which is not a completely public dataset. Thus the need for a current comprehensive dataset.

In this article, we agree that publicly available, current and comprehensive datasets are needed. But more than that, we argue that dataset creation scenarios should be shared too. The rest of this article is organised as follows. In Sec. II, we present other efforts to create a satisfying public dataset. In Sec. III, we present Moirai [10], a tool for sharing and replaying scenarios used for creating datasets. We also show how to transcribe a simple scenario for replay. Finally, we conclude this article in Sec. IV.

II. Related works

In [11], Maxion and Tan show that the performance evaluation of IDS can differ by as much as an order of magnitude depending only on the dataset used for the evaluation. For their test, they use a base dataset to synthesise alternate ones changing only the regularity of the data. This shows that the intrinsic nature of the data can change performance of an IDS. Thus, there is a need for properly created dataset which take into account the variations in nature of the data.

Vasolomanolakis, Cordero et al. have several publications on dataset creation for the evaluation of IDS [12]–[14]. They start from the same observations we do. Namely, IDSs require high-quality public datasets to be evaluated and compared properly. However, these datasets are currently non-existent and the closest we have are either outdated or too sparse. They identify four requirements for new datasets: they must contain labeled attacks; they must contain modern normal and attack data; they must be publicly available or reproducible; and they must be flexible and allow testing different scenarios. They then

propose ID2T, a tool generating labeled datasets from network packet captures, which also makes sure to retain realistic properties. This is really interesting tool and the datasets it produces can be used with any network-based IDS. However, it is limited to network-based IDS and cannot be used to compare host-based IDS.

Małowidzki et al. make the same observations too in [15]. They start by listing datasets used in articles, when they are disclosed. They identify two main shortcomings: stale datasets and unlabeled datasets. They select four criteria as being critical for a good dataset. First and foremost, the dataset must be recent. That dataset should be labeled with sufficient precision and should be rich, capturing different types of malware. And, of course, it must also be correct. Finally, they add a bonus criterion, that of a balanced dataset, including representative samples of both malwares and good data. Analysing existing datasets, they conclude that real correlated datasets collected in production environment are superiors to other kinds of datasets.

From the observations and conclusions of these articles, we can deduce that there is a need for a tool to assemble good, comprehensive datasets. In this article, we go one step further and advocate sharing the scenarios used to create datasets. The technology for doing so, such as virtualisation, exists and there are advantages over sharing the datasets only. The first advantage is that the dataset can be used with any type of IDS. Because anyone can run the scenario while adding their IDS, the IDS collects data from the scenario and every IDS tested on the scenario can be compared. The second advantage is that, by sharing the scenario, anyone can use it as a base to create more complex scenario or simply update it before it becomes obsolete. Little by little, the number of scenarios will increase, thus increasing the quality of testing and comparisons of IDS.

III. Moirai

In light of these findings, we have created Moirai to make sharing and replaying scenarios easier. Moirai leverages existing technologies where possible and only requires one text configuration file in `ini` style to define a scenario. Thus, Moirai is a tool which takes a simple text file as input and orchestrates virtual machines (VM) to play the scenario described in the input file.

For the sharing of VM, Moirai uses Vagrant [16]. Vagrant is originally meant for building portable development environments. It does so using VM, and, being widespread, there is a large library of VM files available already. These can be used as base boxes where we can install whatever software is necessary for the scenario as well as the IDS to test. Sharing VM becomes as easy as specifying a VM to take from Vagrant with a list of software, possibly with their configuration files, to install. For this, Moirai has a `[Cluster]` section with the names of all the `[machine]` sections. Each `[machine]` section then has the name of the Vagrant box to use. In addition, for each scenario, there is a description which indicates which software must be installed on the base boxes. Where possible, we recommend the use of automation tools such as Ansible [17], an automation engine for configuration management, to make the installation painless and faithful to the original one.

Next, Moirai must setup the VM according to the proper network topology. To this end, each `[machine]` section can define the network configuration for that machine. Additionally, Moirai will have its own router VM, shared through Vagrant, to allow for redirections and spoofing. This is useful if, for example, a malware sample is not configurable and will always try to find its controller at an IP that is not part of the scenario. The router can then transparently redirect every attempt to reach this IP to the VM emulating the controller of the malware.

Lastly, Moirai must respect the timing and execution of each action. Moirai's configuration file contains the `[Scenario]` section to list all the tasks required. Each of these `[task]` then contains its timing as well as the target and the actions to execute. In addition, Moirai can automatically send required file before an action and retrieve artifacts after the action is done.

Moirai is available here [10]. It is written in python3 so should be cross-platform. It can control UNIX based machines using ssh and Windows based machines using winrm. We have already implemented two scenarios [18] accross six VM and ten tasks. They are rough draft of what an APT could be like. These scenarios were then sent to a colleague who did not know either Moirai or our scenarios and they were reproduced perfectly. These scenarios were then used for the evaluation in [19].

As an example, let's say we want to create a simple scenario where an attacker exploits the shellshock vulnerability on a webserver. The resulting file is in listing 1. First, we name the machines in the `[Cluster]` section (line 1), and we specify them in their own sections (lines 4 to 10). They both use the same base box and have an IP address. We then define a number of tasks in the `[Scenario]` section (line 12), as well as a maximum run time of 10 minutes. First, we want to start the script which will control the first simple remote access tool (RAT) and make it download a better RAT. This is the `[botmaster]` task, which is started at the beginning of the scenario (line 18) and requires the *botmaster.py* file (line 20) to be uploaded before it can start. One minute in, the `[shellshock]` task (line 22) is started to exploit the vulnerability on the target. Finally, five minutes after the `[shellshock]` task, the `[pupy]` task (line 27) uses the reverse shell opened by the bot master to install the second RAT. Using Moirai, the whole scenario can be executed with one command: `moirai spin`. There is no need for further input. In order to add an IDS to the victim VM, the researcher should download the base VM, add the IDS to it, and use the resulting VM as the new base VM for the victim, the latter of which is done by changing a single line in the `ini` file.

While this is a simple scenario, anyone can clone the repository, add other attacks, other attackers or benign actions and share it by creating a pull-request on the

Listing 1 Example Moirai configuration

```
[Cluster]
machines = attacker, victim

[attacker]
box = TFDuesing/Fedora-20
ip = 192.168.51.5

[victim]
box = TFDuesing/Fedora-20
ip = 192.168.51.100

[Scenario]
tasks = botmaster, shellshock, pupy
duration = 10m

[botmaster]
target = attacker
timing = 0
actions = ./botmaster.py
files = botmaster.py

[shellshock]
target = attacker
timing = 1m
actions = ./shellshock.sh 192.168.51.100

[pupy]
target = attacker
timing = +5m
actions = ./download-pupy.sh
```

repository in [18] with their modifications. Once accepted, everyone will benefit from the improved scenario. As a matter of fact, the scenario which inspired this example is more complex, with four VM and seven tasks.

IV. Conclusion

In this article, we have first argued that in order to have datasets which enable us to compare any IDS, we must not only share the dataset but also the methods in which it was created. In this way, whatever data the IDS require to operate, the data can be collected by replaying the dataset and adding the necessary probes. To this end, we have created Moirai, an open-source tool to make creating and sharing scenarios easy. It leverages existing technologies where possible, such as Vagrant to share and setup VM, and its configuration file is easy to read, understand and modify. This allows researchers to build on each others' scenarios and create ever more and better scenarios. In addition, it is easier to adapt a scenario to replace a malware with a newer one than it is to do the same on already generated data. Thus, scenarios can be kept up to date with the evolving computing landscape.

We have identified two limitations to our approach. The first one is that Moirai leaves the assessment of IDS performance to the person doing the testing. Moirai does not know when attacks occur, does not know how data are collected so cannot label it and does not interact with the IDS so cannot check whether the attack has been detected. This limitation is alleviated by the fact that the user knows all of the above and so should be able to assess the performance relatively easily. The second limitation is that a number of recent threats integrate anti-analysis defenses which often alter the behaviour of the malware when it detects it is being run inside a VM. This is a trade-off that is justified by the fact VM are much easier to copy and share and Vagrant does not have the capability to install VM images to physical machines.

References

[1] R. Lippmann, J. W. Haines, D. J. Fried, J. Korba, and K. Das, "The 1999 darpa off-line intrusion detection evaluation," *Computer networks*, vol. 34, no. 4, pp. 579–595, 2000.

[2] J. W. Haines, R. P. Lippmann, D. J. Fried, M. Zissman, and E. Tran, "1999 darpa intrusion detection evaluation: Design and procedures," DTIC Document, Tech. Rep., 2001.

[3] J. McHugh, "The 1998 lincoln laboratory ids evaluation," in *International Workshop on Recent Advances in Intrusion Detection*. Springer, 2000, pp. 145–161.

[4] M. V. Mahoney and P. K. Chan, "An analysis of the 1999 darpa/lincoln laboratory evaluation data for network anomaly detection," in *International Workshop on Recent Advances in Intrusion Detection*. Springer, 2003, pp. 220–237.

[5] Z. Chen, C. K. Yeo, B. S. L. Francis, and C. T. Lau, "Combining mic feature selection and feature-based mspca for network traffic anomaly detection," in *Digital Information Processing, Data Mining, and Wireless Communications (DIPDMWC), 2016 Third International Conference on*. IEEE, 2016, pp. 176–181.

[6] A. Little, X. Mountrouidou, and D. Moseley, "Spectral clustering technique for classifying network attacks," in *Big Data Security on Cloud (BigDataSecurity), IEEE International Conference on High Performance and Smart Computing (HPSC), and IEEE International Conference on Intelligent Data and Security (IDS), 2016 IEEE 2nd International Conference on*. IEEE, 2016, pp. 406–411.

[7] J. Qian, C. Xu, and M. Shi, "Redesign and implementation of evaluation dataset for intrusion detection system," in *Emerging Trends in Information and Communication Security*. Springer, 2006, pp. 451–465.

[8] R. Koch, M. Golling, and G. D. Rodosek, "Towards comparability of intrusion detection systems: New data sets," in *TERENA Networking Conference*, 2014, p. 7.

[9] DARPA, "Impact," 2016. [Online]. Available: https://www.impactcybertrust.org/

[10] Akheros, "Moirai," 2016. [Online]. Available: https://github.com/akheros/moirai

[11] R. A. Maxion and K. M. Tan, "Benchmarking anomaly-based detection systems," in *Dependable Systems and Networks, 2000. DSN 2000. Proceedings International Conference on*. IEEE, 2000, pp. 623–630.

[12] C. G. Cordero, E. Vasilomanolakis, N. Milanov, C. Koch, D. Hausheer, and M. Mühlhäuser, "Id2t: A diy dataset creation toolkit for intrusion detection systems," in *Communications and Network Security (CNS), 2015 IEEE Conference on*. IEEE, 2015, pp. 739–740.

[13] E. Vasilomanolakis, C. G. Cordero, N. Milanov, and M. Mühlhäuser, "Towards the creation of synthetic, yet realistic, intrusion detection datasets."

[14] E. Vasilomanolakis, "On collaborative intrusion detection," Ph.D. dissertation, Technische Universität Darmstadt, 2016.

[15] M. Małowidzki, P. Bereziński, and M. Mazur, "Network intrusion detection: Half a kingdom for a good dataset," in *Proceedings of NATO STO SAS-139 Workshop, Portugal*, 2015.

[16] HashiCorp, "Vagrant," 2010. [Online]. Available: https://www.vagrantup.com/

[17] Ansible, "Ansible," 2012. [Online]. Available: https://www.ansible.com/

[18] Akheros, "Scenarios for moirai," 2016. [Online]. Available: https://github.com/akheros/moirai-scenarios

[19] G. Brogi and V. Viet Triem Tong, "Terminaptor: Highlighting advanced persistent threats through information flow tracking," in *New Technologies, Mobility and Security (NTMS), 2016 8th IFIP International Conference on*. IEEE, 2016, pp. 1–5.