The *Mouseworld*, a security traffic analysis lab based on NFV/SDN

Antonio Pastor
Telefonica I+D
Madrid, Spain
antonio.pastorperales@telefonica.com

Alberto Mozo
Universidad Politecnica de Madrid
Madrid, Spain
a.mozo@upm.es

Diego R. Lopez
Telefonica I+D
Madrid, Spain
diego.r.lopez@telefonica.com

Jesus Folgueira
Telefonica I+D
Madrid, Spain
jesus.folgueira@telefonica.com

Angeliki Kapodistria
Space Hellas S.A.
Athens, Greece
akapodistria@space.gr

ABSTRACT

Machine Learning (ML) technologies applied to Cybersecurity, especially in the area of network cyber threat detection, are a promising choice, but they require additional research in the applicability of a wide range of available algorithms. Such algorithms usually require training using good-quality and quantitatively significant datasets, which are rarely publicly available. To this end, in this paper we describe a novel experimental framework, that we call the **Mouseworld**, that combines NFV and SDN to create an environment able to (1) blend and transmit real and synthetic traffic and (2) collect and label this traffic in order to be utilised for training and validating ML algorithms that will be applied to the detection of cybersecurity threats. The Mouseworld framework includes a set of traffic generation, collection and labelling modules, jointly with analytics and algorithm training and visualization components. The OSM open-source network orchestrator is utilized to control and manage the framework and to deploy the training and validation scenarios. We present a preliminary result on the area of Security threat detection as a demonstration of the framework viability.

CCS CONCEPTS

• **Security and privacy → Network security**; • **Networks →** *Network services*;

KEYWORDS

Cybersecurity, threat detection, machine learning, NFV

ACM Reference Format:
Antonio Pastor, Alberto Mozo, Diego R. Lopez, Jesus Folgueira, and Angeliki Kapodistria. 2018. The *Mouseworld*, a security traffic analysis lab based on NFV/SDN. In *ARES 2018: International Conference on Availability, Reliability and Security, August 27–30, 2018, Hamburg, Germany.* ACM, New York, NY, USA, 6 pages. https://doi.org/10.1145/3230833.3233283

ARES 2018, August 27–30, 2018, Hamburg, Germany

ACM ISBN 978-1-4503-6448-5/18/08…$15.00
https://doi.org/10.1145/3230833.3233283

1 INTRODUCTION

NFV (network functions virtualization) and SDN (software-defined networking) technologies are being adopted by the industry because they offer a set of benefits compared with legacy technology, in what relates to network infrastructure elasticity, service time-to-market, and reduced total costs of ownership (TCO). ISPs, and companies in general are adopting these technologies to deploy more optimized networks at reduced costs while easing new service creation. Cybersecurity is one topic where these technologies are most promising, because of the capacity to detect and quickly respond to incidents. As a natural consequence, NFV-based Security as a Service (SecaaS) has been proposed [4], [12].

Recently, Machine Learning (ML) techniques have started to gain momentum, especially when Big Data regimes are considered. Cybersecurity and network traffic analysis are one of the most promising areas where to apply ML. It is expected that in the short and middle term, networks are going to provide to end users faster speeds, greater bandwidths, and higher dependability. These enhanced properties will translate into the need of processing and analysing huge amounts of data coming from different sources. In this context, the role of data analytics is twofold, firstly supporting new business opportunities and secondly guaranteeing that 5G networks are rolled out and operational. It is undisputable that in the immediate future the latter will become a strategic functionality of MANO (management and orchestration) stacks specially when Cybersecurity and traffic classification scenarios are considered.

In these scenarios of cognitive management application to orchestration, many of the intelligent and self-learning components are principally based on machine and deep-learning techniques. Lately, many of the deep learning techniques that have proven successful in solving complex Artificial Intelligence problems in domains such as computer vision or speech recognition, come from a specific ML subarea called supervised-learning. What differentiates supervised algorithms from other ML approaches is that they need to be previously trained with a representative collection of examples (datasets) in order to perform with a reasonable accuracy. In addition, when training supervised ML components, it is compulsory to utilize examples containing the expected output (i.e. labelled examples) which imposes a non-negligible limitation to the available datasets for training. Nowadays, gaining access to collections of actual examples is very often not possible and in particular, when cloud and telecom domains are considered, the situation becomes more difficult indeed due to the reluctances of data owners to share their datasets with the competition, and the tight regulations on data protection. Secondly, even when having access to a relevant data source, we need to put a label on each element of the dataset in order to be able to train supervised ML components. Same problem apply in the case of unsupervised ML but in the performance algorithm validation process. In this regard and assuming that we are in a big data regime, it is impractical to manually label a dataset when we consider the size of a representative cloud or network traffic dataset containing hundreds of millions of samples.

To this end, we propose the Mouseworld, a controlled environment for running experiments that will generate realistic labelled datasets for training supervised ML components and validate supervised and unsupervised solutions. This environment is deployed on an NFV-enabled architecture, under the management of an orchestrator (NFVO), extending an NFV MANO stack as necessary.

2 FILLING THE GAPS

2.1 Cybersecurity needs

To foster research on cybersecurity threat detection based on network traffic, it is obvious that threat data generation is required. Frameworks for malware threat identification use sandbox environments that permit malware components to interact with the network and to capture the traffic generated for later analysis. This is a common method to locate malicious servers, identify the specific protocols used and apply fingerprinting techniques. Other threats such as network attacks, volumetric Distributed Denial of Service (DDoS), traffic tunneling, cache poisoning, or Cross site scripting (XSS), need specific tools to generate attacks, and specific honeypots to simulate vulnerable services.

Network-oriented security tools can be applied to the previous scenarios, but they need specific traffic profiles to be setup to solve concrete problems. Deploying scenarios complex enough to cover all previous cases in a realistic way, is extremely costly in time and effort. This problem is aggravated when new types of threats, malware binaries, targets clients and servers need to be considered.

2.2 Machine Learning needs

The design, training and validation of ML algorithms depend on the availability of datasets. Performance of ML algorithms is assessed usually over a limited dataset of reference where cross validation method [8] is applied to obtain objective results. Lack of new fresh datasets can cast doubts about how ML algorithms respond to changes over time or as traffic behavior evolves. A continuous feed of new datasets, encompassing a variety of traffic profiles thus becomes extremely useful, if not essential, for training and validating. This is especially relevant in anomaly detection algorithms, as new anomaly root causes are identified, a priori or posteriori.

There is an additional problem with the application of ML to network traffic scenarios and specifically to cybersecurity cases. The lack of labelled data [3] makes infeasible the use of supervised ML algorithms, where labelled traffic flows are necessary for training and validation processes. In addition, unsupervised algorithms need labelled data for cross validation based on area under the receiver operating characteristic, or AuC metric [5]. Combining network security devices, such IDS, Firewalls, or DPI tools to identify attacks and label the related flow, is the commonly used solution. This labelling approach introduces bias to the ML algorithms during the training process, resulting in loss of the expected generality for detecting threats unknown to that tools, e.g.: Zero-day vulnerabilities without a signature in a DPI.

Next section describes the framework design, using NFV/SDN technology, which can help to alleviate the aforementioned problem.

3 RELATED WORK

The characterization of different algorithms applicable to network flow traffic analysis to detect cybersecurity incidents is a topic with high interest. A system for a generic task of traffic generation and classification was introduced in [1], but only for mobile Android-based classification. In [11], authors present the application of ML for threat detection, based on system logs (web and firewall) and human labelling processes to improve the accuracy of the algorithms. However, to the best of our knowledge, the Mouseworld is the first framework that combines NFV/SDN to automatically train and validate security-related algorithms using labelled datasets.

4 ML ORIENTED NFV/SDN FRAMEWORK

In this section we describe the Mouseworld, a framework that leverages NFV and SDN technologies to offer training and validation experiments of different ML algorithms in order to detect cybersecurity threats.

Fig. 1 shows a global perspective of the Mouseworld framework. This framework is composed by several types of modules: The traffic generators, the network infrastructure, the dataset collector, and the modules for labelling, training and validation. The framework allows to configure different types of applications and deploy customized network architectures to generate the desired traffic over the infrastructure. Some specific probes will collect the traffic packet by packet and group them to network flows summaries in different formats. These network flows will be converted in datasets suitable for ML application by adding a label representing the corresponding security threat to each flow. Finally, the obtained datasets are used for training the ML algorithms and for validating their accuracy and performance.

4.1 Traffic Generators

The traffic generators include all the clients and internal servers that interchange network traffic between their communications. Clients can be standard PCs, mobile devices, IoT devices, etc. Their own nature defines them as the traffic initiators. The software running on the clients can be general purpose operating systems and applications. Fig. 1 represent some examples such as web browsers, video streaming clients, or cloud storage applications. Additionally, cybersecurity related traffic is needed in the client side to cover multiples needs:

- Pentesting tools, vulnerability scanners.
- Exploiting kits to execute new attacks.
- Botnet activity, such as DDoS, control channels, files exfiltration.

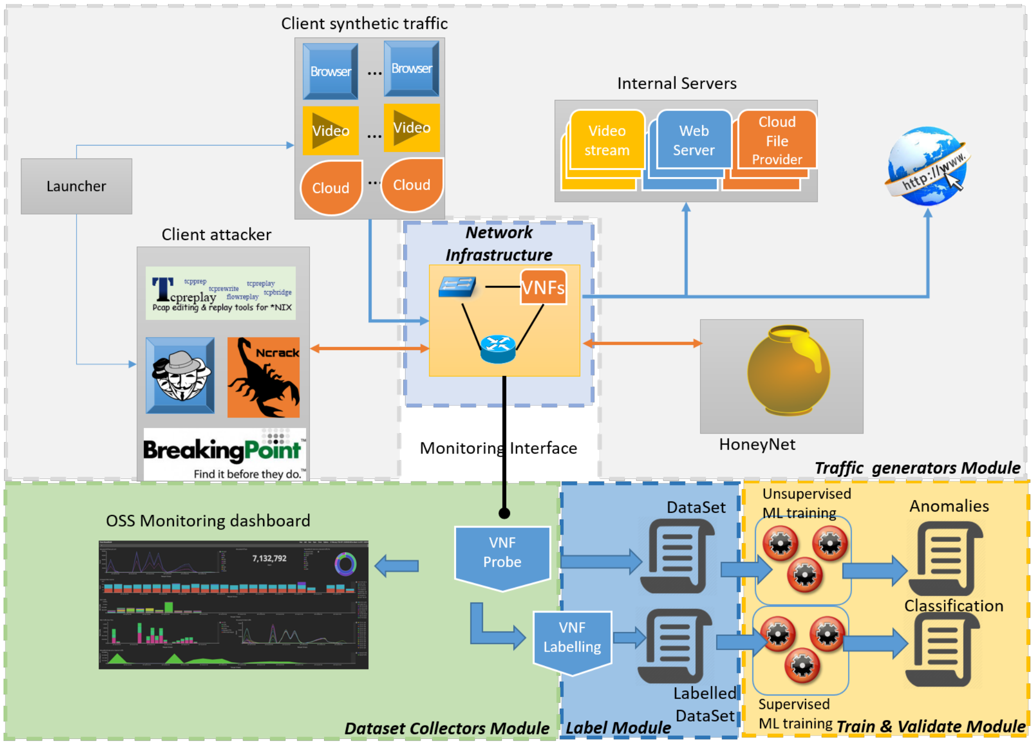


Figure 1: NFV/SDN Framework for Machine Learning applied to Security.

- Traffic capture injection tools, allowing to modify some fields and replay any previously captured traffic. This functionality is essential for some use cases. For instance, malware samples can be executed in a sandbox and all the traffic recorded to be reused by these tools later when it is needed.
- Security protocol testing applications. Traditionally traffic generators allow the injection of traffic to evaluate the performance or to validate the compliance to a standardized protocol. Some of them has included the capability of making fuzzing over protocols or reproducing security attacks at the application layers. One simple example is the generation of DNS cache poisoning packets.

Internal Servers on the framework represents all the necessary applications to address the requests from the clients. In general, they will generate additional traffic as a response to the client stimuli, but not exclusively, being capable in some situations to push information without previous request. All types of clients have a counterpart in the internal servers or in case of interest, an alternative can be the use of real Internet Servers. Note that using external servers, means no access to any metadata in the servers itself, a valuable information for labelling.

The clients and servers should support virtualization and integration as VNFs to be on-boarded and instantiated by a NFV orchestrator. If their nature does not allow such virtualization, i.e. a IoT device, simulation or replay tools can be used, to guarantee a consistent NFV orchestration in the whole framework. All clients and servers generate metadata logs, that identifies the flow generated by the type of client/server. This information is collected by the Labelling Module to add labels automatically to the network traffic flows using the management network (not represented in Fig. 1)

The traffic generation module in the framework includes the Launcher component, in charge of defining the profile of the traffic to be generated. This profile defines the traffic model, e.g. lineal or exponential variation of the connections, the global experiment duration in time, the type of traffic and attacks to include, the number of connections per client, etc. This information is distributed to each traffic generator, before the experiment starts.

4.2 NFV/SDN Network

The network infrastructure can leverage the NFV and SDN technologies and combine physical (PNF) and virtual network functions (VNF) to setup different types of cybersecurity experiments. Fig. 2 highlights the main components in a NFV/SDN architecture. At the physical level, one of the most relevant elements is the storage hardware, where network flows and processed datasets will be stored for their further use in training and validating different ML algorithm. Therefore, a high performance and capacity storage is needed. The SDN controller will be in charge of two main tasks. First, establish the network connections between PNFs and VNFs, and secondly implement the flow paths for the traffic. Two alternatives are possible: mirroring of the traffic between clients and servers to an external probe (Fig. 2a) or passing all the traffic through the capture probe that generates the flows (Fig. 2b). The first approach is less intrusive, where the second one can be useful in experiments where some flows need to be blocked, to alter the client behavior, e.g. block malware Command-and-Control Server to trigger alternative behavior.

The Mouseworld NFV Framework allows multiple configurations for cybersecurity training, based on NFV infrastructure. Fig. 3 shows some examples. Fig. 3a corresponds to a DDoS topology, while the configuration shown in Fig. 3b has been used for a malware sample using resources in Internet. Finally, Fig. 3c illustrates the configuration for any training dataset in which normal traffic is mixed with an attack to generate a more realistic traffic pattern for unsupervised anomaly detection.

4.3 Dataset collector module

Information on network flows is generated by routers, switches or other VNFs, PNFs, or components of the virtualized network infrastructure. Alternatively, based on mirroring ports or tap devices, dedicated probes can collect the traffic and generate network flow information to relieve forwarding functions at all levels from this task. In this framework, Fig. 2 considers both alternatives. The main functionality of the probe in the framework is to collect and transform the flows in a suitable format for ML algorithms, such as multidimensional arrays or CSV files. Using the CSV format as example, each line corresponds to a network flow and each comma separated field is a feature of the network flow, such as source IP address or port number. Deriving new features or normalizing them in range are commonly used transformations useful in general for ML algorithms.

This network flow information is also useful to monitor the experiment traffic, by delivering the identified network flow features to a Console, as shown in Fig. 1, represented by a graphical dashboard.

4.4 Labelling module

As mentioned when analyzing the needs in Section 2, ML requires labelling for the application of supervised training, as well as for any type of formal validation. This module collects the information received from clients and servers and matches each recorded flow to add the appropriate label. Simple labeling can be achieved by assigning specific values to some fields. One example is predefining address pools for clients that generate normal traffic, and other different address range for the specific attack. Therefore, the labelling process can assign attack labels to all the flows from the last address range.

All this information is appropriately stored to be used in ML training and validation processes. Note that the stored datasets could also be useful for future training scenarios. Suppose that a new variant of a malware family or a type of DoS attacks change slightly from the point of view of the network flows, and therefore training an algorithm with historic traffic flows can help the algorithm to detect these new variants.

4.5 Training and Validating Module

The last module is used for the training and validation of ML algorithms. Standard software frameworks, and libraries for BigData management and ML training are used in this module. HPC[1] servers are also recommended in this module. Output of the experiments can be also integrated with the console to perform convenient tracking of the training and validation processes.

5 IMPLEMENTATION

To test the concept of this NFV/SDN Framework applied to ML for cybersecurity, we implemented the Mouseworld Lab. We mainly chose different open source and commercial tools although some additional components, like the Launcher, were developed to cover the gap where there was no software available for the requested task. Finally, we integrated and tested some malware families in the environment.

For the traffic generator module implementation, a set of existing VNFs or compatible software with NFV were selected and integrated. The goal in all the VNFs is to be able to apply automated deployment jointly with remote control and execution with as less human intervention as possible apart from the definition of the dataset generation scenario.

In order to obtain network traffic like the one generated in an ISP PoP or in a typical corporate network, VNFs according to a client and server model were selected and integrated. Client model functions included:

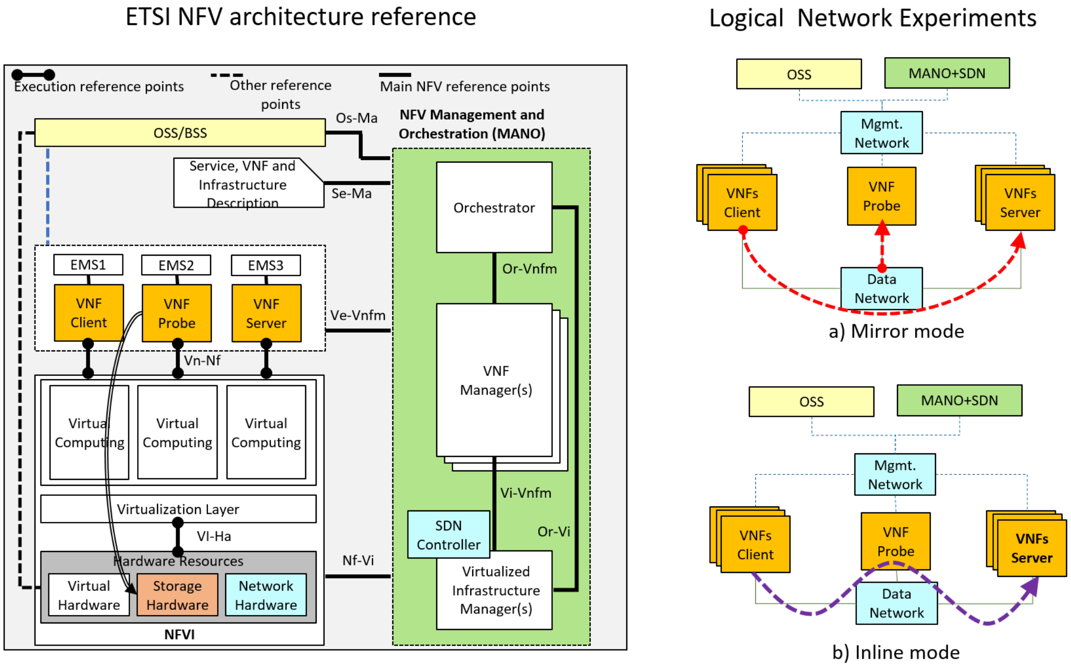
- phantomJS[2] , and Selenium[3] to generate realistic client browsing based on Firefox and Chromium. These tools support all the artifacts of a web page, including tracking connections, and ad banners. Also, video streaming and file transfers. They can be used in headless mode and automatized by scripting.
- VLC[4] client to reproduce video and audio streaming with different qualities. Command line support allows to automatize the generation of request.
- Breaking Point Virtual Edition[5], a commercial traffic generator that can create multiples flows based on predefined profiles, such as ISP mix traffic with P2P, video, browsing, email, and other protocols.

The VNF within the server model included Apache Web servers, OwnCloud servers[6] and VLC servers. In order to discard simple identifications based on server ports, some server instances where configured in not-well-known TCP ports.

The attack VNFs for the selected cybersecurity threats include some specific hacking tools, such password brute forcing tool ncrack[7], for penetration test. The most valuable tool was tcpreplay[8], which allowed us to use some publicly available malware packet captures (pcap files). On the server side, two different VNF honeypots were assembled: cowrie[9] and glastopf[10].

Network infrastructure was intentionally designed as simple as possible and at OSI layer 2, through several physical SDN-enabled

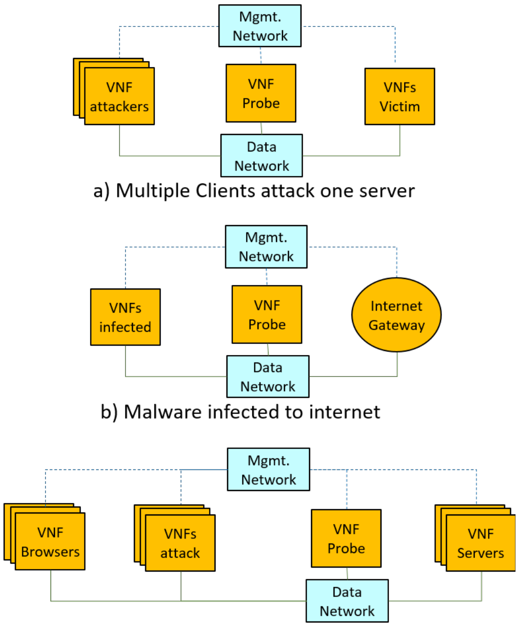
[1] HPC includes GPU for deep neural network training in acceptable times
[2] PhantomJS headless webkit, http://phantomjs.org/
[3] Selenium browser automation, https://www.seleniumhq.org/
[4] VideoLAN VLC, http://www.videolan.org/vlc/
[5] IXIA BreakingPoint Virtual Edition, https://www.ixiacom.com/
[6] OwnCloud file sharing service, https://owncloud.org/
[7] Network authentication cracking tool, https://nmap.org/ncrack/
[8] Pcap editing and replaying utilities, https://tcpreplay.appneta.com/
[9] SSH honeypot, http://www.micheloosterhof.com/cowrie/
[10] Python web application honeypot, http://glastopf.org/



a) Mirror mode

b) Inline mode

Figure 2: NFV reference model vs logical view



a) Multiple Clients attack one server

b) Malware infected to internet

c) Mixed normal and attack traffic

Figure 3: Deployment examples in security training

switches, and an OpenDayLight[11] SDN controller. Network complexity is managed by the orchestrator, allowing different customized scenarios adding flow paths or inline VNFs. For this implementation, a port mirroring of all the traffic was configured to be sent to the VNF probes, following Fig. 2a model. For more complex scenarios the Open Source MANO[12] has been installed in the management network, using OpenStack for the VIM (Virtualised Infrastructure Manager, as per ETSI NFV architecture) functionality.

The dataset collector was implemented with the creation of a VNF that runs nfcapd[13] in daemon mode, listening for all receiving packets. The VNF probe dumps all the flows into two formats: netflow v9 binary files, and csv files. In this initial version of the Mouseworld Lab, we followed a simple design for the labelling module as the datasets were only oriented to validation and not for training. This was the case because the algorithms to be tested were unsupervised, and therefore no previous labeling was required. The labelling process was performed by matching the source and destination IPs used in the attacks. In this implementation, the dataset collector module and the train and validation module were based on two tools, Apache Spot[14] and ELK[15]. The former simplifies the process of data collection, the analysis in BigData environments and it is being actively adopted by other security-related projects, like SHIELD [4]. The latter is commonly used by network service providers to complement network monitoring portfolios. This tandem allows us to monitor the experiments using the Kibana graphic interface, meanwhile the Apache Spot ML module allows us to train, experiment and validate the anomaly detection algorithms.

[11] Open Day Light, https://www.opendaylight.org/

[12] ETSI Open Source MANO, https://osm.etsi.org/

[13] Nfdump toolset, http://nfdump.sourceforge.net/

[14] Apache Spot cybersecurity project, http://spot.incubator.apache.org/

[15] Elasticsearch, Logstash, and Kibana, https://www.elastic.co/elk-stack

Table 1: Features used in algorithms

Feature	Description
FIRST_SEEN	Flow starting date
IPV4_SRC	Source IP address
DURATION	Flow duration in seconds
L4_DST_PORT	Destination IP address
PROTOCOL	TCP/UDP/ICMP to integer
TCP_FLAGS	TCP flags in Netflow format to integer
PKTS	Number of packets per flow
BYTS	Number of bytes per flow
PKTS_SEC	Number of packets per second per flow
BYTS_SEC	Number of bytes per second per flow
BYTS_PKTS	Number of bytes per packet per flow
N_CON	Number of similar flows (IPs, dst port and protocol)

6 EXPERIMENTS

To demonstrate the NFV/SDN Mouseworld framework viability and its applicability in cybersecurity detection, some experiments and measurements were executed. Note that the selected algorithms and features are merely examples to validate the framework, and we do not claim them to be a relevant result of the research described in this paper.

Firstly, some public traffic capture [7] (in pcap format) were chosen. One relevant exploit-kit called RIG and one ransomware called JAFF. Secondly, a network service scenario, similar to Fig. 3c, was deployed. The VNF attacker used was the tcpreplay tool reinjecting previously selected malware pcap files. VNF browsers and VNF servers generated traffic in parallel. The VNF probe, apart from capturing the traffic, generated the netflow format, converted it to a dataset array and filtered and transformed the dataset using the features shown in Table 1.

Three well known unsupervised anomaly detection algorithms were used:

- iForest (Isolation Forest) [6]
- LOF (local outlier factor) [2]
- OCSVM (One class support vector machine) [9]

Validation was made using fresh traffic captures, independent from the traffic used in training. The implemented labelling module was a simple script that uses IP source, IP destination and the periods of time when malware was reinjected and adds the label *anomaly* in those flows. The selected performance indicator was the AuC. Best results obtained with a dataset of 50.000 flows during 12 hours are shown in Table 2, where OCSVM has a superior performance, with the prerequisite that it is trained without malware in the traffic.

Table 2: AuC results for each malware sample

Malware sample	AUC		
	OCSVM	iForest	LOF
Malware RIG EK: Bunitu	0.99539	0,55758	0,49504
Malware RIG EK: Dreambot	0.99507	0,79504	0,49500
Malware RIG EK: Chthonic	0.99546	0,55388	0,49500
Malware JAFF: (zip)	0.99503	0,49501	0,82836
Malware JAFF: (pdf)	0.99502	0,74501	0,74501
Malware JAFF: (doc)	0.99502	0,74501	0,49500

7 CONCLUSIONS AND FUTURE WORK

This paper presents a novel framework for ML training and validation, applicable to different types of cybersecurity threats. This framework is based on the NFV and SDN technologies and allows high flexibility in designing and executing multiple testing scenarios.

The potential capabilities of this framework go beyond the presented experiment and its results. The combination of different network services, threats and security VNFs will help in generating scenarios focused on solving relevant security challenges by ML algorithms. In fact, the name Mouseworld was intentionally taken to highlight that this framework can be used for validating real environment at scale, solving the scarcity of labeled datasets and reproducing real threats.

As the application of OSM to the Mouseworld framework consolidates, we foresee the availability of specific orchestration functions focused on dataset generation, which will be contributed to the OSM community to be included in the upstream project.

ACKNOWLEDGMENTS

The work described in this article has received funding by the European Union Horizon 2020 research and innovation programme, under Grant Agreement no. 700199. The Mouseworld name is inspired in this sci-fi novel [10].

REFERENCES

[1] Giuseppe Aceto, Domenico Ciuonzo, Antonio Montieri, and Antonio Pescapé. 2018. Multi-classification approaches for classifying mobile app traffic. *Journal of Network and Computer Applications* 103 (2018), 131–145.

[2] Markus M Breunig, Hans-Peter Kriegel, Raymond T Ng, and Jörg Sander. 2000. LOF: identifying density-based local outliers. In *ACM sigmod record*, Vol. 29. ACM, 93–104.

[3] Anna L Buczak and Erhan Guven. 2016. A survey of data mining and machine learning methods for cyber security intrusion detection. *IEEE Communications Surveys & Tutorials* 18, 2 (2016), 1153–1176.

[4] G Gardikis, K Tzoulas, K Tripolitis, A Bartzas, S Costicoglou, Antonio Lioy, B Gaston, C Fernandez, C Davila, A Litke, et al. 2017. SHIELD: A novel NFV-based cybersecurity framework. In *Network Softwarization (NetSoft), 2017 IEEE Conference on*. IEEE, 1–6.

[5] Markus Goldstein and Seiichi Uchida. 2016. A comparative evaluation of unsupervised anomaly detection algorithms for multivariate data. *PloS one* 11, 4 (2016).

[6] Fei Tony Liu, Kai Ming Ting, and Zhi-Hua Zhou. 2008. Isolation forest. In *Data Mining, 2008. ICDM'08. Eighth IEEE International Conference on*. IEEE, 413–422.

[7] Malware-traffic-analysis 2017. Retrieved Feb 01, 2018 from http://www.malware-traffic-analysis.net/

[8] Payam Refaeilzadeh, Lei Tang, and Huan Liu. 2009. *Cross-Validation*. Springer US, Boston, MA, 532–538. https://doi.org/10.1007/978-0-387-39940-9_565

[9] Alex J Smola and Bernhard Schölkopf. 2004. A tutorial on support vector regression. *Statistics and computing* 14, 3 (2004), 199–222.

[10] Brian Stableford. 2013. *The cassandra complex*. Tor.

[11] Kalyan Veeramachaneni, Ignacio Arnaldo, Vamsi Korrapati, Constantinos Bassias, and Ke Li. 2016. AIˆ 2: training a big data machine to defend. In *Big Data Security on Cloud*. IEEE, 49–54.

[12] Shao Y. Zhu, Sandra Scott-Hayward, Ludovic Jacquin, and Richard Hill. 2017. *Guide to Security in SDN and NFV: Challenges, Opportunities, and Applications*. Springer.