Semantic Modelling of Network Traffic for Anomaly Detection

PhD 2nd Year Progress Report

Henry Clausen

March 23, 2020

1 Introduction

1.1 Summary of plans outlined in last years proposal

This PhD-project was outlined to use the idea of semantic software models and apply it to network traffic data, and to employ methods from machine learning to automatically learn precise contextual models of network traffic generated by one or more machines. Specific goals of this research project were defined as following:

1. Build an understanding of how contextual structures manifest themselves in network traffic.
2. Develop a semantic model of traffic structures that works on different levels of traffic abstraction (packets or flows) and enables the incorporation of semantic features from the packet level on a flow-level to extend available flow information.
3. Provide a framework to generate traffic data with a high level of ground truth about traffic origins and purposes.
4. Introduce robustness to anomaly-based models by identifying gradual changes in network communication as traffic with common semantic substructures.

These were formulating in last years proposal into the following sub-projects:

1. Modelling of connection establishment via LSTM encoders
2. Traffic generation and data fusion
3. Software evolution and drift
4. Flow-level based modelling

Furthermore, the following (summarised) research questions were defined for each of these goals:

Research Question 1
How well-structured is the space of contextual behaviours observed in the traffic of a machine or a network? How much does noise or input variation blur the observable contextual differences between clearly distinct actions?

Research Question 2
To what degree can contextual structure in network traffic be captured in a model from a training dataset, and how can we achieve this? How can a model adapt to changes of normal contextual structures?

Research Question 3
What is a meaningful representation of traffic structures? What requirements must a labelled traffic generation framework fulfill to provide realistic data?

Research Question 4
What will a contextual model be able to prevent?

There has been substantial progress on research questions 1 and 3, which I outline below. Research question 2 has been answered partly, with research question 4 still to be addressed.

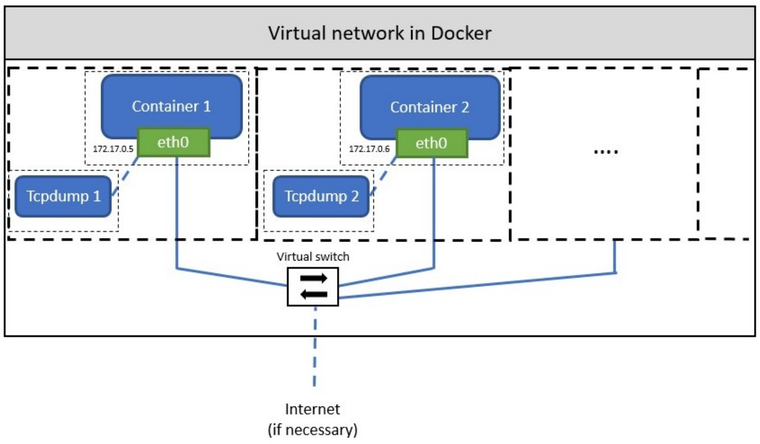


Figure 1: Visualisation of the virtual network in Docker

2 Research progress in the last year

2.1 Traffic Generation using Containerization for Machine Learning

Building contextual models of network traffic means to build an understanding how different network interactions can be distinguished via their traffic trace. However, available network traffic datasets do not contain ground truth labels about the nature of computer interactions and often suffer from a lack of realism. To improve this and ensure that our models extract meaningful sets of sequences that represent these different interactions, we started developing a containerised traffic generation framework to generate traffic with ground truth labels.

This framework, called *DetGen*, generates controlled and isolated network traffic from a variety of applications and tasks. For this, a virtual network was created using the virtualisation program *Docker*. In this network, two or more parties can communicate through containers, which are sandboxes containing programs inside a minimal virtualised operating system. The benefit of this design is that individual containers can only communicate with each other via the virtualised network while the host is in complete control of the parallel execution of tasks in multiple containers. To capture the traffic, every container in the network was complemented with a *tcpdump*[1] container hooked onto the network interface, see Fig. 1. The captured traffic can then be labelled according to the particular scenario it was generated by. We implemented a variety of network service scenarios to capture a diverse set of network traffic.

We emphasised the following particular strengths in the DetGen framework, which makes the generated data particularly suitable for ML-applications:

- traffic variation through a diversity of generation scenarios as well as transmission disturbances,
- ground truth labels through containerised separation of generation scenarios,
- scalability of the amount of generated traffic,

[1]A common packet capture utility

- and modularity of the framework to easily extend and update the set of scenarios.

This work was started in summer 2018 by me and Nikola Pavlov (an LFCS summer intern), and continued in summer 2019 by me and Robert Flood (an LFCS MSc student). My responsibilities lied in overall design of the framework as well as the requirements on the generation features, testing and extending individual scenarios as well as implemention, and the design and conduction of validation experiments.

In autumn of 2019, Robert Flood and I described DetGen in a paper of which I am first author, which was submitted and accepted at the ACSAC DYNAMICS workshop 2019, and will appear in the corresponding proceedings this March.

Due to very positive feedback at the workshop as well as encouragements and suggestions to extend DetGen, we are working further on this project, which is described in Section 4.6.

2.2 Short-term contextual model of network flows using LSTM networks

In the context of the overall research goal of this project, some exploratory work on contextual anomaly detection for network events was conducted before my PhD started by Marc Sabate, Gudmund Grov, Wei Chen, and David Aspinall. This work uses a recurrent neural network to capture meaningful sequences of *NetFlows* and reflect reccurring patterns in a model. For that, recorded NetFlows are grouped according to the generating host. Furthermore, to filter out sequences of flows that are unrelated to each other, a squence of flows that are close in time is grouped into what is called a session as an approximation of the true relation. Each session then serves as a training or test sequence for a behavioural model. Learned contextual behaviour is reflected through the capability of the model to predict traffic protocols and network ports of flows in a session from a smaller subset of flows, with more accurate predictions being rewarded in the training process. Sessions which deviate from previously observed behaviour are then predicted poorly by the model and flagged as potentially malicious.

This work however was not yet complete for successful publication, which is why I took the responsibilty to improve it and bring it to a state suitable for publication. During my work, I made significant changes and additions to both improve the model performance as well as extend the evaluation to highlight the benefits and novelty of this work. In particular, I made the following major changes:

1. I specified the scope of the model to the detection of U2R and R2L attacks, to which it is more suited than high volume attacks. I also exchanged the evaluation datasets to mutiple ones that are more suitable for this task and more realistic in nature.

2. Originally, the model only incorporated the protocol and the port for each flow. I extended this to the direction (from or to host) as well as the size of each flow, which improved detection rates especially for brute-force attacks and sql injections. For this, I came up with an efficient way to process these additional inputs without significant increase of model size.

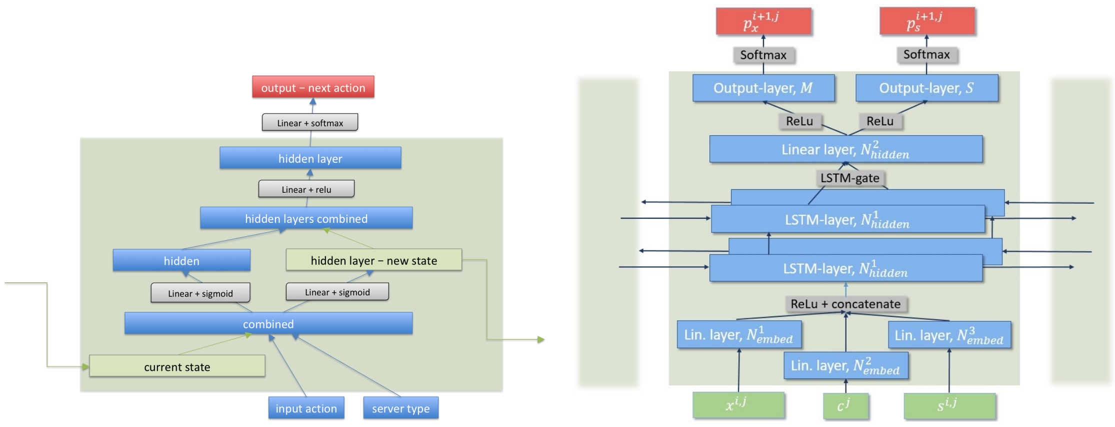


Figure 2: Architectures of the initial model (left) and the extended LSTM network I created (right) for modelling sequences of network flows.

3. Similarly, the original model was a standard recurrent neural network with one layer. I extended this to a bidirectional LSTM network with multiple layers, which required careful parameter calibration and further boosted detection rates. Depicted in Fig. 2 is a comparison between the original model and my extended model.

4. The detection method was changed from an overall host classification to a session classification using a scoring threshold, which is more realistic in a real-world deployment due to the high imbalance between benign and malicious traffic.

5. I replaced the CTU-14-dataset with the more suitable datasets CICIDS-17 and UGR-16, and identified suitable data traces for modelling and analysis.

6. I carved out the novelty and contribution of this work more, and implemented three comparision benchmark models to highlight the benefit of our model. I also highlighted the differences between our work and recent applications of LSTMs to intrusion detection.

The developed model achieves significantly better results than the implemented benchmark models, with 6 out of 7 attacks being detected reliably and an overall AUC (area under ROC-curve) improvement of more than 100% to the next best model.

Since the paper is now in under my responsibility, and due to all the efforts I put into its improvement, we agreed that I would be the first author of this work in the event of publication. Unfortunately, due to the challenges described in Section 4.1, we were so far unable to get this work accepted at a suitable venue. We first submitted the manuscript to the ACSAC conference, where the general consensus was that the application of LSTM networks for network anomaly detection is not novel enough, and that there is not enough connection between the used methodology and the detected attacks. After efforts to improve on these issues, we attempted a revised manuscript to the CODASPY conference. Again, the main criticism here was novelty and scope of the methodology, and scepticism about the translation of results to real-world application. After more thorough revisions, highlighting of the papers novelty as well as the addition of additional modern baseline models, we have submitted a new manuscript to the DIMVA 2020 conference as a final attempt.

Due to these difficulties, I spent a lot more time with this project than I originally intended, in total about 6.5 months. We are awaiting the response of the reviewers at the DIMVA 2020 conference, but do not intend to further pursue any substantial flow-based models.

2.2.1 Datasets and data wrangling

A vital part of this project is the quality and understanding of available datasets. I spent a considerable amount of time, both for this sub-project and in general, gathering different datasets, studying information describing them, and exploring their content to prepare suitable data for each sub-project.

2.3 Stint at BT - Stepping stone detection

In a stepping stone scenario, an attacker launches an attack not from their own computer but from intermediary hosts within an enterprise network that were previously compromised, often using an interactive relay session. A common approach in the literature to detect stepping stones is to identify correlation between two connections on a potential intermediary host. Attackers try to evade detection by inserting chaff packets and delays to make the connection appear uncorrelated. Before starting this 6-week stint, I met with my industrial supervisors where we agreed that the problem of detecting stepping stones is of relevance for them and therefore a suitable topic for my time at BT Labs

The biggest challenge for this problem is that there are no available datasets available that describe stepping stone behaviour. Due to the success of the DetGen framework, I started to implement several scenarios of interactive traffic relays using ssh-tunnels and netcat/netem for chaff and delay insertion, depicted in Fig. 3. With this, I was able to generate significant amounts of traffic with a controllable amount of noise and delay to train and assess correlation models. I furthermore implemented a state-of-the-art method for connection correlation [10] that uses a deep convolutional neural network as a benchmark to compare a future model with. The model was able to identify relayed traffic when the connections were not subject to the mentioned evasion tactics, but degraded significantly when chaff and delays were added, which makes it unsuitable for real-world deployment and does not improve the current state-of-the-art.

After more consultation with a security expert, the scope of this project has been altered, which I describe in Section 4.2

2.4 Modelling of connection setup via LSTM encoders

The distinct contextual behaviour of an applications manifests itself most clearly in the first few packets exchanged in a connection. This is the stage in a connection during which a specific client or server request is transmitted, encryption keys are exchanged, users authenticate themselves with passwords, ports for successive connections are defined, etc. This distinction goes far beyond individual traffic protocols, as even a single application usually can exert different actions which translate to different contextual behaviours in this initial negotiation phase.

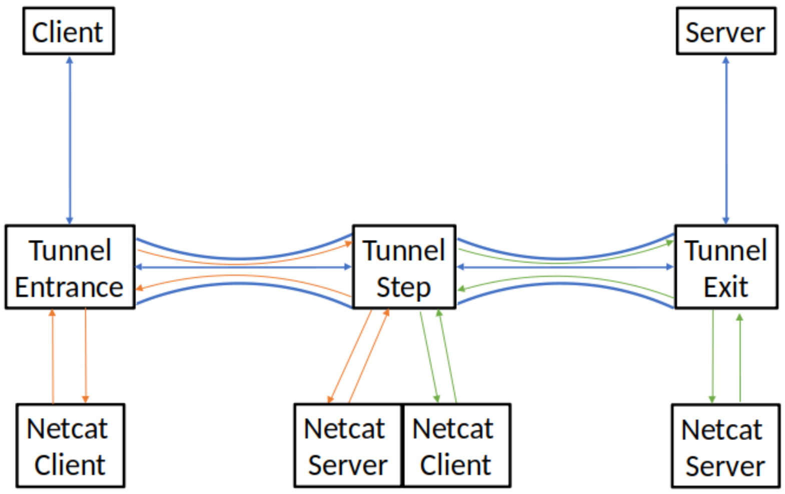


Figure 3: Implementation of the stepping stone scenario for traffic generation.

The significance of the initial negotiation phase is bolstered by two publications on the area of traffic classification that share the same conclusion and train clusters and classifiers on the metadata of the first five to ten packets to distinguish different applications [1, 2].

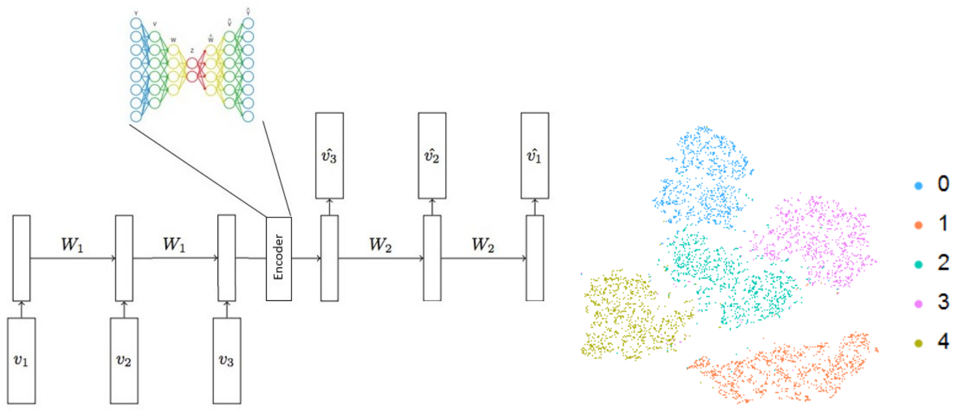
Learning precise contextual representations of this initial negotiation phase from applications running on a computer has several main benefits for this project:

- Several types of attacks exploit loopholes accessible at the negotiation stage. A precise contextual representation model could detect anomalous deviations in particular packet parameters from the learned behaviour. A simple example of such a deviation from expected behaviour would occur during a *buffer overflow attack*, which overflows an input buffer (such as for a password) to directly modify memory locations. This should be observable as a significant deviation in the corresponding packet size.

- Newly installed malware will most likely exhibit different than normal behaviour in this stage, from very different negotiation procedures to just slight deviations in the packet interarrivals.

- Malicious traffic that tries to disguise itself by using common ports can be identified as a mismatch.

In order to generate accurate representations of negotiation phases, I have implemented an LSTM-autoencoder model that can simultaneously work as an anomaly detection model as well as an embedding generator for sequence clustering, see Fig. 4b. I have tested this model on initial test-datasets, on which it achieves excellent results while being able to spot artificial anomalies accurately. I have also added a simple k-nearest neighbour clustering algorithm, however its performance as well as the ability to identify anomalies related to malicious activity will have to be evaluated on the larger CICIDS-17 dataset.



(a) (b)

Figure 4: Visualisation of the design of an LSTM Autoencoder (a), taken from https://machinelearningmastery.com/lstm-autoencoders/, and the embedding of different types of handwritten digits by an autoencoder.

2.4.1 Scope of the project for publication

Due to the difficulties described in Section 4.1 of publishing anomaly-based methods in network intrusion detection, we are currently rethinking the scope this project should take for publication. One idea is to pitch it simply as a traffic classification and clustering method, since the evaluation and applicability demonstration is more straightforward.

3 Developments in related work

Last year, I prepared an extensive literature review on network intrusion and general anomaly-based intrusion detection methods as well as datasets and data sources. Here, I will outline some recent developments in the literature related to this project.

3.0.1 Network intrusion detection

One of the most significant developments recently is the increasing adoption of artificial neural networks to build intrusion detection classifiers or anomaly-based models. LSTM-models

Some anomaly-based methods look at the anomaly of individual events, mostly isolated flows or packets. These approaches are often based on classification or clustering algorithms [6, 4]. Other approaches motivate statistical methods to identify anomalous events in network traffic [8].

3.1 Recurrent Neural Network Applications

A recurrent neural network (RNN) is a directed chain of regular feedforward network, with each individual network cell taking a vector x_k of the input sequence $\{x_1, \ldots, x_n\}$ as input. The RNN is able to retain information about the state of a sequence by also passing a value of the internal layer as additional input to the next network cell in the chain. LSTM (long short-term memory) and GRU (gated recurrent units) networks improve several shortcomings of RNNs such as the vanishing gradient problem, and their performance is generally thought to be superior.

The application of recurrent neural networks to cyber security has risen in popularity recently. Notable work includes Tiresias [?], an LSTM network for security event forecasting with tremendous accuracy. The authors use one-step forecasting of symbolic security event sequences to predict future steps of an attack in a noisy environment. The aim of Tiresias is not intrusion or anomaly detection, but attack forecasting, and the evaluation is therefore done solely in terms of event prediction accuracy of individual events. Methodologically, the authors rely on a pre-existing event vocabulary provided through the data format.

To date, we count eleven published applications of recurrent neural networks (RNNs) to network traffic for intrusion detection. The majority of these approaches relies on labelled attack data for classification.A prominent example of this comes from Kim et al. [?], who classify flow sequences based on 41 numeric input features. Anomaly-based approaches usually rely on iterative one-step ahead forecasts within sequences, with the forecasting error acting as the anomaly indicator. This is for instance done by Bontemps et al. [?], who use sequences of aggregated traffic features as input. Input features in these approaches are mostly numerical, with only [?] creating event vocabularies from flow protocols and sizes. Little exploration of the usage of stacked recurrent layers or input embeddings has been done.

Outside of network traffic, Du et al. [3] proposed DeepLog, an LSTM network to learn a system's log patterns (e.g., log key patterns and parameter values) from normal execution. The authors present a novel log parser to create a sequence of symbolic log keys, which is then also modelled using one-step forecasting. The authors achieve good detection results in regulated environments such as Hadoop with limited variety of events (e.g., 29 events in Hadoop).

To evaluate their ability to model the behaviour of a network and to identify malicious activity and network intrusions, new methods have to be tested using existing datasets of network traffic. This network should ideally contain realistic and representative benign network traffic as well as a variety of different network intrusions. However, as network traffic contains a vast amount of information about a network and its users, it is notoriously difficult to release a comprehensive dataset without infringing the privacy rights of the network users [?]. Furthermore, the identification of malicious traffic in network traces is not straightforward and often requires a significant amount of manual labelling work. Introducing malicious software into an enterprise network would be impossible for ethical and security reasons. For that reason, only about four structured datasets for network intrusion containing real-world traffic and attacks are available openly. The most recent and notable real-world datasets have been released from the Los Alamos National Laboratory (LANL) in 2015 and 2017 [5, 14], and the University of Granada (UGR) in 2016 [7]. Both datasets contain network flow traffic data from a large number of hosts collected over multiple months, giving an accurate representation of medium- to large-scale structures in benign traffic. However, the amount of attack data is small and insufficient for accurate detection rate estimation. Furthermore, packet-level data is not available for both datasets. Other real-world datasets, such as CAIDA 2016 [15] or MAWI 2000 [13], provide packet headers, but are unstructured and contain no labeled attack data at all.

To improve the lack of attack traffic in NIDS datasets, several artificially created datasets have been proposed. For this, a testbed of virtual machines is usually hosted in an enclosed environment to prevent any malicious code from spreading to other machines on other networks. To generate attack traffic, these machines are then subject to a selection of attack carried out by other machines in the environment. Benign traffic is generated using commercial traffic generators such as the *IXIA PerfectStorm tool*, or by scripting a selection of tasks for each machine. Synthetic datasets cover a smaller timeframe and contain traffic from a small number of hosts. Notable examples are the CIC-IDS 2017 dataset from the Canadian Institute for Cybersecurity [11], and the UNSW-NB 2015 dataset from the University of New South Wales [9]. Both datasets contain traffic from a variety of attacks, and are available as packet headers or as network flows with additional features crafted for machine-learning. While the benign traffic for the CIC-IDS 2017 data was generated using scripted tasks from a number of host profiles, the benign data for the UNSW-NB 2015 data is a mixture of captured real traffic from another subnet and traffic generated using a commercial traffic replicator.

We omitted the synthetic KDD-Cup 1999 and the DARPA 1998 datasets along with their derivates from the discussion as they are well-known to be outdated and contain unrealistic benign traffic, artificially high benign/attack data ratios, and artifacts stemming from communication simulations [**?**, **?**], problems which have been addressed by most modern datasets.

Container networks have recently been adopted to conduct traffic generation experiments, such by Fujdiak et al. [**?**] who use containerized web servers to collect DoS-traffic. Furthermore, significant effort has been put into the creation of large-scale virtualization frameworks to provide automatized network testbeds [**?**, **?**].

4 Future plans

4.1 Challenges with publishing in Network Anomaly Detection and ACSAC conference

The original scope of this PhD-project was to built contextual models that represent software behaviour primarily from network traffic and potentially other external data sources, for adaptive anomaly intrusion detection. Over the course of the last year, it became clear that the successful publication of anomaly-based methods network intrusion detection methods is difficult today. This realisation was reflected during our unsuccessful attempts to publish the contextual network flow model described in Section 2.2 at the ACSAC and CODASPY conferences. Despite improving detection rates and false positive rates significantly on U2R and R2L on contemporary datasets, the main criticisms were a lack of novelty in terms of intrusion detection model and a general scepticism of real-world applicability. Prior attempts publication attempts by my supervisor and collaborating researchers yielded similar responses.

This is in line with conversations I had with senior researchers at the ACSAC conferences, who highlighted that anomaly-based methods have been applied to network traffic for over 15 years without convincing results. There seems to be a general scepticism of the detection capabilities and improvements that general-purpose anomaly-detection methods can provide in real-world scenarios due to high false-positive rates, in particular for network traffic. In the context of the proposed project, this raises doubt if research question 4 can be answered adequately for a PhD-project concerning general models of network behaviour. This also made us re-evaluate the scope of the project discussed in Section 2.4.

General advice were to identify very specific applications for which taylored machine learning methods have the potential to yield good results. This is in line with the findings of Sommer and Paxson [12] that anomaly detection methods should be "keeping the scope narrow" and providing an answer for "why the particular choice promises to perform well in the intended setting, considering domain-specific properties". "If one cannot make a solid argument for the relation of the features to the attacks of interest, the resulting study risks foundering on serious flaws."

4.2 Bla

As part of my CASE PhD scholarship, I spent six weeks with my industrial sponsor at BT Labs Adastral Park in August and September 2019. Before starting this stint, I met with my industrial supervisors to define a project to work on that is both related to my PhD and is a relevant problem for BT. During this meeting, we agreed that the problem of detecting stepping stones was a suitable topic.

After finishing the stint at BT Labs, I continued to work on the traffic generation for a bit further before we had a call with Jake Hill, a security expert at BT. This call was meant to provide field knowledge to confirm and/or improve the data generation set-up. However, Jake stated that the problem of attackers launching attacks from intermediary hosts is after all not of great relevance for BT's operations. Instead, his notion of stepping stones described simple proxies relaying services (more on this in Section 4.5). In addition, further investigation suggests that the consensus in the literature is that connection correlation produces too many false positives and is not applicable in real-world scenarios.

4.2.1 Scope of the project for publication

With this information, I decided to scale this project down and produce a simple evaluation of existing connection correlation methods. Since the major contribution of this work so far is the large amount of detailed and varied data I can produce, this allows for the first time an independent assessment of existing methods. So far I have implemented and evaluated four different methods, and am in the process of finishing this project. I am currently producing a small paper from it, which I intend to publish at a smaller workshop or conference.

4.3 Submission of Stepping stone project and negotiation phase model

With work being almost finished for the stepping stone project and the negotiation phase model also being well on the way, I am planning to write each project up into a paper for publication. Since the scope for each project was scaled back a bit, I am intending to submit them to smaller venues with a higher acceptance rate, and do not intend to spent more than three months completing both. A suitable venue for both might be the 2020 Conference on Machine Learning for Cyber Security.

4.4 QUIC anomaly detection

An advice I received at the ACSAC conference from Guofei Gu, the Program Chair, was to identify very specific and novel applications of machine learning methods to intrusion detection in order to produce publications that could be accepted at renowned venues. One such applications could be to the new transport layer QUIC. QUIC seeks to improve performance of connection-oriented web-applications using TCP. It is implemented on top of UDP and implements its own TCP-like packet control mechanisms. It allows for multiplexed HTTP-connections and resolves head-of-line blocking during packet loss as well as reducing latency by minimising round-trips during connection establishment. Furthermore, much of the QUIC implementation is moved from kernel to user space, which allows continuous updates of the protocol.

According to an NGINX-spokesperson, "since the protocol is new and things like stack optimization etc. still to catch up and since it's all user-space, there is a possibility to make changes to protocol very rapidly. This generates a higher attack surface than you would have over more layered approach with TCP and http on top." Initial investigations showed that there exists no research on mitigating intrusions over the QUIC protocol.

Due to the implementation in user-space, QUIC may introduce vulnerabilities that allow remote code execution on a host, such as in *CVE-2017-15407* and *CVE-2017-15398*. The detection of such code executions by building a combined model of QUIC packet exchanges and process starts/system calls therefore seems like a promising project that provides both novelty and relevance.

In particular, I believe the following conditions would allow for interesting results:

- Operation on a host (server or client) level
- Combination of unencrypted packet packet stream and system calls/process starts corresponding
- A sequential model that applies NLP-techniques to packet content and predicts probability of process start or particular system call (which could then correspond to code execution)
- Traffic and system call/process log collection using a containerized framework.

In total, I would assign about six months work in this project to conclude.

One difficulty in this project is that there do not exist many identified vulnerabilities in the QUIC protocol yet, of which none are known to have been used in a successful attack. Therefore, it might be difficult to get enough malicious data for result validation. At this stage of my PhD, beginning a project that is not certain to be successful bears significant risk. I am therefore currently spending a small amount of time reaching out to experts to verify that data for model validation is available and that the scope of this project is of relevance before fully committing to it.

4.5 Service relay detection

As mentioned in Section 2.3, disussion with a security expert at BT Labs indicated that the detection of simple proxy servers that relay protected services to third parties from the perspective of an ISP is a relevant and promising research project. Specific characteristics of this problem are:

- Unlike the original stepping stone problem, these relay proxies operate in a simplistic fashion and do not use evasion tactics. However, content can be forwarded using a different application or protocol than in the first connection.
- Packets are sampled before observation, meaning that only every n-th packet for each connection is observed.
- Benign proxies exist that can increase false-positives.

Initial investigations that I conducted showed that there are no publications yet that address this problem in any way, which indicates the potential for novel contributions. Since there exists no description or analysis of such proxies at all, this however also means that we are reliant on BT for knowledge and guidance on the problem definition and proxy operation mode. Also since no public dataset exists, we would have to implement and generate our own dataset and feedback with BT to check if it resembles the behaviour of such proxies. At the moment it seems that a suitable start would be to focus on video relay. Necessary steps for this project are:

- Implement a docker traffic generation scenario to generate realistic data, and parse them in a sampled manner.
- start designing and implementing detection methods. As there are no evasion tactics involved, simple cumulative sum or moving window statistics with a p-value threshold seem like a good start.
- Test these methods on the data with a suitable background of independent connections as well as benign proxies. Obtaining simple background data is not challenging, however we will have to discuss how and to what extend benign proxies should be included in the background data.
- Evaluate and adjust/improve existing methods.

As we are reliant on the cooperation with BT Labs, we are not completely certain yet if this project will go ahead yet. Currently, they are retrieving data of service relay examples. We will have a further discussion with them later in April, which will bring more clarity.

4.6 Extension of Docker framework

Due to very positive feedback at the workshop as well as encouragements and suggestions to extend the framework, Robert Flood and I are working further to extend the current traffic generation framework. The overall goal is to provide a tool that enables researchers to generate large data quantities of specific services of interest in arbitrary network configurations, complemented by multi-step attack scenarios in differing executions. This would be of specific interests to researchers who apply machine-learning techniques to specific network applications for which there is not sufficient realistic public data available.

Particular goals are:

1. Extend the framework to create datasets that resemble full fledged computer network with variable topology. For this, several subtasks need to be completed:

    (a) Embed all scenarios in a *Mininet* framework in order to allow the fast inclusion of switches, routers, etc.

    (b) Build a mechanism that can generate variable or random network topologies.

    (c) Create a launch mechanism that starts and executes different traffic scenarios for a given network topology at times and locations drawn from appropriate distributions that resemble empirical network behaviour.

2. Include the collection of application logs and system call logs for each container. These will then be matched with the corresponding traffic captures and receive the same ground truth labels.

3. Generate different multi-step attack executions on a set of network topologies to capture the similarities and disimilarities between different attack techniques and tactics.

An advantage of this project is the pre-existing DetGen body, that is being extended here. This means that the timescale of the project depends on how many extensions should be implemented. It should be relatively easy to finish the project quikcly at any given time after about two months of work while having sufficient results to publish.

5 Thesis plan

In the light of the problems described in Section 4.1 that we encountered in the field of anomaly-based intrusion detection, I believe that the scope of this PhD-project has to be altered slightly. As there will not be enough material directly concerning anomaly detection, the formulation of building anomaly-detection models describing software behaviour should be relaxed to general applications of ML (not just anomaly detection) in specific areas related to software defined behaviour. This could be based of the reoccurring theme of traffic (and potentially system/program logs) generation for machine learning. The generated traffic could then be verified as valuable by the implemented intrusion detection applications.

All applications implemented so far are driven (in part) by our traffic generation:

- stepping stone detection
- traffic relay
- LSTM encoder
- QUIC anomaly detection

Overall, all these applications are concerned with software-defined fine-grained structures.

5.1 Introduction and related work

This chapter could largely draw from the existing introduction in the research proposal as well as the extensive literature survey I conducted. Updates on the scope as well as recent developments in related work would have to be added. This should take less then three weeks.

5.2 Background: Anomaly detection and challenges in security

5.3 Background: NLP and its application to security

5.4 Data sources, datasets and data generation

5.5 Traffic structures and effect of malicious behaviour to traffic structures

5.6 Anomaly detection approach and detection models

References

[1] L. Bernaille, R. Teixeira, I. Akodkenou, A. Soule, and K. Salamatian. Traffic classification on the fly. *ACM SIGCOMM Computer Communication Review*, 36(2):23–26, 2006.

[2] M. Crotti, M. Dusi, F. Gringoli, and L. Salgarelli. Traffic classification through simple statistical fingerprinting. *ACM SIGCOMM Computer Communication Review*, 37(1):5–16, 2007.

[3] M. Du, F. Li, G. Zheng, and V. Srikumar. Deeplog: Anomaly detection and diagnosis from system logs through deep learning. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, pages 1285–1298. ACM, 2017.

[4] A. Javaid, Q. Niyaz, W. Sun, and M. Alam. A deep learning approach for network intrusion detection system. In *Proceedings of the 9th EAI International Conference on Bio-inspired Information and Communications Technologies (formerly BIONETICS)*, pages 21–26. ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering), 2016.

[5] A. D. Kent. Cybersecurity Data Sources for Dynamic Network Research. In *Dynamic Networks in Cybersecurity*. Imperial College Press, June 2015.

[6] W.-C. Lin, S.-W. Ke, and C.-F. Tsai. Cann: An intrusion detection system based on combining cluster centers and nearest neighbors. *Knowledge-based systems*, 78:13–21, 2015.

[7] G. Maciá-Fernández, J. Camacho, R. Magán-Carrión, P. García-Teodoro, and R. Therón. Ugr '16: A new dataset for the evaluation of cyclostationarity-based network idss. *Computers & Security*, 73:411–424, 2018.

[8] M. V. Mahoney and P. K. Chan. Learning nonstationary models of normal network traffic for detecting novel attacks. In *Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 376–385. ACM, 2002.

[9] N. Moustafa and J. Slay. UNSW-NB15: a comprehensive data set for network intrusion detection systems (UNSW-NB15 network data set). In *2015 Military Communications and Information Systems Conference (MilCIS)*, pages 1–6, Nov. 2015. 00074.

[10] M. Nasr, A. Bahramali, and A. Houmansadr. Deepcorr: Strong flow correlation attacks on tor using deep learning. In *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security*, pages 1962–1976, 2018.

[11] I. Sharafaldin, A. Gharib, A. H. Lashkari, and A. A. Ghorbani. Towards a reliable intrusion detection benchmark dataset. *Software Networking*, 2018(1):177–200, 2018.

[12] R. Sommer and V. Paxson. Outside the Closed World: On Using Machine Learning for Network Intrusion Detection. In *2010 IEEE Symposium on Security and Privacy*, pages 305–316, May 2010.

[13] C. Sony and K. Cho. Traffic data repository at the wide project. In *Proceedings of USENIX 2000 Annual Technical Conference: FREENIX Track*, pages 263–270, 2000.

[14] M. J. M. Turcotte, A. D. Kent, and C. Hash. Unified Host and Network Data Set. *ArXiv e-prints*, Aug. 2017.

[15] C. Walsworth, E. Aben, K. Claffy, and D. Andersen. The caida ucsd anonymized internet traces 2012,", 2015.