Ahead of the Curve: A Deeper Understanding of Network Threats Through Machine Learning

Joy Nathalie Avelino, Jessica Patricia Balaquit, and Carmi Anne Loren Mora
Trend Micro Core Technology Researchers

A **TrendLabs**SM Research Paper

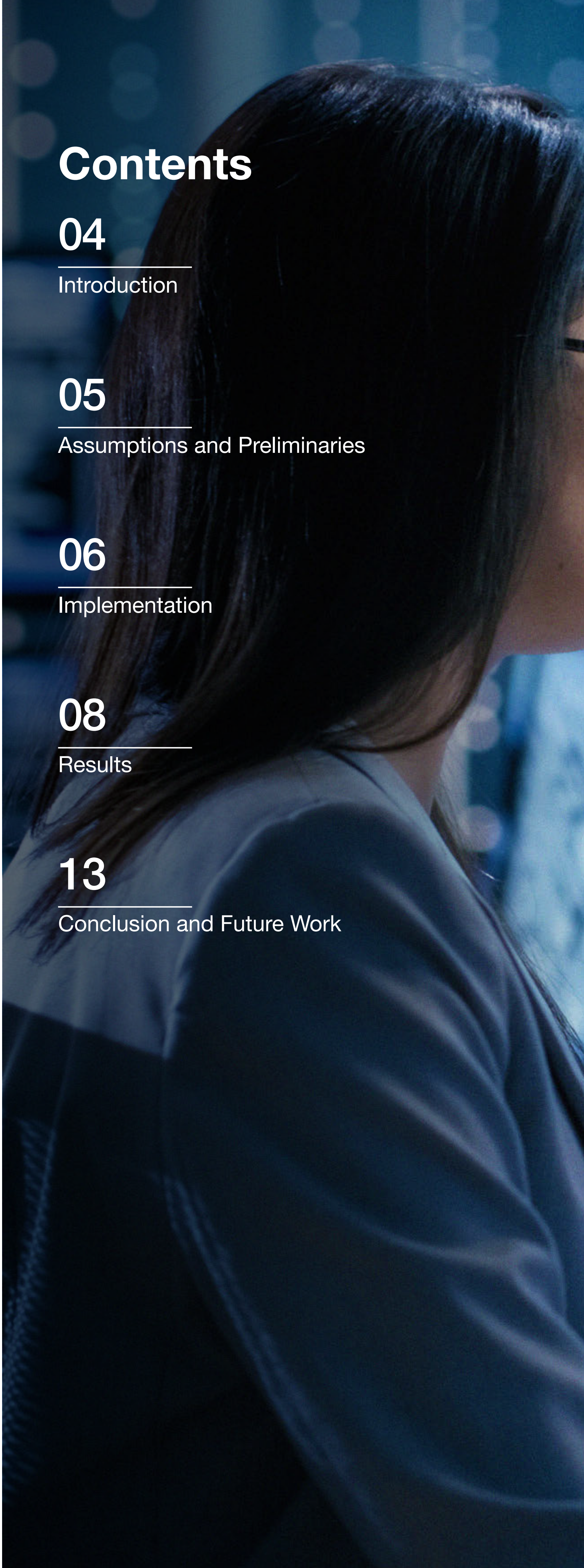
Published by:
Trend Micro Incorporated

Written by:
Joy Nathalie Avelino, Jessica Patricia Balaquit, and Carmi Anne Loren Mora
Trend Micro Core Technology Researchers

Design and layout by:
TrendLabs

Stock images used under license
from Shutterstock.com

For Raimund Genes (1963-2017)



Contents

In the current state of the threat landscape, cybercriminals have increasingly adopted the latest evasion techniques (for example, polymorphism, encryption, and obfuscation) to bypass signature-based detection methods. Since most of these threats propagate through the network, it is important to have proactive techniques to discover an infection before it damages a system.

To address these growing network threats, methods that leverage the power of machine learning should be considered and explored, as big data and machine intelligence continue to gain prominence in information security due to its efficacy in aiding cybersecurity solutions to combat various threats. In this study, we decided to train a machine learning model using header-based information, as well as other characteristics in the HTTP network traffic, in capturing malicious behavior.

Our research discovered that features in the raw byte stream that are augmented with handcrafted features can be useful to understand the characteristics of network threats. In specific clusters formed, it is possible to identify certain threats targeting a specific server, or if there are characteristics that can be observed in the injected code for exploit detection.

This paper will discuss how machine learning can help cybersecurity professionals identify connected malware campaigns and discover valuable insights on future trends, which are necessary actions that can lead to better defenses against network threats.

I. Introduction

A network intrusion attack refers to any compromise in the stability or security of information stored on connected computers.[1] There are many intrusion detection techniques and methods used for detecting network anomalies. The traditional method is to monitor network protocols using signature/behavior-based rules and heuristics.[2, 3, 4] Since not all malicious network traffic happens in post-infection, all attack phases are monitored from pre-infection to post-infection stages. This technique broadens the search parameters for current threats and can be used as a reference on how a threat behaves during an attack.[5] Other techniques include the use of custom sandbox analysis and threat intelligence sharing. Custom sandbox analysis enables the discovery of advanced threats even if a file was not initially detected through the network or in the system. Threat intelligence sharing, on the other hand, enables other security products to quickly contain the threat and prevent further attacks.

While the described technologies can address most network threats, there are still some limitations.[6] Signature-based detection lacks flexibility; if the detected network traffic has a minor change [e.g., the spoofed header has a dynamic, randomly-generated Uniform Resource Identifier (URI)], the signature cannot detect it unless the signature is modified.[7, 8] Additionally, the growing automation of attacks and the sheer amount of attacks make manual inspection by analysts time-consuming.[9] Advanced threats can be covered by behavior-based or heuristic rules; however, they can be potentially aggressive, and, in turn, can lead to false positives.[10, 11]

The lack of flexibility for signature-based detection and the aggressiveness of behavior-based detection and heuristics call for a solution that will address these concerns. This is where machine learning comes in.[12] Machine learning can process data beyond what humans can in a short span of time, and evolve according to the data instances given as input. In this study, the data fed into the machine learning model were from in-the-wild data, which opened the possibility of obtaining insights that can be used to aid in the identification of targeted attacks and advanced threats.

The following sections will discuss, in order, the definitions of network flow data, machine learning, and the dataset used, followed by the implementation of the clustering model, the evaluation of the model, the conclusion formed from this research, as well as insights that can be valuable for future work concerning network threats.

II. Assumptions and Preliminaries

A. Network Flow Data

A flow is defined as a "unidirectional stream of Internet Protocol (IP) packets that share a set of common properties: typically, the IP-five-tuple of protocol, source and destination IP addresses, source and destination flows."[13] Flow data exported by a packet sniffer to a packet capture (PCAP) contains information that is useful for examining the traffic composition of different applications and services in the network. Its intent is not to steal information, but to help secure the network.[14] It can be used to discover and analyze different kinds of network anomalies, such as targeted attacks or presence of botnets.

B. Machine Learning

There are several approaches in the use of machine learning for network data. A standard approach is to use the classification process to identify malicious from legitimate traffic.[15, 16, 17] Classification, which is a type of supervised learning, requires a significant amount of time and resources to sift through the data and label it, since most data encountered in the real world are unlabeled.

Using unlabeled data, however, is efficient in recognizing patterns. This method of exploring unlabeled data, or unsupervised learning, also helps in discovering new relationships between data through clustering, which can be applied in real-time for newly identified threats.

This study used a semi-supervised learning approach to maximize the labeling and processing of large amounts of unlabeled data through clustering. These labels are used to find relationships between different malware families and to know how they differ from one another.

C. Dataset

As prior researches have discussed, the dataset is a critical component in utilizing machine learning on malicious network flow. Previous studies used existing public datasets[18, 19] or datasets generated in a controlled environment.[20, 21]

This study utilizes in-the-wild network dataset from PCAPs of recent threats tagged as malicious by Trend Micro's network detection engine and may potentially contain new and never-been-seen threats. The network flows that are processed are Hypertext Transfer Protocol (HTTP) traffic, since HTTP is commonly used as medium for malicious activity.[22] The goal of this study is to get further information from the clustering results in order to provide timely and relevant coverage of the network threat landscape.

III. Implementation

A. Data Preprocessing

In a PCAP, malicious flows are often mixed with normal flows, making it susceptible to noise. The large volume of network data present indicates that manual clean-up is a resource-intensive task. To ensure that the clusters are representative of the current threat landscape, the collected data should be filtered as much as possible.[23]

In this study, data capture is split into multiple streams to mitigate the noise, with each stream considered as an individual data point. Given that this study deals with malicious network flow, it is expected that non-standard headers and formatting aberrations may be found in the data and preprocessing should be taken into account.

B. Feature Engineering

Stream headers and other relevant information were used to generate the features fed into the clustering model. Some of the features used were taken from previous academic papers studying features for anomaly detection, such as byte entropy, distribution, and standard deviation of the headers and payload.[24, 25] Concrete attack instances were carefully abstracted, as reliance on these would overfit to the malware present in the dataset and would prevent the model from fitting well to novel attack instances.[26]

The features in this study were crafted to reflect the subject matter expertise of network threat detection experts, and to discriminate between certain types of malware. Some were only target-specific server types, while others manifest characteristics that hint on the kind of malicious content being delivered to a machine.

While there is great potential for machine learning in security research applications, translating network traffic to an acceptable input format for a machine learning model remains a challenge.[27, 28] Since considering an ad-hoc versus an automated approach poses considerable trade-offs, both methods of feature engineering were employed in this study. The features selected have undergone scaling and normalization before being fed to the clustering model.

C. Choosing the Clustering Algorithm

To determine the ideal algorithm for use in this type of problem, three clustering implementations were considered: k-means, Density-based Spatial Clustering of Applications with Noise (DBSCAN), and Hierarchical DBSCAN (HDBSCAN). k-means and DBSCAN from the scikit-learn[29] library, as well as HDBSCAN from a standalone library by Leland McInnes, John Healy, and Steve Astels,[30] were implemented using Python language.

In k-means, clustering requires prior knowledge of the number of clusters involved. It may also output different results depending on where the initial point was placed, which makes the clustering unstable. Since the number of clusters is unknown, there is a need for an algorithm that can estimate the number of clusters. In the threat landscape where new types of threats are continuously emerging, this model may have to be adjusted periodically.

Thus, only DBSCAN and HDBSCAN were used. In both density-based algorithms, the number of clusters is determined by its neighboring points. This solves the problem with setting the number of clusters whenever a new threat is discovered.

The final analysis was generated using the HDBSCAN algorithm. It extends DBSCAN by using a hierarchical approach before extracting the stable clusters. When compared to DBSCAN, the results produced by HDBSCAN are consistent with the understanding of the threats as reviewed by the domain experts. In semi-supervised learning, validation of the cluster still involves human intervention. The hierarchical approach that HDBSCAN employed is useful in understanding the results and helped augment human expertise, which will be illustrated in the next section.

IV. Results

The preliminary results in the utilization of the clustering model to cluster similar types of malicious network flows are favorable. The following cluster visualization is produced using Embedding Projector.[31]

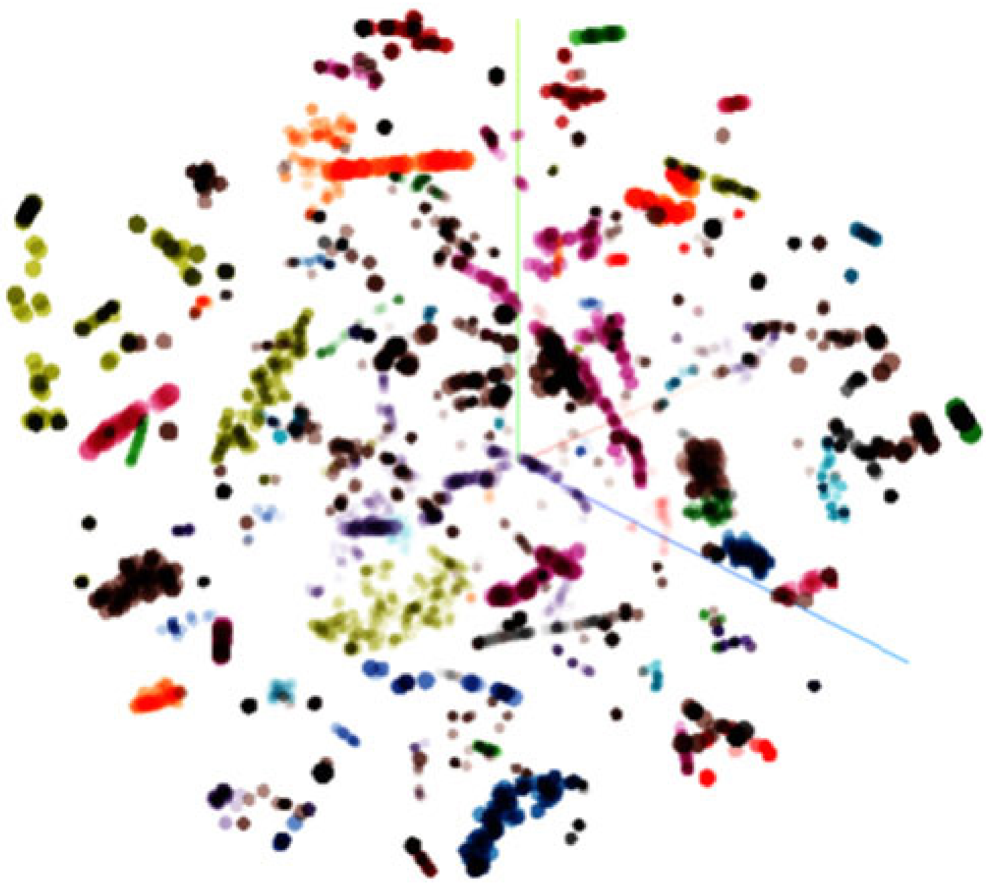


Figure 1. Clusters formed by malicious HTTP streams. Each color represents one cluster.

One of the clusters inspected predominantly consists of network flows indicating ransomware[32] infection. The web plays an important part for ransomware because it requires a connection to the Command and Control (C&C) server to send an infection report, or receive the encryption key. Upon inspecting the ransomware cluster, most of the similarities occur in the Uniform Resource Locator (URL) found in various headers (for example, URI and location).

Another interesting result comes from clusters comprised entirely of exploit kit[33] detections as seen in Figure 2. When these clusters were examined, one of the features most relevant to the clustering turned out to be those which concern file types. This makes sense since exploit kits are known to exploit through file formats e.g. Shockwave/Flash, PDF, and JavaScript (JS), among others.

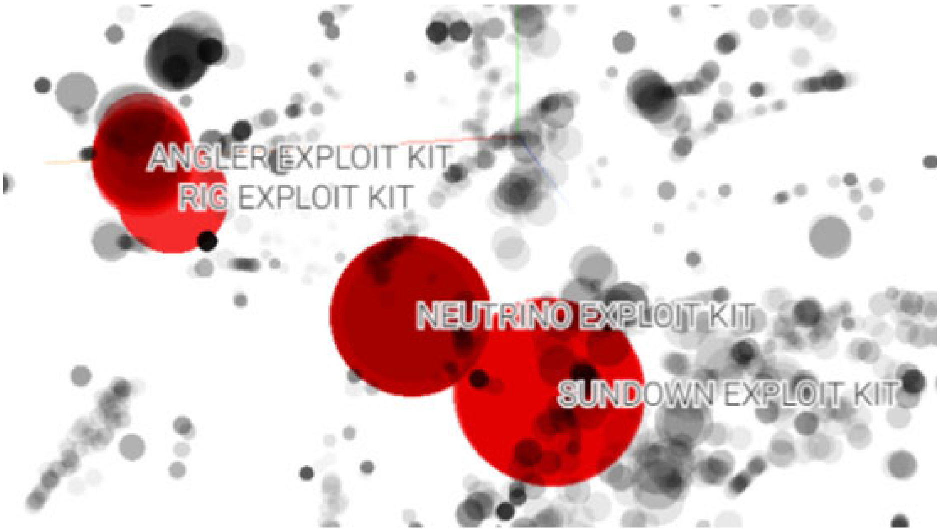


Figure 2. Clusters comprising different exploit kits.

Figure 3 illustrates the different network characteristics of five malware families: Rig, FlashPack, Angler, Neutrino, and Blacole. The different colors correspond to structural attributes determined by the features passed to the model. For signature-based detection, one rule will be created for each family due to varying flow characteristics present in the network. Since signature-based detection lacks flexibility, having a slight change in the network traffic can render the rule unusable, unless the signature is modified.

Figure 3. Raw network data of each malware family. Each color represents one characteristic.

Nevertheless, the clustering model was able to find similarities in the network flows, allowing them to be grouped together. From the multiple characteristics seen in each malware family, as illustrated in Figure 3, the clustering model was able to identify which ones constitute a certain profile that correlates among the similar samples. Figure 4 shows an analogy of how the clustering model sees the similar characteristics among the malware families.

Figure 4. Network flows as seen by the clustering model

Blacole seems like an outlier for the reason that it was categorized as a Trojan and not specifically as an exploit kit in the dataset labelling However, when its network traffic was examined, we found out that the key similarity that links Blacole to some exploit kits is that its malware routine takes advantage of JS vulnerabilities. This emphasizes the fact that exploit kits can be identified without tailoring features to a specific attack instance.

Another analysis was conducted to variants of Gh0st RAT (Remote Administration Tool) – a well-known Trojan affiliated with the GhostNet bot network.[34] A number of Gh0st RAT variants have emerged over time since its source code is publicly available.[35]

```
7hero  FWAPR  Heart  Level  QWPOT    X6RAT  xhjyk  cyl22   kaGni  Wh0vt
Adobe  FWKJG  IM007  Lover  Spidern  XDAPR  00000  DrAgOn  light  Snown
B1X6Z  GWRAT  ITore  Lyyyy  Tyjhu    Xjjhj  ABCDE  EXXMM   LkxCq  SocKt
BEiLa  Gh0st  KOBBX  MYFYB  URATU    ag0ft  apach  Eyes1   lvxYT  Super
BeiJi  GOLDt  KrisR  MoZhe  W0LFKO   attac  Assas  Gi0st   Naver  Sw@rd
ByShe  HEART  LUCKK  MyRat  Wangz    cb1st  Blues  GM110   NIGHT  v2010
FKJP3  HTTPS  LURK0  OXXMM  Winds    https  chevr  Hello   NoNul  VGTLS
FLYNN  HXWAN  LYRAT  PCRat  World    whmhl  CHINA  httpx   Origi  wcker
wings  X6M9K  xqwf7  YANGZ  QQ_124971919
```

Figure 5. Variants of Gh0st RAT[36]

Figure 6 illustrates the streams that were clustered across multiple versions of Gh0st RAT because they contain similar payloads.

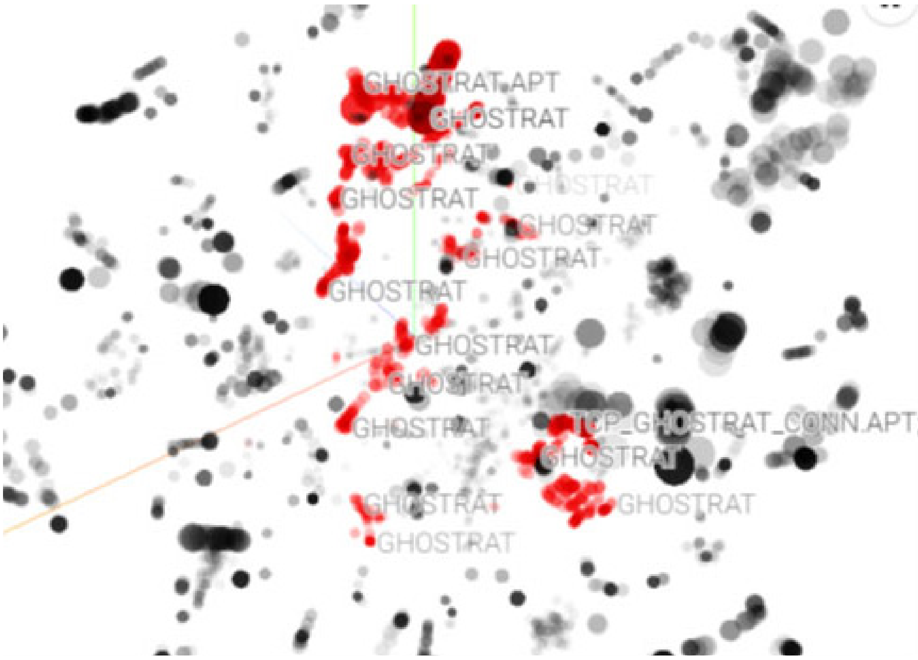


Figure 6. Gh0st RAT clusters

With threats reusing old malware to carry payload for backdoor capabilities (as seen in Figure 7), cryptocurrency mining[37] (see Figure 8), and targeted attacks,[38] machine learning can associate incoming traffic to future Gh0st RAT variants.

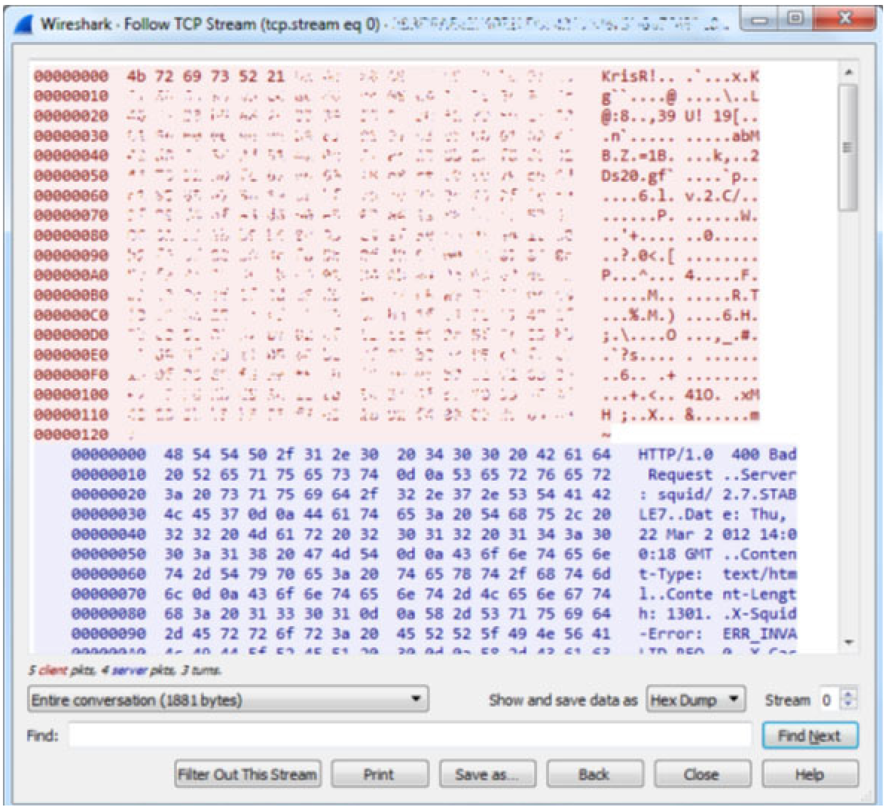


Figure 7. Hex dump of Gh0st RAT variant KrisR

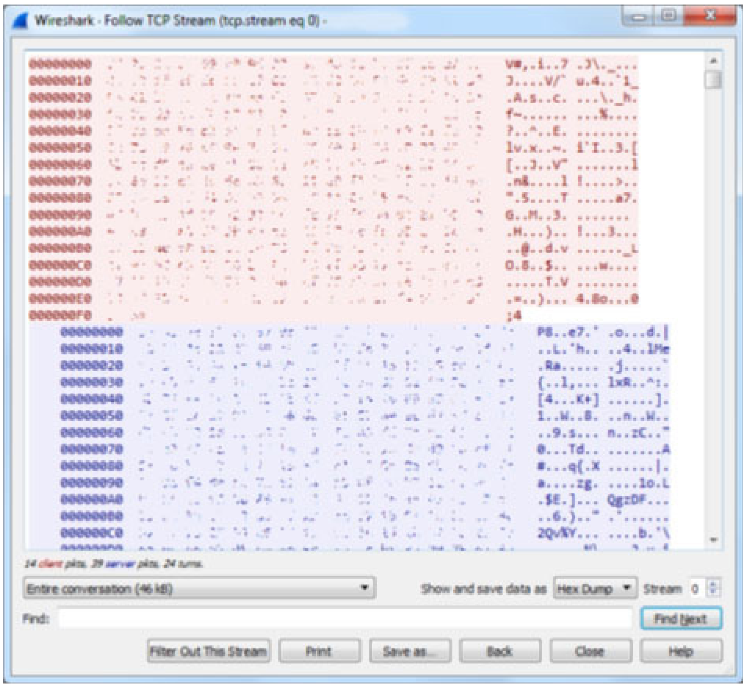


Figure 8. Hex dump of Monero cryptocurrency mining payload

V. Conclusion and Future Work

Clustering malicious network flows with features from the raw byte stream, when augmented with handcrafted features as input from the data, can provide insights on different network patterns from malicious traffic. It can also show similar characteristics between different malware families, albeit within the same classification—such as exploit kits. Indeed, this approach is useful in augmenting signature creation for detecting network malware.

As shown in this research, machine learning plays a key role in the process of successfully clustering network threats. Using machine learning for analysis vastly improves the speed at which data is organized and conclusions are obtained. In addition, the results show how machine learning can be used to efficiently identify a widely used vulnerability as it is spreading, or to recognize a certain vulnerability used in a novel way as part of another malware campaign.

For analysts who would conduct similar studies in the future, we recommend feature refinement—an approach that can lead to better modelling of malicious flow clustering. At this stage, the model would benefit most from taking a closer look at URLs in the streams, and from other experimentation with other features extracted from the header contents, such as measuring string randomness. In order for real-time detections to be more accurately clustered, the model must be equipped, in future iterations, with the capacity to handle sequential data. This will also bolster its capabilities to cluster flows from other protocols than HTTP.

Acknowledgement

We would like to acknowledge the following people from Trend Micro for supporting us in this project: Threat Research Director Mary Ong; Head of Machine Learning Group Brian Cayanan; Jameson Ong; machine learning consultants Jon Oliver, Abraham Camba, Jayson Pryde, Robert Tacbad, and Joa Suico; the Quality Assurance Team that pioneered this project; and the Deep Discovery Inspector Group that provided their threat expertise in network analysis.

References

1. Leonid Portnoy, Eleazar Eskin, and Sal Stolfo. “Intrusion detection with unlabeled data using clustering.” In *In Proceedings of ACM CSS Workshop on Data Mining Applied to Security (DMSA-2001*, Citeseer, 2001.

2. Ibid.

3. Gabriel Sanchez. (February 16, 2016). *SANS Institute.* “Don’t always judge a packet by its cover.” Last accessed on 9 October 2018 at https://www.sans.org/reading-room/whitepapers/access/dont-judge-packet-cover-36745.

4. Hung-Jen Liao, Chun-Hung Richard Lin, Ying-Chih Lin, and Kuang-Yuan Tung. “Review: Intrusion detection system: A comprehensive review.” In *Journal of Network and Computer Applications 36: 16-24,* January 2013.

5. Trend Micro. (11 July 2013). *Trend Micro Research Paper.* “Lateral Movement: How Do Threat Actors Move Deeper Into Your Network?” Last accessed on 9 October 2018 at http://about-threats.trendmicro.com/cloud-content/us/ent-primers/pdf/.

6. Trend Micro. (June 2015). *Trend Micro Research Paper.* “Inside the wire: Why Perimeter-centric Monitoring Leaves You Vulnerable. Last accessed on 9 October 2018 at https://www.trendmicro.de/media/wp/ddi-network-vs-perimeter-wp-en.pdf.

7. G. Sanchez. (February 16, 2016). *SANS Institute.* “Don’t always judge a packet by its cover.” Last accessed on 9 October 2018 at https://www.sans.org/reading-room/whitepapers/access/dont-judge-packet-cover-36745.

8. Yang Xin, Lingshuang Kong, Zhi Liu, Yuling Chen, Yanmiao Li, Hongliang Zhu, Mingcheng Gao, Haixia Hou, and Chunhua Wang. “Machine learning and deep learning methods for cybersecurity.” In *IEEE Access, page1*, May 2018.

9. Konrad Rieck. “Computer security and machine learning: Worst enemies or best friends?” In *2011 First SysSec Workshop, pages 107–110,* July 2011.

10. Y. Xin, L.Kong, Z. Liu, Y. Chen, Y. Li, H. Zhu, M.Gao, H. Hou, and C. Wang. “Machine learning and deep learning methods for cybersecurity.” In *IEEE Access, page1,* May 2018.

11. Curtis Cade. (13 July 2015). “Understanding Heuristic-based Scanning vs. Sandboxing.” 2015. https://www.opswat.com/ blog/understanding-heuristicbased-scanning-vs-sandboxing.

12. Trend Micro. (October 2016). *Trend Micro Research Paper.* “There is no silver bullets: The strengths and weakness of today’s threat-protection techniques and why a multi-layered approach to endpoint security is a must.” Last accessed on 9 October 2018 at https://www.trendmicro.com/content/dam/trendmicro/global/en/business/products/user-protection/endpoint/wp_XGen-Silver-Bullet.pdf.

13. Gerhard Munz and Georg Carle. “Real-time analysis of flow data for network attack detection.” In *2007 10th IFIP/IEEE International Symposium on Integrated Network Management, pages 100–108,* May 2007.

14. Mohammed Abdul Qadeer, Arshad Iqbal, Mohammad Zahid, and Misbahur Rahman Siddiqui. “Network traffic analysis and intrusion detection using packet sniffer.” In *2010 Second International Conference on Communication Software and Networks, pages 313–317,* February 2010.

15. Blake Anderson, Subharthi Paul, and David McGrew. “Deciphering Malware’s use of TLS (without Decryption).” In *Journal of Computer Virology and Hacking Techniques, pages 1–17,* July 2016.

16. Dmitri Bekerman, Bracha Shapira, Lior Rokach, and Ariel Bar. “Unknown malware detection using network traffic classification. In *2015 IEEE Conference on Communications and Network Security (CNS), pages 134–142,* September 2015.

17. Jonathan J. Davis. (2017). “Machine Learning and Feature Engineering for Computer Network Security (doctoral dissertation).” Queensland University of Technology, Brisbane, Queensland, Australia.

18. L. Portnoy, E. Eskin, and S. Stolfo. “Intrusion detection with unlabeled data using clustering.” In *In Proceedings of ACM CSS Workshop on Data Mining Applied to Security (DMSA-2001,* Citeseer, 2001.

19. R. P. Lippmann, D. J. Fried, I. Graf, J. W. Haines, K. R. Kendall, D. McClung, D. Weber, S. E. Webster, D. Wyschogrod, R. K. Cunningham, and M. A. Zissman. “Evaluating intrusion detection systems: the 1998 darpa off-line intrusion detection evaluation.” In *Proceedings DARPA Information Survivability Conference and Exposition. DISCEX’00, vol. 2, pages 12–26,* January 2000.

20. B. Anderson, S. Paul, and D. McGrew. “Deciphering Malware’s use of TLS (without Decryption).” In *Journal of Computer Virology and Hacking Techniques, pages 1–17,* July 2016.

21. Shuai Zhao, Mayanka Chandrashekar, Yugyung Lee, and Deep Medhi. "Real-time network anomaly detection system using machine learning." In *2015 11th International Conference on the Design of Reliable Communication Networks (DRCN), pages 267–270,* March 2015.

22. Niels Provos, Dean McNamee, Panayiotis Mavrommatis, Ke Wang, and Nagendra Modadugu. "The ghost in the browser analysis of web-based malware." In *Proceedings of the First Conference on First Workshop on Hot Topics in Understanding Botnets (HotBots'07). USENIX Association, page 4,* 2007.

23. Trend Micro. (May 2017). *Trend Micro Research Paper.* "Machine Learning and Next-Generation Intrusion Prevention System (NGIPS): Building a smarter NGIPS." Last accessed on 9 October 2018 at https://documents.trendmicro.com/assets/wp/WP01_Machine_Learning_170608US.pdf.

24. B. Anderson, S. Paul, and D. McGrew. "Deciphering Malware's use of TLS (without Decryption)." In *Journal of Computer Virology and Hacking Techniques, pages 1–17,* July 2016.

25. J. J. Davis and Andrew John Clark. "Data preprocessing for anomaly based network intrusion detection: A review." In *Computers & Security 30(6-7): 353–375,* September 2011

26. K. Rieck. (2009). "Machine Learning for Application-Layer Intrusion Detection (doctoral dissertation)." Technische Universität Berlin, Berlin, Germany.

27. J. J. Davis. (2017). "Machine Learning and Feature Engineering for Computer Network Security (doctoral dissertation)." Queensland University of Technology, Brisbane, Queensland, Australia.

28. Hao Dong, Jin Shang, David Yu, and Chenghuai Lu. "Beyond the blacklists: Detecting malicious URL through machine learning." In *Proceedings of BlackHat Asia 2017,* March 2017.

29. F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. "Scikit-learn: Machine learning in Python." In *Journal of Machine Learning Research 12: 2825–2830,* November 2011.

30. Leland McInnes, John Healy, and Steve Astels. "hdbscan: Hierarchical density based clustering." In *The Journal of Open Source Software 2,* March 2017.

31. Daniel Smilkov, Nikhil Thorat, Charles Nicholson, Emily Reif, Fernanda B. Vi´egas, and Martin Wattenberg. "Embedding Projector: Interactive Visualization and Interpretation of Embeddings." arXiv:1611.05469, November 2016.

32. Trend Micro. (n.d.). *Trend Micro*, "What is Ransomware?" Last accessed on 9 October 2018 at https://www.trendmicro.com/vinfo/us/security/definition/ransomware.

33. Joseph C. Chen and Brooks Li. (2015). *Trend Micro.* "Evolution of Exploit Kits." Last accessed on 9 October 2018 at https://www.trendmicro.de/cloud-content/us/pdfs/security-intelligence/white-papers/wp-evolution-of-exploit-kits.pdf.

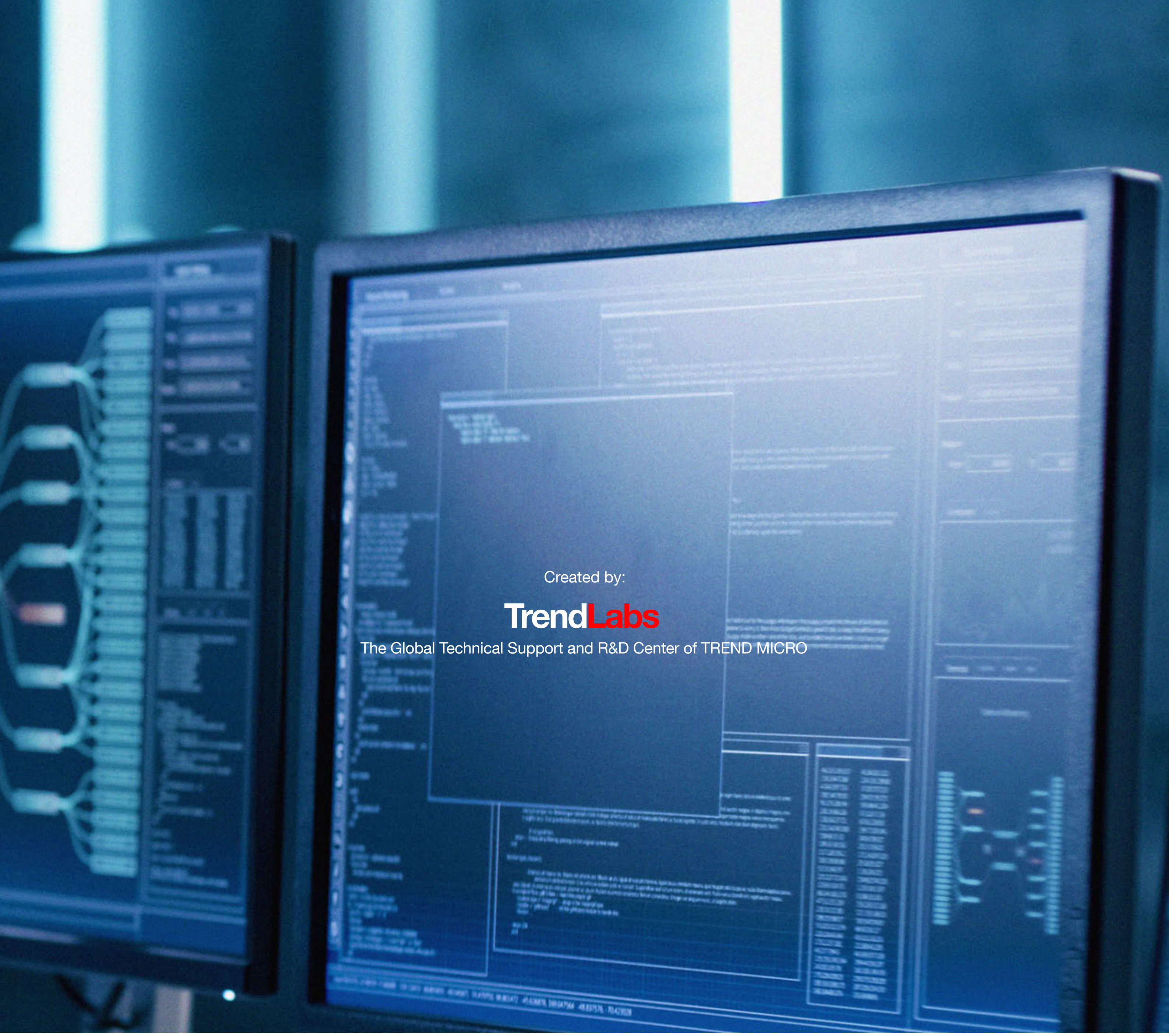
34. Trend Micro. (21 September 2012). *Threat Encyclopedia.* "GHOSTRAT." Last accessed on 9 October 2018 at https://www.trendmicro.com/vinfo/us/threat-encyclopedia/malware/ghostrat.

35. Snorre Fagerland. (2012) *Norman ASA.* "The many faces of gh0st rat: Plotting the connections between malware attacks." Last accessed on 9 October 2018 at http://download01.norman.no/documents/ThemanyfacesofGh0stRat.pdf.

36. Ibid.

37. Nikolaos Pantazopoulos. (17 April 2018). *NCC Group.* "Decoding network data from a Gh0st RAT variant." Last accessed on 9 October 2018 at https://www.nccgroup.trust/uk/about-us/newsroom-and-events/blogs/2018/april/decoding-network-data-from-a-gh0st-rat-variant/.

38. Ziv Chang, Kenney Lu, Aaron Luo, Cedric Pernet, and Jay Yaneza. (2015) "Operation Iron Tiger: Exploring Chinese Cyber-Espionage Attacks on United States Defense Contractors." Last accessed on 9 October 2018 at https://www.erai.com/ CustomUploads/ca/wp/2015_12_wp_operation_iron_tiger.pdf.



TREND MICRO™

Trend Micro Incorporated, a global leader in cybersecurity solutions, helps to make the world safe for exchanging digital information. Our innovative solutions for consumers, businesses, and governments provide layered security for data centers, cloud environments, networks, and endpoints. Optimized for leading environments, including Amazon Web Services, Microsoft®, VMware®, and more, our solutions enable organizations to automate the protection of valuable information from today's threats. All our products work together to seamlessly share threat intelligence and provide a connected threat defense with centralized visibility and investigation, enabling better, faster protection. With almost 6,000 employees in over 50 countries and the world's most advanced global threat intelligence, Trend Micro enables organizations to secure their connected world. For more information, visit www.trendmicro.com.

TREND MICRO™

Securing Your
Connected World