Malware Propagation in Fully Connected Networks: A Netflow-Based Analysis

Kayla M. Straub†, Avik Sengupta†, Joseph M. Ernst†, Robert W. McGwier†,
Merrick Watchorn*, Richard Tilley‡, and Randolph Marchany‡
†Hume Center for National Security and Technology, Virginia Tech, Blacksburg, VA 24060
*Cyber Directorate, SAIC, Inc., McLean, VA 22102
‡IT Security Laboratory, Virginia Tech, Blacksburg, VA 24060
Email: {kstraub, aviksg, jmernst, rwmcgwi}@vt.edu, merrick.s.watchorn@saic.com, {brad,marchany}@vt.edu

***Abstract*—Malware attacks have become ubiquitous in modern large data-centric networks. Therefore advanced malware threat detection and related countermeasures are an important paradigm in cybersecurity research. This work studies malware propagation in fully connected networks, where network topology plays a minimal role in lateral spread within the network. The live netflow and perimeter alert data used in this study contrasts with other previous works due to the unavailability of ground truth for any attack type. Important features calculated from the netflow data as well as a novel ring-based flow model are described. These are helpful in tracking possible malware flow within the network. The results show that relevant features can be used to draw inferences about the propagation of certain classes of malware attacks.**

***Index Terms*—malware, lateral propagation, netflow.**

I. Introduction

The privacy and security of shared content in modern data-centric computer networks is under threat from malware-based attacks. Malware poses a severe threat to the integrity of industrial and private computer networks across the globe. As networks become denser and interactions between geographically diverse systems increase, it has become more difficult to protect networks against sophisticated and ever-evolving threats. Malware that communicates over low volume traffic has the dubious distinction of being able to slip under the radar of protective signature-based detection techniques to invade networks. Of particular interest are botnet-based attacks, where an attacker uses multiple command and control centers to direct malicious bots (computers) as an underground computing resource to perform illicit activities [1].

As a result, significant research has gone into detecting and stopping botnet-based malware flows. Historically, epidemiological models, like the Susceptible/Infected/Recovered (SIR) model [2], have been used in conjunction with network sensor alerts and forensics data sources to identify rates of spread and recovery from malware within a network [3]–[7].

Such models have been largely used as a post-attack forensics tool. This entails curve fitting of the observed data to find the best model parameters that can accurately model the spread of the malware post-attack. While theoretical modeling and detection studies [5]–[9] provide a technical framework for malware detection, tracking and modeling the spread of malware within networks in near-real-time still remains a largely open task. This is primarily due to the diversity of networks and constantly evolving attack methodologies. Anti-virus approaches to malware detection have been superseded by more comprehensive intrusion detection software (e.g., Snort, FireEye), which provide perimeter alerts to large networks for possible malware intrusion. However, signature-based virus removal tools still remain the primary solution for rooting out malware once it breaches the network perimeter. Since modern malware is capable of evolving to avoid signature-based detection, a more fundamental approach to malware eradication, rooted in malware behavior, is required [10].

In this work a computer network under a fully connected setting is considered where topology plays a minimal role in propagation. This research and the proposed solutions are based on stringent data-related constraints since only commonly available sensor data in the form of network perimeter alert reports (i.e., Snort and FireEye) are used to identify possible malware intrusions. In this case, there is no availability of ground truth pertaining to actual attacks, which is a realistic scenario for zero-day attack detection. As a result, an unsupervised system design is adopted which can lead to autonomous detection. In this work, network information is captured as netflow data from the ARGUS sensor. One of the main drawbacks of netflow data in real analysis is the lack of packet payloads [11]. Netflow records only contain flow parameters such as flow duration, number of packets, and the number of bytes transferred thereby making it difficult to design detection signatures to prevent malware incursions. Netflow data are half-duplex and capture each direction of the flow as separate entries. This work formalizes an approach that can classify malware flow within the network by efficiently parsing and filtering the flow data, whereby the detector must be capable of recognizing distinguishing patterns for malicious flows as opposed to legitimate network traffic. This paper studies unsupervised malware detection techniques to identify the propagation of malware that may occur from initial points of infection. The main contributions of this work are:

This research is based upon work supported by the National Science Foundation under Grant No. 1134843.

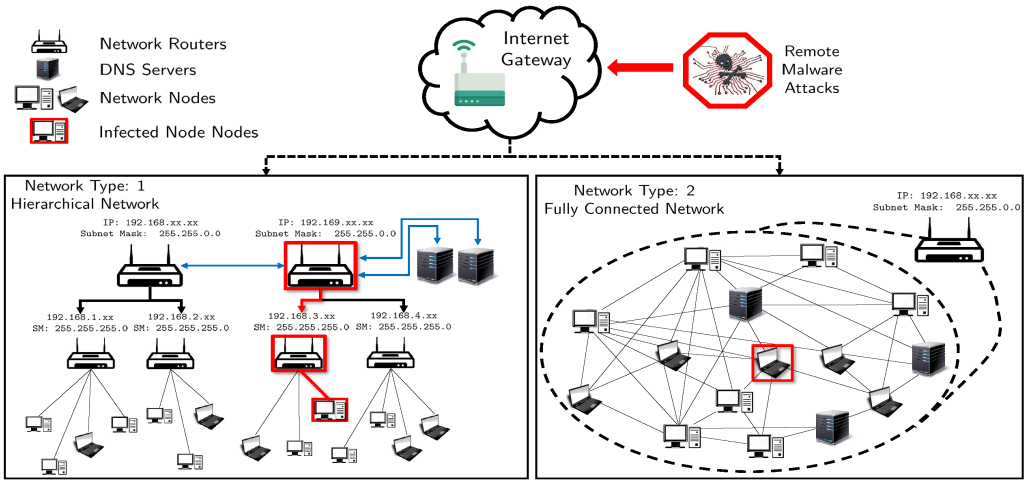


Fig. 1. Two different network topologies, hierarchical and densely connected, are illustrated.

- The introduction of the fully connected Virginia Tech computer network including the associated perimeter alert and netflow monitoring system.
- A unique anonymized netflow dataset is used to model malware flow within the network based on initial points of attack collected in real time from the perimeter sensors. The proposed feature extraction strategies can help identify flow characteristics for different types of malware.
- Passive classification of common attacks that exhibit lateral flow within the network. The proposed ring-based malware propagation model tracks aggregate flows and is verified using live netflow data.

The paper is organized as follows: Section II discusses the network and threat models under consideration. Section III introduces the Virginia Tech (VT) computer network dataset that is used exclusively in this work. Section IV describes feature extraction methodology for modeling malware flow. The ring-based propagation model is detailed in Section V while Section VI presents the results of applying this model to the real-world data. Section VII concludes the paper and points to future work in this area.

II. Network Threat Model

This section details the network model used in this work. It is necessary to first examine the network connectivity and available sensor resources. These are the key motivators behind the choice of approach for tracking malware flow within the network. Types of common malware that may infect the network and pose major security threats are considered in the network model. Finally, this section discusses the ideology of modeling malware spread in real-time computer network.

A. The Virginia Tech Computer Network

There exists a major disparity between topological networks and densely (fully) connected networks, which are illustrated in Fig. 1. An Internet gateway connects a network to the Internet where a malicious command and control (C&C) server directs malware attacks at the network. The network model on the left is a topological or hierarchical model with sub-networks that are connected to the Internet gateway via multiple routers. When a node is infected in such a network, the topology of the network plays a significant role in the spreading path of the malware. Forensics can track a malware path within the network based on the topological characteristics. Generating a dependency graph to utilize graphical algorithms is an effective approach to identifying malicious users within a hierarchical network [12]. The network on the right, however, has a densely connected structure. The nodes are all connected through a common router to the Internet. In this case the topological effect is completely eliminated making tracking and detection a challenging task. This work examines the VT computer network, which falls under the second category of topology-free, fully connected networks.

The VT network consists of roughly 131 072 distinct nodes, including three DNS servers that are open to the Internet. To expand the available address space, the VT network uses IPV6 campus-wide. The network has firewall perimeter sensors in the form of FireEye and Snort sensors that report any suspicious activity going into and out of the network. Due to the sheer volume of network traffic, ground truth on true attacks within the network cannot be reliably obtained. To facilitate this analysis, it is assumed that the perimeter sensors are tuned such that relatively few misdetections occur (which may cause an elevated false alarm rate). The network also has a network flow monitoring sensor, ARGUS, which returns structured netflow data for every intra- and inter-network communication flow. It is infeasible to perform FireEye and SNORT type analysis on high data volumes produced in the ARGUS netflow records. Thus, the VT network presents an interesting real-world scenario, where, unlike ideal theoretical models, there are scarce real-time resources to track malware flow. A wealth of netflow data is available that can only be leveraged by intelligent system design.

B. Types of Malware

This section discusses the basic types of malware under consideration. There are a multitude of malware threats to the network that are diverse in modes of attack as well as in

TABLE I
ARGUS NETFLOW DATA EXAMPLE

FIELD	VALUE
stime	2015-08-29 23:49:01.956461
protocol	1
source_add	A05D4F49B1339EB8BCC345326...
sport	0x0008
dir	<—>
dest_add	2860AAF84D1FC2FB9ED454455...
dest_port	0x4124
pkts	2
bytes	188
state	ECO
field10	0
field11	0

scale. This work concentrates on botnet-based malware attacks that are difficult to detect using real-time analysis. Botnets generally consist of a bot controller who gains control of an Internet Relay Channel (IRC) to set up multiple command and control centers (C&C). These in turn attack susceptible network nodes. The infected machines are then converted into *bots* that communicate with the C&C server to coordinate further attacks and malware spread throughout the network. The most common types of botnet attacks are *Trojans via email spam* and *Distributed Denial-of-Service* attacks.

C. Malware Spreading

The spread of botnets within the network can occur through low data-rate traffic that can easily slip under the perimeter sensor alerts based on how stealthy the botnet commander wants the attack to be. Thus, botnet type attacks form a main focus of the presented analysis. Application-level attacks, which can be easily disguised as legitimate communication, are also considered. Application-layer attacks are the most difficult to defend against since the vulnerabilities encountered rely on complex user inputs that are hard to define with a detection signature. Thus, it is of interest to track the path of such malicious flows across the network in real-time.

III. DATASET: THE VIRGINIA TECH NETWORK DATA

The VT IT Security Lab has provided a unique network database pooled from the ARGUS, Snort and FireEye sensors installed in the VT computer network. The Snort and FireEye data represent actual alerts generated from flagged flows at the network perimeter. The dataset is completely anonymized (i.e., every IP address in and out of network is hashed using a SHA224/HMAC). Any other data that may identify users or machines are obscured, including information about the type of device associated with each address.

A. Netflow Data

The netflow data is obtained from the ARGUS sensor that tracks every IP flow within the VT network. An example flow is presented in Table I, which shows the anonymized IP addresses. Of particular interest are the source and destination address, time, source and destination port, and state fields. The netflow data has heavy volume i.e., one day's worth of flow data has roughly 600 million flows and requires approximately 100 GB of storage for the raw text. In this work, MongoDB handles and processes queries on the dataset.

TABLE II
SNORT PERIMETER ALERT EXAMPLE

FIELD	VALUE
code1	2213
code2	1:2021630:1
src_addr	7DD3C1F42D225EBBD00D06...
classification	A Network Trojan was Detected
alert	Snort Alert [1:2021630:0]
priority	1
mode	TCP
time	Aug 21 06:30:42
src_port	55581
dst_addr	15FB2F5578E1099FE67A2E...
dst_port	3389

TABLE III
A SINGLE ALERT FROM THE FIREEYE SENSOR.

FIELD	VALUE
spt	1053
cn3Label	cncPort
cn2Label	sid
cs6Label	channel
rt	Sep 27201514 : 09 : 09 UTC
proto	tcp
dst	5F32CAB52BA7B07C4C1B410506A603...
externalID	5207926
dvchost	xxx.xxx.xxx
alert	5207926
date	Sep 2709 : 54 : 11
cs4Label	link
cs1Label	sname
src	B4A9770AFE4F1A46E912AC6C260679...
dpt	80
cn2	78010979
cn3	80
cn1	0
cs5Label	cncHost
request	hxxp://a.w.duod.cn/_rfv/i1n2i3t4.jsp
xmac	xx:xx:xx:xx:xx:xx
cn1LAbel	vlan
act	notified
cs1	Android.Riskware.Nqshield

B. Perimeter Alert Data

Example FireEye and Snort perimeter sensor alerts are shown in Table II and Table III. Both types of alerts identify the source and destination addresses, the alert time, the ports, and protocol. The Snort alert also analyzes the flow behavior and provides a further classification which describes the type of attack. For example, the alert in Table II classifies the attack as a network Trojan.

The FireEye sensor additionally has a sandbox functionality whereby it lets the possibly malicious applications run within a virtual sandbox to determine its behavior and only then report the infection. However, there might be malware that can escape the honeypot detection of FireEye and therefore the additional alerts produced by Snort are equally valuable for tracking possible malware flows in an unsupervised system.

IV. FEATURE EXTRACTION

Machine learning approaches to botnet tracking have been successfully implemented [13]–[17], but these methods do not scale well to large fully-connected networks [18]. The raw

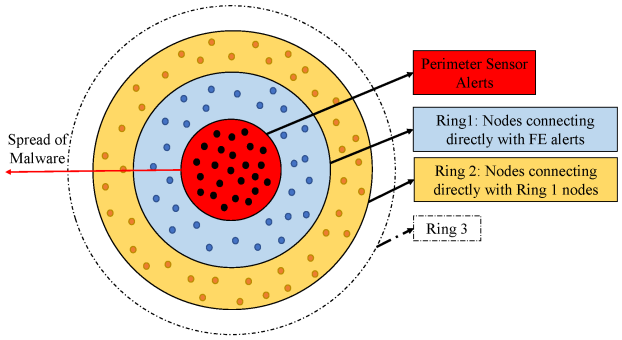


Fig. 2. A ring-based malware spread model.

netflow data volume from the VT network is also too large to use for any signal-based real-time analysis. For denser networks such as these, feature extraction techniques borrowed from the machine learning literature can be used to condense the data to a manageable size.

The Snort alert classifications, as discussed in Section III-B, can provide additional insight into malware behaviors. Studying each alert category separately demonstrates how the propagation behaviors differ between the various types of attacks. The feature analysis in this work focuses on the Snort dataset to incorporate this additional categorization.

The IP addresses from the alerts are matched with the corresponding ARGUS Netflow data to expand the amount and types of information available about each IP address. Incorporating this information helps construct a full behavior profile for each node, and then the features are extracted on this per-IP address basis.

The feature construction is motivated by the potential malware flows to be classified. Some features are calculated by counting the different kinds of flows per IP address, such as the number of flows using various protocol values. Other features compare the number of packets or bits associated with each flow, including the average, minimum, maximum, and standard deviation of both packets and bits per flow. More sophisticated features examine the inter-arrival time of flows for each IP address, including the average, minimum, maximum, and standard deviation of the inter-arrival times. Another feature considered was the average number of packets per connection between any two machines within the VT Network. In total, 39 features were calculated from the Snort alert data.

V. A Ring-Based Spread Model

This section proposes a ring-based model for tracking malware flow within a fully connected network based on perimeter alerts and netflow analysis. A major challenge is that for the flow data, no ground truth on attacked nodes is available. This renders ineffective a large majority of anomaly detection-based approaches. The model used instead is based on the mold of epidemic spread modeling of network malware [7].

In the absence of accurate sensing within the network, the main objective of the proposed model is to combine the netflow data with the sensor alerts from the Snort data to form a coherent spreading pattern. A new ring-based protocol is used to identify malware flows in terms of aggregate distributions. Fig. 2 shows the proposed ring-based model. The innermost ring highlights the nodes present in the Snort reports for suspicious activity. The next ring contains nodes that are directly connected to these nodes in the innermost ring. The outer rings represent the nodes to which the nodes in the previous ring are connected. Using this model, aggregate statistics are calculated for each ring. For malware types that spread through the network, Ring 1 is expected to represent malicious network behavior, while Ring 2 and onwards should represent behavior related to ebbing malicious flows, which should subside by proceeding through the outer rings.

VI. Results

Examining the extracted features through the ring-based model identifies features that are useful for tracking malware flow once the network has been penetrated. The first part of these results compares the various features to determine which hold the most meaningful information. The second section examines these features with respect to the ring-based model.

A. Feature Results

To identify the most important features, the feature histograms for the Snort-flagged nodes are compared to the feature histogram for uninfected IP addresses. Features that exhibit divergent distributions between the uninfected and infected nodes are indicators of suspicious behavior. Fig. 3 shows the result of comparing feature histograms across the various alert types. The histograms are normalized such that the y-axis represents a proportion of the total alerts, enabling comparisons between different-sized alert sets. In particular, the average inter-arrival time and the minimum inter-arrival time show a sharp contrast between the flagged nodes and the clean nodes in the first row. These features seem to form entirely different distributions than their uninfected counterparts. The average inter-arrival time histograms for miscellaneous attack, detection of a network scan, and attempted denial of service alerts peak at the same value. For all three of these distributions, the average inter-arrival time mode is approximately 300 seconds. This value is also a local maximum for the network trojan attack as well. In contrast, the uninfected nodes form a markedly different distribution that peaks in the first bin. This insight into the behaviors of these types of attacks could be leveraged by tracking software for faster and more accurate malware detection.

The remaining features in Fig. 3 show weaker dissimilarity from the uninfected feature plots but have the potential to be useful. The network trojan and the attempted denial of service alerts are the only alerts that exhibit meaningful patterns when considering total flows or total packets sent. The following analyses and results in Section VI-B focus on the most important feature as identified from the figure, namely the average inter-arrival time.

B. Malware Spread Results

Further analysis of these statistics can be performed by applying the ring model detailed in Section V and inspecting

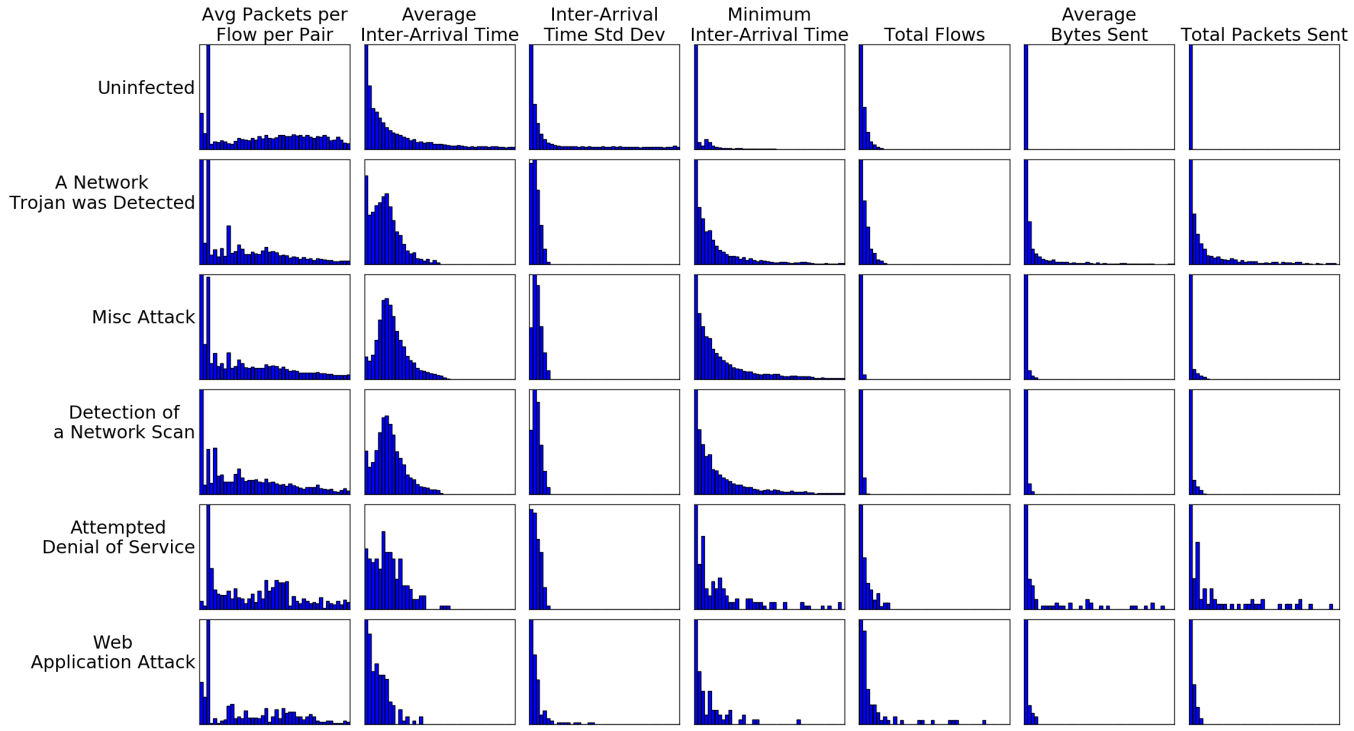


Fig. 3. Feature histograms plotted for various Snort alert classifications. The plots for each attack type were compared to the uninfected-IP values, shown in the first row. In particular, the average and minimum inter-arrival times seem to contain information that could be used to track malware flow.

the behavior progression of the rings. The clearest example of ring-based behavior found in the data is demonstrated by the flow of miscellaneous attacks within the network. "Miscellaneous attacks" refers to a Snort alert classification typically associated with compromised or hostile host traffic. For this alert type, Fig. 4(a) displays the average inter-arrival time distributions for each of the first four rings. The ring in the top left plot represents Ring 0, or the infected nodes for this type of attack. This group shows a Gaussian-shaped distribution of average inter-arrival times. By Ring 1, the peak has shifted to zero, as it is for the uninfected case. The second and third rings both resemble the average inter-arrival time of the uninfected nodes, as shown in Fig. 5. This behavior suggests that the effect of the attack dissipates as hosts become further removed from the infected node. This same behavior pattern was also observed for network trojan and attempted denial of service attacks.

From Fig. 4(a), it is clear that the distribution distinctly changes across the rings for this attack type. This shows that while for the collective attacks it is not possible to classify the flow statistics under the ring-based approach, given the type of attack, it is possible to model a flow-based progression through the rings. The figure shows that the outer rings slowly revert back to the *normal* network behavior (i.e., the behavior of an uninfected node as shown in Fig. 5).

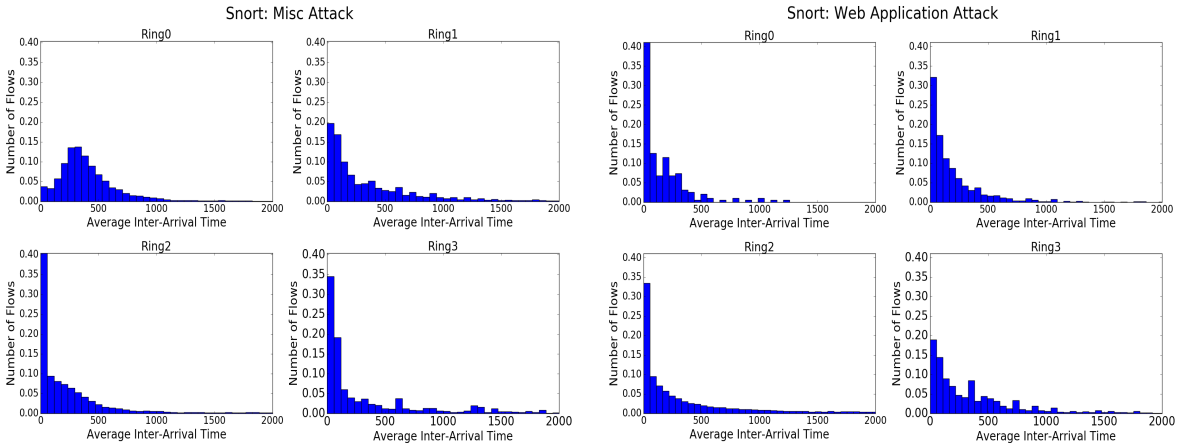
However, this model does not seem to apply to all attack types. It is clear from Fig. 3 that the Web Application attack distribution for average inter-arrival time does not significantly differ from that for the uninfected IP addresses. Inspecting this feature value through the ring model in Fig. 4(b) shows the distributions changing very little from one ring to the next. In contrast to Fig. 4(a), there is no clear transition to the uninfected distribution through the flows from the alert to Ring 3. Another type of Snort alert that did not support the ring-based model view was attempted administrative privilege gain. For certain feature-attack type combinations, the ring-based model can effectively model malware spread, but other combinations do not exhibit this behavior.

VII. Conclusions and Future Directions

This paper presents exploratory malware propagation analysis on a real-world dataset that consists of the perimeter-sensing flow data of a fully connected network. Identifying malware flow patterns can be used to track and manage malware that has infiltrated the network after being flagged at the perimeter. An intelligent tracking system could predict movement of an infection through the network in real time to allow the system to respond immediately in preventing further spread by isolating users in the projected path.

This work has shown that by utilizing appropriate features, meaningful patterns can be extracted from the network data without ground truth knowledge of attacks. A novel ring-based paradigm is introduced that, combined with meaningful features, accurately models the flow of certain malware types. Preliminary results indicate that applying these methods could aid detection of botnet attacks in real time.

There are opportunities to expand on the material presented here by adding new features, further ring-based model analysis, and applying these findings to other networks. This includes conducting a deeper investigation into the other



(a) (b)

Fig. 4. Distributions of average inter-arrival time for (a)Miscellaneous attacks: This feature applied to this alert exhibits ring-based propagation behavior; (b) Web Application attacks: This feature applied to this alert does not exhibit ring-based propagation behavior.

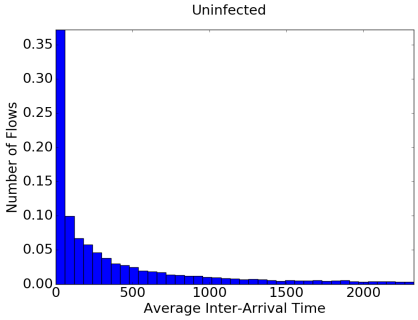


Fig. 5. Distribution of average inter-arrival times for uninfected hosts within the network. An uninfected host is one that does not receive any flows marked as suspicious by the Snort perimeter alert.

features identified in Fig. 3, particularly in analyzing how these features behave in the context of the ring-based model. Additional features could be generated considering the pairwise connections, as was the case with the pairwise average packets per flow feature. Further work should investigate quantifying and testing these observed distributional differences associated with malware attacks. In order to make general claims, it is necessary to perform similar studies using other datasets to substantiate the results presented in this work. The eventual application of this research would be to implement these methods in a real-time environment to evaluate the botnet detection accuracy and how effectively such a tracking system can manage an attack.

REFERENCES

[1] M. Abu Rajab, J. Zarfoss, F. Monrose, and A. Terzis, "A multifaceted approach to understanding the botnet phenomenon," in *Proceedings of the 6th ACM SIGCOMM conference on Internet measurement*. ACM, 2006, pp. 41–52.

[2] J. Kim, S. Radhakrishnan, and S. K. Dhall, "Measurement and analysis of worm propagation on internet network topology," in *Computer Communications and Networks, 2004. ICCCN 2004. Proceedings. 13th International Conference on*. IEEE, 2004, pp. 495–500.

[3] M. E. J. Newman, "Spread of epidemic disease on networks," *Physical Review*, vol. E 66, no. 1(2012):016128, 2002.

[4] J. O. Kephart and S. R. White, "Measuring and modeling computer virus prevalence," in *IEEE Computer Society Symposium on Research in Security and Privacy,*, May 1993, pp. 2–15.

[5] J. O. Kephart, "C. langton, ed., artificial life iii. studies in the sciences of complexity," in *IEEE Computer Society Symposium on Research in Security and Privacy,*, 1994, pp. 447–463.

[6] J. O. Kephart and S. R. White, "Directed-graph epidemiological models of computer viruses," in *IEEE Computer Society Symposium on Research in Security and Privacy,*, May 1991, pp. 343–359.

[7] K. J. Hall, "Thwarting network stealth worms in computer networks through biological epidemiology," 2006.

[8] J. J. Blount, D. R. Tauritz, and S. A. Mulder, "Adaptive rule-based malware detection employing learning classifier systems: A proof of concept," in *IEEE 35th Annual Computer Software and Applications Conference Workshops (COMPSACW)*, July 2011, pp. 110–115.

[9] J. François, S. Wang, T. Engel *et al.*, "Bottrack: tracking botnets using netflow and pagerank," in *NETWORKING 2011:Lecture Notes in Computer Science*. Springer, 2011, vol. 6640, pp. 1–14.

[10] F. Daryabar, A. Dehghantanha, and H. G. Broujerdi, "Investigation of malware defence and detection techniques," *International Journal of Digital Information and Wireless Communications (IJDIWC)*, vol. 1, no. 3, pp. 645–650, 2011.

[11] L. Bilge, D. Balzarotti, W. Robertson, E. Kirda, and C. Kruegel, "Disclosure: detecting botnet command and control servers through large-scale netflow analysis," in *Proceedings of the 28th Annual Computer Security Applications Conference*. ACM, 2012, pp. 129–138.

[12] S. Wang, R. State, M. Ourdane, and T. Engel, "Riskrank: Security risk ranking for ip flow records," in *Network and Service Management (CNSM), 2010 International Conference on*. IEEE, 2010, pp. 56–63.

[13] C. Livadas, R. Walsh, D. Lapsley, and W. T. Strayer, "Usilng machine learning technliques to identify botnet traffic," in *Local Computer Networks, Proceedings 2006 31st IEEE Conference on*. IEEE, 2006, pp. 967–974.

[14] W. Glodek and R. Harang, "Rapid permissions-based detection and analysis of mobile malware using random decision forests," in *Military Communications Conference, MILCOM 2013-2013 IEEE*. IEEE, 2013, pp. 980–985.

[15] M. Antonakakis, R. Perdisci, D. Dagon, W. Lee, and N. Feamster, "Building a dynamic reputation system for dns." in *USENIX security symposium*, 2010, pp. 273–290.

[16] Z. Berkay Celik, R. J. Walls, P. McDaniel, and A. Swami, "Malware traffic detection using tamper resistant features," in *Military Communications Conference, MILCOM 2015-2015 IEEE*. IEEE, 2015, pp. 330–335.

[17] C.-T. Lin, N.-J. Wang, H. Xiao, and C. Eckert, "Feature selection and extraction for malware classification," *Journal of Information Science and Engineering*, vol. 31, no. 3, pp. 965–992, 2015.

[18] M. Thomas and A. Mohaisen, "Kindred domains: detecting and clustering botnet domains using dns traffic," in *Proceedings of the companion publication of the 23rd international conference on World wide web companion*. International World Wide Web Conferences Steering Committee, 2014, pp. 707–712.