A containerised approach to labelled C&C traffic

Markus Leira Asprusten[1], Julie Lidahl Gjerstad[1,2], Gudmund Grov[1,2], Espen Hammer Kjellstadli[1], Robert Flood[3], Henry Clausen[3], and David Aspinall[3,4]

[1] Norwegian Defence Research Establishment (FFI), Kjeller, Norway,
{Markus.Asprusten,Gudmund.Grov,Espen-Hammer.Kjellstadli}@ffi.no
[2] University of Oslo, Oslo, Norway, julielgj@ifi.uio.no
[3] University of Edinburgh, Edinburgh, United Kingdom,
{s1784464,Henry.Clausen,David.Aspinall}@ed.ac.uk
[4] The Alan Turing Institute, United Kingdom

Abstract. A challenge for data-driven methods for intrusion detection is the availability of high quality and realistic data, with ground truth at suitable level of granularity to train machine learning models. Here, we explore a container-based approach for simulating and labelling C&C traffic of real malware through a proof-of-concept implementation.

1 Introduction & motivation

Data-driven methods for intrusion detection rely on high quality and realistic data in which to infer their detection models from. A SOC or CERT will typically have access to large quantities of log or network data which could be utilised for unsupervised or self-supervised learning. However, evaluation of such models is challenging without an evaluation set with ground truth, while supervised methods require that all data points used for learning are labelled.

One way to achieve such labels is active learning, where a security analyst is in-the-loop during learning. Another option is to label existing data, e.g. from historic incidents or labelling data points from scratch. Due to the shear size of the data, and skills required to do so, manual labelling from scratch is not feasible and would instead require automated labelling techniques, such as Snorkel [6], which provides lower quality *weak labels*. In this paper, we focus on a third approach, which is to simulate benign and adversarial behaviour, and use the knowledge from setting up the simulation to label the data. We limit the work to network-based intrusion detection systems (NIDS), and detection of command and control (C&C) beaconing traffic – a detection problem where NIDS are known to be applicable [4].

An important property of applied machine learning is the ability to generalise beyond the training data, and a known problem for NIDS is that good results on training sets often does not generalise well to an operational setting. Existing tools for C&C, often developed for penetration testing, are often used in real malware [7]. We therefore utilise such existing C&C tools in our simulation. In addition to provide realistic traffic, it means that even if a machine learning model does not generalise well beyond the tools used for simulation, it may still be valuable operationally as the tool may be used in real malware. There are several such tools: Simuland[5] focuses on Microsoft Defence products, with a mapping to the Mitre ATT&CK Framework[6]; Cobalt Strike[7] provides a number of red team activities, from generating phishing emails to browser pivoting; Metasploit[8] supports the full attack scenario, from scanning for vulnerabilities to collecting credentials and generating a final report of the attack; PoshC2[9] focuses on post-exploitation and lateral movements with encrypted C&C traffic and features extensive logging of every action and response; Covenant[10] is a .NET C&C framework; Sliver[11] is a C&C red teaming tool; Atomic Red Team[12] is a library of simple detection tests mapped to the MITRE ATT&CK framework; and Merlin[13] is a popular post-exploitation C&C Tool.

None of these tools provide ground truth and a second challenge is correct labelling captured data with suitable granularity. One common approach, used e.g. by Garcia et al [3], is to label all data from malware infected machines as 'malicious' and everything else as either 'benign' or 'background'. This will entail that some benign data is erroneous labelled as malicious. A different approach is taken by Landauer et al [5], which combines knowledge of attack time with domain knowledge of the attack steps to label post-simulation – a similar approach is also taken by Buchanan et al [1]. Their approach does not target NIDS, and in addition, the labelling quality will depend on the domain knowledge and how it is implemented in the labelling process.

In our work, we instead build on the *DetGen*-tool by Clausen et al [2], where we can achieve finer grain control of the labelling compared with Garcia et al without the need for encoding domain expertise of each attack steps. Here, we encapsulate the malware in a container and label at the container-level, thus separating traffic arising from the malware from traffic arising from other processes in a machine. We extend [2] by encapsulating the Merlin C&C simulation tool in the container. Note that whilst Cobalt Strike was the most common tool used in malware in Recorded Future's 2021 report [7], it requires a licence. We therefore use Merlin, which is also a popular tool for simulating C&C.

2 An experiment using DetGen[Merlin] with Ghost

The Merlin C&C-framework has two main components: a server and a client. The server is configured to listen for HTTP-connections from the client, and sends C&C-commands to the client over this connection. The client software runs post-exploitation on a system you wish to control and will repeatedly connect to the server with a certain interval, also called a heartbeat. To avoid detection, the interval can be skewed to vary the interval.

The DetGen framework is built around Docker-Compose[14]. Each component in DetGen runs in a container, with a separate associated container to capture

[5] https://github.com/Azure/SimuLand

[6] https://attack.mitre.org/

[7] https://www.cobaltstrike.com

[8] https://www.metasploit.com/

[9] https://poshc2.readthedocs.io/en/latest/

[10] https://github.com/cobbr/Covenant

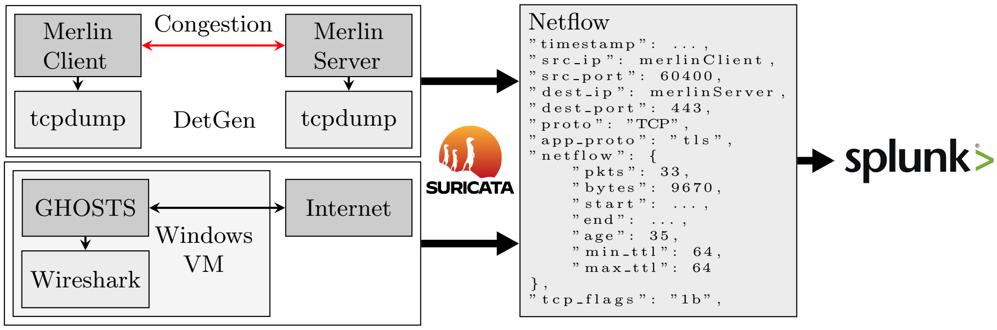
[11] https://github.com/BishopFox/sliver/

[12] https://github.com/redcanaryco/atomic-red-team

[13] https://github.com/Ne0nd0g/merlin

[14] https://docs.docker.com/compose/

the network traffic using `tcpdump`. This separation into container is then utilised when labelling traffic. In the experiment, the Merlin client and server ran in separate containers, with their associated ”`tcpdump`-containers” connected directly to the network interface of the Merlin containers In addition, the DetGen framework adds congestion and other small errors to make the simulation more realistic. This is illustrated in figure 1 (top-left).



```
”timestamp”: ...,
”src_ip”: merlinClient,
”src_port”: 60400,
”dest_ip”: merlinServer,
”dest_port”: 443,
”proto”: ”TCP”,
”app_proto”: ”tls”,
”netflow”: {
    ”pkts”: 33,
    ”bytes”: 9670,
    ”start”: ...,
    ”end”: ...,
    ”age”: 35,
    ”min_ttl”: 64,
    ”max_ttl”: 64
},
”tcp_flags”: ”1b”,
```

Fig. 1. Experimental setup of Merlin with DetGen, and GHOSTS

The setup is sufficient to train a ”signature-model” which can recognise C&C-traffic. However, if our aim is to use supervised learning to train a classifier, we also need benign traffic. To achieve this, we used a framework called General Hosts (GHOSTS) [8] to simulate benign traffic. We configured GHOSTS to visit a list of domains known to be benign and record the traffic (using Wireshark), meaning, as with Merlin, we are capturing HTTP-traffic. Due to technical reasons and time constraints, we did not integrate GHOSTS into the DetGen framework, and instead captured GHOSTS traffic separately in a Windows virtual machine. This is sufficient for our proof-of-concept but will need to be integrated in the future. GHOSTS was then used to to connect to live domains with real world congestion applied. Figure 1 shows the full experimental setup for Merlin and GHOSTS. After running the simulation we changed the IP address of the Merlin Client to be the same as the GHOSTS Client. We then used Suricata[15] to convert the data sets to Netflow before combining them and importing them to Splunk[16] for visualisation. Up to this point, the C&C and Merlin traffic were in separate files, which we exploited when labelling during import into Splunk.

Splunk can then be used to train classification models, which can further be applied to real data. Here, we only visualise the traffic to illustrate how the labels can separate the traffic, as shown in figure 2. The plot shows both the number of bytes transmitted from the Merlin Client to the Merlin Server and benign traffic generated by GHOSTS (in logarithmic scale). It is easy to see the heartbeat in the graph from the Merlin Client. Note that there is some discrepancies initially

[15] https://suricata.io/

[16] https://www.splunk.com/

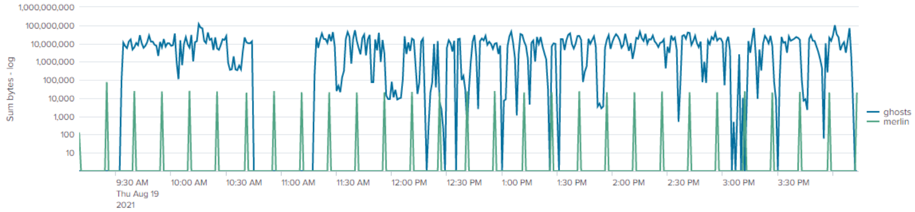


Fig. 2. #bytes of Merlin C&C traffic compared to traffic from GHOSTS in Splunk.

in the heartbeat due to some issues during setup, while a sudden dip in GHOSTS is likely caused by a need for user input to a CAPTCHA or similar.

3 Discussion and further work

We have shown that C&C tools used in practice can be simulated and labelled in a way that it can be separated from benign traffic in a SIEM with a fine grain of atomicity, which can further be utilised to train machine learning models for NIDS. Whilst the scientific contribution presented here may be limited, we believe our approach is promising for applying underlying research in an operational setting. This will require that the simulations are ran over longer time periods, using different C&C tools, different configuration and different architectures. This is also the case for the simulated benign traffic, where GHOSTS need to be integrated into the DetGen framework.

References

1. Buchanan, M., Collyer, J.W., Davidson, J.W., Dey, S., Gardner, M., Hiser, J.D., Lang, J., Nottingham, A., Oprea, A.: On generating and labeling network traffic with realistic, self-propagating malware. arXiv preprint arXiv:2104.10034 (2021)
2. Clausen, H., Flood, R., Aspinall, D.: Traffic generation using containerization for machine learning. arXiv preprint arXiv:2011.06350 (2020)
3. Garcia, S., Grill, M., Stiborek, J., Zunino, A.: An empirical comparison of botnet detection methods. computers & security **45**, 100–123 (2014)
4. Hutchins, E.M., Cloppert, M.J., Amin, R.M., et al.: Intelligence-driven computer network defense informed by analysis of adversary campaigns and intrusion kill chains. Leading Issues in Information Warfare & Security Research **1**(1), 80 (2011)
5. Landauer, M., Skopik, F., Wurzenberger, M., Hotwagner, W., Rauber, A.: Have it your way: Generating customized log datasets with a model-driven simulation testbed. IEEE Transactions on Reliability **70**(1), 402–415 (2020)
6. Ratner, A., Bach, S.H., Ehrenberg, H., Fries, J., Wu, S., Ré, C.: Snorkel: Rapid training data creation with weak supervision. In: Proc. of the VLDB Endowment. vol. 11, p. 269. NIH Public Access (2017)
7. Recorded Future by Insikt Group: Adversary infrastructure report 2020: A defender's view. Tech. Rep. CTA-2021-0107 (2021)
8. Updyke, D., Dobson, G., Podnar, T., Earl, B., Cerini, A.: Ghosts in the machine: A framework for cyber-warfare exercise npc simulation. Tech. Rep. CMU/SEI-2018-TR-005, SEI/CMU (December 2018)