AIMS 2018

June 4-6, 2018
Munich, Germany

Proceedings

AIMS 2018 is technically co-sponsored by:
AIMS 2018 Preface

The International Conference on Autonomous Infrastructure, Management, and Security (AIMS 2018) is a single-track event integrating regular conference paper sessions, hands-on lab, and a PhD student workshop into a highly interactive event. Within the security and network and service management community, AIMS is focused on PhD students and young researchers. One of the goals of AIMS is to provide early researchers with constructive feedback by senior scientists and give them the possibility to grow in the research community by means of targeted lab sessions on technical and educational aspects of their research activity. This focus on early-stage researchers is immediately observable in the program, which is dominated by PhD sessions, where young students present their research.

AIMS 2018 – which took place during June 4-6, 2018 in Neubiberg, Germany and was hosted by the Universität der Bundeswehr München, Germany – was the twelfth edition of a conference series on management and security aspects of distributed and autonomous systems. It followed the established tradition of unusually vivid conference series, after successful events in Zürich, Switzerland in 2017, Munich, Germany in 2016, Ghent, Belgium in 2015, Brno Czech Republic in 2014, Barcelona, Spain in 2013, Luxembourg, Luxembourg in 2012, Nancy, France in 2011, Zürich, Switzerland in 2010, Enschede, The Netherlands in 2009, Bremen, Germany in 2008, and Oslo, Norway in 2007.

AIMS 2018 was focused on the fundamental impact that the data explosion and the subsequent need for better automation will have on management and security of networks. This theme was addressed by both the technical and PhD sessions by papers focusing on next-generation network monitoring, automated threat identification and prevention, network management and resilience, and data protection. AIMS 2018 was organized as a 2-day program consisting of a lab session on the first day and a technical session and two PhD sessions on the second day.

The lab session addressed the current topic of the security of IoT devices, which are being widely adopted in multiple application domains including Industry 4.0, smart homes, smart cities, and intelligent transport. The lab presented a detailed overview of IoT devices, protocols, and their security issues and provided participants with hands-on experience in analyzing their network traffic. Additionally, AIMS 2018 featured a research keynote on “Thirty years of Incident Response: What have we learned?” by Thomas Schreck, head of Siemens CERT.

The technical program consisted of one full paper session, which addressed many of the AIMS 2018 topics, namely network monitoring, malware mitigation, and data protection. The session included three papers, which were selected after a thorough reviewing process out of seven submissions.

The PhD sessions are a venue for doctoral students to present and discuss their research ideas, and more importantly to obtain valuable feedback from the AIMS audience about their planned PhD research work. For AIMS 2018 two PhD sessions had been arranged for, focusing on threat identification and prevention, and on network management of future networks. A total of eight papers were presented and discussed. These papers were selected after a separate review process out of sixteen submissions.

These proceedings include all papers presented at AIMS 2018 as defined within the overall final program. It demonstrates the European scope of the conference, as most of the accepted papers originate from European research groups. The ratio of PhD workshop submissions to full paper submissions clearly showed that AIMS is mainly a conference for young researchers, which give them a unique opportunity to share and discuss their research.
The editors would like to thank all the people that helped make the AIMS 2018 a successful event. First and foremost, our thanks go to the authors, who submitted their papers, and to the members of the AIMS Technical Program Committee for reviewing those papers and providing highly valued feedback. Many thanks are extended to the keynote speaker, Thomas Schreck, to the lab lecturers, Abdelkader Lahmadi and Frederic Beck, and to Sofiane Lagraa, who helped to organize the lab session. The thanks are as well in order for Burkhard Stiller for preparing these proceedings and many thanks are due as well to the local organizers for handling all the logistics and hosting the AIMS 2018 event.

Finally, special thanks go to the AIMS 2018 supporter, Universität der Bundeswehr München.

June 2018

Jérôme François
Martin Drašar

AIMS 2018 TPC Co-chairs
AIMS 2018 Organizing Committee

General Chair
Gabi Dreo Rodosek, Universität der Bundeswehr München, Germany

Technical Program Co-chairs
Martin Drašar, Masaryk University, Czech Republic
Jérôme François, INRIA, France

Lab Chair
Sofiane Lagraa, University of Luxembourg, Luxembourg

Publication Chairs
Burkhard Stiller, University of Zürich, Switzerland

Steering Committee
Anna Sperotto, University of Twente, The Netherlands
Filip De Turck, Ghent University - iMinds, Belgium
Rémi Badonnel, INRIA, France
Daphné Tuncer, University College London, United Kingdom
Burkhard Stiller, University of Zürich, Switzerland
Robert Koch, University of Federal Armed Forces Munich
Pavel Celeda, Masaryk University, Czech Republic
AIMS 2018 Technical Program Committee

Abdelkader Lahmadi, University of Lorraine, France
Aiko Pras, University of Twente, The Netherlands
Alexander Clemm, Huawei, U.S.A.
Alexander Keller, IBM Global Technology Services, U.S.A.
Alva Couch, Tufts University, U.S.A.
Anandha Gopalan, Imperial College London, U.K.
Bertrand Mathieu, Orange Labs, France
Bruno Quoitin, Université de Mons, Belgium
Burkhard Stiller, University of Zurich, Switzerland
Clarissa Marquezan, Huawei Technologies, Germany
Corinna Schmitt, University of Zurich, Switzerland
Daniele Sgandurra, Royal Holloway, University of London, U.K.
David Hausheer, OVGU Magdeburg, Germany
Edmundo Monteiro, University of Coimbra, Portugal
Guillaume Doyen, UTT, France
Guy Leduc, University of Liege, Belgium
Henning Sanneck, Nokia Bell Labs, Germany
Idilio Drago, Politecnico di Torino, Italy
Isabelle Chrisment, TELECOM Nancy – Université de Lorraine, France
Jan Korenek, Brno University of Technology, Czech Republic
Jan Vykopal, Masaryk University, Czech Republic
Jérôme François, INRIA Nancy Grand Est, France
Kurt Tutschku, BTH – Belkinge Technical University, Sweden
Lisandro Zambenedetti Granville, UFRGS, Brazil
Marinos Charalambides, University College London, U.K.
Martín Barrère, Imperial College London, U.K.
Martin Zadnik, Brno University of Technology, Czech Republic
Michele Nogueira, Universidade Federal do Paraná, Brazil
Olivier Festor, INRIA Nancy - Grand Est, France
Paulo Simoes, University of Coimbra, Portugal
Pavel Celeda, Masaryk University, Czech Republic
Philippe Owezarski, LAAS-CNRS, France
Ricardo Schmidt, University of Passo Fundo, Brazil
Robert Koch, Universität der Bundeswehr München, Germany
Roberto Riggio, FBK CREATE-NET, Italy
Shingo Ata, Osaka City University, Japan
Steven Latré, University of Antwerp – imec, IDLab, Belgium
Tim Wauters, Ghent University – imec, IDLab, Belgium
Table of Content

Martin Drašar, Jérôme François:
AIMS 2018 Preface .............................................................................................................. iii
AIMS 2018 Organizing Committee .......................................................................................... v
AIMS 2018 Technical Program Committee ............................................................................. vi
AIMS 2018 Table of Content ................................................................................................... vii

PhD Session I
Olivier van der Toorn, Anna Sperotto:
Threat Identification Using Active DNS Measurements ....................................................... 1
Stanislav Spacek, Pavel Celeda:
Advanced Threat Detection Through Network Flows and Log Correlation ......................... 6
Katharina Kuhnert, Jessica Steinberger, Harald Baier:
Botnet Detection in Anonymous Networks ........................................................................ 11
Raphael Labaca Castro, Gabi Dreo Rodosek:
Black Box Attacks using Adversarial Samples against Machine Learning
Malware Classification to Improve Detection ....................................................................... 16

Full Paper Session
Tomas Cejka, Petr Velan, Jiří Havránek, Pavel Benáček:
Enhanced Flow Monitoring with P4 Generated Flexible Packet Parser ............................ 21
Mohamed Nassar, Haidar Safa:
Throttling Malware Families in 2D ..................................................................................... 33
Paulo Silva, Lukas Kencl, Edmundo Monteiro:
Data Privacy Protection - Concealing Text and Audio with a DNA-inspired Algorithm ...... 46

PhD Session II
Moritz Mueller, Roland van Rijswijk-Deij:
Towards an Independent and Resilient DNS .................................................................... 60
Godfrey Kibalya, Joan Serrat, Juan Luis Gorricho:
RAN Slicing Framework and Resource Allocation
in Multi-domain Heterogeneous Networks ..................................................................... 65
Eder John Scheid, Burkhard Stiller:
Leveraging Smart Contracts for Automatic SLA Compensation -
The Case of NFV Environments ......................................................................................... 70
Nils Rodday, Aiko Pras, Gabi Dreo Rodosek:
Towards European Network Sovereignty ........................................................................... 75

List of Authors ......................................................................................................................... 79
Threat Identification Using Active DNS Measurements

Olivier van der Toorn and Anna Sperotto
University of Twente, Enschede, The Netherlands
{o.i.vandertoorn,a.sperotto}@utwente.nl

Abstract. The DNS is a core service for the Internet. Most uses of the DNS are benign, but some are malicious. Attackers often use a DNS domain to enable an attack (e.g., DDoS attacks). Detection of these attacks often happens passively, which leads to a reactive detection of attacks. However, registering and configuring a domain takes time. We want to proactively identify malicious domains during this time. Identifying malicious domains before they are used allows to pre-emptively stop an attack. We aim to accomplish this goal by analysing active DNS measurements. Through the analysis of active DNS measurements there is a window of opportunity between the time of registration and the time of an attack to identify a threat before it becomes an attack. Active DNS measurements allows us to analyse the configuration of a domain. Using the configuration of a domain we can predict if it will be used for malicious intent. Machine Learning (ML) is often used to process large datasets, because it is efficient and dynamic. This is the reason we want to use ML for the detection of malicious domains. Because our results are predictive in nature, methodology for validation of our results need to be developed. At the time of the detection ground truth is not (yet) available.

1 Introduction

A core part of the Internet infrastructure is the Domain Name System (DNS). This system performs the translation from a domain name to an IP address. Most uses of the DNS are benign, but there are malicious uses of the DNS. This malicious activity may have a devastating impact on the Internet at large. A chief example is the Distributed Denial of Service (DDoS) attacks against the service and DNS provider Dyn [1], which made a large portion of Web sites inaccessible. However, examples like the 2016 DDoS attack against an unnamed European media organization which peaked at 363Gbps [2] demonstrate what kind of traffic abused DNS servers can generate.

Many of these attacks require a domain name. For example, as a (controlled) means of amplification (in a DDoS attack), or as a dynamic way of hosting a Command and Control (C&C) server for botnets. Research in the field of detection and mitigation has already indicated that the DNS plays a central role as source of data for security. The work in [3, 4], for example, analyzes user-generated DNS traffic to identify botnet command-and-control traffic. In practice, there exists a plethora of mechanisms to protect us from cyberthreats (e.g., firewalls,
blacklists, ACL, IDS, DDoS protection services etc.). What all these solutions have in common is that they are reactive. Threat intelligence based on DNS is frequently based on passive DNS measurements. These systems require client DNS activity before they are able to detect attacks.

We want to change the approach from reactive to pro-active threat identification. There is a window of opportunity between the registration of a domain and when it is first used in an attack. We want to identify malicious domains during this time. We aim to do so by combining active DNS measurements with Machine Learning.

2 Goal, Research Questions and Approach

The goal of this research is to develop methodology for pro-active threat identification of malicious domains through active DNS measurements. We want to detect malicious domains between registration time of a domain, and the first attack. Rather than when the attack is on-going. Because we want to predict if a domain is malicious or not, we also need to develop a method of validation.

We hypothesize that active DNS measurements allow us to do pro-active detection of malicious domains. Since attackers frequently register a DNS domain and configure it in a specific manner before an attack begins. Active DNS measurements allows us to analyse the configuration of domains, and make a prediction about the nature of the domain. To achieve our goal we have defined the following research question.

RQ_M: How can we use active DNS measurements to pro-actively identify malicious domains, and what are the benefits of such an approach?

Our approach in answering RQ_M will be part measurement-based and part experiment based. Figure 1 shows the envisioned path from start to finish. Each node represents a sub-research question, which is discussed in the following sections.

2.1 Survey the landscape

There exists a plethora of cyber-attacks, for example botnets, DDoS attacks, and worms. While we would like to detect all of them, if an attack makes no use of the DNS, our approach will not be feasible in the detection this attack. We need to research which attacks make use of the DNS, and how. To this end we have defined the following research question.

RQ_M.1: Which cyber-attacks make use of DNS and how do they use it?
To answer this research question we want to perform an extensive literature survey. Additionally, we wish to interview DNS operators to obtain a complete picture of the landscape. From this research question we are able to define a set of use-cases, attacks, which will help confine the further research.

2.2 Passive Versus Active DNS Measurements

Figure 2 shows a typical DNS setup. On the left are clients querying a resolver. This resolver, in turn, queries the authoritative name servers to answer the queries of the clients. Two types of DNS measurements are visible in this Figure. Passive DNS measurements typically measure the traffic between resolver and authoritative name servers. Whereas active DNS measurements essentially emulate the clients by performing queries themselves.

State of the art (malicious) domain identification is typically based on passive DNS measurements [5, 6]. While effective they lead to a reactive detection.

To investigate the benefits, and drawbacks, of an active approach we have defined to following research question.

*RQ_M.2: What are the strengths and weaknesses of both types of DNS measurements with respect to the attacks?*

To approach *RQ_M.2*, we will study both facets, active and passive, in detail. As a starting point for this study we aim to use Entrada [7], a passive DNS measurement system, and OpenINTEL [8], an active DNS measurement system, to evaluate the differences between the two types of measurements.

We want to study how well both approaches fare in the detection of attacks. The attacks we will evaluate stem from research question *RQ_M.1*.

2.3 Detection and validation

We want to be able to do detections of the entire DNS namespace. Therefore the detection process needs to be efficient in the processing of large amounts of data (e.g. OpenINTEL collects 2.2 billion data points from 207 million domains every day). However, since attacks are dynamic in nature [9] the detection process should also be flexible. Machine Learning (ML) is often used in (large-scale) spam detection [10, 11] which is dynamic in nature. For these reasons we aim to use ML for the detections. There are many different ML algorithms [12] we need to investigate which algorithm is suited to our detection problem.
Additionally, we face the problem of validation. Because we want to do pro-
active detection there is, at the time of the detection, no ground truth available
to validate our detections against.

To reach these goals we have defined the following sub-question.

**RQ_{M.3}: How can we perform efficient, large-scale, detections using
Machine Learning and how do we validate these detections?**

To approach this question we want to evaluate different classifier algorithms.
Subsequently, to validate the detections we need to compare our results with
well-known blacklists over a long period of time. That way we are able to assess
the quality of the results and evaluate the time advantage of our method.

### 2.4 Clustering

Finally, we want to explore if we are able to infer information about the
parties behind the domains by clustering domains together. For example, in [13]
the authors were able to cluster a CEO fraud campaign together, since the per-
petrators used the same configuration to target multiple victims. To investigate
what the benefits are of clustering approaches, such as the identification of the
parties behind the act, we have defined the following sub-question.

**RQ_{M.4}: What additional information can be obtained by clustering
similar domain-configurations together?**

The approach we will take is clustering the domains we have detected together.
And investigate the differences, and similarities, between the clustered domains.

### 3 Ethical considerations

Ethical considerations play a role throughout this project. For both types of
measurements we have to take ethical best practices [14] into account. Especially
with passive measurements, since these are in closer contact with user behaviour.
The same counts when dealing with requests to be removed from the blacklist,
these may be motivated by personal gain.

### 4 Preliminary Steps

We have already made a couple of preliminary steps in this project. In [15]
we have applied our method to identify snowshoe spam domains. With snowshoe
spam the spammer spreads the sending of spam over many hosts. Thus each
individual host leaves a shallow imprint, like a snowshoe. This makes snowshoe
spam hard to detect. Many spammers make use of Sender Policy Framework
(SPF) to make their email appear more legitimate and increase their chance
of success. SPF requires spammers to have a domain with a record for each
sending host. Thus, snowshoe spam domains have many records (e.g. more than
200 MX records). Using active DNS measurements we were able to analyse the
configuration of domains. By applying Machine Learning on this dataset we were
able to detect snowshoe spam domains more than a hundred days before they
appeared on regular blacklists. This research formed the basis for this Ph.D.
project.
Acknowledgements

This research is funded by SIDNfonds\(^1\). SIDNfonds is an independent fund on the initiative of SIDN, the registrar for ‘.nl’ domains.

References

1. Hilton, S.: Dyn Analysis Summary Of Friday October 21 Attack. https://dyn.com/blog/dyn-analysis-summary-of-friday-october-21-attack/ (October 2016)
2. Constantin, L.: Attackers use DNSSEC amplification to launch multi-vector DDoS attacks. https://www.computerworld.com/article/3097364/security/attackers-use-dnssec-amplification-to-launch-multi-vector-ddos-attacks.html (July 2016)
3. Choi, H., Lee, H., Kim, H.: BotGAD: Detecting Botnets by Capturing Group Activities in Network Traffic. In: COMSWARE. (2009)
4. Choi, H., Lee, H.: Identifying botnets by capturing group activities in DNS traffic. Computer Networks (2012)
5. Antonakakis, M., Perdisci, R., Lee, W., Vasiloglou, I., Dagon, D.: Detecting Malware Domains at the Upper DNS Hierarchy. In: Proc. of USENIX Security ’11. (2011)
6. Bilge, Leyla and Kirda, Engin and Kruegel, Christopher and Balduzzi, Marco: EXPOSURE: Finding Malicious Domains Using Passive DNS Analysis. In: NDSS 2011. (2011)
7. Maarten Wullink, Giovane C. M. Moura, Muller, M, and Cristian Hesselman: ENTRADA: a High Performance Network Traffic Data Streaming Warehouse. In: NOMS 2016. (April 2016)
8. van Rijswijk-Deij, R., Jonker, M., Sperotto, A., Pras, A.: A High-Performance, Scalable Infrastructure for Large-Scale Active DNS Measurements. IEEE JSAC 34(6) (2016)
9. Miller, R.: The Changing Face of Cyber-Attacks. Technical report, CA Technologies (2013) http://www3.ca.com/-/media/Files/whitepapers/the-changing-face-of-cyber-attacks.pdf.
10. Gudkova, D., Vergelis, M., Demidova, N., Shcherbakova, T.: Spam and phishing in Q1 2017. https://securelist.com/spam-and-phishing-in-q1-2017/78221/ (05 2017)
11. Bhowmick, A., Hazarika, S.M.: Machine Learning for E-mail Spam Filtering: Review, Techniques and Trends. CoRR (2016)
12. Brownlee, J.: A Tour of Machine Learning Algorithms. https://machinelearningmastery.com/a-tour-of-machine-learning-algorithms/ (November 2013)
13. Sperotto, A., van der Toorn, O., van Rijswijk-Deij, R.: TIDE: Threat Identification Using Active DNS Measurements. In: Proceedings of the SIGCOMM Posters and Demos. SIGCOMM Posters and Demos ’17 (2017)
14. Dietrich, S., v. d. Ham, J., Pras, A., v. R. Deij, R., Shou, D., Sperotto, A., v. Wynsberghe, A., Zuck, L.D.: Ethics in data sharing: Developing a model for best practice. In: 2014 IEEE Security and Privacy Workshops. (2014)
15. van der Toorn, O., van Rijswijk-Deij, R., Sperotto, A.: Melting the Snow: Using Active DNS Measurements to Detect Snowshoe Spam Domains. In: NOMS 2018. (apr 2018)

\(^1\) https://www.sidnfonds.nl/
Threat Detection Through Correlation of Network Flows and Logs

Stanislav Spacek\textsuperscript{1,2} and Pavel Čeleda\textsuperscript{2}

\textsuperscript{1} Faculty of Informatics, Masaryk University, Brno, Czech Republic
\textsuperscript{2} Institute of Computer Science, Masaryk University, Brno, Czech Republic
\{spaceks|celeda\}@ics.muni.cz

Abstract. A rising amount of mutually interconnected and communicating devices puts increasing demands on cybersecurity operators and their tools. With the rise of end-to-end encryption, it is becoming increasingly difficult to detect threats in network traffic. With such motivation, this Ph.D. proposal aims to find new methods for automatic detection of threats hiding in encrypted channels. The focus of the proposal is on correlating the data still available in the encrypted network flows with the data contained in the logs of network applications. Our research is in the initial phase and will contribute to a Ph.D. thesis in four years.

Keywords: intrusion detection, network flows, network logs, encrypted traffic

1 Introduction

The network traffic constantly evolves as new user applications and services are being introduced. On the one hand, they provide new possibilities for users, on the other hand, they pose new challenges for cybersecurity experts and the Network Intrusion Detection Systems (NIDS) designed for network threat detection. One of the latest challenges the NIDS must cope with is encrypted communication. We detected that approximately 60\% of the traffic generated by nearly 43 000 devices on our campus network used encryption in September 2017. This share will surely rise even higher in the future \cite{2}.

Current NIDS mostly use threat detection based on the network flow analysis \cite{12}. It is a fine strategy suitable for detection of previously unknown attacks, but as attacks evolve and patterns in network traffic change, it must be continuously adapted to remain reliable. Initially, the network flow analysis was only able to analyze network traffic based on its source, destination, timestamp and the amount of transferred data. Recently, it allows for more fine-grained analysis based on the data from the application layer \cite{3}. This way, it can identify applications and protocols in transfer and thus better distinguish between malicious and benign traffic.

Unfortunately, content analysis is not straightforward for encrypted traffic. The lack of information lost in encryption is recently approached from two directions. The first approach lies in the analysis of information that remain
unencrypted, for example, initial handshake to establish encryption parameters. The second approach encompasses the analysis of statistical data that we can measure even for encrypted traffic. The measurements include parameters like length and volume of the transmission, packet inter-arrival times and frequency of sent and received packets. Kovanen et al. provide a whole set of statistical features, which might indicate a threat [7]. The experiments with a combination of the above approaches were conducted by Anderson et al. [1]. We plan to achieve better results by adding another data source.

A different approach to security analysis of network traffic is event log monitoring. Logs are generated either on the key infrastructure elements such as the servers and routers or on the endpoints such as user devices. Some of these devices like IoT and mobile devices might be unable or unwilling to provide their logs. However, they still leave traces in the logs of the network infrastructure. Security Information and Event Management (SIEM) systems currently in use are designed specifically for threat detection by log analysis. They monitor and correlate events that are generated inside their constituency. This way, they are able to not only detect anomalous behavior, but also reconstruct a sequence of actions, for example, all actions of a user in a specific time window.

To our knowledge, intrusion detection is currently done separately in network flows and logs. Gu et al. [5] combined both approaches for the purpose of botnet detection. However, logs and flows were analyzed separately to achieve better detection accuracy. In this Ph.D. proposal, we intend to use the logs of network applications as an additional information input along the network flow data. When appropriately correlated, this new base of data will provide new insight into network traffic and extend the range of detectable attacks. Specifically, we expect that including logs will improve detection rate for attacks in encrypted traffic.

2 Research Questions

The main objective of the proposed research is to detect advanced threats in evolving network traffic. We formulated the following research questions that need to be answered in order to achieve our goal.

1. **How can we ameliorate the network flow analysis with logs to detect threats in a constantly evolving environment?**
   The monitoring of network flows provides insight into network events from a different perspective than the monitoring of network logs. We plan to investigate possibilities of correlating information contained in both network flows and logs so that a common base of data can be established. The parameters we identified for the correlation so far are the timestamp and source and destination addresses that can help to identify specific service and its corresponding log. We expect that when starting from this extended base, detection methods can achieve higher accuracy and detect threats that currently remain hidden.
2. **How can the network flow and log correlation improve the threat detection rates in encrypted traffic?**

The content analysis and the deep packet inspection is not possible for encrypted communication, but information can still be gained from the unencrypted handshakes and statistical analysis. For example, the SSL/TLS handshake may be extracted from a flow of a client-server communication and correlated with the corresponding server log based on the client IP address and timestamp. The handshake then provides list of available cyphers and the log provides the service name and the actions executed by the client, thus extending the base of data available for analysis.

3  **Proposed Approach**

3.1 **Stream Representation of Logs and Flows**

There are many different ways to store logs, and log entries themselves differ from application to application as they usually combine runtime variables with human-readable text. Therefore, the logs must be first transformed into a unified form and then directed to central storage where further analysis takes place. The unified form must contain all the information from the original log entry. It must be processable automatically and, should be similar to the form in which network flows are processed.

A log processing model is naturally represented by the event-driven architecture [4]. When the event-driven architecture is applied, a log entry corresponds to an event entity. The application that generates network logs fits the role of an event producer, and the central log storage acts as an event consumer.

The event entity unifies log entries generated by diverse log producers through the use of parameters. The parameters are either generic, for example the timestamp and source, or producer-specific. The producer-specific parameters differ by a producer and also by log entry type. For example, an SSH log producer may generate authentication success, failure, disconnection and other messages that each have a different set of specific parameters. Nevertheless, the result is a unified data structure that represents various log entries. It can be automatically processed and does not omit any information contained in the original log entry.

Typically, network applications generate a large number of log events. These events may be grouped into streams by the originating network application or by the actual device that was the source. This allows specifying the scope of monitoring either locally or on the whole network. The possibilities of event grouping into streams were researched by Tovarňák [11].

The stream-based processing of network flows is possible, as shown by Jirsík et al. [6]. The log event stream can be correlated to the flow stream based on similar properties that will be the subject of our research. A possible approach is through timing-based correlation since timestamp is a parameter present both in flows and log events. However, time synchronization is rarely accurate over the network so a tolerance scope must be defined. While a large tolerance scope will lead to an inaccurate association of logs to flows, a too small tolerance scope will
struggle to associate any logs to flows. This is also an issue that we plan to focus on in our research, along with exploring other parameters for correlation.

3.2 Threat Detection in Encrypted Traffic

The challenge of the encryption poses for threat detection might be demonstrated on botnets. Botnets often use domain names instead of IP addresses to communicate [9]. In network flows, the domain name might be learned either from the Server Name Indication (SNI) parameter of application layer or the DNS communication. Currently, the SNI is unencrypted in SSL/TLS communication. However, recent drafts of TLS 1.3 propose encrypting the SNI to improve communication privacy [10], and DNS communication might also be encrypted by DNSSEC. Adding logs to flow analysis helps to resolve this issue. The domain name can be obtained by correlating the communication flows with the DNS log.

Most of the current threat detection methods for encrypted traffic use machine learning techniques, particularly clustering algorithms, to identify anomalies in the network flow [13]. The k-means and k-nearest neighbor algorithms are the most and second most used one respectively. The popularity of k-means is due to its variability, which allows it to be fine-tuned for various purposes. The algorithm variability is an important property since the detection method must meet several criteria. First, it must be fast enough to process a huge amount of data supplied by the monitoring infrastructure. Second, it must be accurate enough to detect as many attacks as possible. It will pose a challenge to balance speed and accuracy, as both may prove mutually exclusive.

3.3 Testing Methodology

The most prevalent public datasets currently used for detection method testing are DARPA98 and KDD99, released in 1998 and 1999 respectively [12], despite the fact that the limitations of the DARPA dataset were extensively covered by McHugh et al. [8]. Since network traffic changed drastically during last 20 years, a newer dataset will be used to better represent the current environment. The testing will be done by injecting manually labeled malware samples into background traffic and logs captured on a real network. This approach allows the creation of different datasets by changing the background traffic and belittles the issue of overfitting. We will share the samples to ensure results reproducibility in the future.

4 Conclusion

In this research, we aim to expand the base of data used by current detection methods to discover threats in network traffic. We consider logs as an information-rich source that, when properly correlated with network flow data, will provide new insight into the network traffic. We expect that our research will bring following contributions. Firstly, we will define new methods for log and flow correlation. Secondly, we will use the correlated data to improve NIDS threat
Threat Detection Through Correlation of Network Flows and Logs

detection rates. Lastly, our testing datasets will be available for future detection methods verification.

Acknowledgement

This research was supported by the Security Research Programme of the Czech Republic 2015 - 2020 (BV III / 1 VS) granted by the Ministry of the Interior of the Czech Republic under No. VI20172020070 Research of Tools for Cyber Situation Awareness and Decision Support of CSIRT Teams in the Protection of Critical Infrastructure.

References

1. Anderson, B., McGrew, D.: Identifying Encrypted Malware Traffic with Contextual Flow Data. In: Proceedings of the 2016 ACM Workshop on Artificial Intelligence and Security. pp. 35–46. AISec ’16, ACM, New York, NY, USA (2016)
2. Cisco Systems: Encrypted Traffic Analysis. Whitepaper (January 2018), https://www.cisco.com/c/dam/en/us/solutions/collateral/enterprise-networks/enterprise-network-security/nb-09-encrypted-traffic-analysis-whitepaper.pdf, accessed in April 2018.
3. Claise, B., Trammell, B., Aitken, P.: Specification of the IP Flow Information Export (IPFIX) Protocol for the Exchange of Flow Information. RFC (September 2013), https://tools.ietf.org/html/rfc7011, accessed in April 2018.
4. Etzion, O.: Event Processing in Action. Manning, Greenwich Conn (2011)
5. Gu, G., Perdisci, R., Zhang, J., Lee, W., et al.: BotMiner: Clustering Analysis of Network Traffic for Protocol-and Structure-Independent Botnet Detection. In: USENIX security symposium. vol. 5, pp. 139–154 (2008)
6. Jirsik, T., Cermák, M., Tovarnák, D., Čeleda, P.: Toward Stream-Based IP Flow Analysis. IEEE Communications Magazine 55(7), 70–76 (2017)
7. Kovanen, T., David, G., Hämäläinen, T.: Survey: Intrusion Detection Systems in Encrypted Traffic. In: Internet of Things, Smart Spaces, and Next Generation Networks and Systems, pp. 281–293. Springer (2016)
8. McHugh, J.: Testing Intrusion Detection Systems: A Critique of the 1998 and 1999 DARPA Intrusion Detection System Evaluations as Performed by Lincoln Laboratory. ACM Transactions on Information and System Security 3(4), 262–294 (nov 2000)
9. Nazario, J., Holz, T.: As the net churns: Fast-flux botnet observations. In: Malicious and Unwanted Software, 2008. MALWARE 2008. 3rd International Conference on. pp. 24–31. IEEE (2008)
10. Rescorla, E., Huitsem, C.: SNI Encryption in TLS Through Tunneling. RFC Draft (March 2018), https://tools.ietf.org/html/draft-ietf-tls-sni-encryption-02, accessed in April 2018.
11. Tovarnák, D.: Normalization of Unstructured Log Data into Streams of Structured Event Objects. Dissertation thesis, Masaryk University, Faculty of Informatics, Brno (2018), https://ia.muni.cz/th/rtfsq/12. Umer, M.F., Sher, M., Bi, Y.: Flow-based Intrusion Detection: Techniques and Challenges. Computers & Security 70, 238–254 (2017)
13. Velan, P., Cermák, M., Čeleda, P., Drašar, M.: A Survey of Methods for Encrypted Traffic Classification and Analysis. International Journal of Network Management 25(5), 355–374 (2015)
Botnet detection in anonymous networks

Katharina Kuhnert*, Jessica Steinberger*,† and Harald Baier*

* da/sec - Biometrics and Internet Security Research Group, Hochschule Darmstadt, Darmstadt, Germany
{Katharina.Kuhnert}@crisp-da.de
{Jessica.Steinberger,Harald.Baier}@h-da.de
† Design and Analysis of Communication Systems (DACS) University of Twente, Enschede, The Netherlands
{J.Steinberger}@utwente.nl

Abstract. Botnets are a major threat to the Internet landscape and have been responsible for large scale distributed attacks on online services. To make take down measures more difficult, Botnet operators started to incorporate anonymous networks into their software to protect their users and their Botnets. Although privacy and anonymity is an important goal of such services, cyber criminals also benefit from these properties, hiding their identity and location in the shadow of anonymity. This work investigates Botnets in anonymous networks and proposes mitigation strategies against such threat. This research is still in its initial state and is planned to result in a Ph.D. thesis.

Keywords: Botnet, detection, anonymous networks, Tor, I2P.

1 Introduction

In recent years, research of anonymous networks gained increasing interest as more and more people, businesses and governmental agencies worldwide make use of them to protect their identity and communication. Online censorship prohibits users from accessing services and information on the Internet [1], and projects like Tor and I2P are vital contributions to allow individuals to communicate without risking their privacy [2].

However, the use of such anonymization networks by cyber criminals also became more popular and their malicious activities within these networks affect service operators and legitimate users. One approach to mitigate the effects of malicious activities within anonymous networks currently relies on an entire block of the use of anonymous networks and thus any hidden service [3,4].

Moreover, Botnets have evolved to a major threat to the Internet landscape causing damage by cyber criminals launching Distributed Denial of Service (DDoS) attacks, stealing data or distributing spam [5]. Within the shadow of anonymity and confidentiality, Botnet operators have started to move their communication and recruitment of Botnet-zombies to anonymous networks and thus make it difficult to take down [6,7]. Blocking specific malicious services such as
Botnet controllers hosted on hidden services in the dark web, is difficult to perform and the identification of a Botnet target still remains a challenge [5,8,9,10]. By using anonymous and encrypted communication channels, detecting and blocking Botnet activity within anonymous networks is still in its initial phase [8]. Existing take down measures rely on identifying the Command & Control (C&C) servers to block Botnet traffic. With the increase of Botnet activity within anonymous networks, detection methods are required to perform mitigation [4].

The aim of this work is to provide an extensive characterization of Botnet activities within anonymous networks and to propose mitigation solutions against Botnets using these networks. In addition, this research will contribute to the understanding of what anonymous network services are used by Botnets and if a single Botnet uses multiple anonymous networks to obfuscate its communication.

2 Research Goal and Questions

The overall research goal of this work is to propose methods against the rising threat of Botnets within anonymous networks. To achieve this goal, we split up the overall research goal and investigate four research questions as depicted in Figure 1.

Traditionally, Botnets have been controlled via protocols like Internet Relay Chat (IRC) [11]. These protocols are used by Botnet operators to hide their communication, avoid take down measures and obfuscate their identity [10]. We investigate the use cases of Botnet operators using anonymous networks and its services. We also analyze how Botnet operators integrate such networks into their applications. Thus, the first research question is as follows:

**RQ1: Do Botnet operators use anonymous services and if yes, how?**

Communication in anonymous networks is routed in such way which makes it difficult to identify the end-to-end communication partners [12,10]. Due to the encryption of all network traffic within an anonymous network, it is not easily possible to gain insight into transmitted data [10]. To gain an understanding of Botnet communication, we will characterize Botnet traffic within anonymous
networks and services. As a result of the characterization, we will derive features that serve as input parameters to build automatic classifiers. Thus, the second research question is as follows:

**RQ2: What are the characteristics of Botnet traffic using anonymous networks?**

Besides the development of a classification model, the detection of Botnet activities within anonymous networks is required to propose mitigation strategies. Traditional Botnet C&C traffic detection methods are using honeypots, active and passive traffic analysis techniques [10] and are based on the use of state of the art algorithms [13,14,15,16]. Although network traffic within anonymous networks is encrypted, behavior of Botnets can be detected with the aforementioned approaches by using their characteristics [14]. Thus, the third research question is as follows:

**RQ3: How can Botnet traffic in anonymous networks be detected?**

To make detection and take down measures more difficult, we assume that Botnet operators make use of multiple anonymous networks and services for a particular Botnet. Therefore, we investigate if Botnet operators distribute their anonymous communication over several anonymous networks. In case distributed anonymous Botnet communication is used, we analyze its benefits and how the Botnet traffic is distributed. Thus, the fourth research question is as follows:

**RQ4: Does a Botnet use multiple anonymous networks and if yes, how?**

3 Proposed methodology

In this Section, we present the approach of this work to answer the research questions.

3.1 Analysis of Botnets in anonymous networks

To answer RQ1, we perform a systematic literature review according to the well-established review methodology described by the Cochrane Community [17]. This systematic review results in an overview presenting Botnets that make use of well-known anonymous networks and its service. In particular, we focus on the most famous anonymous networks Tor and I2P. Furthermore, the systematic review results in an overview presenting the current state of the art Botnet detection methods in context of anonymous networks.

In addition to the systematic literature review, we find evidence of how Botnets use anonymous networks by studying source code of Botnets. Source code of Botnets have become available on the Internet and can be studied by researchers [18,19]. Such implementation details will give insight if and how Botnet communication is actually made by using anonymous networks. The contribution of the results of RQ1 will be an extensive insight of Botnets in anonymous networks and how they communicate.
3.2 Collect traffic data and characterize anonymous Botnet traffic

An important part of this work are measurements of real-world traffic within anonymous networks. Therefore, we deploy measurement nodes within the anonymous networks to collect traffic using OnionScan [20]. In addition, we will create a testbed for anonymous networks using the available Botnet software to create fingerprints of anonymous Botnet traffic. The recorded data allows the classification and detection of Botnet traffic to answer RQ2.

3.3 Create a model to detect Botnet traffic in anonymous networks

The results of the Botnet traffic characteristics serve as input parameters to design a model that distinct between Botnet traffic and non-Botnet traffic in anonymous networks. We create a method for Botnet detection based on state of the art algorithms [13,14,15,16]. To evaluate and optimize our approach, we first test our classifier within our test environment and second, deploy our detection algorithms on real-world anonymous networks. The results of this approach will answer RQ3.

3.4 Investigate in multiple anonymous networks

In this work, we have presented the use case of Botnets hiding their presence in anonymous networks. Using the results of RQ1 to RQ3, we investigate if a single Botnet uses multiple anonymous networks to hide their communication. We attempt to find correlations of Botnet traffic within multiple anonymous networks to answer RQ4.

4 Early results and Final Considerations

Early results of our research revealed that Botnets are using anonymous networks to hide their communication [18,19]. In addition, an initial web search identified the availability of Botnet source code to study Botnet communication and evaluate our detection model. This research is still in its initial phase and the main goal of this work, as described previously, must be achieved within a period of four years, as part of a Ph.D thesis.

Acknowledgment. This work was partly supported by the German Federal Ministry of Education and Research (BMBF), the Hessen State Ministry for Higher Education, Research and the Arts (HMWK) within the Center for Research in Security and Privacy (CRISP) and by the Netherlands Organisation for Scientific Research (NWO) Distributed Denial-of-Service Defense: Protecting Schools and other public organizations (D3) Project.
References

1. Aceto, G., et al.: Analyzing internet censorship in pakistan. In: IEEE 2nd International Forum on Research and Technologies for Society and Industry Leveraging a better tomorrow. (Sept 2016)
2. Cuthbertson, A.: Iran internet censorship forces protesters to turn to dark web. http://www.newsweek.com/iran-internet-censorship-sees-protesters-turn-dark-web-772182 Accessed: 3. April 2018.
3. Prince, M.: The trouble with tor. https://blog.cloudflare.com/the-trouble-with-tor (2016) Accessed: 31. March 2018.
4. Hopper, N.: Challenges in protecting tor hidden services from botnet abuse. In: Financial Cryptography and Data Security, Springer Berlin Heidelberg (2014)
5. Spamhaus Malware Labs: Spamhaus botnet summary 2016. https://www.spamhaus.org/news/article/733/spamhaus-botnet-summary-2016 (2017) Accessed: 8. April 2018.
6. Blog, G.D.S.: Botnet command server hidden in tor. https://www.gdatasoftware.com/blog/2012/09/24033-botnet-command-server-hidden-in-tor Accessed: 30. March 2018.
7. Paganini, P.: Botnet, pro & cons of using tor networks. http://securityaffairs.co/wordpress/8678/cyber-crime/botnet-pro-cons-of-using-tor-networks.html Accessed: 30. March 2018.
8. Spamhaus Malware Labs: Spamhaus botnet threat report 2017. https://www.spamhaus.org/news/article/772/spamhaus-botnet-threat-report-2017 (2018) Accessed: 8. April 2018.
9. Yang, Y., Leung, C.: Botnets drilling away privacy infrastructure. CoRR abs/1512.06447 (2015)
10. Dharna, P.S.: Analysis and detection of botnets and encrypted tunnels. International Journal of Computer Science Trends and Technology (2017)
11. Sanatinia, A., Noubir, G.: Onionbots: Subverting privacy infrastructure for cyber attacks. In: 45th Annual IEEE/IFIP International Conference on Dependable Systems and Networks. (June 2015)
12. Koch, R., et al.: How anonymous is the tor network? a long-term black-box investigation. Computer (Mar 2016)
13. Asha, S., et al.: Analysis on botnet detection techniques. In: International Conference on Research Advances in Integrated Navigation Systems. (May 2016)
14. Miller, S., Busby-Earle, C.: The role of machine learning in botnet detection. In: 11th International Conference for Internet Technology and Secured Transactions. (Dec 2016)
15. Su, Y.H., et al.: The forward-backward string: A new robust feature for botnet detection. In: IEEE Conference on Dependable and Secure Computing. (Aug 2017)
16. Kapre, A., Padmavathi, B.: Adaptive behaviour pattern based botnet detection using traffic analysis and flow interaval. In: International conference of Electronics, Communication and Aerospace Technology. (April 2017)
17. Higgins, J.P., Greens, S.: Cochrane handbook for systematic reviews of interventions. http://handbook-5-1.cochrane.org/ Accessed: 08. April 2018.
18. WhitePacket: The open tor botnet (zib). https://github.com/whitepacket/ZIB-trojan Accessed: 6. April 2018.
19. touyachrist: Zeus evolution. https://github.com/touyachrist/evo-zeus Accessed: 6. April 2018.
20. OnionScan: Onionscan: Investigating the dark web. https://onionscan.org (2018) Accessed: 16. May 2018.
Black box attacks using adversarial samples against Machine Learning malware classification to improve detection

Raphael Labaca Castro and Gabi Dreo Rodosek
Department of Computer Science, CODE
Universität der Bundeswehr München, Germany
{raphael.labaca, gabi.dreo}@unibw.de

Abstract. Machine learning algorithms are every day more accurate and are getting ready to be deployed ubiquitously. In fact, many technologies have been using them for years such as credit card fraud predictions and spam detection whilst others are starting to evaluate implementations like autonomous vehicles and different applications in IT security. However, the accuracy of these algorithms has prevailed over its security implication and hence the latter was not a priority when improving this technology. Therefore, adversarial learning is important in general to ensure AI safety because it forces the model to act in unexpected ways, which allows to observe its behavior under worst case input. The goal of researching adversarial machine learning on malware classifiers is not only to predict how a bad actor could cause security issues through malware misclassification but also to generate adversarial samples that can help retrain these models to improve its detection.

Keywords: Adversarial learning · Malware classifier · Machine learning

1 Introduction

Machine learning per se is nothing new in malware detection. Some companies have been working with neural networks and behavioral approaches for over a decade [1] in the security industry. The technology has been improving ever since and nowadays is a popular approach to detect a big part of unseen malware as these algorithms, when properly trained, are able to generalize and classify a portable executable (PE) sample between benign or malicious. Implementations to detect malicious executables also date back since early 2000’s, when Schultz et al. created a dataset and its labels with the help of a McAfee antivirus [2] and extracted the list of DLLs, DLL function calls and strings as features. Later in 2004, Kolter et al. implemented an approach [3] taking two datasets of clean and malicious executables of similar size and encoded them using millions of byte level n-grams as features. Further approaches focused on extracting only seven format specific features to predict malicious samples [4] and afterwards to categorize PE samples based on functions of their payloads. However, most of the
efforts focused on improving performance by measuring different metrics such as accuracy or area under the ROC curve as well as keeping false positive and false negative rates low. That means, most classifiers were only measured by its ability to tell apart malicious from benign without considering that attackers might be able to target the classifier forcing it, for instance, to misclassify samples. In other words to increase its false-positive rate using adversarial techniques. The rest of this paper is structured as follows: section 2 is the background and current state-of-the-art, section 3 a problem description followed by section 4 where we present our approach and section 5 the preliminary results. Ultimately, final considerations are given in section 6.

2 Background

Early research on adversarial learning has been conducted analyzing different attacks on an IDS learning algorithm that could cause it to misclassify events [5]. Additional contributions by Szegedy et al. focused on deep neural networks for image classification. They proposed to inject perturbations to images from some datasets, such as ImageNet [6], that were imperceptible for the human eye but mislead the model to classify the images incorrectly [7]. A gradient-based approach was proposed by Goodfellow et al. and focused on neural networks’ linearity [8]. Further research by Papernot et al., on adversarial image classification in deep neural networks introduced a Jacobian Matrix that provided high evasion results with a low rate of modification of the input features [9]. Though, in order to identify which features have a stronger impact on classification, the attacker needs knowledge of the model’s architecture. Similarly, adversarial examples with full knowledge of the model and access to the parameters were implemented on trained feed forward neural networks in order to deceive an Android malware classifier [10].

Malware is by nature a different problem than image classification as the input is rather binary than continuous and perturbations, which are invisible to the human eye, are no longer relevant. Xu et al., proposed a method to search for PDF adversarial samples by manipulating API calls from PE files and probing them against two PDF malware classifiers [11] by leveraging its score. A more recent approach from Hu et al., proposed a generative adversarial network (GAN) [12]. The idea is to create a substitute model that fits the black box classifier and a second model called generative network, which is trained to create adversarial samples and hence minimize detection rates on the substitute model [13]. Then it performs cross-evasion from the model to the actual black box classifier and reports perfect evasion for the samples analyzed. Similar to previous approaches, it requires to have deep knowledge from the target model. Recently, Anderson et al. proposed to target a more realistic scenario, where the attacker does not have such a knowledge and hence it is only able to probe the classifier in order to receive a binary output whether a sample is malicious or benign [14]. On the other hand, given the complexity of the scenario the success rates reported are modest, around 16%.
3 Problem description

The aim of the research is to use reinforcement learning (RL) to inject perturbations on malicious samples in order to deceive a malware classifier. The intention is to probe a static PE antivirus with perturbed adversarial samples and analyze its behavior under worst case input until it misclassifies a malicious sample as benign or a defined discrete number of stages is reached. Afterwards, these adversarial samples will be used to retrain the classifier in order to improve detection against adversarial techniques. The problem takes the following steps:

- Agent observes state $s_t$ at discrete time step $t$: $s_t \in S$
- Learning agent selects action $a_t$ at step $t$: $a_t \in A(s_t)$
- Agent receives reward $r_t = r(s_t, a_t)$ and next state $s_{t+1} = \delta(s_t, a_t)$
- Updates policy $\pi$ at step $t$: $\pi_t(s, a)

![Fig. 1. Representation of the reinforcement learning problem.]

To each policy a value can be assigned $V^\pi(s_t) = \sum_{i=0}^{\infty} \gamma^i r_{t+1}$, where $V^\pi(s_t)$ is the cumulative value that an agent receives by following a policy $\pi$ and $\gamma$ is a discount factor for future rewards. Therefore, the agent’s optimal policy is defined as:

$$\pi^* = \arg \max_{\pi} V^\pi(s), (\forall s)$$

The states will be mapped using static features that include, for example, PE headers and sections, imports and exports, resources. The actions need to be defined carefully focusing on preserving the sample’s functionality. Thus, we will only add features, and not remove, mainly because we should not get higher misclassification rates at the expense of losing functionality.

In summary, we have identified the following research questions, which outline the main aspects of the research:

1. At which extent can an attacker create automatic perturbations on malware samples to avoid detection without affecting the malware’s functionality?
2. How successful can be an attacker in transferring results from one classifier to another?
3. How to protect classifiers against adversarial attacks?
4. How much adversarial input does a classifier need to improve its robustness against adversarial samples?

4 Proposed approach

The goal of this research is to extensively analyze static features and actions to perform on adversarial samples in order to misclassify malicious PEs as benign by using reinforcement learning without knowledge of the target model. Then, feed the classifier back with these new malware mutations to analyze how detection techniques can be improved. Current research usually approaches attacks with some knowledge of the model, which is more a theoretical case than a real scenario. On the other hand, research that indeed targets black box models are on its early stages and hence success rates are still low.

Research question 1 aims to understand modifications on a PE sample without affecting its structure. Unlike image classification, keeping integrity is one of the biggest challenges in adversarial research on malware samples. Therefore, changes need to be subtle to avoid affecting internal functionality but strong enough to pass the classifier.

Research question 2 focuses on assess how cross-evasion from target model to further classifiers or products would work. Given the fact that an attack will adjust to bypass a target model, how likely it is to use the same malicious mutations across different classifiers successfully.

Research question 3 is about remediation. Adversarial samples will prove that these models are not reliable under worst case input but how to make them safer still remains an open issue. We expect to address whether retraining using adversarial samples is effective against these threats or hardening machine learning antimalware engines might be a better option.

Finally, research question 4 aims to evaluate how much would the agent need to reinforce its policy until the model is effectively deceived or a certain number of iterations without modifying the PE structure is achieved.

5 Preliminary results

We started by collecting data and testing perturbations that can be successfully integrated into specific samples and do not affect its functionality whilst it gets misclassified by scanners. One way to apply the modifications is to use a library that provides representations of PE files into raw data like LIEF (Library to Instrument Executable Formats). It consists of a cross platform library, which allows to parse and modify PE files. Once the file is rendered into bytes, manipulations can be performed and later on the new sample compiled. Initial testing showed that the library might return errors for some samples, which may lead to apply a pre-filtering step on the input data as well as to analyze further libraries.
to parse PE files. Additionally, we are testing different RL frameworks trying to find the best solution to implement perturbations efficiently.

6 Final considerations

Adversarial examples are important to help us be able to trust machine learning predictions. Even without the presence of a bad actor adversarial samples can provide a better understanding about a classifier when unexpected input is given. Although our approach is to understand behavior under pressure despite of whether an adversary is able to replicate an attack today, we will focus on keeping the scenario as realistic as possible as this could be an issue in the future.

Unlike adversarial attacks, defending a model is much more complex and therefore it will probably be an additional challenge to our approach. However, we expect a better understanding by researching both offensive and defensive techniques combined.

References

1. Machine Learning in anti-malware companies. Last accessed 31 Mar 2018. https://www.welivesecurity.com/wp-content/uploads/2017/08/NextGen_ML.pdf
2. Schultz, Matthew G., et al. Data mining methods for detection of new malicious executables. Security and Privacy, 2001. S&P. Journal Proceedings. IEEE Symposium (pp. 38-49). IEEE (2001).
3. Kolter, J. Z., & Maloof, M. A. Learning to detect malicious executables in the wild. Journal Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining (pp. 470-478). ACM (2004).
4. Shafiq, M. Zubair, et al. A framework for efficient mining of structural information to detect zero-day malicious portable executables. Technical Report, TR-nexGINRC-2009-21, available at http://www.nexginrc.org/papers/tr21-zubair.pdf (2009).
5. Barreno, Marco, et al. Can machine learning be secure? Proceedings of the ACM Symposium on Information, computer and communications security. ACM (2006).
6. Deng, Jia, et al. Imagenet: A large-scale hierarchical image database. Computer Vision and Pattern Recognition. CVPR. IEEE Conference on. IEEE (2009).
7. Szegedy, Christian, et al. Intriguing properties of neural networks. arXiv preprint arXiv:1312.6199 (2013).
8. Goodfellow, Ian J., Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. arXiv preprint arXiv:1412.6572 (2014).
9. Papernot, Nicolas, et al. The limitations of deep learning in adversarial settings. Security and Privacy (EuroS&P). IEEE European Symposium on. IEEE (2016).
10. Grosse, Kathrin, et al. Adversarial perturbations against deep neural networks for malware classification. arXiv preprint arXiv:1606.04435 (2016).
11. Xu, Weilin, Yanjun Qi, and David Evans. Automatically evading classifiers. Proceedings of the Network and Distributed Systems Symposium (2016).
12. Goodfellow, Ian, et al. Generative adversarial nets. Advances in neural information processing systems. (2014).
13. Hu, Weiwei, and Ying Tan. Generating adversarial malware examples for black-box attacks based on GAN. arXiv preprint arXiv:1702.05983 (2017).
14. Anderson, Hyrum S., et al. Learning to Evade Static PE Machine Learning Malware Models via Reinforcement Learning. arXiv preprint arXiv:1801.08917 (2018).
Enhanced Flow Monitoring with P4 Generated Flexible Packet Parser

Tomas Cejka¹, Petr Velan¹, Jiri Havranek², and Pavel Benacek¹

¹ CESNET, a.l.e.
Zikova 4, 160 00 Prague 6, Czech Republic
{cejkat,petr.velan,benacek}@cesnet.cz

² CTU in Prague, FIT
Thakurova 9, 160 00 Prague 6, Czech Republic
{havraj6}@fit.cvut.cz

Abstract. Passive network flow monitoring provides visibility into network traffic. It is necessary for many applications such as accounting, network management, and security. As its origins are in packet switching and routing devices, the common flow exporter implementations process only necessary packet headers. Link layer protocols are often skipped, and only the first network and transport layer headers are used to construct flow records. However, the network traffic is gradually becoming much more complex as new protocols are being used in practice. We present a novel multi-layer flow monitoring approach that handles complex protocol encapsulation. To process packets with an arbitrary number of protocols, we have created a new packet parser based on the P4 language, which is easily extensible and widely used in SDN networks. We argue that the new multi-layer flow monitoring approach provides more precise and detailed statistics about the traffic of overlay networks at a backbone level.

1 Introduction

Traffic monitoring is necessary for many daily objectives such as accounting, operational awareness, or network security. Without enough network visibility, it is very difficult to detect network-wide malicious behavior and breach attempts. Flow monitoring has been used for this purpose for more than two decades, and significant research effort has been invested to attack detection based on flow data [24].

Flow records, which contain information about communication between two network devices, are exported by monitoring probes (flow exporters) and stored (and processed) by flow collectors. Each flow record is usually identified by an n-tuple consisting of timestamps, network addresses, transport protocol, and transport ports. The IP Flow Information eXport (IPFIX) standard [8] describes not only the protocol for flow export from the probes to the collectors but the flow monitoring architecture as well [23]. A straightforward approach, and also a commonly used one, is to get the flow record identifiers from the first visible
network and transport layers. The rest of packets is assumed to be a payload, which is aggregated to just a number of bytes.

In practice, low-level network infrastructure does not change fast enough as the new technologies evolve. As a result, we can see multiple network layer protocols encapsulated in a single packet. The reason is simple, the current infrastructure and logical network are used to carry a new overlay infrastructure. The aim of this paper is to provide more detailed information about such encapsulated traffic with the (almost) same flow processing tools so that it is possible to use existing analysis and detection methods.

Besides monitoring highly encapsulated traffic, we also deal with a lack of flexibility of the current flow exporters. The set of supported protocols and, in addition, the set of extracted header fields is fixed and hard-coded in the source codes of flow exporters. Usually, the functionality and the range of provided information might be extended by external plugins (linkable objects); however, we show an entirely different approach which uses the Programming Protocol-independent Packet Processors (P4) language to construct a highly flexible flow exporter. The presented flow exporter provides flexibility of P4 and also allows for deep packet inspection and traversing multiple encapsulated layers of the packets.

This paper provides a novel approach to monitoring of a network traffic with multiple encapsulated protocols. Firstly, we present a generic parser based on the P4 language that is capable of easy extension of the supported protocols at the input of the flow exporter and extension of information contained in flow records. Secondly, we use this parser to construct a flow exporter that can process an arbitrary number of encapsulated layers. The new flow exporter creates a tree of flow records so that each encapsulated layer can be analyzed with minimal or no change to existing flow collectors and tools for analysis.

The rest of the paper is structured as follows. Sec. 2 presents related work in the area of flow exporters and P4 described network devices and explains the difference between the existing approaches and our approach. Sec. 3 describes usage of P4 for definition of the supported protocols, and construction of automatically generated flexible packet parser capable of recursive extraction of the encapsulated protocols. Sec. 4 proposes a multi-layer monitoring approach and describes its application in the new flow exporter and other existing flow-based tools. Sec. 5 discusses benefits and possible applications of the presented approach. Sec. 6 concludes the paper and lists our future work.

2 Related Work

Packet header parsing is the basic functionality of every network device. Nowadays, devices support lots of protocols; however, the purpose of the parsing is to use just information contained in the headers for switching, routing, or any other actions. This is the reason why the parsing usually ends up with the first needed occurrence of the protocol in the packet. This approach is the main principle of encapsulation and decapsulation used in the ISO/OSI model.
The first part of this section surveys existing work on flow monitoring of encapsulated traffic. The second part describes packet parsers and focuses especially on the P4 language.

2.1 Flow Exporters

An approach similar to the ISO/OSI principle is used for the current flow exporters, which usually utilize the first network and transport layers found in a packet. Additionally, it is possible to skip certain lowest layers such as VLAN, TRILL, or MPLS in order to analyze packets transferred through the network infrastructure. However, current implementations of flow monitoring tools do not support the representation of repeated protocols in a single packet.

A special category of devices and their parsers are L7 filters and analyzers (e.g., [9, 22, 27]). Such devices are capable of advanced traversing through the encapsulated header, however, they usually support only a limited number of combinations of the protocols in a packet. This can work in practice where the parsing and header traversal is partially done by the underlying layers of the L7 device. Automatic generation of protocol parsers is not a novel approach. In 2006, Pang et al. proposed a high-level language called binpac to describe parsers of L7 protocols in [21]. Contrary to binpac, this paper is focused on lower layers and their multiple occurrences in a packet.

Elich et al. in [11] investigate IPv6 tunneled traffic in a real network. The authors of the paper extend an existing flow exporter to examine and report values of specific fields of chosen IPv6 tunneling protocols. The additional fields are exported as a part of the original flow record that contains the tunneled traffic as a payload. The authors define custom IPFIX elements to represent the new fields and extend a flow collector to process these elements. Our work aims to provide support for general packet encapsulation, not only IPv6 tunnels. Moreover, we choose a different method of exporting information about encapsulated traffic.

The complexity of link layer protocols is steadily increasing. Traffic encapsulation is quite common whenever virtual networks are created or IPv6 transition mechanisms are applied. For instance, VLANs or similar mechanisms help to segregate a network into isolated parts due to many reasons such as security or routing. Most flow exporters provide the capability to process MPLS and VLAN tags, fewer provide mechanisms to unwrap tunneling mechanisms such as the GRE protocol. Although there are several open-source flow exporters, such as [6, 10, 15, 17] as well as many commercial ones [7, 12], we are currently not aware of any flow exporter that would allow processing of traffic with arbitrary encapsulation. Moreover, the structure of flow records is usually flat and adding dynamic fields for every encountered encapsulation layer is not supported by flow collectors either.
2.2 Packet Parsers

Software packet parsers and parsers created for operation in hardware switches and routers are different in their nature. Both can be optimized for speed; however, the hardware can take advantage of a parallel data processing which allows more efficient analysis of the packets. The hardware parsers usually need a parse graph of protocols that are supported and which shows the expected order of their presence in the packets. This limitation makes it difficult for packet parsers in networking devices to process arbitrary protocol encapsulation.

Gibb et al. studied design of packet parsers for hardware targets in [14]. The paper shows trade-offs in parser designs, identifies design principles for switches and routers, and describes a parser generator that outputs the source code in Verilog. The authors show that the parse graphs can be either fixed or programmable. However, even though the programmable parse graphs are easy to modify, they must be known at a compile time.

The authors of [16] present a parser design which allows processing of multiple headers in a single bound. They use a fast header field extractor with a lookahead function and implement a translation for NetFPGA devices. However, parser instructions are compiled from parse graphs as well, which limits the parser versatility.

There are many software packet parsing libraries for various programming languages. The ones which implement the parsers directly are not easily extensible and are not useful for our purpose. An example of such a library is libtins [13], which is a high-level, multiplatform C++ network packet sniffing and crafting library. However, some parsers can be generated from a high-level description or grammar, which allows new protocols to be added easily and efficiently. For example, Bangert and Zeldovich created a tool called Nail [2] for parsing and generating data formats. The main purpose of the tool is to generate safe parsers. By eliminating semantics actions, the authors were able to simplify the parsers and create generators as well. However, the expressive power of their grammar was reduced.

P4 [4] is an open source high-level language which was firstly introduced by Bosshart et al. in 2014. The language allows us to describe all aspects of a packet processing device (i.e. structure of the supported protocols, parsing packet headers and processing packets). The language is designed to be platform independent which makes it suitable for a general description of a network device. The P4 language is the next step in an evolving ecosystem of Software Defined Networks (SDN). The strongest idea of P4 is to provide an easy way for a rapid data plane prototyping, while the former implementations of SDN switches typically contained only a static set of supported actions and protocols. Even though P4 improves the flexibility of SDN switches, due to performance requirements, it is not usual to map the described parser of protocols with a limited number of supported layers.

The P4 language is the major domain-specific language for the description of network devices. The P4 language Consortium [19] provides several tools such as a P4 compiler frontend [20], a software tool for running translated programs [18],
and others. Current works show that a P4 description can be mapped on various technologies. For instance, Benacek provided a theory about the translation of P4 programs to FPGA target in [3], whereas the aim is to process high-speed (more than 100 Gb/s) network traffic. Open vSwitch [1], which is an open source software implementation of an OpenFlow switch, can be used as a software target of P4 programs.

3 Flexible Packet Parser Based on P4

A general architecture of the proposed software flow exporter is inspired by existing tools and our many-years experiences. The high-level architecture of the flow exporter is shown in Fig. 1 and can be divided into three main parts: packet parser, flow cache, and flow export. The main focus of this section lies in the packet parser block.

The flexibility of the flow exporter is mainly in the packet parser that is generated from a high-level description in P4. For our purpose, a subset of the language features is used to specify the list of supported protocols and possible transitions between them.

Fig. 1. The process of P4 program compilation and packet parser usage.

3.1 New Backend for P4-16 Compiler

P4 Language Consortium provides and maintains an open source compiler for the P4 language at the GitHub repository [20]. It must be noted, that there are two versions of P4 compiler, whereas the latest one (for the P4 version 16) is used in our work. Fig. 1 shows the compiler consisting of frontend, midend, and backend. The frontend is a parser of the supplied P4 source codes, which creates an internal tree representation (IR). The midend is responsible for optimizations
of IR driven by architecture dependent policies and, subsequently, IR is passed into the backend. The backend generates output for the target.

Originally, the P4 compiler contained some existing backend modules; however, for our purposes, we needed a different kind of output — source codes of a linkable parser that can be used in the flow exporter. Therefore, a new backend\(^3\) has been developed in order to automatically generate source codes of the packet parser in the C language\(^4\). As a result, generated *.c and *.h files contain a packet parser function, declaration of known protocol headers (\texttt{struct} in C), and useful constants. The source codes can be easily linked to any C project to handle packet parsing.

The generated parser is implemented as a finite state machine (inspired by existing eBPF backend) that expects the size of the packet and the array of its bytes as an input. A state is represented as a block of source code that consists of a header extraction and next state decision logic. The aim of the parser is to extract header fields from the packet. Since the description of transitions between different protocol headers might be complicated and dependent on bytes around the current position of a "packet cursor" (offset in the packet that is being processed), there are several function constructs in P4: \texttt{lookahead} (read bits without moving packet cursor), \texttt{advance} (move packet cursor without reading bits), and \texttt{extract}. These functions must be supported by the backend and are translated into the equivalent functionality of the packet parser in order to support complex header traversal.

### 3.2 Output of the Packet Parser

Besides flexibility of the set of supported protocols defined by P4, it is also possible to specify a list of protocol headers that can be returned as an output of the packet parser. Other protocol headers, which are defined in P4 but are not explicitly listed for output, can be parsed and skipped in a packet.

The output of the packet parser is shown in Fig. 2. Contrary to a traditional approach, which stops at the first recognized transport layer, the presented flexible parser returns all identified protocol headers as a linked-list. The linked-list output allows for a multi-layer approach of the flow exporter (more details are in the following sections).

### 4 Multi-Layer Flow Monitoring

We have presented a flexible packet parser based on P4 which allows us to process encapsulated network traffic, extract packet headers relevant to the flow monitoring, and skip headers that should be ignored. This section describes the implementation of a flow exporter based on this parser. We show how to extend

---

\(^3\) Publicly available at [5].

\(^4\) Former P4 compiler (version 14) was able to generate a packet parser in the form of some C source codes, however, functionality was limited compared to our new packet parser.
Fig. 2. Example of mapping the encapsulated multi-layer packet onto multiple flow records.

Enhanced Flow Monitoring with P4 Generated Flexible Packet Parser

4.1 Flow Exporter Implementation

We have considered two options of representation of the encapsulated traffic in a flow cache — Extended flow records and Tree structure. Fig. 3 shows two packets with multiple IPv4 headers and both representations as well. The option of Extended flow record is to keep a single flow record based on the outermost headers (e.g., src IP 192.168.1.1 in Fig. 3) and represent the upper layers by added information fields. This can be easily implemented by defining flow keys to span over all layers that are present in the packet. The packets with the same values of flow keys belong to the same flow; therefore, the outer flow is split naturally. This approach was used by Elich et al. in [11]. The disadvantage of this approach is that each flow record has to be dynamically sized for it has to be able to contain information from an arbitrary number of layers. Moreover, the processing of such flow records on the flow collector would be cumbersome,
and the flow collectors would have to be significantly modified to support queries over multiple layers in the same flow record.

The second option, the preferred one, is to represent the multi-layer hierarchy of encapsulated flows as a tree with the outermost flow being the root of the tree. Each layer is represented by a standalone flow record that contains an identification of the parent flow. A flow record of the outermost layer has naturally undefined parent flow, i.e., it can be set to 0 or some other constant. This representation of flows has several advantages. The first is that each flow record can have the same structure with a fixed length. This makes the implementation of the flow cache easier since a simple hash table can be used. The only change to the flow cache is that each flow record must now contain an identifier of its parent flow or an indication that no such parent flow exists, i.e., the flow is the outermost one. This allows utilizing the existing flow cache implementations with minimal change.

We have decided to use the second option for the reasons listed above. Therefore, we need to find a unique identifier for each flow record so that it can be referenced from the child flows. A simple sequence of numbers could be used for this purpose; however, its state would have to be kept so that a restart of the flow exported does not reset the counter which would create duplicated identifiers. A simple solution to keep the state and solve collisions is to use a timestamp as a start of a sequence. Using a timestamp of the start of the flow as a key would work as well. However, the timestamp precision has to be high enough to ensure that no two packets could receive the same timestamp on a given medium (the timestamp must increase faster than the packets are received). Note that the

![Diagram of flow records and tree structure]

Fig. 3. Two options of representing encapsulated flows in a flow cache.
flow identifier collisions need to be handled by a collector as well since it can receive data from multiple exporters.

The current trend in flow monitoring is to gather as much information from the traffic as possible. The flexible P4 packet parser allows extracting values of any header field of the supported protocols. The values are then accessible in the linked list of the extracted headers and can be easily added to the flow record. For example, if we decide to analyze TCP window size, it is very easy to make the value accessible just by adding the attribute to the TCP header definition in the P4.

Unfortunately, P4 is not designed to process application layer headers which are often text-based or without a fixed structure. However, a pointer to the application layer is easily extracted by the P4, and the parsing process can be complemented by the use of any of the number of available application parsers.

4.2 Flow Collector Compatibility

When a flow collector receives a flow record with unknown elements, it usually discards only these elements and processes the rest of the flow. Therefore, multi-layer flow records can be easily processed by standard flow collectors. However, without the access to the parent ID element outermost flow records cannot be distinguished, which could have an undesirable impact on further processing, such as computing statistics. As a workaround and indication of the outermost flows can be exported as a supported yet otherwise unused field, that can be processed by the collector. However, this approach may lead to confusion, errors and is better to be avoided by using a more flexible flow collector.

To make a flow collector fully compatible with the multi-layer flow monitoring, it needs to be flexible to allow a definition of new elements. flow ID and parent ID elements must be properly received from the flow exporter, stored and must be accessible for later processing. An example of such a flexible collector is IPFIXcol [26] with FastBit storage backend and associated fbitdump query tool [25]. To avoid collisions during parent flow lookup, a flow source identifier must be used together with the parent flow identifier since there might be several flows with the same flow identifier created by different flow exporters.

5 Discussion

Since we have designed our flexible packet parser as a C library, it can be used for other purposes than flow monitoring. A network diagnostics systems might use the parser to automatically discover issues in encapsulated traffic. Host-based firewalls are another example of devices which can make use of the parser. Filter rules to match encapsulated packets are not usually supported, and they have to be specified using byte sequences as the encapsulated traffic is considered a payload by the firewalls. Our parser can enable firewalls to inspect all protocol headers and match even the encapsulated traffic easily.
We believe that a deployment of multi-layer flow monitoring in practice will bring new insights to the network traffic and help to identify hidden threats and issues. Since the required modifications to the flow collectors are minimal, it should not be difficult to deploy. However, in order to get multi-layer flow monitoring widely adopted, the existing IPFIX flow definition, which is widely used, might need to be updated.

The performance of our parser should be increased before the multi-layer flow monitoring is deployed. Our parser is not optimized for speed at this time; therefore it is not suitable for high-speed applications. However, it can still be used to monitor local networks and analyze packet captures. The overhead of storing more layers of flows in the flow cache can hinder performance since encapsulated traffic requires an update of multiple flow records for a single packet. The real impact of this overhead depends on the structure of the traffic and should be evaluated before the multi-layer flow monitoring is deployed in a high-speed network.

6 Conclusion

Flexibility is one of the key features of modern network devices. The P4 language, which was used in our work, was developed in 2014 in order to make the packet processing much more extensible and adaptable according to application needs. This paper is focused on applying benefits of the high-level P4 description in the domain of flow-based monitoring.

Sec. 2 describes the current state-of-the-art in the area of flow exporters, which are the base components of a flow-based monitoring infrastructure. Our research and many-years experiences from practice show that flow exporters, as well as many other network devices, do not parse the whole packets to look up all known protocol headers. This approach is natural and logical since any additional parsing costs processing power, which is limited and expensive in the network devices. On the other hand, analysis of the higher layers of the protocol headers increases visibility that is very important for various use-cases such as performance analysis or network security.

The main contributions of this paper lie in the usage of the P4 language and improvement of a flow exporter that applies a “multi-layer” monitoring approach. The P4 language is used for automatic creation of a packet parser with a flexible set of the supported protocols. This flexible packet parser is able to traverse through the all known protocol headers in a packet and pass the gathered information to the rest of the flow exporter.

Since the official compiler of the latest version of P4 (P4-16) does not support output format that we could easily use in our flow exporter, a new backend for this compiler was created, and it is available at our GitHub repository [5]. The backend generates C source codes that can be used in any project.

A novel approach of the “multi-layer” monitoring and its representation in a standard IPFIX format was explained in Sec. 4. It is based on the linked-list of identified protocol headers, which is the output of the flexible packet parser.
The goal is to represent different layers of encapsulation in a single packet using slightly extended IPFIX flow records. Despite the flow records being extended, existing tools might be still used for processing. In addition, this approach allows us to gain more information from a single packet compared to a standard IPFIX record.

As a future work, we plan to elaborate on the performance of the generated packet parser because the maximal throughput is a key feature that affects deployment of the flow exporter in high-speed networks. A list of supported protocols and extracted header fields can be defined in P4, however, the whole concept of partially generated flow exporter can be improved to support changes at runtime.

Acknowledgment

This work was supported by the project No. TH02010214 Platform for Acceleration of Network Functions Virtualization funded by the Technology Agency of the Czech Republic, internal grant No. SGS17/212/OHK3/3T/18 of CTU in Prague, and the project Reg. No. CZ.02.1.01/0.0/0.0/16_013/0001797 co-funded by the MEYS and ERDF.

References

1. A Linux Foundation Collaborative Project: Open vSwitch, http://openvswitch.org/
2. Bangert, J., Zeldovich, N.: Nail: A practical tool for parsing and generating data formats. In: Proceedings of the 11th USENIX Conference on Operating Systems Design and Implementation. pp. 615–628. OSDI’14, USENIX Association, Berkeley, CA, USA (2014), http://dl.acm.org/citation.cfm?id=2685048.2685098
3. Benacek, P.: Generation of High-Speed Network Device from High-Level Description. Ph.D. thesis, Czech Technical University in Prague (2017), https://github.com/benycze/PhD-Thesis
4. Bosshart, P.; et al.: P4: Programming Protocol-independent Packet Processors. SIGCOMM Comput. Commun. Rev. pp. 87–95 (2014), http://doi.acm.org/10.1145/2656877.2656890
5. CESNET, a.l.e.: Flexible Packet Parser — Backend for p4c, https://github.com/CESNET/cesnet/cesnet/p4c-backend
6. CESNET, a.l.e.: flow_meter, https://github.com/CESNET/Nema-Modules/tree/master/flow_meter
7. Cisco Systems, Inc., San Jose, CA and USA: Cisco Application Visibility and Control (AVC), https://www.cisco.com/c/en/us/products/routers/avc-control.html
8. Claise, B., Trammell, B.: Specification of the IP Flow Information Export (IPFIX) Protocol for the Exchange of Flow Information. RFC 7011 (Sep 2013), https://rfc-editor.org/rfc/rfc7011.txt
9. ClearFoundation: l7-filter (2017), http://l7-filter.clearos.com/
10. Deri, L.: nProbe: an Open Source NetFlow Probe for Gigabit Networks. In: Proceedings of the TERENA Networking Conference, TNC’03. Zagreb, Croatia (2003), http://luca.ntop.org/nProbe.pdf
11. Elich, M., et al.: An investigation into teredo and 6to4 transition mechanisms: Traffic analysis. In: 38th Annual IEEE Conference on Local Computer Networks - Workshops. pp. 1018–1024 (Oct 2013)

12. Flowmon Networks: Flowmon Probe, https://www.flowmon.com/en/products/flowmon/probe

13. Fontanini, M.: libtins, http://libtins.github.io/

14. Gibb, G., Varghese, G., Horowitz, M., McKeown, N.: Design principles for packet parsers. In: Architectures for Networking and Communications Systems. pp. 13–24 (Oct 2013)

15. Inacio, C.M., Trammell, B.: YAF: Yet Another Flowmeter. In: Proceedings of the 24th International Conference on Large Installation System Administration. pp. 1–16. LISA’10, USENIX Association, Berkeley, CA, USA (2010), http://dl.acm.org/citation.cfm?id=1924976.1924987

16. Kozanitis, C., et al.: Leaping multiple headers in a single bound: Wire-speed parsing using the kangaroo system. In: Proceedings of the 29th Conference on Information Communications. pp. 830–838. INFOCOM’10, IEEE Press, Piscataway, NJ, USA (2010), http://dl.acm.org/citation.cfm?id=1833515.1833654

17. Miller, D.: Softflowd, https://www.mindrot.org/projects/softflowd/

18. P4 Language Consortium: Behavioral Model Repository (Oct 2017), https://github.com/p4lang/p4c

19. P4 Language Consortium: P4 (2017), https://p4.org/

20. P4 Language Consortium: p4c (2017), https://github.com/p4lang/p4c

21. Pang, R., Paxson, V., Sommer, R., Peterson, L.: binpac: A yacc for writing application protocol parsers. In: Proceedings of the 6th ACM SIGCOMM Conference on Internet Measurement. pp. 289–300. ACM (2006)

22. Radware, Ltd.: Web Application Firewall (WAF) & Network Security Solution (2017), https://www.radware.com/Products/AppWall/

23. Sadasivan, G., et al.: Architecture for IP Flow Information Export. RFC 5470 (Informational) (Mar 2009), https://www.rfc-editor.org/rfc/rfc5470.txt, updated by RFC 6183

24. Umer, M.F., Sher, M., Bi, Y.: Flow-based intrusion detection: Techniques and challenges. Computers & Security 70, 238 – 254 (2017), http://www.sciencedirect.com/science/article/pii/S0167404817301165

25. Velan, P.: fbitdump, https://github.com/CESNET/ipfixcol/tree/master/tools/fbitdump

26. Velan, P., Krejčí, R.: Flow Information Storage Assessment Using IPFIXcol. In: Dependable Networks and Services. Lecture Notes in Computer Science, vol. 7279, pp. 155–158. Springer Berlin Heidelberg, Heidelberg (2012), http://dx.doi.org/10.1007/978-3-642-30633-4_21

27. Wireshark Foundation: Wireshark (2017), https://www.wireshark.org/
Throttling Malware Families in 2D

Mohamed Nassar and Haidar Safa

Department of Computer Science
American University of Beirut
Beirut, Lebanon

Abstract. Malicious software are categorized into families based on their static and dynamic characteristics, infection methods, and nature of threat. Visual exploration of malware instances and families in a low dimensional space helps in giving a first overview about dependencies and relationships among these instances, detecting their groups and isolating outliers. Furthermore, visual exploration of different sets of features is useful in assessing the quality of these sets to carry a valid abstract representation, which can be later used in classification and clustering algorithms to achieve a high accuracy. In this paper, we investigate one of the best dimensionality reduction techniques known as t-SNE to reduce the malware representation from a high dimensional space consisting of thousands of features to a low dimensional space. We experiment with different feature sets and depict malware clusters in 2-D. Surprisingly, t-SNE does not only provide nice 2-D drawings, but also dramatically increases the generalization power of SVM classifiers. Moreover, obtained results showed that cross-validation accuracy is much better using the 2-D embedded representation of samples than using the original high-dimensional representation.

Keywords: Malware · t-SNE · Visualization · SVM.

1 Introduction

Security breaches are executed through malwares and are a major threat to the Internet today. There are several forms of malware ranging from viruses and spam bots to trojan horses and rootkits [12]. Recently, the Petya ransomware [1] crashed shipping companies, ports, and law agencies. This malware targets the master boot record of a machine and prohibits the operating system from normal execution. It then spreads and encrypts all the system files. A message appears on the screen stating the amount of ransom to decrypt the files. The payment is through crypto-currencies. The prominent expansion of malwares is due to their metamorphic and polymorphic techniques that give the ability to change their code as they propagate. In addition, malwares adopt new ways to detect the environments where they are running, hence hindering their detection and making dynamic analysis difficult if not impossible.
Visual analytics provides approaches to obtain an understanding from complex data. It aims at developing methods that allow analysts to examine the processes underlying the data. Visual exploration of malware families is a pre-processing step of a more in-depth malware family analysis, as it allows for the development of intuitions and hypotheses about the discriminative power of a set of contextual or behavioral features. However, visualizing malware families in low dimensional space (2-D, 3-D) is a topic that received little attention in the literature. Malware data are fundamentally different than text and images which motivates investigating ways to adapt existing approaches or inventing new ones. For instance a byte in malware has different meanings in different contexts, in contrast to a byte representing pixel intensity in an image.

In this paper, we experiment with the best low-dimensional embedding technique known as t-SNE (Student-t distribution – Stochastic Neighborhood Embedding) for depicting malware clusters and features. We propose a pipeline for feature extraction and selection, followed by visualization. We compare the raw classification accuracy at the high-dimensional and low-dimensional spaces for n-grams features. Finally We propose a new first-insight classifier based on t-SNE and SVM. Note that our goal is not to propose a very high accuracy classifier or to compete with extensive feature selection approaches. Instead, we aim at exploring the visualization space of malware families and to which extent such a pre-processing procedure might be useful for analyzing a typical malware dataset. The remaining of the paper is organized as follows. Section 2 surveys relevant related work. Section 3 presents our proposed methodology. In Section 4, we discuss implementation and results. We finally conclude in Section 5 and present future research directions.

2 Background and Related Work

Classifying and Clustering malware families were addressed in many recent work in the literature. In [7], the current automated approaches for malware clustering were summarized. The paper considered the high accuracy obtained by six commercial anti-viruses as biased since unbalanced datasets were used where most malware instances are easy to classify. A plagiarism detector algorithm was applied on the same dataset and yielded the same accuracy results compared to those of the anti-viruses, though the plagiarism detector does not have any expert knowledge about malwares. In [2] a scalable, behavior-based malware clustering approach was proposed. This approach aims to isolate outliers that exhibit a novel behavior to be further analyzed. It used a recent technique called taint tracking to build behavioral profiles and locality sensitive hashing for clustering these profiles. Malware instance visualization was proposed in [9]. This approach suggested transforming the binary into a vector of 8-bit integers, which can be reshaped into a matrix and therefore viewed as a gray-scale image. This technique proves useful in increasing the accuracy of malware classifiers. The work presented in this paper is different such that we focus on visualizing

\footnotesize{\url{http://www.jsylvest.com/blog/2017/12/malconv/}}
families of malwares as scatter plots using t-SNE \cite{van2008visualizing}, an embedding technique that allows visualizing high-dimensional data by giving each datapoint a location in a two or three-dimensional map. The technique is a variation of Stochastic Neighbor Embedding (SNE).

SNE starts by converting the high-dimensional Euclidean distances between datapoints into conditional probabilities that represent similarities. The similarity of datapoint $x_j$ to datapoint $x_i$ is the conditional probability, $p_{ji}$, that $x_i$ would pick $x_j$ as its neighbor if neighbors were picked in proportion to their probability density under a Gaussian centered at $x_i$. For the low-dimensional counterparts $y_i$ and $y_j$ of $x_i$ and $x_j$, it is possible to compute a similar conditional probability $q_{ji}$. SNE aims to find a low-dimensional data representation that minimizes the mismatch between $p_{ji}$ and $q_{ji}$. To do so, SNE minimizes the sum of Kullback-Leibler divergences over all datapoints using a gradient descent method. t-SNE differs from the old SNE in two ways: (1) it uses a symmetrized version of the SNE cost function with simpler gradients and (2) it uses a heavy-tailed Student-t distribution rather than a Gaussian distribution to compute the similarity between two points in the low-dimensional space. This allows t-SNE to alleviate both the so-called crowding problem and the optimization problems of SNE \cite{van2008visualizing}.

The dataset we use is the training set from Microsoft Malware Classification Challenge \cite{malware_classification} released in 2015, which since has been studied by researchers in more than 50 publications targeting feature engineering, deep learning, clustering and classification approaches. For instance, \cite{convolutional_neural_networks} proposes using convolutional neural networks for feature extraction and classification based on the binary and reconstructed assembly files. Our work takes another direction and focuses on the visualization question.

\section{Proposed Methodology}

In this section we describe our proposed methodology which copes with relatively large data sets. We have used the training dataset from Microsoft malware classification challenge (Big 2015) having 10868 labeled instances \cite{malware_classification}. The standard version of t-SNE having quadratic complexity in terms of the number of instances $O(N^2)$ might be applied. For larger data-sets, other versions of t-SNE are proposed such as random-walk based sampling of landmark points or using specialized data structures leading to $O(N \log N)$ complexity \cite{maaten2008}. We particularly used the version of scikit-learn with the Barnes-Hut approximation running in $O(N \log N)$ time.

Fig. 1 shows the pipeline of our proposed methodology. Starting from a labeled corpus of files containing the malwares payload in hexadecimal format, we extract n-byte grams (3, 4 and 5). Note that the number of features grow exponentially such that for 3-byte grams we have $2^{8 \times 3}$ possible words, for 4-byte grams it is $2^{8 \times 4}$ and so on. Most current machine learning libraries cannot handle this number of features even in sparse format. For example, Scikit-learn accepts feature indices less than a positive 4-bytes signed integer ($2^{31} - 1$). Therefore, in
the proposed methodology, we hash the feature indices to 22-bits integers. We also proceed by early removal of rare words that appear less than k times (we assumed k = 3). These words probably represent addresses in memory or literals and have little differentiation power. This technique is very efficient in reducing the storage amount of the features set. We store the resulting features along with the instance label in sparse format (LibSVM/SVMLight format) as one line per instance in the output text file. Each feature is represented by an index (the hash of the n-bytes gram) and a value which is the number of occurrences of this n-bytes gram in the malware instance. This stage is implemented by using the Sally tool [10] which is an efficient feature extraction tool that generates n-grams besides other features such as TF-IDF [3].

We proceed with a first feature selection stage using the $\chi^2$ statistical test-based selector where our target is to reduce the number of features to 1,000. This limit reduces the complexity of computing the pair-wise distances in t-SNE. However, we are starting with a much larger number ($2^{22} = 4,194,304$ feature-space). The time complexity of the $\chi^2$ selector is $O(n_{classes} \times n_{features})$. The features that are the most likely to be independent of class and therefore irrelevant for classification are removed.

We also can reduce the space complexity of this stage (mainly because of memory limitations) by sampling the instances in equal proportions to their family sizes. If sampling is used than we must use the generated $\chi^2$ selector model to transform the complete dataset and keep the top 1,000 features of each instance. For the dataset in question we did not use sampling.

![Fig. 1. Pipeline for malware family visualization.](image)

Optionally, we apply a PCA (Principal Component Analysis) transformation to reduce further the number of features to the range of 30-50 features. This speeds up the computation of pairwise distances between the data-points in the next stage and suppresses some noise without severely distorting the inter-point distances [5]. t-SNE then embeds these features in 2 dimensions. The malware
instances are depicted as scatter plots. t-SNE outperforms other data embedding techniques such as PCA, Sammon mapping, Isomap and LLE.

4 Implementation, Preliminary Results, and Interpretation

In this section, we describe the implementation environment and setup, the dataset used, the hyper parameters then we discuss the obtained results.

4.1 Setup, implementation, and tools

All our experiments were performed using a commercial off-the-shelf laptop with a 64-bit Ubuntu 16.04 LTS operating system, an Intel core i5-5200U CPU (4 cores, 2.20GHz) 8 GB RAM and 1 TB Hard disk. Our implementation was based on Python v3.6, numpy v1.13, scipy v0.19 and the Scikit-learn v0.19 library. The plots are generated using the BokehJS v0.12 library. We have used Sally [10] for feature extraction. Our code is available at https://github.com/mnassar/malware-viz under form of Python Jupyter notebooks for further exploration and result reproducibility.

4.2 Dataset

The dataset is the training set from Microsoft Malware Classification Challenge [11], which includes 10868 labeled samples. For each sample, the raw data and meta data are provided. The raw data contains the hexadecimal representation of the file’s binary content, without the Portable Executable (PE) header to ensure sterility. The metadata manifest is a log containing various metadata information extracted from the binary, such as function calls, strings, etc. This was generated using the IDA disassembler tool. In our implementation, we focused solely on the raw data. However, we consider augmenting our visualization with the meta data for future work. The raw training data volume is about 41 GBytes. However, our feature selection method reduces this to about 6.4 GBytes for the sum of three feature sets (3, 4 and 5-grams). The dataset contains malwares belonging to the following 9 families: Ramnit, Lollipop, Kelihos Ver. 3, Vundo, Simda, Tracur, Kelihos Ver. 1, Obfuscator.ACY and Gatak. One challenge of this data set is the unbalanced sizes of different families. The distribution of instances for the training dataset is shown in Table 1. It is shown that classes 4, 5, 6 and 7 are underrepresented as compared to the other families.

4.3 t-SNE parameters

t-SNE implementation in Sklearn has many tunable parameters. We show the most important ones with a short description in Table 2. Note that the t-SNE method is known to be little sensitive to these parameters. Nevertheless, a good tuning enhances the quality of the obtained clusters sometimes.
Table 1. Distribution of samples among the families.

| Class | Family    | Type          | Nb. Of Instances | Percentage (%) |
|-------|-----------|---------------|------------------|----------------|
| 1     | Ramnit    | Worm          | 1541             | 14.20          |
| 2     | Lollipop  | Adware        | 2478             | 22.80          |
| 3     | Kelihos,ver 3 | Backdoor     | 2942             | 27.07          |
| 4     | Vundo     | Trojan        | 475              | 4.37           |
| 5     | Simda     | Backdoor      | 42               | 0.39           |
| 6     | Tracur    | Trojan Downloader | 751           | 6.91           |
| 7     | Kelihos,ver 1 | Backdoor   | 398              | 3.66           |
| 8     | Obfuscator.ACY | Obfuscated malware | 1228       | 11.30          |
| 9     | Gatak     | Backdoor      | 1013             | 9.32           |

Table 2. Tunable Parameters in t-SNE.

| Parameter                      | Description                                                                 | Typical/default value |
|--------------------------------|-----------------------------------------------------------------------------|-----------------------|
| Random state                   | This is the seed of random initialization in the embedded space. It is an important parameter to accomplish comparisons under same initial conditions (used with init = ‘random’) | No typical value. We have chosen 42 as our default. |
| N_iter                         | Maximum number of iterations for the optimization.                          | Default: 1000         |
| Perplexity                     | The perplexity can be interpreted as a smooth measure of the effective number of neighbors. | 5 – 50 Our default is 40 |
| Early exaggeration             | Larger values of this parameter tend to start with distant clusters in the embedded space | Default: 12           |
| metric                         | The distance between instances where each instance is a feature array.      | Euclidean             |
| Learning rate (lr)             | It must not be too low or too high. If the cost function gets stuck in a bad local minimum increasing the learning rate may help. | 10 – 1000 We have chosen 200 as our default. |
4.4 Sample of results

Due to lack of space we do not show all obtained visual plots. However, we depict and interpret the most important ones. Fig. 2 shows the scatter plot of the 9 classes for 4-byte grams (no PCA). In this figure, it is directly noticed that each family is composed of one or multiple clusters. No one clear cluster per family exists as in the case of the MNIST dataset [6]. This is expected due to the polymorphic nature of malware data which has much more noise than what can be carried by handwritten digits or letters. Common evasion and obfuscation techniques may also explain why some families have intersection regions in between their clusters. Still we can identify big clusters for each of the big families (1, 2, 3, 8, 9) and even some smaller clusters for the families 4 (Vundo) and 7 (Kelihos_ver 1). It is also important to notice the existence of some outliers.

To give a chance to the underrepresented families, we started over with only their instances in the pipeline. Obtained results are shown in Fig. 3. These results are much better, and we can observe clear groups for each of these small families. Next, we take only the largest two classes (2 and 3) to compare plots between different sets of features (3-bytes grams, 4-bytes grams and 5-bytes grams). Obtained results are illustrated in Fig. 4. They show that to some extent, 5-grams (Fig. 4(c)) have less overlap between clusters than 3-grams (Fig. 4(a)) or 4-grams (Fig. 4(b)). We wanted to correlate this with the classification accuracy on the original dataset (the one after feature selection which is 5420 instances * 1000 features) and the transformed dataset (5420 instances * 2 features). Results in Table 3 shows that training accuracy is almost perfect for all three feature sets. However, training accuracy is often not a good metric since the classifier might overfit the training set.

We compute the two-fold cross validation accuracy which is much more indicative of the generalization power of the classifier. Astonishingly, the two-fold cross validation accuracy is very poor on the original dataset and dramatically much better on the embedded dataset. This can be explained by the fact that t-SNE groups the datapoints into separate clusters in a low dimensional space, which is at the origin of the design of support vector classifiers with the Radial Basis Function (RBF) kernel. These classifiers shine under this kind of settings. The SVC accuracy (%) for the three feature sets with default Sklearn parameters (RBF kernel, \(C = 1.0, \gamma = 1/\text{features} \)) are shown in Table 3. Better SVC accuracy results are usually obtained after a grid search for the best hyperparameters \(C \) and \(\gamma \).

Next, we examine the performance of t-SNE on unbalanced families. We take the extreme case by choosing the largest family (Kelihos_ver 3 – class 3) and the smallest family (Simda – class 5). Results that are presented in Fig. 5 show that t-SNE can isolate the class 5 in a small cluster. They also show that 4-grams and 5-grams perform better than 3-grams in isolating clusters of the two classes. Fig. 5(d) shows that choosing a bad perplexity value might degrade the clustering quality.
Fig. 2. 4-grams, no-PCA, 9 classes, perplexity=40, lr=200.

Fig. 3. 4-grams, no-PCA, classes 4,5,6,7, perplexity=40, lr=200.
Fig. 4. (a) Classes 2 and 3 (3-grams, perplexity=40, lr=200); (b) Classes 2 and 3 (4-grams, perplexity=40, lr=200); (c) Classes 2 and 3 (5-grams, perplexity=40, lr=200)

| Feature Set     | Training Accuracy (1000-d) | Training Accuracy (2-d) | Two-fold cross-Validation Accuracy (1000-d) | Two-fold cross-Validation Accuracy (2-d) |
|-----------------|-----------------------------|-------------------------|---------------------------------------------|-----------------------------------------|
| 3-byte grams    | 99.98                       | 99.98                   | 67.91                                       | 99.57                                   |
| 4-byte grams    | 100.00                      | 99.96                   | 64.06                                       | 99.88                                   |
| 5-byte grams    | 99.98                       | 99.98                   | 68.13                                       | 99.92                                   |
Fig. 5. (a) Classes 3 and 5 (3-grams, perplexity=40, lr=20); (b) Classes 3 and 5 (4- 
grams, perplexity=40, lr=200); (c) Classes 3 and 5 (5-grams, perplexity=40, lr=200); 
(d) Classes 3 and 5 (5-grams, perplexity=5, lr=200)
t-SNE shows similar performance in enhancing the classification accuracy as shown in Table 4. Note that since the classes are severely unbalanced, an accuracy of 98.60 would be simply obtained if the classifier considers all the data points a belonging to the majority class 5. The two-fold cross validation accuracies on the original dataset are bad in this sense. However, we notice that the two-fold cross validation accuracy is much better in the embedded space. 3-grams features perform the worst as it can be expected by examining the corresponding scatter plot.

| Feature Set | Training Accuracy (1000-d) | Training Accuracy (2-d) | Two-fold cross-Validation Accuracy (1000-d) | Two-fold cross-Validation Accuracy (2-d) |
|-------------|----------------------------|-------------------------|--------------------------------------------|--------------------------------------|
| 3-byte grams | 100.00                     | 99.93                   | 98.52 (bad)                                | 99.83                                |
| 4-byte grams | 100.00                     | 99.93                   | 98.62 (bad)                                | 99.90                                |
| 5-byte grams | 100.00                     | 99.97                   | 98.62 (bad)                                | 99.90                                |

Finally, we want to validate this hypothesis on the complete dataset (9-classes). Results are shown in Table 5. The training accuracy in 2-d is a bit smaller than in 1000-d but allows much better generalization of the classification model as clearly inferred from the cross-validation accuracy results.

| Feature Set | Training Accuracy (1000-d) | Training Accuracy (2-d) | Two-fold cross-Validation Accuracy (1000-d) | Two-fold cross-Validation Accuracy (2-d) |
|-------------|----------------------------|-------------------------|--------------------------------------------|--------------------------------------|
| 3-byte grams | 99.66                     | 96.84                   | 56.76                                      | 94.26                                |
| 4-byte grams | 99.58                     | 96.22                   | 55.26                                      | 93.13                                |
| 5-byte grams | 98.25                     | 95.69                   | 60.24                                      | 92.58                                |

### 4.5 Testing Accuracy

In this section we further assess the idea of squeezing the dimensions into a small hyperspace using t-SNE than expanding it back to infinite dimensional space using the RBF kernel with SVM. We wanted to estimate the testing accuracy using the raw unlabeled test dataset (10873 instances). Note that t-SNE is a non-parametric mapping, therefore we cannot use the learnt model to map the test datapoints to the embedded space which is formed by the train datapoints. As an alternative we run t-SNE on the full dataset composed of both train and test datapoints (Another approach is to train a multivariate regressor to predict the map location from the input data [13]). We then fit an SVC model solely
based on the train embedded datapoints. This SVC model is used to estimate probabilistic predictions of the membership of each embedded test point to each of the 9 possible classes. The pipeline of this approach is depicted in Fig. 6.

\[
\text{logloss} = -\frac{1}{10873} \sum_{i=1}^{10873} \sum_{j=1}^{9} y_{ij} \log p_{ij}
\]

where \(y_{ij} = 1\) if \(i\) belongs to class \(j\) and 0 otherwise, \(\log\) is the natural logarithm and \(p_{ij}\) is the probability that \(i\) belongs to class \(j\) as given by the classifier.

Our classifier achieves a testing logloss of 0.1719. This is fairly acceptable given the simplicity of the employed feature set (1000 best features among 1,2,3,4 and 5 n-bytes grams) and without recurring to any involved feature engineering. A clueless classifier scores 2.1972. For this experiment we have used an m5.4xlarge AWS EC2 instance (64 GB RAM) and a 500 GB volume.

5 Conclusion

In this paper, we have successfully applied feature extraction, selection, embedding and visualization over a recent malware dataset. We have proposed a pipeline that can cope with dataset of similar or larger size. We use the t-SNE
algorithm to embed the malware datapoints in 2D and visualize them as scatter plots. A very interesting result is that compressing the data using t-SNE dramatically enhances the cross-validation accuracy of Support Vector Machines classifiers. t-SNE shapes the clusterability of datapoints in the embedded space, which is very appealing to SVM classifiers with the RBF kernel.

In future work, we aim to experiment with other feature sets, for instance by analyzing the assembly data files. We also want to assess the viability of the SVM–t-SNE classifier over other data sets. Another direction is to work on implementing t-SNE in a 3D WebGL framework and integrate it in Jupyter notebooks. Available GPUs can also be used to bear some of the tedious computations.

Acknowledgement

This work was supported in part by a grant from the University Research Board of the American University of Beirut, Lebanon.

References

1. Akkas, A., Chachamis, C.N., Fetahu, L.: Malware analysis of wannacry ransomware (2017)
2. Bayer, U., Comparetti, P.M., Hlauschek, C., Kruegel, C., Kirda, E.: Scalable, behavior-based malware clustering. In: NDSS. vol. 9, pp. 8–11. Citeseer (2009)
3. Chowdhury, G.G.: Introduction to modern information retrieval. Facet publishing (2010)
4. Ellis, G., Mansmann, F.: Mastering the information age solving problems with visual analytics. In: Eurographics. vol. 2, p. 5 (2010)
5. Gibert Llauradó, D.: Convolutional neural networks for malware classification. Master’s thesis, Universitat Politècnica de Catalunya (2016)
6. LeCun, Y., Bottou, L., Bengio, Y., Haffner, P.: Gradient-based learning applied to document recognition. Proceedings of the IEEE 86(11), 2278–2324 (1998)
7. Li, P., Liu, L., Gao, D., Reiter, M.K.: On challenges in evaluating malware clustering. In: International Workshop on Recent Advances in Intrusion Detection. pp. 238–255. Springer (2010)
8. Maaten, L.v.d., Hinton, G.: Visualizing data using t-sne. Journal of machine learning research 9(Nov), 2579–2605 (2008)
9. Nataraj, L., Karthikneyan, S., Jacob, G., Manjunath, B.: Malware images: visualization and automatic classification. In: Proceedings of the 8th international symposium on visualization for cyber security. p. 4. ACM (2011)
10. Rieck, K., Wressnegger, C., Bikadorov, A.; Sally: A tool for embedding strings in vector spaces. Journal of Machine Learning Research 13(Nov), 3247–3251 (2012)
11. Ronen, R., Radu, M., Feuerstein, C., Yom-Tov, E., Ahmadi, M.: Microsoft malware classification challenge. arXiv preprint arXiv:1802.10135 (2018)
12. Stallings, W.: Cryptography and network security: Principles and practice. Pearson (2016)
13. Van Der Maaten, L.: Learning a parametric embedding by preserving local structure. RBM 500(500), 26 (2009)
14. Van Der Maaten, L.: Accelerating t-sne using tree-based algorithms. Journal of machine learning research 15(1), 3221–3245 (2014)
Data Privacy Protection - Concealing Text and Audio with a DNA-inspired Algorithm

Paulo Silva¹, Lukas Kencl², and Edmundo Monteiro¹

¹ Department of Informatics Engineering, University of Coimbra, Coimbra, Portugal
{pmgsilva,edmundo}@dei.uc.pt
² R&D Centre for Mobile Applications (RDC), Czech Technical University in Prague, Czech Republic
lukas.kencl@rdc.cz

Abstract. Nowadays, with an increasing amount of personal and confidential data being transmitted and stored online, entities who store and manage data need to assure certain guarantees of data privacy protection. As such, we start by presenting a state of the art review of anonymization and concealing techniques. Their characteristics and capabilities are described, as well as metrics and tools to implement and evaluate data anonymization and concealing. Then, an evaluation of the applicability of the DNA-inspired information concealing algorithm is made. Usually, various metrics are used to measure aspects like disclosure risk or utility of the anonymized data. In this work, we use the Cosine Similarity metric to measure the similarity between the original data and respective versions after application of the algorithm. The evaluation is made by analyzing the output of the algorithm as well as the performance of the algorithm itself. With the final results and analysis, it is possible to determine its overall applicability with text and audio files. There is a discussion with advantages and disadvantages of this and other algorithms, as well as an identification of problems and respective suggestions for improvements on data privacy protection methods.

Keywords: Data Concealing · Data Privacy

1 Introduction

Over time different technologies and solutions are offered to fill the needs and requirements of privacy protection in IT. Solutions that range from privacy policies to security measures, authentication methods to anonymization techniques, or even laws and regulations. All play their role in the process of providing data privacy protection.

Twenty years ago, when those algorithms started to emerge, different approaches have been presented in order to provide the necessary anonymization and concealing of data. Nowadays, the scope of the methods is similar. Varies from making a complete modification of the data, making slight changes or just concealing according to certain rules. However, one aspect remains since the beginning: the more anonymized or concealed the data is, the less utility it could
provide. The opposite also applies. Ultimately, the goal of the privacy protection research field is to give data privacy protection and provide data with useful retrievable information.

This article presents the experiments performed with the DNA-inspired information concealing algorithm [1]. Most anonymization techniques and algorithms work based structured data inputs. Like network traces, logs or tables which is not the case of the DNA-inspired information concealing algorithm that conceals structured and unstructured data like text or audio for instance. The algorithm, by preserving and maintaining families of repeats, is capable of concealing data in a different fashion than the remaining methods. Therefore, the experimental work also aims to provide an analysis and evaluation of the method regarding its applicability and performance. The metric used to evaluate the results, the Cosine Similarity, shows its value not only in information retrieval or text mining applications but also in the analysis of concealed and anonymized files. Several text and audio files with different characteristics were used as an input of the algorithm. Accordingly, an analysis of the main findings and characteristics of the concealing process is presented.

Our motivation is not only to present methods and metrics but also to present actual data anonymization results and analysis. Therefore, our main objective is the evaluation of the applicability of the referred algorithm over certain data types: such as various text and audio files. Furthermore, aligned with the experimental analysis we perform an identification of open issues and problems in the field. Moreover, suggestions and possible solutions (aligned with future work) are also addressed.

The article continues with the following sections: Anonymization and Concealing Solutions (Section 2), Experimental Work (Section 3), Discussion (Section 4) and Conclusion (Section 5).

2 Anonymization and Concealing Solutions

In this section, existent anonymization and concealing algorithms are presented. Anonymization metrics and tools are equally described.

2.1 Anonymization Algorithms and Techniques

The term *anonymization* is commonly used to describe anonymization and concealing processes. There are several anonymization algorithms and techniques currently available. There is also a variation of their specifications, performance, data inputs or capabilities. Considering what was mentioned before, there are algorithms inherently more suitable than others to certain data types or applications (e.g. structured data, unstructured data, offline application, real-time, reversible or non-reversible).

The Black Marker technique [2] is a strong anonymization algorithm as a result of the replacement of fields by *NULL* or 0. Thus, it is an effective way of
hiding sensitive information. However, it does not provide good usability levels. Methods like Suppression [3] and Time Unit Annihilation [2] operate similarly.

A Permutation [3] is a one-to-one mapping of values. It is a direct substitution technique that replaces each value with some other value selected within a possible range, resulting in a unique anonymized value for each original value. This method is useful when is necessary to preserve the count or the order of the datasets, without preserving the information of the values themselves.

Hashing functions [2] can be very useful for anonymization of both text and binary data. What a hash function does is the mapping of each value to a new value. Not necessarily unique, as the permutation. The limitation with binary data, for example, is that truncating the result of a hash function to the shorter length of the value is often required. Consequently, the hash function is weaker and suitable for more collisions.

Generalization [3] is a way of transforming a more sensitive or revealing attribute in a more general information. There are several ways of making that modification. In the case of ZIP codes, for example, it is possible to group the specific numbers and group them as state or province, which is attributing a single value to a group of sensitive fields.

The K-anonymity concept [4] tries to answer the following: Given person-specific field-structured data, produce a release of the data with scientific guarantees that the individuals who are the subjects of the data cannot be re-identified while the data remain practically useful. To answer that problem, K-anonymity provides data privacy by ensuring that the sensitive attributes are repeated k times, with K being always greater than one, in order to make more difficult the identification of individual values. To achieve k-anonymity, this algorithm relies on the combination of both generalization and suppression. The MinGen algorithm [5] was developed having as a default condition the adhesion to K-Anonymity with a minimal generalization.

Differential Privacy [3] aims to provide means to maximize the accuracy of queries from a given dataset while minimizing the chances of identifying its records. There are similarities with the noise addition. In this case, anonymization is assured by the addition of Laplace noise to the queries performed on the dataset. Therefore, it is not possible to distinguish if a certain value was modified or not.

The DNA-inspired information concealing algorithm [1] (used in the experimental work (Section 3) is able to conceal information based on the introduction and maintenance of families of repeats, as DNA itself is able, at a different level. Consider the information concealing problem where a certain sequence \( w \) and a small integer \( k \) are given and \( |w| \) represents the length of the sequence. One wants to transform \( w \) into a new sequence \( wF \) so that is computationally hard to reconstruct \( w \) from \( wF \); the length of \( wF \) is linear in \( |w| \) and if \( s \) would be a segment of \( w \), when \( |s| \leq k \), then \( s \) would be a segment of \( wF \).

With a concealing problem, an attacker problem is inherent: how much information about the private sequence \( w \) can an attacker reconstruct from the final and concealed sequence \( wF \). To provide a solution to this problem, it was
proposed an algorithm composed of five procedures. Before the application of the procedures, the input sequence is turned into a cyclic sequence and only then, the five procedures can begin to be applied. A cyclic sequence is a sequence that has the last item connected to the first. Even though the first procedure has preparatory function, the five of them have a basic pattern: partition the input sequence into consecutive disjoint blocks; in front of each block, the terminal part of its predecessor is added (overlap); dust can be added at the end of each block; rearrange the blocks into an output sequence \( wF \). In this application, dust is considered a random part of the sequence itself and its length is related to the length of the sequences to preserve (K).

### 2.2 Anonymization Metrics

To assess how well the data is anonymized or concealed, there is a need to use certain metrics. Sometimes more than one at a time. The methods that can be used in this area range from common descriptive statistics for more advanced clustering algorithms. The utilization of descriptive statistics is a general but also an effective way of getting to know the amount of privacy granted to data or the usability of it. With this method, several measures can be taken to analyze the anonymized data. Mean, standard deviation, variance, covariance, dispersion or others, are some of the values that can be measured to quantify the distortion between anonymized and original data.

With the Classification Error Metric [3] there is a process similar to the descriptive statistics, for example, i.e. measuring the classification error returned and compare it to the original data, being the trade-off between data privacy and usability present also. In this case, both original data and anonymized data are passed through a machine learning algorithm which returns a classification error for original and anonymized data.

The Shannon’s Entropy [6] is a way of measuring the amount of information in a particular block of information. It returns the amount of information based on the uncertainty or randomness of data.

When considering Mutual Information (MI) [7] it is possible to observe the great utility it can provide. Using this metric, there are several ways to improve the assumptions taken of the privatized data and, with the same principle, provide better anonymizations when the MI is used in the anonymization algorithm.

The Pearson’s Correlation Coefficient [3], or correlation metric, measures the level of a linear correlation between two datasets, the original dataset and the anonymized one. It measures, as well, the direction of the correlation, being positive or negative. This method returns values between -1 and 1. The signal indicates the direction of the correlation; if it is positive or negative; and the value indicates the strength of the relation.

Like the Euclidean Distance, the Davies Bouldin Index [8] is used to evaluate how good the clustering of data is. There are three main factors to have in account with this metric: the quantification of how good the clustering is the main one. Furthermore, there is the distance between clusters and the distances within the cluster which can be useful for further analysis (e.g. Euclidean Distance).
The Cosine Similarity \([9]\) is the inner product of two vectors, divided by the product of their lengths. Also, the angle between the two vectors is represented by \(\theta\). This generates a normalized value between zero and one. The files \(f_1\) and \(f_2\) (or vectors) being compared have the same information if the Cosine Similarity is one. On the other hand, the files are completely different if the Cosine Similarity is zero. This is the metric used in the experimental work and can be defined as:

\[
\cos(\theta) = \frac{f_1 \cdot f_2}{|f_1||f_2|}, \quad \theta \in [0, 1]
\]  

\(2.3\) Anonymization Tools

There is quite a choice of available software and tools to make data anonymization. They work on different data types, data formats and provide different end solutions. Thereby, in this section, there will be presented solutions that offer a wider range of anonymization options and metrics as well.

ARX Data Anonymization Tool \([10]\) is very complete due to the wide range of algorithms implemented. The scalability feature was considered since its early development. Capable of analyzing data utility and re-identification risks, it supports privacy models, such as k-anonymity, l-diversity, and t-closeness. Semantic privacy models as differential privacy. Data transformation techniques like generalization, suppression and top/bottom coding as well as global and local recoding.

sdcMicro \([11]\) or Statistical Disclosure Control Methods for Anonymization of Microdata and Risk Estimation is an up-to-date ‘R’ package used for the generation of anonymized data. In addition, it also includes metrics and estimation processes, which provides a better and more complete analysis of the data. Fitted with a graphical user interface, there is a wide variety of available techniques to apply in the anonymization process. If compared with other tools as \(\mu\)-Argus (which does not provide metrics), this has all the techniques plus a few more.

There are other tools available, for example, TIAMAT, UTD Anonymization Toolbox, the Cornell Anonymization Toolkit or SECRETA \([12]\).

\(3\) Experimental Work

The DNA-inspired information concealing algorithm was used to perform experiments over data. The objective was to conceal text and audio using a series of repetitions and permutations with the rules of the algorithm.

\(3.1\) Methodology, Tools and Data

To perform the experiments several runs of each test were executed. For each batch of runs, there was a variation of the parameters used in the input of the algorithm. Each time the algorithm was executed with a certain set of parameters, a different seed (used on the cut of the blocks) was used. The reason was to
provide different outputs in every execution. In total, each set of parameters was executed five times in order to provide consistent results. For a posterior analysis of the results, averages and standard deviations are calculated over each set of executions. Matlab and Python scripts were used to support the experiments and data analysis.

The types of data used in the experiments were text (.txt) and audio (.wav) files. Several text files, as well as audio files, were used in the experiments. It was important to have different text files available for the experiments. The different files allowed a more complete analysis in cases where different contents, authors or types of texts were being concealed. In these files, there were novels, formal text or email contents. There were different styles of writing (different authors) and works from the same authors, in order to analyze the differences in those cases.

**Text Files** To make the analysis of the generated files containing concealed information, we used the cosine similarity metric (described in section 2.1). The cosine similarity measure was taken in four different ways, for all the files. Following there is a description of those measures of similarity. For all cases, all the characters were considered. It was assumed that all the characters matter for the analysis. Therefore, there was no striping of spaces, punctuation marks or other characters.

*Fixed length sequences* - One of the characteristics of the English language is its word length. It was shown that the English language has an average length of 5.1 letters per word [13]. Considering the language average length, one way of analyzing the data was by fixing a word length and verify all the sequences of characters with that specified length. Considering what was described, a fixed length of 5 characters was defined. All the sequences of characters in the original and generated documents are identified and represent a term. Additionally, for comparison purposes, measures with lengths of 25 and 50 characters were also taken.

*Term Frequency* - The notion of term frequency is used to refer to all the sequences of characters separated by a space. With the term frequency analysis, all the sequences of characters separated by space are identified and represent a term.

*Two consecutive terms* - Similar to the previous definition, with the exception that each term is separated by a space. All the sequences of characters separated by one space, represent one term in this case.

*Three consecutive terms* - Analogously, all the sequences of characters separated by two spaces, represent one term in this case. This case, as well as the previous one, allows the analysis of how different the term construction and precedence is in both files being compared.
To measure the cosine similarity in such a way, several Python programs had to be developed. The developed programs started by reading both original and generated files, in the four ways previously described. After that, for each one of the four ways of analysis, the cosine similarity was manually implemented. Later, after all the calculations and file analysis, the results were gathered and treated for validation and respective analysis.

There were parameters that could be specified when running the algorithm. Besides K (length of the sequence to preserve), it was possible to choose the block size (B), which is the length of blocs to conceal and the concealing type: weak or strong. Some internal parameters of the algorithm (such as overlap, lower bound, and upper bound) depend on each other. Nonetheless, the concealing method, block size, and sequence length are independent. Regarding the values used in the text experiments, below there is a description of the values used in those parameters and how they were modified in this case (text files).

**Block Size (B)** - Length of the blocks used by the algorithm. As described before (Section 2.1), the algorithm cuts the input into blocks and performs the needed operations over those blocks. The B parameter varied between 64, 128, 256, 512, 1024 and 2048. The reasoning behind these values are variations in average email lengths (character count). The block size parameter is presented as B64, B128, B256, B512, B1024 and B2048. Example: Block size 64 - 'The number of characters or letters in a text like this is 64...'

**Length of sequences to preserve from input (K)** - Length of sequences to be preserved from the input file. Characters in case of text, samples in case of audio (section 3.1. It preserves fixed length sequences of the input file. The K parameter varied between 3, 4, 5, 7 and 10. The K parameter is presented as K3, K4, K5, K7 and K10. Example of text input: K3 - 'The', 'nu', 'mb', 'r o', 'f c', 'har', 'act', 'ers', 'or', 'let', 'ter', 's i', 'n a', 'te', 'xt', 'lik', 'e t', 'his', 'is', '64', '...'

**Concealing type** - During the experiments there were two types of concealing used. The Weak concealing type applies the transformations and operations to the input without adding dust and overlaps. On the other hand, the Strong concealing type besides providing a stronger concealing, it also prevents and difficult even more de-concealing attempts. This type adds dust and overlaps of the information based on the input itself. As well as different characteristics for the cuts and permutations performed.

**Audio Files** Regarding the audio files, the parameters had to be different due to the nature and characteristics of the inputs. All the audio files used in the experiments had a sample rate of 44100 samples per second. The reason to choose this value for the sample rate is due to its proliferation. This is the most commonly used sample rate. For instance, CDs use this sample rate. By having a 44100 sample rate, a 20 kHz maximum frequency is achieved. Which is generally
the highest frequency audible by humans, so it makes sense to use this rate. If compared with text, that would mean 44100 characters. For this reason, the values of K and Block Size, are higher. To conceal the audio files, the following parameters were used on the algorithm:

**Block Size (B)** - The B parameter varied between 2048, 20480, 204800 and 2048000. The block size parameter is presented as B2048, B20480, B204800 and B2048000.

**Length of sequences to preserve from input (K)** - The K parameter varied between 10, 100, 1000 and 10000. The K parameter is presented as K10, K100, K1000, and K10000.

**Concealing type** - The Weak and Strong concealing types were used in the audio experiments.

The cosine similarity measure was taken in two different ways in this case. Following, there is a description of those measures of similarity: sample frequency and fixed length sequences. The notion of sample frequency is used to refer all the samples of audio. With the sample frequency analysis, it is considered that each second of the audio file has 44100 samples. Due to the high value, this analysis is made for comparison purposes. It would not be feasible to compare audio files with such a high level of detail. The scope needs to be reduced by using sequences of samples to compare. For instance, comparing sequences of 10, 100, 1000 or 10000 samples.

Audio and text files change considerably in terms of file characteristics. In a text file, a single word, or sequence of characters has much more meaning than a single sample of the audio signal - each second has 44100 samples. For this reason, fixed length sequences should be used for the audio comparison. It is fairly easy for a human to identify audio segments with, at least, approximately half a second or less of duration. However, for a computer, the analysis is more complex. Several segments of audio can be compared or mined.

In a first stage, the concealed files are compared with the original, using all the samples individually. Additionally, sequences of 10 and 100 samples are also analyzed by the computer, verifying the similarity with the original files. On a second stage, a perceptive analysis is made. By listening to the original and generated files with different values of K, it is determined if it is possible to recognize certain tracks of the original files in the generated files.

### 3.2 Results

It was found that the similarity between texts from the same author (Table 1) tends to have higher similarities than others from different authors (Table 2). It was equally found that when analyzing text with pairs of words, or even triplets, the results are more specific. The usage of this type of measure provides a way of comparison of files with specific authors or file creators. The file size was meaningful in the evaluation. The shorter the texts, the higher the difference between
Table 1. Cosine Similarity between William Shakespeare’s novels

| File          | F. Length (5) | Term Freq. | Two Cons. | Three Cons. |
|---------------|---------------|------------|-----------|-------------|
| A1 vs Hamlet  | 0.764         | 0.925      | 0.480     | 0.037       |
| A1 vs Macbeth | 0.724         | 0.902      | 0.409     | 0.021       |
| Hamlet vs Macbeth | 0.720     | 0.921      | 0.378     | 0.022       |
| Average       | 0.736         | 0.916      | 0.422     | 0.027       |

Table 2. Cosine Similarity between works of different authors - James Joyce and William Shakespeare

| File            | F. Length (5) | Term Freq. | Two Cons. | Three Cons. |
|-----------------|---------------|------------|-----------|-------------|
| Ulysses vs Hamlet | 0.718        | 0.862      | 0.454     | 0.027       |
| Dubliners vs A1 | 0.652        | 0.781      | 0.346     | 0.016       |
| A2 vs Macbeth   | 0.688        | 0.844      | 0.315     | 0.013       |
| Average         | 0.686        | 0.829      | 0.372     | 0.019       |

them. However, increasing the text length shows an increase of similarity, in all the cases due to a higher frequency of terms.

Nevertheless, the evaluation of the DNA-inspired information concealing algorithm revealed interesting findings. It was possible to show values of K which would lead to values of similarity in line with those obtained in the comparison of random texts. For instance, in the four types of analysis - term frequency, fixed length, two and three consecutive terms - the weak concealing method can preserve sequences up to five characters and still presents similarity results like random files.

In Figure 1 it is possible to observe that K3, K4 and K5 present similarity values between original and concealed files not greater than the reference values taken from the comparison between not concealed files. In the same situation, the strong concealing method could present similar results and preserve sequences up to seven or even ten characters in some cases.

Concerning the block size, as it is possible to observe in Figures 2 and 3, it was found that there were no significant changes in the similarity values obtained. The higher variations occurred when the similarity values were too low, evidencing the (small) differences. Analogously, the block size in the case of the audio experiments did not reveal substantial differences.

The execution time of the strong concealing method showed to be, in average, twice as much the time as the measured in the weak concealing method. Both taking longer execution times with small values for the K parameter. One of the findings that it is not as positive as expected, is the file size. The weak concealing method produces files three times the size of the original file (which can be acceptable). However, the strong concealing type due to its inherent concealing characteristics generates files almost twenty-four times the size of

1 All’s Well That Ends Well
2 A Portrait of the Artist as a Young Man


**Fig. 1.** Example of weak concealing method over the file Emails - Two Consecutive Terms. In the blue and pink areas, are the original (unconcealed) similarity values for files of the same author and different authors, respectively.

**Fig. 2.** Weak Concealing, Term Frequency - Cosine Similarity (Hamlet)

**Fig. 3.** Strong Concealing, Term Frequency - Cosine Similarity (Hamlet)

the original files - representing a considerable setback in terms of storage and memory demand.

Regarding the audio files, the experiments could confirm that file size ratio of the generated files is identical to text. Despite being a different file type, since the treatment given by the algorithm is identical, the ratio remains at three and twenty-four times bigger files, for the weak and strong concealing methods, respectively.

It was also shown that the cosine similarity measures taken between audio files from the same artist or different artists are higher with files from the same artist. On Table 3 it is possible to observe the similarity values between audio files from the same artist (considering all the samples). While on Table 4 it is possible to observe the similarity values between audio files from different artists.

Additionally, the experiments performed with different file lengths showed that the duration of the file has significance when verifying how similar two files
Data Concealing with a DNA-inspired Algorithm

Table 3. Cosine Similarity (all samples) between audio files from the same artist

| File                          | Cos. Sim. |
|-------------------------------|-----------|
| Man’s World / I Got You       | 0.86      |
| Man’s World / Get Up ...       | 0.96      |
| I Got You / Get Up ...         | 0.94      |
| Let’s Dance / Starman          | 0.86      |
| Let’s Dance / Life of Mars     | 0.40      |
| Starman / Life on Mars         | 0.75      |
| Interstellar / Inception       | 0.61      |
| SR1 / SR2                      | 0.99      |
| Average                        | 0.796     |

Table 4. Cosine Similarity (all samples) between audio files from different artists

| File                          | Cos. Sim. |
|-------------------------------|-----------|
| Man’s World / Let’s Dance      | 0.21      |
| Man’s World / Interstellar     | 0.61      |
| Get Up ... / Inception         | 0.69      |
| Get Up ... / Life of Mars      | 0.89      |
| Man’s World / SR1              | 0.45      |
| Get Up / SR1                   | 0.53      |
| Life on Mars / SR1             | 0.66      |
| Starman / SR1                  | 0.82      |
| Inception / SR1                | 0.86      |
| Average                        | 0.635     |

are. As in the text experiments. Also, *Man’s World* had its four sections similar to each other, resulting in high similarity values when comparing different sizes of the file. Considering the similarity between original and generated files, due to the larger number of samples, the values are close to 1. This happens with weak and strong concealing.

When the audio analysis was made by listening to the tracks and trying to identify some characteristic of the concealed files, the values of K1000 and K10000 showed that after these values it starts to be possible to recognize and differentiate characteristics. For instance, differentiate whether it is music or a person speaking.

4 Discussion

Considering what was described and analyzed in the previous sections, there are some points to consider regarding the usage of this algorithm. In terms of advantages, the following can be considered: accepting unstructured data as input, unlike many other algorithms and techniques which demand structured data like CSV or XML with organized contents; It is not necessary to define specific attributes conceal. For instance, other methods require a preparation and definition of attributes such as name or address to prepare the anonymization process; Easy and simple to deploy - choosing the concealing type (weak or strong), the value for K and Block size and the algorithm does the rest; Ability to conceal audio files and conceal the files using local data only. Not adding new symbols or samples to the file alphabet. However, there are disadvantages as well. For instance, generating files 3 times (using the weak concealing method) and twenty-four times (using the strong concealing method) larger than the original. This could be an obstacle is storage is limited for instance. Another disadvantage is the fact of accepting only mono audio files when nowadays most audio files are at least dual channel.
The execution time, two times longer in the strong concealing, it is not placed in either the previous groups. The reason is that it may be a disadvantage in some cases, and may not, in some other cases. For scenarios where performance is a must-have requirement (e.g. cloud environments), then it could represent a disadvantage. However, many of the final applications of these processes do not demand immediate data availability (i.e. real-time anonymized data). Therefore, the execution time would not be necessarily classified as a disadvantage.

Another interesting insight provided by this work is the potential usage of term frequency analysis to identify languages, due to the alphabet of the files. Although this was not experimented in this work, comparing the generated files with certain files would provide higher or lower values of similarities if the languages match, or not. Thereby, suggesting language differences or not.

In this article, we presented and discussed results of literature datasets like Shakespeare or James Joyce novels. Moreover, we obtained similar results with datasets such as email correspondence and scientific publications. As such, we consider that the undertaken experiments are representative enough.

4.1 Open Issues

Although there are several methods available, one should not search for a "one fits all" solution. The particular method we analyzed has its pros and cons (as most methods). Nevertheless, there are other aspects that should be considered when this or other methods are enforced. One aspect equally important is ensuring proper access control management before, during and after applying data anonymization methods on the private information. There are many specific situations that can be thought of. Nonetheless, there are two points of view that generally apply: the user’s side and the entity that keeps the user’s private data.

On the user’s side, taking web navigation as an example, one can always try to use safer methods like Virtual Private Network (VPN) services, anonymous browsing or not providing sensitive information at all. However, there are cases whether the user needs to provide sensitive information, or it cannot control the process (e.g. medical records or voter’s information). This part is where entities treating the data could act and provide privacy to data. Which is something that does not happen every time. Based on the aforementioned cases, there can be scenarios where either the infrastructure is secure, and the data is not treated in terms of anonymization processes. It could happen that in a possible point of failure of the infrastructure, data can be compromised.

A situation that can also happen, is an incorrect anonymization process. Although certain tables or files could be anonymized by an anonymization or concealing method, if an attacker can cross information from different sources (or files), it could be possible to identify sensitive fields. The process of anonymization or concealing data, based on the tools presented before, needs preparation. Certain tools can only anonymize data with specific formats or specific file types. This is a problem if data is obtained from different sources or has a different type than the ones supported for instance.
4.2 Suggestions and Possible Solutions

It could be beneficial if additional file types and formats would be supported. Not only for the algorithm analyzed in this work but for others mentioned before. Although performance evaluations were not done on other algorithms, as the trend with many services might be the migration to cloud computing services, the faster the better. An ideal scenario in these cases would be a near real-time anonymization or concealing process.

Prevention of data crossing is a point approach, but difficult to tackle. Cross-referencing data has the potential to look at several sources and data repositories with the attempt of discovering confidential information or unveiling identities for instance. Other than trying to guarantee that all the fields and information available are protected according to the amount of information publicly available, there is not much a person could do in order to avoid this situation. This is assuming that one or more fields would always be left partially disclosed (e.g. researching purposes). Reversibility is a feature that could be useful, depending on the use cases. However, is it not of as great importance as other possible solutions.

On the fly anonymization and concealing would be a very complete scenario. Suppose a solution merging both authentication systems and privacy requirements (such as anonymization or/and concealing) is offered. A user, data curator or system administrator could use its credentials to authenticate in a certain system and based on those credentials, have access to certain types of information from different confidential levels. The system (ideally) would then execute the necessary data privacy protection procedures automatically and generate data with the requested levels of privacy or confidentiality.

This type of methods has advantages when compared with encryption or denying access to data. By limiting or blocking the access to data or encrypting them, makes them unusable, for any situation. Which is not the intended situation for the academic community (e.g. researchers). Certain types of data should be made available for study considering the aspect of data privacy protection, which is what this kind of anonymization and concealing techniques do. Additionally, the supervision of a regulator about the enforcement and application of privacy policies could be beneficial in terms of data privacy protection.

5 Conclusion

Considering the importance of data privacy in our society, in this article, we focused on anonymization and concealing methods as well as actual experiments. We presented algorithms, metrics, and tools that can be used to perform data anonymization. One of the methods presented, the DNA-inspired information concealing algorithm, was used to conduct concealing experiments over text and audio files. The ability (not present in many algorithms) to conceal unstructured data demonstrated its usefulness not only by concealing a variety of text files, but also audio files. One of the main characteristics of the algorithm being analyzed
was the ability to deal with unstructured data, instead of tables, XML or CSV files. However, other than the analysis of the algorithm, it is evident there is still room for improvements and new ideas. Suggestions that can improve privacy protection were given. Including the example of a solution that could integrate an authentication system with anonymization and concealing methods. This suggestion is something that will be studied and analyzed in a future work. Along with a more comprehensive analysis of the state of the authentication systems and methods.

Acknowledgments. This work has been partially supported by the projects EUBRA-BIGSEA and ATMOSPHERE, funded by the European Commission under the EU/BR Cooperation Programme, Horizon 2020 grant agreement no 690116 and 777154.

References

1. Kencl, L. and Loebl, M.: DNA-inspired information concealing: A Survey, Computer Science Review (2010)
2. Slagell, A. and Lakkaraju, K. and Luo, K.: FLAIM: A Multi-level Anonymization Framework for Computer and Network Logs. Large Installation System Administration Conference (2006)
3. Mivule, K. and Anderson, B.: A Study of Usability-Aware Network Trace Anonymization. Science and Information Conference (2015)
4. Sweeney, L.: K-anonymity: A Model for Protecting Privacy. International Journal on Uncertainty, Fuzziness and Knowledge-based Systems (2002)
5. Sweeney, L.: Achieving k-anonymity privacy protection using generalization and suppression. International Journal on Uncertainty, Fuzziness and Knowledge-based Systems (2002)
6. Shannon, C.: A mathematical theory of communication. ACM SIGMOBILE Mobile Computing and Communications Review (2001)
7. Kraskov, A. and Stgbauer, H. and Andrzejak, R. and Grassberger, P.: Hierarchical Clustering Based on Mutual Information. IEEE Transactions on Pattern Analysis and Machine Intelligence (2003)
8. Davies, D. and Bouldin, D.: A Cluster Separation Measure. IEEE Transactions on Pattern Analysis and Machine Intelligence (1979)
9. Singhal, A.: Modern information retrieval: A brief overview. IEEE Data Eng. Bull. 24.4 (2001)
10. Prasser, F. and Kohlmayer, F.: Putting Statistical Disclosure Control into Practice: The ARX Data Anonymization Tool. Medical Data Privacy Handbook (2015)
11. Templ, M. and Kowarik, A. and Meindl, B.: Statistical Disclosure Control for Micro-Data Using the R Package sdcMicro. Journal of Statistical Software (2015)
12. Poulis, G. and Gkoulalas-Divanis, A. and Loukides, G. and Skiadopoulos, S. and Tryfonopoulos, C.: SECRETa: A System for Evaluating and Comparing RElational and Transaction Anonymization algorithms. 17th International Conference on Extending Database Technology (2014)
13. Bochkarev, V. and Shevlyakova, A. V and Solovyev, V.: Average word length dynamics as an indicator of cultural changes in society. arXiv preprint arXiv:1208.6109 (2012)
Towards an Independent and Resilient DNS

Moritz Müller and Roland van Rijswijk-Deij

Stichting Internet Domeinregistratie Nederland (SIDN)
and University of Twente, The Netherlands
{m.c.muller, r.m.vanrijswijk}@utwente.nl

Abstract. The Domain Name System (DNS) is the crucial naming system of the Internet. Before clients can establish a connection with a service they look up its address in the DNS. Therefore, DNS operators have a large responsibility and a lot of power: if the DNS is not available, clients effectively cannot establish a connection to a service. Also, DNS operators can track and manipulate requests by clients. This can leak personal information, lead to censorship and cause security issues. DNS is designed to be in the hands of many. This makes it more resilient against outages and independent from parties that misuse their power. In practice, however, the DNS becomes concentrated at a few providers – often with negative consequences: e.g., when the DNS provider Dyn suffered an outage in 2016, many services on the Internet went offline. In this PhD research we aim at studying and improving the resilience and independence of the DNS infrastructure. We analyze where on the Internet the DNS is concentrated and discuss the impact of these points of concentration on the resilience and independence. Further, we propose measures to mitigate these impacts.

1 Introduction

The Domain Name System (DNS) is the naming system of the Internet. It is critical because a lookup in the DNS precedes almost every connection setup.

By design, the DNS is a distributed system. Its name-space is hierarchical, where each zone (the root, .com, and example.com) is served by authoritative name servers. These are queried by recursive resolvers, which clients use to lookup information in the DNS.

Figure 1 demonstrates such a lookup. Assume the client wants to visit the website example.com. To do so, it first needs to look up the IP address of example.com in the DNS. It employs a recursive resolver that walks through the DNS hierarchy until it reaches the name server that is authoritative for the requested domain. The server responds with the IP address and the resolver returns it to the client. Finally, the client uses the address to connect to the website.

This architecture gives DNS operators a lot of responsibility but also a lot of power: on one side, operators of zones like the root, .com, or example.com have the responsibility to keep their name servers available all the time. If in Figure 1, none of the name servers of example.com are reachable, the services of example.com become effectively unavailable. This responsibility becomes even larger if the operator manages zones with many domains, like the root or a TLD.
Fig. 1. An example of a DNS query and its involved components and stakeholders.

On the other side, operators of recursive resolvers have a lot of power over their clients. First, operators can track every DNS request, which can reveal sensitive and possibly confidential information [1]. Second, because a DNS request precedes virtually every connection setup on the Internet, resolver operators are gatekeepers. They can block access to certain domains or serve their clients a wrong answer. Last, they also need to keep their resolvers available all the time. If clients cannot reach a resolver to serve their queries, they become effectively offline.

To distribute the responsibility and power, the DNS is designed to be in the hands of many [2]: as shown in Figure 1, clients can select from multiple resolvers. Also, each domain can have multiple name servers and each server can be operated by a different provider.

This design should make the DNS resilient against outages and avoids that every component is controlled by the same organization. If the authoritative name server x in Figure 1 of example.com fails, then there are still two other servers available. Also, if the client does not trust resolver 1 to serve the correct answers it can choose resolver 2 instead.

Incidents in the past, however, indicate that the actual implementation of the DNS does not follow these principles anymore. A major Distributed Denial of Service attack on the DNS provider Dyn in 2016 rendered many popular web services unavailable for hours [3] and affected domains like twitter.com, linkedin.com and airbnb.com. All of which had multiple name servers configured,
but every one of them was hosted at Dyn. When Dyn became unavailable, so did every name server.

Similar trends have been observed at recursive resolvers as well. Internet service providers (ISPs) are forced by governments to block DNS requests, e.g. to adult content [4]. At the same time they make it harder for their customers to choose other resolvers than the one provided by the ISP [5]. If the ISP has to block certain queries at their resolvers a (less tech-savvy) user has no other choice than to accept this censorship.

This indicates, the DNS in fact becomes more concentrated at a few organizations. This apparently has negative implications for the resilience and independence of the DNS and thereby, the Internet as a whole.

Several proposals have tried to address at least parts of these issues but usually involved a complete re-design of the DNS and did not see any wide spread deployment[6].

In this PhD research we carry out a comprehensive study of all aspects that influence the resilience and independence of the DNS. Our goal is to analyze (RQ1) how concentrated the DNS is, (RQ2) study its implications on the resilience and independence and discuss (RQ3) countermeasures that can be applied to the existing DNS architecture.

Previous research shows that name servers of popular domain names are largely concentrated at a few providers [7]. We choose a broader view: we carry out a comprehensive study of the domain name space and take all aspects that influence the resilience and availability of the DNS into account.

2 Goal, Research Questions, and Approach

The ultimate goal of this research is to:

**Goal** Study and (further) improve the resilience and independence of the DNS infrastructure.

To achieve this goal we have to answer three research questions: in RQ1 we lay the foundation for RQ2 and RQ3.

**RQ1** How are components of the DNS distributed across networks, organizations and states?

The purpose of this research question is to measure if and where on the Internet the DNS is concentrated and examine its drivers. Concentration can occur on a technical, organizational and national level and can be motivated by technical, economical or social drivers.

We will study the following components: (a) the recursive resolvers (boxes in the middle column in Figure 1), (b) the authoritative name servers (right column), (c) the network links that connect the components (bold lines), and (d) the managing organizations and countries.
We, among others, use the active DNS measurement platform OpenINTEL to map which name servers are authoritative for the domain space. We want to understand in which networks these servers are located, who is responsible for running the servers, and under which jurisdiction those organizations fall.

Similar research is necessary for recursive resolvers. In order to study their concentration, we need to understand, which resolvers are “important” for the DNS eco-system. Based on passive DNS query data, collected at TLDs and the root, as well as query data of recursive resolvers, we will develop and evaluate a methodology to identify resolvers that serve large parts of the Internet users [8,9].

As soon as we have identified these resolvers, we measure where they are located in the network infrastructure and who is responsible for them. This helps us to answer the question:

**RQ2** Is the current DNS infrastructure a threat to the resilience and independence of the DNS?

We develop threat scenarios and apply them to the points of concentration, identified in RQ1. In these scenarios the availability and integrity of the DNS, but also the privacy of its users are threaten. We base them on realistic threats observed in the wild, for example: 3rd-parties that want to take down critical components of the DNS, but also the operators of these components. They manipulate the DNS, e.g. for their own interest or because they are forced by a nation state, and can cause unintentional outages or changes in the DNS through mis-configuration.

Because we have identified the important components of the DNS in RQ1 we can measure the impact of the identified threats on the users of the DNS. These threats lead us to the third and last research question:

**RQ3** Which countermeasures are suitable to (further) increase the resilience and independence?

We focus on countermeasures that address the most harmful and probable threats, identified in RQ2 and discuss countermeasures on a technical, operational and organizational level. In our approach, we assess the effect and practicality of existing measures using testbeds and production systems (e.g. at the Dutch ccTLD operator). Further, our goal is to design novel approaches as well that have not been proposed at standardization organizations or implemented before. Countermeasures include mechanisms to make resolvers more resilient against outages of name servers, methodologies to reduce the risk of operational failures, or regulatory measures that split up points of concentration.

We demonstrate that we have achieved our research goal by using qualitative and quantitative metrics. We deploy our proposed countermeasures at resolvers and name servers in production and can rely on anecdotal evidence from operators to prove their efficiency.

---

1. [https://www.openintel.nl](https://www.openintel.nl)
3 Preliminary Results

In the previous months, we took first steps to answer RQ2 and RQ3. To identify threats in RQ2 we need to understand how resolvers interact with name servers. We measured their behavior and observed that more than 5% of the resolvers in the wild send every query to only one name server [10]. If this server becomes unavailable the resolvers are, temporarily, unable to receive an answer to their query. These resolvers threaten the availability of the DNS.

One reason for unavailable name servers are mis-configurations of the DNS Security Extensions (DNSSEC). DNSSEC is designed to protect the integrity of the DNS but can also cause outages if not configured correctly. We developed and tested a methodology that reduces the risk of outages for DNSSEC operators [11]. Thereby, we increase the resilience of the DNS and contribute to RQ3.

Acknowledgment This PhD research is carried out in part time with the support of SIDN (https://www.sidn.nl) and is supported by SIDN Labs and NLnet Labs through the project Self-managing Anycast Networks for the DNS (SAND) project, phase 3 (http://www.sand-project.nl).

References

1. Bortzmeyer, S.: DNS Privacy Considerations. RFC 7626 (Informational) (2015)
2. Mockapetris, P.: Domain names - concepts and facilities. RFC 1034 (November 1987)
3. Dyn Inc.: Incident Report: DDoS Attack Against Dyn Managed DNS. https://www.dynstatus.com/incidents/nlr4yrr162t8 (2016)
4. BBC: Q&A: UK filters on legal pornography. http://www.bbc.com/news/technology-23403068 (2013)
5. Edmonds, Robert: Disappearing Choice of Recursive DNS Services in Home Networks. https://indico.dns-oarc.net/event/28/session/11/contribution/58/material/slides/0.pdf (2018)
6. Kalodner, H.A., Carlsten, M., Ellenbogen, P., Bonneau, J., Narayanan, A.: An empirical study of namecoin and lessons for decentralized namespace design. In: WEIS. (2015)
7. Bates, S., Bowers, J., Greenstein, S., Weinstock, J., Zittrain, J.: Evidence of Decreasing Internet Entropy: The Lack of Redundancy in DNS Resolution by Major Websites and Services. Working Paper 24317, National Bureau of Economic Research (2018)
8. Wullink, M., Moura, G.C., Müller, M., Hesselman, C.: ENTRADA: A high-performance network traffic data streaming warehouse. In: Network Operations and Management Symposium (NOMS), 2016 IEEE/IFIP, IEEE (2016) 913–918
9. DNS OARC: DITL Traces and Analysis. https://www.dns-oarc.net/oarc/data/ditl2017 (2017)
10. Müller, M., Moura, G.C.M., de O. Schmidt, R., Heidemann, J.: Recursives in the Wild: Engineering Authoritative DNS Servers. In: Proceedings of the 2017 Internet Measurement Conference. IMC ’17, New York, NY, USA, ACM (2017) 489–495
11. Müller, M., Chung, T., van Rijswijk-Deij, R.: Rolling with Confidence: Managing the Complexity of DNSSEC Operations (Poster). In: ICT-Open Proceedings. (2018)
RAN SLICING FRAMEWORK AND RESOURCE ALLOCATION IN MULTI-DOMAIN HETEROGENEOUS NETWORKS

Godfrey Kibalya, Joan Serrat and Juan-Luis Gorricho
Universitat Politècnica de Catalunya, 08034 Barcelona, Spain
Godfrey.mirondo.kibalya@upc.edu, serrat@tsc.upc.edu, juanluis@entel.upc.edu

Abstract. The Third-Generation Partnership Project (3GPP) has identified network slicing as one of the key technologies to address the complex communication scenarios in the 5G network. With the 5G commercialization target dates fast approaching, there are a plethora of challenges that need to be addressed regarding 5G slicing especially in the Radio Access Network (RAN). Consequently, this paper highlights the key research challenges being addressed by the Ph.D. thesis proposal. Specifically, we propose a virtualized RAN slicing framework considering a multi-RAT, multi-domain heterogeneous network and a two-level resource allocation scheme at each domain. Within a slice, we adopt a novel concept of slice profiles to address the heterogeneity challenges associated with the 3GPP notion of a slice. We also propose an entity called cross-domain coordinator to coordinate inter-domain resource sharing. The paper discusses the approaches to be adopted in formulating the intra-domain and inter-domain resource sharing algorithms.

Keywords: RAN Slicing, Multi-Domain Resource Sharing, Heterogeneous Network Resource Allocation

1 Introduction

The International Telecommunication Union (ITU) has already classified 5G mobile network services into three categories which are [1]: Enhanced Mobile Broadband (eMBB), Ultra-reliable and Low-latency Communications (uRLLC), and Massive Machine Type Communications (mMTC). Considering that these service categories have different characteristics and technical requirements, the 3GPP proposes to extend a logical dedicated network in form of network slice suitable for meeting the technical requirements of each of the above service classes [2].

Network slicing and virtualization presents a plethora of challenges. Besides the complexity related to how the virtual resources are mapped onto the physical resources, another challenge is on how to effectively allocate the virtualized resources to the different slices with differing requirements. Moreover, the access network wireless resources are highly dynamic and variable both in temporal and spatial domains. This requires that resource allocation mechanisms capture the dynamicity of the physical...
resources. The problem will further be complicated by the emergency of heterogeneous networks constituted by multiple Radio access technologies (RATs). The different RATs offer different capabilities towards the services supported by the different slices in terms of technical requirements. Consequently, the Base Station (BS) association problem needs to map the slice technical requirements with BS capabilities. Moreover, the different BSs may belong to different operators or domains, which brings in another dimension of inter-domain resource sharing of the virtualized resources. Another unexplored challenge is the heterogeneity within the aforementioned service categories. Within each of these categories, there are users with differing technical requirements hence requiring innovative approaches if these users are to exist in same slice.

The problem to be addressed in the thesis involves how to create and manage slices consisting of a set of radio resources in a multi-technology and multi-domain wireless access network so that services can be deployed on top of these resources. The proposed framework exhibits the following key characteristics: 1) Isolation to safeguard against mutual interference between slices. 2) Elasticity by enabling flexible and dynamic allocation, reclaiming, and release of resources to optimize temporal and spatial resource usage. 3) Authentication to enable authorized use of the virtual resources between slices.

This paper is structured as follows: Section one has briefly introduced the slicing concept and the key research issues unexplored in literature. Section two presents state of the art in RAN slicing resource allocation. Section three introduces the proposed slicing framework and a summary of proposed contributions indicated in section four. The paper is concluded in section five.

2 State of the art

A big number of works treat a slice as a virtual network (VN) belonging to Mobile virtual network operator (MVNO) and the aim is to allocate resources to meet the minimum data-rate requirements in the different VNs as in [3] and references therein, or match the financial contribution of the network operator as in [5]. Besides not addressing the heterogeneity within the 3GPP slice notion, these works do not consider a multi-RAT setting nor do they consider multi-domain resource allocation. Also, the Base station is not considered as a resource to be allocated to users of the different VNs. In addition, these works adopt a single level resource allocation scheme. The work in [12] and the associated references propose a two level slicing model for allocating resources to multiple tenants under heterogeneous network. These works however, consider heterogeneity in terms of only BS coverage thus Macro and small cells and not on basis of RATs. These works also do not address the heterogeneity challenges in the slicing concept. The work considering Multi-RAT network is found in [14] and references therein. These also however, consider the RAT effect in terms of their capacity contribution to the network and not in terms of their capability towards different services.

To the best of our knowledge, there is no work addressing multi-domain resource allocation in RAN slicing. Another novelty in our work is the concept of slice profiles to address the heterogeneity challenges in the slicing concept. Moreover, different from other works we consider the effects of the different RATs towards services supported by slices.
3 Proposal of a slicing management architecture

We propose a two-level resource allocation framework as shown in Fig1. As indicated, the physical radio resources of a given domain are virtualized into a VRRP in a 3-D grid of time, frequency and space. The VRRP of a given domain is under the control of the domain slice manager (DSM) with each domain being characterized by a single DSM.

The proposed framework supports Inter-domain resource sharing through the cross-domain coordinator (xDC) connected to the different DSMs. In particular, whenever the DSM runs out of resources or wants to establish connection in a different domain, it sends request to the xDC specifying possible constraints on the required resources such as type of resources, SLA with its clients etc. The xDC then queries the auction databases of the different DSMs for any resources meeting the specified constraints. If resources are found, then the request is served, otherwise request is rejected. The auction database contains details of resources such as type and amount that a given DSM is willing to auction to another domain at any given instant. This database can be updated upon request from the xDC or updated automatically by the DSM. The DSM may have resource sharing agreements with specific domains and with different sharing terms. Within this regard, we propose to develop an inter-domain resource sharing policy and algorithm capturing these possible aspects.

The DSM is assumed to have global view of the virtual resources available within its domain through the different monitoring modules. It manages resource allocation at upper level by dynamically allocating resources to different slices through the local slice manager (LSM) of the slices. In order to consider isolation between slices, each slice is modeled with maximum fraction of available resources it can use under full domain resource utilization. But when a given slice is not fully utilizing its resource share, these can be borrowed by other slices. In such a case, we shall define an optimal pre-emption mechanism to guide on release of the resources when the entitled slice requests for them. We intend to formulate the resource allocation problem as an optimization problem so as to maximize resource utilization across the different slices subject to slice requirement constraints and available resources.

At the lower level, the resource allocation is performed by the LSM and this is modeled into three stages thus, BS assignment to the user, associating a user to the appropriate profile of the slice, and assigning radio resources to the user. We envisage two approaches for BS association. In the first approach, the association problem is modeled as a knapsack problem. The second approach will adopt an intelligent online technique to capture more automated intelligence and to enable multi-attribute analysis. We model the effects of different RATs towards the technical requirements of a user belonging to a given slice. With respect to this level, the thesis will develop algorithms for BS and radio resource allocation to the different profiles and users.

The LSM manages a number of modules which are not shown in Fig1 due to space constraints. These include: 1) The monitoring module which captures the network condition such as number and QoS of admitted users, resource utilization etc. This information is stored in local database and used by the learning module to update decisions regarding the BS association, profile association and resource allocation. 2) The RAT selection module responsible for selecting the BS to which to associate the different users considering the user service requirements, RAT capability etc. This module runs
the BS association algorithm. 3) The Profile selection module that interacts with the monitoring module and the learning module to decide the slice profile to which to associate the user. Moreover, it is possible to move the user from one profile to another depending on the prevailing network conditions. 4) The resource assignment module in charge of managing resource assignments from the slice resource pool to the different profiles and withdraw of resources from one profile to another 5) The learning module which is an intelligent module that exploits the monitoring information to influence the next decisions regarding user admission, profile selection and resource allocation.

Fig. 1. RAN Slicing Framework

4 Proposed Innovations

The thesis is being developed working on a number of proposals targeting the implementation of RAN slicing in 5G networks. Specifically, we propose the following:

A practical RAN slicing framework that is commensurate with the slicing concept as envisioned by the 3GPP while addressing the heterogeneity challenge that arises within the concept. In particular, we introduce the concept of profiles within a slice. In the profiling concept, users of a slice that have same technical requirements are assigned to the same profile that is associated with a share of virtual resources and priority level.

A two-level resource allocation algorithms. At the upper level, the virtualized resource pool is under the control of an entity called a Domain Slice Manager (DSM) that allocates resources to the different slices of its domain. At the lower slice level, the Local Slice Manager (LSM) is in charge of allocating resources received from DSM to the users of the different profile. We propose algorithms for each of these levels.

A dynamic algorithm to determine the profile to which to attach an admitted user. The algorithm will exploit intelligent learning techniques to capture the dynamic nature of the slice conditions. Moreover, an approach based on prioritization and pre-emption
mechanism for congestion and emergency management of critical services is also proposed.

A dynamic online resource sharing policy for sharing resources across multiple domains where the domain belong to different entities. Additionally, we propose an embedding algorithm for embedding the virtual radio resources to physical resources.

Different from other works, we consider a heterogeneous environment comprising of both small cells and macro cells belonging to different RATs. Towards this direction, we propose an online BS association algorithm that captures the capability of different RATs towards the users of different slices.

4. Conclusion

The slicing of the RAN is envisaged as a key step towards achieving end-to-end network slicing leading to a reduction in CAPEX and OPEX for operators. The slicing solutions should be tailored towards a practical 5G environment in which the slicing is bound to be implemented. However, slicing solutions in literature do not consider the heterogeneity of future networks in terms of cell sizes, RAT types and multi-domain ownership. Moreover, the works do not appreciate the diversity of services that will characterize the 5G network. Towards this direction, once our approaches proposed in this paper are realized, it will contribute towards realizing network slicing in the 5G RAN.

Acknowledgement

This work has received funding from the European Union’s Horizon 2020 research and innovation programme under grant agreement No 777067 (NECOS project). This work is also funded by the national project TEC2015-71329-C2-2-R (MINECO/FEDER).

References

1. EU H2020 5G NORMA, “D2.1: Use cases, scenarios, and requirements,” September 2015
2. K. Samdanis et al., “5G Network Slicing: Part 1 – Concepts, Principles, and Architectures.”, IEEE Communications Magazine, Vol. 55, pp 70-71, May 2017
3. S. Parsaeefard, R. Dawadi, M. Derakhshani, T. Le-Ngoc,: Joint User-Association and Resource-Allocation in Virtualized Wireless Networks. IEEE Access, Vol.4, pp. 2738 –2750 (2016). Doi: 10.1109/ACCESS.2016.2560218
4. P. Caballero et al.: Multitenant Radio Access Network Slicing: Statistical Multiplexing of Spatial loads. In IEEE/ACM Transactions on Networking, vol.25, no.5, pp.3044-3058, Oct.2017. DOI:10.1109/TNET.2017.2720668
5. Y. L. Lee, J. Loo, T. C. Chuah and L. C. Wang,: Dynamic Network Slicing for Multitenant Heterogeneous Cloud Radio Access Networks. In IEEE Transactions on Wireless Communications, vol.17, PP, no. 99, pp. 1-1. IEEE Transactions on Wireless Communications, vol. 17, no. 4, pp. 2146-2161, April 2018. Doi: 10.1109/TWC.2017.2789294
6. S. Khatibi and L.M.Carreia,: A model for virtual radio resource management in virtual RANs. EURASIP J.Wireless Commun.New.vol. 2015, no.68, pp.1-12, Mar.2015. DOI 10.1186/s13638-015-0292-7
Leveraging Smart Contracts for Automatic SLA Compensation – The Case of NFV Environments

Eder John Scheid and Burkhard Stiller

Communication Systems Group (CSG), Department of Informatics (IFI)
University of Zürich (UZH), Binzmühlestrasse 14, CH 8050 Zürich, Switzerland
{scheid,stiller}@ifi.uzh.ch

1 Introduction and Motivation

Service Level Agreements (SLA) describe the requirements that Service Providers (SP) must meet when delivering a specific service to clients [3]. For example, in cloud computing, an SLA can describe that a database server must have 99.99% of availability. Moreover, for the same database server, an SLA can state that the server throughput must be greater than or equal to 1Mbps; otherwise, the client receives 10% of its payment back. The interaction between a client and an SP when specifying SLAs is performed within Operational Support Systems (OSS) and Business Support Systems (BSS). However, even with the employment of such systems, some tasks, e.g., SLA specification, and compensation, still require manual effort and interaction to be accomplished. Such manual interaction hinders service agility and is prone to errors.

If an SLA is violated (e.g., throughput less than 1Mbps) during the contract duration, then the client is entitled to compensation (e.g., 10% payment back). The compensation process is costly, bureaucratic, and involve dedicated personnel in case of a dispute [6]. Moreover, this process requires trust between the involved parties. On the one hand, the client must provide evidence (i.e., data) that the SP is not delivering what was agreed, and thus have to allocate resources for monitoring the contracted service. On the other hand, the SP must also monitor its services to provide evidence that they are being properly provided. Therefore, reducing the number of third-parties in SLA compensation process, while providing trust in the data to involved parties, is important to reduce costs and minimize unnecessary resource allocation.

Blockchains and Smart Contracts (SC) remove trusted third parties by decentralizing and replicating the data throughout all participants in the network. A blockchain is an append-only ledger where the data can only be inserted, but cannot be removed or changed. A SC is an executable code that runs on a given blockchain to facilitate, execute, and enforce agreements between untrusted parties [2]. Moreover, both technologies (blockchains and SCs) rely on cryptography, and consensus mechanisms to secure the data against tampering.

Blockchain-based SCs may be an option for the management of SLAs by automating compensation-related tasks while providing trust. The employment of blockchain-based SCs addresses two crucial aspects: the guarantee of contract
enforcement, and the immutability of the data. In the SLA compensation context, the former aspect assures the subscriber that it will receive compensation in case of an SLA violation. In contrast, this aspect also assures the SP that the client will pay the subscription fee. Moreover, the latter aspect assures to both parties, in the case of a trusted monitoring solution, that the monitored data cannot be altered after being appended to the blockchain. Thus, both aspects can aid to simplify the compensation in the case of an SLA violation.

2 Problem Description

Currently, the process of managing SLAs is cumbersome and involves different stakeholders, such as system administrators, incident response teams, managers, and end-users. The time that it takes between SLA violation detection and payment of monetary compensations impacts directly on the operational expenses of the involved organizations.

Taking the Amazon Compute SLA [6] as an example, the compensation process (i.e., requesting credits) for the violation of the uptime SLA involves: (i) opening a formal case in the AWS Support Center, (ii) submitting a claim with “SLA Credit Request” in the subject line, (iii) informing dates and times of each unavailability incident, and the affected services (instances or volumes), and (iv) sending request logs (replacing or removing sensitive information) that document the errors and that corroborate the claimed outage. If the request is confirmed by the responsible team, then the service credit will be paid (within one billing cycle following the month in which the request was confirmed). If one fails to provide any necessary information or the team does not acknowledge the claim, then the credit will not be paid. This process is inefficient and prone to errors because of the lack of trust between both parties, the manual interaction required by the subscriber, and its complexity.

2.1 Research Questions

The present work aims to explore the use of blockchain-based SCs to automate SLA compensation process, which currently is manually performed and cumbersome. Within this context, three major aspects are to be investigated during the development of the Ph.D.: (i) feasibility to automate the process, (ii) complexity of the process, and (iii) integrity of the data involved in the process. Thus, it is expected to answer the following research questions over the Ph.D course:

(i) How long does it take to detect an SLA violation and pay the defined compensation to the subscriber using SCs?
(ii) How complex, regarding human-computer interaction, is the proposed approach compared to existing solutions?
(iii) How to monitor the terms of the SLA and resources while providing trust in the monitored data to involved parties?
3 Approach

To address the issues related to the management of SLAs, an approach to automatically manage the payment and compensation of SLAs using SCs running on a blockchain is presented herein. To provide a straightforward proof-of-concept, the approach is described using NFV-related SLAs as an example. Nevertheless, the presented approach can be applied to different types of contexts.

Blockchain-based SCs can be programmed to execute immutable agreements between two parties that do not necessarily trust each other, such as a subscriber and an SP. Thus, it is logical that SLAs can be translated into SCs. The proposed approach considers the use of a blockchain capable of running a Turing-complete language, i.e., it is possible to implement any computational program using the provided language. Therefore, this Turing-complete SC language increases the spectrum of functions of the compensation process that can be implemented in an SC, such as implementing a formula to calculate the exact reimbursement amount based on a variable number of inputs. One example of such a Turing-complete language can be found in [6]. Figure 1 depicts an overview of the architecture of the proposed approach.

![Figure 1. Overview of the proposed general architecture](image)

In the proposed approach, a subscriber can request the desired resources to an SP and specify the SLAs that must be complied. The SP, in turn, have to deploy an SC in the blockchain (using the Blockchain Adapter) that contains, amongst other information: (i) the translated SLAs in the form of code, (ii) the details of the requested resources, and (iii) the duration and price of the services. This SC will only be enforced when the subscriber transfers the necessary funds to the SC address; this means that it is accepting the terms (i.e., SLAs) of the SC and subscribing to the service. Once the deposit transaction is accepted by the network, included in the blockchain, and acknowledged by the SP, the SP dispatches an event to the Resource Manager to deploy the requested resources in the Physical Infrastructure. When resources are deployed, the Resource Manager can update the SC, so that the subscriber is aware that the deployment
process is completed and that it can start to use resources. This interaction between the Resource Manager and the SC occurs using the Blockchain Adapter component, which implements an Application Programming Interface (API) to send transactions to the blockchain. Moreover, the Blockchain Adapter is placed within the OSS/BSS to support different Resource Managers and Resource Orchestrators. Thus, this component is agnostic to the technology used by the SP Resource Infrastructure to deploy resources in the Physical Infrastructure. One open issue of this approach is the monitoring of resources to ensure that the agreed SLAs are not being violated, this issue is discussed in [6].

In the Network Function Virtualization (NFV) [1] context, which proposes to virtualize and host physical middleboxes on standard hardware, SLAs can specify that Virtualized Network Functions (VNF) or a chain of such VNFs must provide a specific throughput to achieve a required Quality of Service (QoS). For instance, a subscriber of a security-related service chain (e.g., a firewall, and a Deep Packet Inspection (DPI)) can define an SLA stating that this chain should process $x$ packets per second. Otherwise, if the condition is not met, a financial compensation must be paid.

A high-level proposal for the interaction steps of the approach using VNFs as a resource example is described in [6]. The first step is the request of desired VNFs by the subscriber to the SP, informing VNF Descriptors and SLAs. The SP in turn codes and deploys the SC in the blockchain containing the information received from the subscriber. After the SC is appended in the blockchain, the SP sends the address of the SC to the subscriber which deposits the funds in the SC, subscribing to the service and agreeing on the terms of the SLAs. Once the SP acknowledges the deposit in the SC, it requests for the deployment of VNFs by the NFV Infrastructure (NFVI). The NFVI deploys the VNFs and updates the SC with the information regarding the deployed VNFs (e.g., IP address, port number, and status). With the information stored in the SC, the subscriber can redirect its network traffic through the VNFs. Trusted monitoring agents within the NFVI continuously monitor the VNFs checking for any SLA violation, if any violation is encountered, they send this information to the SC so that the subscriber can receive the defined compensation, if no violation is encountered and the contract is not expired they continue the monitoring. Otherwise, if the contract has expired, then the SP can receive the funds stored in the SC.

4 State-of-the-Art

Regarding the application of SCs to support the management of SLAs, [4] proposes a formal description of Web API alongside with the specifications of related SLAs. Moreover, the authors propose an SLA contract method built on top of Ethereum. However, the proposed solution focuses on Web API-related SLAs only. Moreover, [5] proposes an SC, also on top of Ethereum, to implement SLAs in the Small-Cell-as-a-Service context. The SLA implemented in the SC relates to an individual home or business user providing a service for mobile network
operators. Even though this solution implements SLA in SCs, the authors do not delve into the details of how the monitoring of the terms is performed.

The efforts of these two approaches point towards that it may be feasible to translate SLAs into SC and to automate some SLA management steps that are being currently performed manually, such as the compensation process previously described. However, some aspects still need to be researched, especially regarding the complexity of employing blockchain-based SCs to solve this issue, and the integrity of the monitored data utilized to verify whether the SLAs are being complied or not.

5 Discussion and Next Steps

The approach herein presented is based on SCs for an automated compensation of SLAs. It automatically enforces the payment of the defined subscription fee or the monetary compensation in case of an SLA violation. Supporting the proposal of the approach, it was presented a proof-of-concept describing the SLA compensation process between a subscriber and an NFV SP being performed through an SC. Moreover, different facets of SLA management can be incorporated in the approach, such as SLA negotiation, and contract definition, enhancing the features provided by current OSS/BSS. Thus, it is expected that with the development of this approach further aspects of SLA management can be improved, reducing the bureaucracy, costs, and involved parties.

Furthermore, the research on the employment of SCs to automate the compensation of SLAs and payment of subscriptions contributes not only to the SLA management area but the overall study on blockchains and related applications, which is still in its infancy. Next steps include, but are not limited, to implement the approach, conduct a qualitative and quantitative evaluation of the implementation, and seek to answer the listed research questions.

References

1. Network Functions Virtualisation (NFV). White Paper (October 2012). Available at https://portal.etsi.org/nfv/nfv_white_paper.pdf Accessed 08 January, 2018
2. Alharby, M., van Moorsel, A.: Blockchain-based Smart Contracts: A Systematic Mapping Study. CoRR abs/1710.06372 (2017)
3. Mubeen, S., Asadollah, S.A., Papadopoulos, A.V., Ashjaei, M., Pei-Breivold, H., Behnam, M.: Management of Service Level Agreements for Cloud Services in IoT: A Systematic Mapping Study. IEEE Access PP(99), 1–1 (2017)
4. Nakashima, H., Aoyama, M.: An Automation Method of SLA Contract of Web APIs and Its Platform Based on Blockchain Concept. In: 2017 IEEE International Conference on Cognitive Computing (ICCC), pp. 32–39 (June 2017)
5. Pascale, E.D., McMenamy, J., Macaluso, I., Doyle, L.: Smart Contract SLAs for Dense Small-Cell-as-a-Service. CoRR abs/1703.04502 (2017)
6. Scheid, E.J., Stiller, B.: Automatic SLA Compensation based on Smart Contracts. Technical Report IFI-2018.02 https://files.ifi.uzh.ch/CSG/staff/scheid/extern/publications/IFI-2018.02.pdf, Universität Zürich, Zürich, Switzerland (April 2018)
Towards European Network Sovereignty

Nils Rodday\textsuperscript{1,2}, Aiko Pras\textsuperscript{2}, and Gabi Dreo Rodosek\textsuperscript{1}

\textsuperscript{1} Universität der Bundeswehr München, Neubiberg, Germany
\{nils.rodday, gabi.dreo\}@unibw.de
\textsuperscript{2} University of Twente, Enschede, Netherlands
\{aiko.pras\}@utwente.nl

Abstract. Almost all major remote communication in Europe today is digital and routed through the Internet. With a very limited set of enterprise networking gear vendors, Europe’s dependency on foreign networking technology is shockingly high. Through the Snowden revelations we have learned that foreign manufacturers have to collude with intelligence services to install backdoors and make their hardware remotely accessible. This PhD proposal offers a way towards European Network Sovereignty, by first designing a fingerprint to detect the router manufacturer, second finding alternative routes to avoid non-trusted routers in the routing infrastructure and third investigating the potential of Open Source concepts in the enterprise networking market.

1 Introduction

Today’s world is heavily relying on the Internet as a medium to communicate and exchange information. Critical infrastructures, cars, smart homes and, of course, almost every person in the modern world that is using a smartphone, tablet or PC is connected to the Internet of Things (IoT) that is estimated to consist of more than 20 billion devices by 2020 \cite{1,2}. The routing infrastructure of the Internet itself is built upon networking devices that are being produced by very few international companies including e.g. Cisco (U.S.), Juniper (U.S.), HPE (U.S.), and Huawei (Chinese) \cite{3}. Since these countries follow their own national interests and some of the systematic state sponsored wiretapping and eavesdropping programs have become public \cite{4}, the security of the infrastructure has to be questioned, as not only single entities but whole countries are totally dependent on its confidentiality, integrity and availability.

We have seen in recent years, especially through the Snowden leaks, how government agencies ensure their access to information. Several backdoors being present for years in networking devices have been documented \cite{5,6}. Moreover, it was reported, that backdoors are actively installed by the NSA in networking devices that are shipped to clients outside of the U.S. \cite{4}. On the other hand, the U.S. government is suspecting other governments to do likewise and banned Chinese networking gear as well as Russian anti virus products from use in U.S. government facilities \cite{7,8,9}. Additionally, fiber optic cables transferring data...
between Internet nodes have been proven to be tapped on by government agencies \cite{6}. Considering the huge amount of unveiled surveillance activities, it is very likely that networking hardware that is actively used these days in European data centers is compromised and can therefore be monitored or shut down by external parties, e.g. via a kill switch, to interrupt communication.

2 Research Problem and Questions

To get closer to European Network Sovereignty we first need to be able to identify the manufacturers of Internet routers between source and destination. If a manufacturer is deemed untrustworthy the second step is to establish an alternative route only containing trusted networking devices. To solve the overall problem of trust issues between a country using foreign networking equipment and the vendor under the influence of his government, the third step is to change the current model of how networking gear is manufacturered and where responsibilities lie. We believe, that the concept of Open Source is a solution to this problem. We therefore define the following research questions:

1. (Detect) How should a digital fingerprint be designed that makes the detection of the manufacturer of an enterprise networking router/switch possible?
2. (Avoid) How can routing decision in the Internet be influenced to avoid untrusted networking devices?
3. (Replace) What benefit can Open Source technology provide in the reduction of dependencies and resolve of trust issues?

3 Approach

In order to answer research question number one, we will look at existing tools and techniques of detecting and fingerprinting IT hardware within networks in general, and then drill down on the specifics of enterprise networking equipment. The most important tool to mention here is Nmap, an open source network security scanning tool which is considered the swiss army knife within the network security community \cite{6}. One challenge that can already be foreseen at the current state and has to be overcome is that enterprise providers like to keep their infrastructure details confidential and therefore try to block any scanning related activities within their networks. Another approach is to fingerprint routers by observing their normal behaviour (e.g. ICMP messages), instead of scanning the device itself (TCP/UDP ports). One study looked at initial TTL values of ICMP replies in response to certain packets to fingerprint routers \cite{7}. Moreover, the same authors used again ICMP replies to detect middleboxes by comparing the original packet with a copy of the packet encapsulated in the ICMP reply \cite{8}. Apart from initial TTL values, there has no other literature been found on other features that could be utilized to fingerprint a router, but the authors in \cite{7} suggest additional research potential. Also, we plan to utilize projects such
as PlanetLab \[14\] with numerous vantage points to perform scans of Internet Service Provider (ISP) networks in order to build an inventory of currently used devices.

Once an identification of enterprise networking hardware is possible, the next step is to investigate if we are able to influence the routing infrastructure in such a way that untrusted routers on the route to our packet destination can be avoided. This answers research question number two. As the routing protocol Border Gateway Protocol (BGP) is required of most ISPs to establish routing between one another, the approach to answer this research question is to carry out a review of existing techniques and methods on BGP and investigate whether any of these can be leveraged to aid in influencing BGP in such a way that non-trusted routers can be avoided.

To answer research question number three, we will first look at the current model of how enterprise routers and switches are build. They are shipped as one device where hardware and software are produced by the same manufacturer. It is not possible for the buyer to deviate from the preinstalled Network Operating System (NOS). A thriving paradigm that separates the data plane (hardware) from the control plane (software) is Software Defined Networking (SDN) \[15\]. It allows the user to separately purchase hardware and software (just like in the PC/server market). A router from one manufacturer can be bought and software from another manufacturer be installed. From a security perspective SDN has many advantages, the most important one is that software is more likely to be released as Open Source and therefore can be checked and altered by everybody themselves. The concept of Open Source resolves trust issues, as it is not necessary anymore to trust the vendor. Also, more flexibility is introduced, as the buyer can make modifications to the existing NOS. The use of SDN allows innovation in both, the hardware and software segment independently from each other and smaller companies have a lower entry barrier to enter the enterprise networking market as they can only focus on one part instead of having the need to provide a complete solution. A tool that will be utilized in order to evaluate SDN architectures is Mininet \[16\]. We plan to evaluate the current status of SDN and its future potential. Moreover, we would like to build prototypes to answer this research question.

### 4 Final Considerations

The PhD research proposed is aiming at developing a fingerprint for Internet routers and exploring possibilities of finding alternative routes containing only trusted devices. Also, we will investigate the potential of Open Source concepts in the networking market. Together, the proposed approach is reducing Europe’s dependency on foreign enterprise networking equipment and therefore paving the way towards European Network Sovereignty.
5 References

[1] Mohamed Ali Feki et al. “The internet of things: the next technological revolution”. In: Computer 46.2 (2013), pp. 24–25.
[2] Sam Lucero et al. “IoT platforms: enabling the Internet of Things”. In: IHS Technology white paper (2016).
[3] Christian Canales, Joe Skorupa, and Naresh Singh. “Market Share Analysis: Enterprise Network Equipment, Worldwide, 2016”. In: Gartner Database ID: G00328325 (2016).
[4] Glenn Greenwald. No Place to Hide: Edward Snowden, the NSA, and the U.S. Surveillance State. USA: Metropolitan Books, 2014. ISBN: 162779073X, 9781627790734.
[5] Stephen Checkoway et al. “A systematic analysis of the Juniper Dual EC incident”. In: Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security. ACM. 2016, pp. 468–479.
[6] Bill Hau, Tony Lee, and Josh Homan. SYNful Knock - A Cisco router implant. 2015. URL: https://www.fireeye.com/blog/threat-research/2015/09/synful_knock_-_acis.html
[7] Rob LeFebvre. New bill seeks to ban Huawei from any US government contracts. 2018. URL: https://www.engadget.com/2018/01/13/bill-ban-huawei-us-government-contracts/
[8] H.R.4747 - Defending U.S. Government Communications Act. 2018. URL: https://www.congress.gov/bill/115th-congress/house-bill/4747
[9] Dustin Volz. Trump signs into law U.S. government ban on Kaspersky Lab software. 2017. URL: https://www.reuters.com/article/us-usa-cyber-kaspersky/trump-signs-into-law-u-s-government-ban-on-kaspersky-lab-software-idUSKBN1E62V4.
[10] Mike Wheatley. How The NSA Taps Undersea Fiber optic Cables. 2013. URL: https://siliconangle.com/blog/2013/07/19/how-the-nsa-taps-undersea-fiber-optic-cables/
[11] Gordon Fyodor Lyon. Nmap network scanning: The official Nmap project guide to network discovery and security scanning. Insecure, 2009.
[12] Yves Vanaubel et al. “Network fingerprinting: TTL-based router signatures”. In: Proceedings of the 2013 conference on Internet measurement conference. ACM. 2013, pp. 369–376.
[13] Gregory Detal et al. “Revealing middlebox interference with tracebox”. In: Proceedings of the 2013 conference on Internet measurement conference. ACM. 2013, pp. 1–8.
[14] Planet Lab Project. 2007. URL: https://www.planet-lab.org/
[15] Diego Kreutz et al. “Software-defined networking: A comprehensive survey”. In: Proceedings of the IEEE 103.1 (2015), pp. 14–76.
[16] Mininet Team. Mininet: An instant virtual network on your laptop (or other pc). 2018. URL: http://mininet.org/
List of Authors

Harald Baier ......................................................................................................................... 11
Pavel Benáček .......................................................................................................................... 21
Raphael Labaca Castro ................................................................................................. 16
Tomas Cejka .......................................................................................................................... 21
Pavel Celeda .......................................................................................................................... 6
Martin Drašar ........................................................................................................................ iii
Jérôme François ......................................................................................................................... iii
Juan Luis Gorricho ............................................................................................................. 65
Jiří Havránek ......................................................................................................................... 21
Lukas Kencl ........................................................................................................................... 46
Godfrey Kibalya ..................................................................................................................... 65
Katharina Kuhnert ................................................................................................................... 11
Edmundo Monteiro .............................................................................................................. 46
Moritz Mueller ......................................................................................................................... 60
Mohamed Nassar ................................................................................................................... 33
Aiko Pras ................................................................................................................................ 75
Nilis Rodday ........................................................................................................................... 75
Gabi Dreo Rodosek ............................................................................................................... 16, 75
Haidar Safa ............................................................................................................................. 33
Eder John Scheid .................................................................................................................... 70
Joan Serrat ............................................................................................................................... 65
Paulo Silva ............................................................................................................................... 46
Stanislaw Spacek .................................................................................................................... 6
Anna Sperotto .......................................................................................................................... 1
Jessica Steinberger ................................................................................................................... 11
Burkhard Stiller ....................................................................................................................... 70
Olivier van der Toorn ............................................................................................................ 1
Roland van Rijswijk-Deij ...................................................................................................... 60
Petr Velan ................................................................................................................................. 21
