COMPUTERS & SECURITY 80 (2019) 306–316

Available online at www.sciencedirect.com

ScienceDirect

journal homepage: www.elsevier.com/locate/cose

Computers & Security

Identifying infected users via network traffic

Margaret Gratian[a], *Darshan Bhansali*[a], *Michel Cukier*[a], *Josiah Dykstra*[b,*]

[a] *University of Maryland, College Park, United States*
[b] *Department of Defense, United States*

ARTICLE INFO

Article history:
Received 8 June 2018
Revised 8 September 2018
Accepted 13 October 2018
Available online 22 October 2018

Keywords:
User behavior
Cybersecurity
Network traffic analysis
Feature engineering
Supervised learning
Unsupervised learning
Principal component analysis

ABSTRACT

There has been increasing interest in deeper understandings of users and user behavior to tailor and strengthen cybersecurity. Just as in the field of medicine where symptoms of a disease trigger early doctor intervention, we investigate the feasibility of extracting indicators from network traffic that can be used for proactive identification of user infection, which in turn can prompt proactive intervention from a network administrator. Using two months of wireless traffic from a large, public university, we attempt to differentiate between 1923 users with infected devices and random samples of uninfected users. We extract 36 features from the traffic and apply various dimensionality reduction techniques, unsupervised clustering methods, and supervised learning algorithms. Principal Component Analysis suggests 10 features that contribute most to explaining about 92.8% of variance in user behavior, revealing 26 features less useful for understanding users. K-means clustering partitions users in distinct groups that in the majority of cases separate the infected and uninfected users, with some clusters being composed of up to 100.0% of only one type of user. Finally, supervised learning yields accuracy values up to 79.0% and ROC AUC values up to 86.0% when classifying users as either infected or uninfected. Our work shows there is potential to derive infection 'symptoms' from network traffic.

Published by Elsevier Ltd.

1. Introduction

In both research and development there has been extensive work to use network traffic to proactively identify entities that are susceptible to either becoming infected or spreading infection. Network traffic analysis of IP addresses, binaries, and devices has become an invaluable process for proactive identification of security threats; anomaly and network intrusion detection systems often boast over 99% accuracy rates for detecting suspicious, malicious, or compromised entities.

In recent years, there has been increasing interest in deeper understanding of individual users and user behavior to tailor and strengthen cybersecurity. Motivated by the wealth of information that the network traffic originating from an IP address provides about the nature of the IP, we explore whether insights from user network traffic can provide an important new perspective from which to understand users.

Several authors have observed that user network traffic contains discernable patterns that can act as user fingerprints (Verde et al., 2014; Alotibi et al., 2015; Pang et al., 2007). However, work in this area is limited and generally focuses on the goal of uniquely identifying users from their traffic flows. Our research presents a complementary approach: given a corpus of network traffic and ground truth data about the mapping of traffic records to the unique user who generated the traffic, we explore whether features from a user's 'network fingerprint' can be used to differentiate between users who are more likely to become victims of cyber compromise and users who are less likely to become victims.

* Corresponding author.
E-mail addresses: mgratian@umd.edu (M. Gratian), darshan.bhansali@rhsmith.umd.edu (D. Bhansali), mcukier@umd.edu (M. Cukier), jdykstra@ltsnet.net (J. Dykstra).
https://doi.org/10.1016/j.cose.2018.10.007
0167-4048/Published by Elsevier Ltd.

Specifically, we investigate the feasibility of extracting indicators from user network traffic that are correlated with device infection. Just as in the field of medicine where symptoms of a disease trigger early doctor intervention, we aim to identify network traffic-based indicators that can be used for proactive identification of user infection, which in turn can prompt proactive intervention from a network administrator.

We analyze two months of wireless network traffic at a large public university for all 1923 infected users and random samples of uninfected users. In this study, we consider a user to be represented by a collection of devices and wireless traffic records on the network, and define an infected user as someone who has a record of device compromise in the university's intrusion detection/protection system logs.

We make the following contributions:

- We extract 36 features from network traffic and apply Principal Component Analysis to identify 13 principal components that explain about 92.8% of the variance in user behavior. Analysis of these 13 components reveals 10 features that best explain user behavior, suggesting 26 features that may be less useful for understanding users.
- Through unsupervised learning techniques, we show that there are differences in users revealed by their network traffic that can be used to partition users into distinct groups, and in some cases these clusters separate the infected and uninfected users, with some clusters having a composition up to 100.0% of only one type of user.
- We apply 10 different supervised learning techniques using both the original 36 features and the 13 principal components and evaluate the accuracy, area under the receiver operating characteristic curve (ROC AUC), false positive rate, and false negative rate with which we can classify users as either infected or uninfected. We show that using network traffic alone to classify users, we can achieve accuracy values up to 79.0% and ROC AUC values up to 86.0%. However, the lowest false positive rate is 16.8% and the lowest false negative rate is 21.2%, suggesting additional features or sources of data may be necessary to further refine the models.
- We show that there is potential to identify signals of infection status from network traffic patterns without knowledge of the actual content that a user is accessing.

The remainder of this paper is organized as follows. Section 2 presents the related work and research questions. Section 3 presents the methodology for feature engineering, dimensionality reduction, unsupervised clustering, and supervised learning. Section 4 presents the results. Section 5 discusses our results. Section 6 discusses the limitations. Finally, Section 7 concludes the paper and presents some areas of future work.

2. Background and related work

Network traffic, either raw or summarized in a form such as Cisco's NetFlow, is commonly analyzed for the purpose of anomaly or network intrusion detection (Bhuyan et al., 2014). Since hostile traffic generally looks different from benign traffic, these approaches often involve extracting various features about an IP address from the network traffic - such as byte or packet values, temporal patterns, source and destination location information - and then using some statistical, classification, clustering, or rules-based algorithm to distinguish between malicious and benign IP addresses (Bhuyan et al., 2014; Mahoney, 2003; Hammerschmidt et al., 2016; Hammerschmidt et al., 2016; Evangelou & Adams, 2016; Münz et al., 2007; Tjhai et al., 2010).

Several authors have observed that grouping and analyzing traffic flows from a human user-centric perspective versus an IP or host-centric perspective provides a more nuanced understanding of a network and may prove useful for identifying threats to the network, especially because users typically generate traffic on multiple devices and a single IP can map to thousands of unique users (Verde et al., 2014; Alotibi et al., 2015, 2016; Clarke et al., 2017). In Verde et al. (2014), the authors observed that traffic generated by an individual can act as a biometric signature and developed a system that could infer the identity of users from NetFlow records even when thousands of users were hidden behind a few IP addresses. The authors used eight hours of NetFlow records to profile five users on a large metropolitan WiFi network with a total of 200,000 users and an average of 1000 users NAT' d behind two IPs addresses. When attempting to identify these users from a set of 100 million raw NetFlow records collected over a 24 h time period, the system correctly identified the users with true positive rates over 90% and false positive rates below 0.08.

In Clarke et al. (2017), the authors examined the possibility of user identification using network packet metadata and neural networks. They collected network traffic consisting of fields such as time stamps, source and destination IP and port information, and packet length for 46 users over a two-month period. They then examined whether or not they could use traffic metadata to identify specific users on nine popular services. Results varied depending on the user and the service. For example, on Google, the neural net achieved a true positive identification rate of 90% for one user, but a true positive identification rate of only 48.3% for another. Similarly, in Alotibi et al. (2016), the authors developed user-behavioral profiles from raw traffic network metadata and found that they could accurately identify users on Skype, Hotmail, and BBC with 98.1%, 96.2%, and 81.8% accuracy, respectively. In Alotibi et al. (2015), the authors examined how to derive high-level behavioral features from low-level network traffic metadata in order to build profiles that indicate the services used by an individual and how they use the service; when the authors evaluated their profiling technique on Facebook, they were able to distinguish between different user activities using the raw traffic alone.

In Brown et al. (2014), the authors developed a tool that used HTTP traffic from four users in a domestic household network to identify the user generating the traffic with a true-positive rate of 64% and a false-positive rate of 28%. In Tomšů et al. (2017), the authors profiled 25 synthetic users with 90% accuracy based on web transactions and observed that user web transactions are fairly consistent and exhibit little novelty over time. In Pang et al. (2007), the authors examined how effectively they could identify whether a 802.11 traffic sample came from a user and found that 'implicit identifiers' such

as network destinations (IP address and port pairs) and SSIDs were highly useful in distinguishing users.

Our work is most similar to the work presented in Canali et al. (2014), in which the authors analyzed the web browsing behaviors of users to predict who was at risk of being exposed to malware. Using telemetry data from a major AV company containing 100,000 users and millions of URLs visited by the users, the authors extracted 74 features summarizing user behavior, to include volume of activity, temporal patterns of activity, number and type of websites visited, and variability of browser activity. Correlation analysis was performed to understand the relationship between these features and the probability of the user encountering a malicious web page. The authors defined *at risk* users as those who visited at least two distinct malicious URLs or at least three blacklisted domains during the 2-month study period; *safe* users never visited malicious URLs or blacklisted domains; and the *uncertain* users were those who did not fall into either category. The authors found that there was a significant increase in malicious URLs visited by users on the weekend; *at risk* users spent more time online at night; and volume of activity was one of the best predictors of being *at risk*. The authors then used insights from their correlation analysis to build logistic regression models; the model classified users as *at risk* with 74% accuracy and an 8% false positive rate.

An important distinction between our work and the work in Canali et al. (2014) is that our work is better suited to network administration contexts where an administrator would like to be able to forecast which users will become victims of threats or vulnerabilities as identified by the monitoring tool implemented on the network, but due to the privacy expectations of users cannot instrument user devices, require specific antivirus products, or know the actual content users are accessing on the web. Our work attempts to identify indicators and make predictions about users in a privacy-preserving manner using network patterns alone. The actual specifics of our data will be discussed in detail in Section III.

Based on the related work that has shown how features can be extracted from network traffic to either uniquely identify or profile different types of users, the first research question for this study is:

RQ1. *Are there features that can be extracted from the network traffic that best explain the differences in user behavior?*

Motivated by the extensive related work that has shown how features extracted from historic network traffic can be used to develop profiles that distinguish between malicious and benign hosts or IP addresses (and in the case of Canali et al. (2014), *at risk* users), we examine whether the features we extract to explain user behavior can also be used to distinguish between infected and uninfected users. The second research question for this study is therefore:

RQ2. *How well can we differentiate between infected and uninfected users on the network using features derived from user network traffic?*

We evaluate RQ2 using both an unsupervised clustering and supervised learning approach. In the context of unsupervised clustering, we evaluate cluster purity, i.e. the percent of users in a cluster that represent only one type of user (infected or uninfected). In the context of supervised learning, we evaluate accuracy, area under the receiver operating characteristic curve (ROC AUC), false positive rate, and false negative rate. The technique to compare these four attributes is discussed later in the Methodology section.

3. Methodology

This section presents an overview of the dataset, the steps taken to clean and process the data, and the experiments to test our research questions.

3.1. *Data overview*

The data for this experiment consists of the wireless network traffic and wireless network threat events collected by a Palo Alto Networks Intrusion Detection/Protection System at a large, public university. The security division of the university's Information Technology (IT) department manages this system. From September 26 to October 18, the system version was Pan OS 7.0.10. It was upgraded to version 8.0.4 on October 18th and to 8.0.5 on October 25th. The system is configured to detect threats using the default Palo Alto signatures and detection rules. The IT department modifies the list of domains and IPs that the system's firewall is set to block based on a feed of active hostile threats updated every 15 minutes.

The IT security office provided us with a random sample of traffic records from each hour of each day for the period of September 26, 2017 through November 21, 2017. They provided us with a complete record of the threats during this time period.

Of all the traffic generated by users at the university, 65% is wireless. Users of the wireless network include students, faculty, staff, and guests at the university. Users must authenticate onto the wireless campus network using a unique user ID that is recorded by the Palo Alto system.

To guarantee privacy of the users, the IT security office replaced user IDs with unique hashes. Anonymous sessions or sessions associated with a router or device that did not map back to a specific user were discarded. The resulting data was then pushed to a secure server hosted at the university for our analysis. After the process of discarding and anonymization, the dataset consisted of 66,561,686 wireless traffic records and 14,621 threat records.

The traffic logs record wireless traffic sessions on the network, with a session representing a complete conversation between a source and destination IP address. That is, a single record in a traffic log captures the request sent from a source IP to a destination IP and the response received by the source IP from the destination IP. The traffic logs record 54 fields associated with the session, including the source user ID, time the session began, the source and destination IP, number of bytes sent and received, the number of packets sent and received, the length of the session, and so on.

The threat logs record events that occurred during wireless traffic sessions that were identified as malicious or suspicious by the Palo Alto system. Therefore, every record in the threat logs can be matched with a record in the traffic logs representing the session during which the threat occurred. If

a threat is identified by the system during a wireless traffic session, once the session is complete the associated source and destination IP addresses and ports are recorded in the threat log, along with information about the protocol, time of the threat, type of threat, severity level, and so on. Severity levels are ranked as informational, low, medium, high, or critical. Our threat dataset consisted of the threats ranked medium, high, or critical, as these are the events that trigger a response from the IT department.

More information about the Palo Alto system and its traffic logs and threat logs can be found at Palo Alto Networks (2018).

3.2. *Data validation and preprocessing*

This section discusses the steps that were taken to validate and prepare the data for our analysis. Validation and preprocessing were done using Python, with Pandas (2018) and NumPy (2017) packages.

The first step involved identifying the number of unique users in our data. Filtering by unique user hash, we determined there were 53,165 unique users who appeared in the traffic logs. In the threat logs, we identified 1935 unique users. Since every threat record is associated with a traffic session, any user who appears in the threat logs should also appear in the traffic logs. Comparing the users in the threat logs to the users in the traffic logs, we identified 12 users who never appear in the traffic logs. Because data is taken directly from the IDS/IPS threat logs, we assume these 12 users represent a corner case in the threat recording process. One possible explanation is if a threat is identified at the start of a session and is immediately blocked, a completed traffic session never occurs and therefore is never recorded in the traffic log. These 12 users and their records in the threat logs were dropped from the analysis, resulting in 1923 unique users in the threat logs. We denote the 1923 users who appeared in the threat records – in other words users who have created threats assigned a severity level of medium, high, and/or critical – as the *infected users*.

The next step was ensuring that we could identify the traffic sessions associated with the threat records for these 1923 users. For each infected user threat record, we filtered the traffic logs by the user's unique user hash. From this set of potential traffic sessions, we then filtered by the recorded time of the traffic session and the threat record. Since traffic records only appear in the traffic logs when the conversation between the source and destination IP has been completed or terminated and because some sessions may last several days, we first tested filtering the traffic sessions by a 48-hour window before the threat log was recorded and a 48-hour window after the threat log was recorded. From this filtered set of records, we checked if there was an IP address in the traffic record that matched an IP address in the threat record. After doing this process for all the threat records, there were only 28 records created by a total of 11 unique users that were not matched. By expanding the time window from 48 hrs to 96 hrs before and after a threat was recorded and then checking for a match on either source or destination IP address, we matched six more records associated with five unique users. By expanding our search to month-long windows and then searching for a match on source or destination IP, we matched three records created by two unique users. By relaxing the condition on matching a source and destination IP, we matched the remaining 19 records created by four unique users by first searching a 96-hour window and then expanding to a month-long window.

After this phase of preprocessing, there were 1923 unique infected users and 51,242 unique uninfected users. Because the uninfected users represent about 3.6% of the total users in our sample, using the entire set of uninfected users in our analysis would result in artificially high accuracy results when attempting to classify users as infected or uninfected. So, we conducted our analysis on two different ratios of infected and uninfected users. One ratio represented a 50:50 split of infected to uninfected users, so 1923 infected and 1923 uninfected. The uninfected users represented a random sample of the total uninfected population. To ensure that results were not the product of the random sample, we produced three different random samples of uninfected users and repeated our analysis of the 50:50 ratio for each sample.

In order to also have a dataset with an imbalance of infected and uninfected users, we also selected a 30:70 ratio of infected to uninfected, so 1923 infected and 4487 uninfected. Again, we conducted our analysis on three random samples containing 4487 uninfected users each.

The final step in our data-preprocessing phase was to analyze and clean the fields in the actual traffic sessions. In the traffic logs, there are 54 fields recorded for a single traffic session. These fields are defined by Palo Alto so not all of them are relevant on the university network. We reduced this list of 54 fields to ten fields after dropping fields that contained missing values, always recorded the same value, or provided only minimal information about a traffic session. The final fields retained for feature engineering were: *Start Time, Elapsed Time, Source IP, Source Port, Destination IP, Destination Port, Bytes Sent, Bytes Received, Packets Sent,* and *Packets Received*.

3.3. *Feature engineering*

The ten features were then transformed into numerical features that summarized all of a user's traffic records. Feature engineering was driven both by exploratory data analysis and related work that has identified useful features for differentiating between malicious and benign IP addresses (Bhuyan et al., 2014; Münz et al., 2007; Evangelou and Adams, 2016; Tjhai et al., 2010) and the work in Canali et al. (2014) to identify users at risk of malware encounters. This process resulted in 36 different features, presented in Table 1.

To account for the fact that some features are quantified using different units of measurement (for example, frequency count vs. time) and to ensure each variable received equal weight, feature values were standardized prior to analysis using the process described in The Pennsylvania State University (2018).

3.4. *Principal component analysis*

Principal Component Analysis (PCA) is a statistical dimensionality reduction technique that converts a set of potentially correlated features into a smaller set of orthogonal

Table 1 – The network traffic features.

Code	Feature	Code	Feature
NREC	Number of user records in the traffic logs	*0600*	Number of sessions initiated by a user between 0600 and 0659
LEN	Average session length	*0700*	Number of sessions initiated by a user between 0700 and 0759
TDIF	Average time difference between session start times	*0800*	Number of sessions initiated by a user between 0800 and 0859
BYS	Average number of bytes sent	*0900*	Number of sessions initiated by a user between 0900 and 0959
BYR	Average number of bytes received	*1000*	Number of sessions initiated by a user between 1000 and 1059
PAS	Average number of packets sent	*1100*	Number of sessions initiated by a user between 1100 and 1159
PAR	Average number of packets received	*1200*	Number of sessions initiated by a user between 1200 and 1259
SIP	Number of unique source IP addresses	*1300*	Number of sessions initiated by a user between 1300 and 1359
SP	Number of unique source ports	*1400*	Number of sessions initiated by a user between 1400 and 1459
DIP	Number of unique destination IP addresses	*1500*	Number of sessions initiated by a user between 1500 and 1559
DP	Number of unique destination ports	*1600*	Number of sessions initiated by a user between 1600 and 1659
NAPP	Number of applications used	*1700*	Number of sessions initiated by a user between 1700 and 1759
0000	Number of sessions initiated by a user between 0000 and 0059	*1800*	Number of sessions initiated by a user between 1800 and 1859
0100	Number of sessions initiated by a user between 0100 and 0159	*1900*	Number of sessions initiated by a user between 1900 and 1959
0200	Number of sessions initiated by a user between 0200 and 0259	*2000*	Number of sessions initiated by a user between 2000 and 2059
0300	Number of sessions initiated by a user between 0300 and 0359	*2100*	Number of sessions initiated by a user between 2100 and 2159
0400	Number of sessions initiated by a user between 0400 and 0459	*2200*	Number of sessions initiated by a user between 2200 and 2259
0500	Number of sessions initiated by a user between 0500 and 0559	*2300*	Number of sessions initiated by a user between 2300 and 2359

linearly uncorrelated features called the principal components that explain as much variance in the data as possible (Jolliffe, 1986). PCA has been used in the related work (Evangelou and Adams, 2016) to identify subsets of features from larger sets of features derived from traffic record data that are the most useful for understanding normal IP/host behavior.

PCA was performed using Python's sklearn.decomposition.PCA library (scikit-learn developers 2017a) on the standardized data. Scree plots were produced for each sample to determine the number of principal components that should be retained for analysis. We examined the Scree plots to see at what number of components the graphs began to level off. For the three samples with a 50:50 ratio, the graphs all leveled off at 13 components, providing us with percent variance explained ratio values of 90.01%, 90.22%, and 98.09%, respectively. Using similar reasoning for the 30:70 ratio samples, we also retained 13 components for all three cases, achieving percent variance explained ratios of 90.27%, 90.39%, and 97.71%. Due to space constraints we do not present the Scree plots here.

We then examined how each of the original 36 features contributed to the percent variance explained and the principal components. Fig. 1 displays the average percent variance explained by each feature for the three samples in the 50:50 dataset and the average percent variance explained by each feature for the three samples in the 30:70 dataset.

The correlation values between each feature and each of the 13 principal components were also calculated for each sample using the sklearn.decomposition.PCA library (scikit-learn developers 2017a). Absolute values of the correlation values were then calculated, because based on the method we used for calculating these variables, magnitude, not direction, is used for determining variable importance (Minitab 18 Support 2017). Again, due to space constraints the full tables are not reported here, but the Results section discussions features with correlation values over 0.5.

3.5. *Unsupervised clustering*

Motivated by related work (Münz et al., 2007; Tjhai et al., 2010) that has explored the feasibility of using K-means clustering to differentiate between normal and anomalous traffic, we apply K-means clustering to our six samples using both the 36 original features and the 13 principal components as input. K-means clustering works by grouping entities into a user-specified number of clusters based on similarities between the features of the entities.

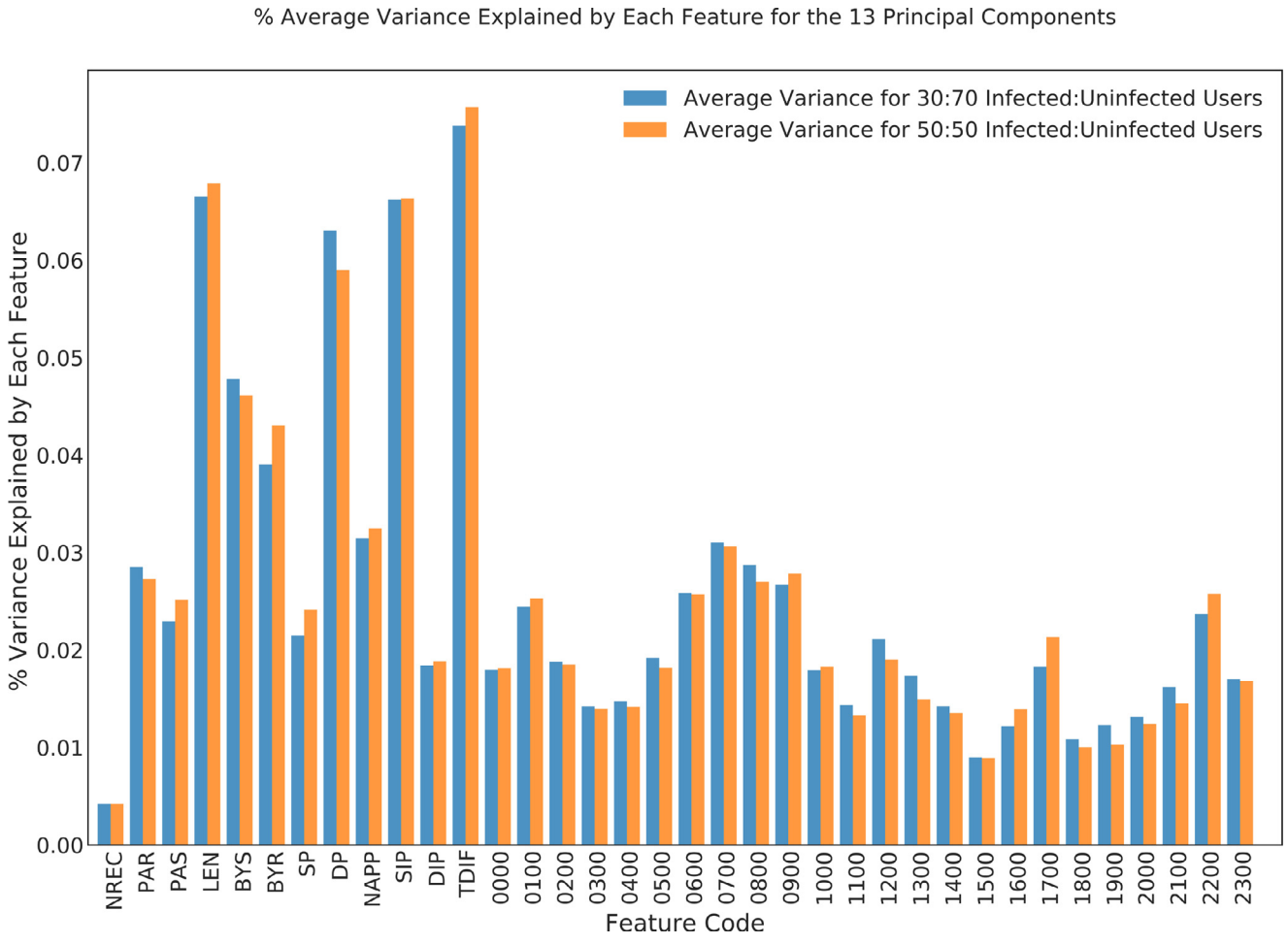

Fig. 1 – % Average variance explained by each feature for the 13 principal components for both ratios.

We evaluated K-values ranging from Verde et al. (2014) and Mahoney (2003) and for each K, calculated the sum of squares difference between the cluster center and each observation in the cluster. This value is 0 when all observations in a cluster are identical, so using this approach, the 'best' value of K is the one in which the sum of squares distance is minimized (Stanford University, 2018). Based on the sum of squares difference, the best value was K = 4.

Once the algorithm partitioned users into clusters, we examined how many infected and uninfected users appeared in each cluster. Note that K-means is an unsupervised technique, so the algorithm did not have knowledge of these labels. Table 2 presents the results of the percent and number of infected and uninfected users in each cluster for K = 4 for all three samples for each of the 50:50 and 30:70 ratios.

3.6. *Supervised learning*

To use our dataset for supervised learning, infected users were labeled with a 1 and uninfected users were labeled with a 0. Each sample in the 50:50 and 30:70 infected to uninfected ratio datasets were both split into training and testing sets. We evaluated two common train/test splits: 70/30 and 80/20.

Table 2 – K-means clustering results for K = 4 for each data sample using the original 36 features.

% Infected (# infected) in each cluster % Uninfected (# uninfected) in each cluster			
Cluster 1	*Cluster 2*	*Cluster 3*	*Cluster 4*
Original 36 features, 50:50 Ratio			
88.54% (340) 11.46% (44)	**25.27% (514)** 74.73% (1520)	**74.84% (1068)** 25.16% (359)	**100.0% (1)** 0.0% (0)
25.01% (513) 74.99% (1538)	**89.09% (343)** 10.91% (42)	**75.66% (1066)** 24.34% (343)	**100.0% (1)** 0.0% (0)
29.49% (681) 70.51% (1628)	**0.0% (0)** 100 (1)	**0.0% (0)** 100% (1)	**80.91% (1242)** 19.09% (293)
Original 36 features, 30:70 Ratio			
12.15% (489) 87.85% (3537)	**76.39% (356)** 23.61% (110)	**56.18% (1077)** 43.82% (840)	**100.0% (1)** 0.0% (0)
10.95% (416) 89.05% (3382)	**71.97% (534)** 28.03% (208)	**51.37% (938)** 48.63% (888)	**79.55% (35)** 20.45% (9)
14.47% (635) 85.53% (3754)	**0.0% (0)** 100.0% (1)	**63.79% (1288)** 36.21% (731)	**0.0% (0)** 100.0% (1)

Table 3 – K-means clustering results for K = 4 for each data sample using the Principal Components.

% Infected (# infected) in each cluster
% Uninfected (# uninfected) in each cluster

Cluster 1	*Cluster 2*	*Cluster 3*	*Cluster 4*
13 Principal Components, 50:50 Ratio			
81.85% (496)	**26.33% (465)**	**94.12% (16)**	**64.93% (946)**
18.15% (110)	73.67% (1301)	5.88% (1)	35.07% (511)
79.28% (1213)	**51.28% (20)**	**30.32% (690)**	**0.0% (0)**
20.72% (317)	48.72% (19)	69.68% (1586)	100.0% (1)
78.46% (1235)	**30.31% (688)**	**0.0% (0)**	**0.0% (0)**
21.54% (339)	69.69% (1582)	100.0% (1)	100.0% (1)
13 Principal Components, 30:70 Ratio			
12.19% (424)	**80.0% (16)**	**51.85% 1483)**	**0.0% (0)**
87.81% (3053)	20.0% (4)	48.15% (1377)	100.0% (53)
60.07% (1309)	**14.19% (595)**	**18.18% (4)**	**93.75% (15)**
39.93% (870)	85.81% (3598)	81.82% (18)	6.25% (1)
61.03% (1339)	**13.86% (584)**	**0.0% (0)**	**0.0% (0)**
38.97% (855)	86.14% (3630)	100.0% (1)	100.0% (1)

Eight supervised algorithms were implemented using Python's sklearn libraries: K-Nearest Neighbors (scikit-learn developers 2017b), Logistic Regression (scikit-learn developers 2017c), Random Forest (scikit-learn developers 2017d), AdaBoost (scikit-learn developers 2017e), Gradient Boost (scikit-learn developers 2017f), Extra Tree (scikit-learn developers 2017g), Bagging (scikit-learn developers 2017h), Voting (Majority Rule) (scikit-learn developers 2017i). For all the algorithms except Voting (which is built on the output of the other algorithms), Grid Search (scikit-learn developers 2017j) was used to tune the hyper-parameters and stratified 10-fold cross validation (scikit-learn developers 2017k) was conducted to prevent model overfitting. Two neural networks were implemented using the keras.io library (Keras Documentation, 2018), one with two layers, denoted Neural Network 1, and one with four layers, grid search to tune parameters, and 10-fold cross validation to prevent overfitting, denoted Neural Network 2. For all models, we calculate the accuracy, false positive rate, false negative rate, and area under the receiver operating characteristic curves produced by varying the decision threshold on balancing the true positive rate with the false positive rate.

In Tables 4 and 5, we present the average accuracy, ROC AUC, false positive rate (FPR), and false negative rate (FNR) across all three samples for the 50:50 ratio and 30:70 ratio, respectively. Due to space constraints, we do not present the results of all classifiers, but rather we present the *non-dominated* classifiers when comparing classifiers on four attributes: accuracy and ROC AUC (both of which we aim to maximize) and false positive rate and false negative rate (both of which we aim to minimize.

Non-dominated classifiers were determined using the dominance choice strategy (Herrmann, 2015), a technique commonly used in engineering disciplines to select the best alternative using multiple attributes as the criteria. To apply the technique, we eliminated classifiers if there was another classifier that performed better on all four attributes until we were left with a list of classifiers that could no longer be eliminated. For example, when identifying the non-dominated classifiers for the 50:50 ratio using the 36 original features and an 80/20 train/test split, Neural Network 1 was eliminated because Gradient Boost had a higher accuracy, higher ROC AUC, lower false positive rate, and lower false negative rate than it did. Gradient Boost was not eliminated because there was no other classifier that performed better across *all* four attributes. Note that the dominance calculations were made using six significant digits, but the tables present the values rounded to three significant digits for clarity and simplicity.

Table 4 – 50:50 Ratio average supervised algorithm results.

Classifier	*Accuracy*	*ROC AUC*	*FPR*	*FNR*
50:50 Ratio, 36 original features, 70/30 train/ test split				
Ada Boost	0.768	0.836	0.214	0.250
Gradient Boost	0.784	0.851	0.215	0.217
50:50 Ratio, 36 original features, 80/20 train/ test split				
Ada Boost	0.775	0.849	0.201	0.249
Gradient Boost	0.784	0.860	0.213	0.218
Bagging	0.784	0.854	0.211	0.221
Neural Network 2	0.788	0.853	0.210	0.214
50:50 Ratio, 13 principal components, 70/30 train/ test split				
Ada Boost	0.737	0.837	0.175	0.351
Gradient Boost	0.783	0.847	0.215	0.218
Extra Tree	0.752	0.810	0.203	0.294
50:50 Ratio, 13 principal components, 80/20 train/ test split				
Gradient Boost	0.787	0.854	0.212	0.214
Extra Tree	0.710	0.822	0.172	0.413
Neural Network 2	0.790	0.852	0.208	0.212

Table 5 – 30:70 Ratio Average Supervised Algorithm Results

Classifier	*Accuracy*	*ROC AUC*	*FPR*	*FNR*
30:70 Ratio, 36 original features, 70/30 train/ test split				
KNeighbors	0.784	0.851	0.211	0.229
Gradient Boost	0.784	0.853	0.215	0.217
Extra Tree	0.781	0.832	0.168	0.340
30:70 Ratio, 36 original features, 80/20 train/ test split				
Gradient Boost	0.785	0.850	0.214	0.217
Extra Tree	0.775	0.830	0.169	0.360
30:70 Ratio, 13 principal components, 70/30 train/ test split				
Random Forest	0.781	0.809	0.211	0.238
Ada Boost	0.780	0.845	0.211	0.239
Gradient Boost	0.780	0.849	0.220	0.221
Extra Tree	0.770	0.821	0.176	0.358
30:70 Ratio, 13 principal components, 80/20 train/ test split				
Extra Tree	0.774	0.824	0.196	0.230
Neural Network 2	0.781	0.847	0.218	0.220

4. Results

This section provides a more in depth discussion of the results of Principal Component Analysis (PCA), K-means clustering, and supervised learning.

4.1. *Principal component analysis*

For the datasets with a 50:50 ratio of infected to uninfected users, the average variance in the data explained by the 13 principal components was 92.77%. As was shown in Fig. 1, for all three samples there were about five different features that contributed the most variance to the overall variance: average session length, average number of bytes sent, number of unique destination ports, number of unique source IP addresses, and average time difference between session start times.

By examining the correlation values between the principal components and the original 36 features, we can further understand the network traffic features that contribute most to explaining the differences in users. Determining the threshold for a correlation value to be large enough for a feature to be considered important is a subjective decision (Minitab 18 Support 2017); higher values reduce the number of features considered important, while lower values increase the number of features.

For the 50:50 ratio, if we consider 0.4 to be the threshold on importance we retain 17 features; a threshold of 0.5 retains 10 features; 0.6 retains six features; 0.7 retains two features; and 0.8 retains one feature, a correlation value of 0.885 between average number of bytes sent and principal component 7 (PC7).

Setting the threshold at 0.5, the ten features considered important are: average number of packets received, average number of packets sent, average session length, average number of bytes sent, average number of bytes received, number of unique source ports, number of unique destination ports, number of unique applications used, number of unique source IP addresses, and average difference between session start times. Five of these features overlap with the five features that contribute the most variance to the overall variance.

For the datasets with a 30:70 ratio of infected to uninfected users, the average variance in the data explained by the 13 principal components was 92.79%. The same five features as in the 50:50 ratio dataset contributed the most variance to the overall variance: average session length, average number of bytes sent, number of unique destination ports, number of unique source IP addresses, and average time difference between session start times.

Again, if we consider 0.4 to be the threshold on importance we retain 17 features; a threshold of 0.5 retains nine features; 0.6 retains six features; 0.7 retains four features; and 0.8 retains two features, with the highest correlation value being 0.895 again between average number of bytes sent and PC7.

Setting the threshold at 0.5, the nine features considered important are: average number of packets received, average number of packets sent, average session length, average number of bytes sent, average number of bytes received, number of unique destination ports, number of unique applications used, number of unique source IP addresses, and average difference between session start times. Nine of these features overlap with the ten features presented in the 50:50 ratio; in this ratio, only number of unique source ports is no longer considered important.

In Research Question 1, we asked if we could derive features from network traffic that could explain the difference in user behavior, motivated by existing research that has suggested network traffic patterns can function as user fingerprints. Overall, comparing the PCA results of the two different ratios, there is a total of 10 unique features that, either based on the percentage of total variance explained or correlation values above 0.5 with individual principal components, are most important in shaping the final principal components and explaining user behavior.

4.2. *Unsupervised clustering*

We performed K-means clustering using both the original 36 features as input (see Table 2), and the 13 principal components as input (see Table 3). For all data samples the partitioning of users created by the K-means algorithm often aligned with the infected/ uninfected labels that we assigned to users.

Examining the clusters created using the 36 original features as input, for the 50:50 ratio there are a few instances where clusters consist of a single user; we interpret Cluster 4 in the first and second samples and Clusters 2 and 3 in the third sample as potential outliers. Excluding these instances, we see that cluster purity ranges from about 70.51% (1628 uninfected users in Cluster 1 in the third data sample) to 89.09% (343 infected users in Cluster 2 in the second data sample). For the 30:70 ratio, we similarly see a majority of high cluster purity instances and three outlier instances. We also see instances where the cluster composition is split almost evenly between infected and uninfected users; for example, in the second sample, 51.37% of Cluster 3 represents infected users and 48.63% represents uninfected users.

When we examine the results of K-means using the 13 principal components, for the 50:50 ratio there are three outlier instances and one cluster instance that splits infected and uninfected users evenly. Besides these instances, cluster purity values range from 64.93% (946 infected users in Cluster 4 in the first sample) to 94.12% (16 infected users in Cluster 3 in the first sample). For the 30:70 ratio, there are two outlier instances and one instance where a cluster is evenly split. Besides these instances, cluster purity ranges from 60.07% (1309 infected users for Cluster 2 in the second sample) to 100.0% (53 uninfected users for Cluster 4 in the first sample).

By the nature of unsupervised clustering, we do not have insight into how the algorithm made its decisions to partition users. Our interest in applying K-means was to understand if there was a signal of infection status present in the network traffic data; since the algorithm had no insight into user infection status, users were partitioned according to whatever features or combinations of features K-means determined provided the strongest signal for separating users into clusters.

Our results suggest that a signal for infection status is present in both the 36 features we extracted from the network traffic records and in the 13 principal components derived from these features. This implies both may have potential for our long-term goal to develop a set of user infection symptoms. However, because some clusters were partitioned almost evenly between infected and uninfected users, to validate the strength of the infection signal derived from network traffic, further analysis over longer periods of time and for additional samples of users is necessary.

Recall that Research Question 2 asked what is the highest cluster purity of infected or uninfected users we can achieve when using unsupervised clustering to differentiate between users. Using the network traffic, the highest purity we achieve is 89.09% infected users; using the principal components, the highest purity we achieve is 100.0% infected users.

4.3. *Supervised learning*

For the 50:50 ratio of infected to uninfected users, classification accuracies of the top performing models ranged from 71.0% to 79.0%, ROC AUCs ranged from 81.0% to 86.0%, false positive rates ranged from 17.2% to 21.5%, and false negative rates ranged from 21.2% to 41.3%. While Bagging is listed as a top performing classifier (see Table 4), its training accuracy was at least 10% higher than its testing accuracy, suggesting that the model may have been overfitting.

For the 30:70 ratio of infected to uninfected users, classification accuracies of the top performing models ranged from 77.0% to 78.5%, ROC AUCs ranged from 80.9% to 85.3%, false positive rates ranged from 16.8% to 22.0%, and false negative rates ranged from 21.7% to 36.0%.

In deciding which algorithm results to include in the paper, we chose to consider the four different attributes of equal importance. In practice of course, a decision-maker would need to make a determination about what attribute(s) is most important. If the goal is to supplement a firewall or anti-virus system, i.e. a disease confirmation test, with indicators of user infection, i.e. warning symptoms that trigger early intervention – as is the intended goal of our work - then high false positive rates may not be as detrimental or concerning as high false negative rates and the number or rate of true positives may be considered most important. In this case, an algorithm like Neural Network 2 for the 50:50 ratio using the 13 principal components and an 80/20 train/test split performs quite well, with an accuracy of 79.0%, an ROC AUC of 85.2%, an FPR of 20.8%, and an FNR of 21.2%.

Recall that Research Question 2 asked how well we can classify users as infected or uninfected using supervised learning. For the 50:50 ratio, there is no one model that outperforms all the others on every attribute, but examining the attributes individually and excluding the Bagging classifier because of overfitting: 79.0% accuracy by Neural Network 2; 86.0% ROC AUC by Gradient Boost; 17.2% FPR by Extra Tree; and 21.2% FNR by Neural Network 2. Examining attributes individually for the 30:70 ratio: 78.5% accuracy by Gradient Boost; 85.3% ROC AUC by Gradient Boost; 16.8% FPR by Extra Tree; and 21.7% FNR by Gradient Boost.

5. Discussion

We extracted 36 features from the traffic records based on exploratory analysis and an extensive review of the related work. We identified features that summarize the data sent and received during traffic sessions, source and destination IP and port information, and temporal patterns of user network activity. Our features are suited to a network administration context where it is not possible to instrument devices or know the specific content that users are accessing on the web due to the privacy expectations of users. We therefore investigate the feasibility of understanding user infection status in a context-*unaware* manner.

We applied PCA to identify 13 principal components that best explain the variance in user behavior. These 13 components are able to explain 92.77% variance on average for the three data samples with a 50:50 ratio of infected to uninfected users; for the three data samples with the 30:70 ratio, the components explain on average 92.79% of the variance. After analyzing how each of the original 36 features contributed to the overall variance, we identified five features that may be of more importance than others in explaining the difference in user behavior: average session length, average number of bytes sent, number of unique destination ports, number of unique source IP addresses, and average time difference between session start times. After analyzing the correlation values between the original features and the 13 principal components and using a correlation value of 0.5 as a threshold for feature importance, we confirmed these five features and identified five additional features: average number of packets received, average number of packets sent, average number of bytes received, number of unique source ports, and number of unique applications used.

K-means clustering was performed using both the 36 original features and the 13 principal components. After the algorithm partitioned the users in to 4 distinct clusters based on the optimal K value, we identified the number and percent of infected and uninfected users that were partitioned into each cluster. Despite a few clusters consisting of a single user and a few clusters split evenly between infected and uninfected users, cluster purity ranged from 60.07% up to 100%, suggesting a signal of infection status exists in network traffic patterns.

Finally, we also tested an array of supervised learning algorithms for both data ratios and two different training/ testing splits to ensure any results we obtained were not just the product of our selected parameters. Though at a minimum our false positive rates are 16.8% and false negative rates are 21.2% - in other words, quite high - we were able to classify users as either infected or uninfected with accuracies up to 79.0% and ROC AUCs up to 86.0%, further showing that features extracted from the network traffic records of users can differentiate between the infected and uninfected.

6. Limitations

Our study has several limitations. First, we used traffic records flagged for containing threat activity by the university's IDS/IPS as ground truth for defining infection status. We do not know the specifics of how the system determines threats, and we also do not know the system's false positive rate or false negative rate. While knowing the labeling process might give us more insight into why users do or do not appear in the threat logs, understanding the nuances of the system is not relevant to our goal, as the IDS/IPS is simply what triggers a response from the network team at the university. Revisiting the disease analogy we presented earlier, developing a method to confirm a disease is a separate problem from identifying symptoms of the disease; the threat labeling pro-

cess is analogous to a disease confirmation test. Moreover, the system makes infection determinations with insight into the content a user is accessing, to include specific websites and downloads, while we are making determinations using only network traffic patterns.

Another limitation is how we chose to define the infected users. As noted previously, the Palo Alto IDS/IPS tags each threat with a severity level of informational, low, medium, high, or critical. We labeled users as 'infected' if they created medium, high, or critical threats and did not differentiate between infection severities. 1858 users created medium severity threats only, 41 created high severity threats only, and 11 created critical threats only. If we consider users who created high and/or critical threats as the infected population, the differences between the infected and uninfected may be more pronounced. However, our sample of critical and high severity threat users is currently too small to effectively apply the unsupervised or supervised algorithms presented here.

The next limitation is that we cannot guarantee completeness of the traffic data. Though the traffic logs should capture every session that occurs on the university network and this data is stored in a redundant manner by the IT division, due to the sheer volume of data we cannot validate that we have every record created by every user during our 2-month window of data.

Finally, our study focused on users at a university network. This was motivated by the availability of ground truth data to match users with network traffic flows. Therefore, though our methodology generalizes to other networks, it is possible that the network indicators identified in this experiment do not. However, the advantage of using an open university network is that there is the possibility of discoveries about users that would not be possible on a closed network. Unlike many corporate networks where extensive security checks and restrictions are in place, at the university users maintain their own devices, there are no restrictions on the type of devices or the software on them, and almost no websites or services are blocked. This enabled us to analyze how users behave on a network on which they have almost complete freedom.

7. Conclusion

In this paper, we conducted an empirical study on the feasibility of extracting indicators from the network traffic of users that can be used as early warning symptoms of infection. Motivated by work that has used insights from network traffic to proactively identify malicious IPs or infected hosts and more recent work that has suggested users have unique network traffic fingerprints, we analyzed two months of wireless network traffic for 1923 infected users and random samples of uninfected users at a large public university.

The long-term goal of our work is not to replace an intrusion detection system, but rather to supply security researchers and network administrators with network traffic indicators (or techniques to attain indicators) that support better user understandings. Being able to identify infected users early or identify users who are not yet infected but behave similarly to currently infected users can enable administrators to take proactive steps to mitigate infection, through added protections and safety mechanisms or security training for users who appear to be more at-risk than others to becoming infected.

Our future work involves analyzing the identified 36 features and 13 principal components over a longer period of time to confirm which of these features or combinations of features can be used as early infection 'symptoms' in practice, in addition to evaluating techniques to reduce our false positive and false negative rates. We will also examine how narrowing our definition of infection by treating only the high and critical threat users as infected impacts our clustering or supervised learning results. We also intend to incrementally incorporate additional sources of data and insights about users into our models. Finally, once we have identified features that provide a strong signal about infection status, we will explore causation by analyzing why these features reveal infection status and whether certain network behaviors and traits of users make some users more likely than others to be infected.

REFERENCES

Alotibi G, Li F, Clarke N, Furnell S. Behavioral-based feature abstraction from network traffic. In: Proceedings of the 10th international conference on cyber warfare and security: (ICCWS'15). Academic Conferences Limited; 2015. p. 1–9.

Alotibi G, Clarke N, Li F, Furnell S. User profiling from network traffic via novel application-level interactions. In: Proceedings of the 11th international conference for internet technology and secured transactions (ICITST'16). IEEE; 2016. p. 279–85. doi:10.1109/ICITST.2016.7856712.

Bhuyan MH, Bhattacharyya DK, Kalita JK. Network anomaly detection: methods, systems and tools. Commun Surv Tutor IEEE 16 2014;1:303–36. doi:10.1109/SURV.2013.052213.00046.

Brown A, Mortier R, Rodden T. An exploration of user recognition on domestic networks using NetFlow records. In: Proceedings of the 2014 ACM international joint conference on pervasive and ubiquitous computing: adjunct publication (UbiComp '14 Adjunct). New York, NY, USA: ACM; 2014. p. 903–10. doi:10.1145/2638728.2641560.

Canali D, Bilge L, Balzarotti D. On the effectiveness of risk prediction based on users browsing behavior. In: Proceedings of the 9th ACM symposium on Information, computer and communications security (ASIA CCS '14). New York, NY, USA: ACM; 2014. p. 171–82. doi:10.1145/2590296.2590347.

Clarke N, Li F, Furnell S. A novel privacy preserving user identification approach for network traffic. Comput Secur 2017;70:335–50. doi:10.1016/j.cose.2017.06.012.

Evangelou M, Adams NM. Predictability of NetFlow data. In: Proceedings of the 2016 IEEE conference on intelligence and security informatics (ISI'16). IEEE; 2016. p. 67–72. doi:10.1109/ISI.2016.7745445.

Hammerschmidt C, Marchal S, State R, Pellegrino G, Verwer S. Efficient learning of communication profiles from IP flow records. In: Proceedings of the IEEE 41st conference on local computer networks (LCN'16). IEEE; 2016. p. 559–62. doi:10.1109/LCN.2016.92.

Hammerschmidt Christian, Marchal Samuel, State Radu, Verwer Sicco. Behavioral clustering of non-stationary IP flow record data. In: Proceedings of the 2016 12th IEEE international conference on network and service (CNSM'16). IEEE; 2016. p. 297–301. doi:10.1109/CNSM.2016.7818436.

Herrmann JW. Engineering decision making and risk management. John Wiley & Sons; 2015.

Jolliffe IT. Principal component analysis. New York, NY: Springer; 1986.
Keras Documentation. 2018 Keras: the python deep learning library. Retrieved March 8 2018 from https://keras.io/.
Münz G, Li S, Carle G. Traffic anomaly detection using k-means clustering. Proceedings of the GI/ITG workshop MMBnet, 2007.
Mahoney MV. Network traffic anomaly detection based on packet bytes. In: *Proceedings of the 2003 ACM symposium on applied computing (SAC '03)*. New York, NY, USA: ACM; 2003. p. 346–50. doi:10.1145/952532.952601.
Minitab 18 Support. 2017. Interpret the key results for principal component analysis. Retrieved March 8 2018 from https://support.minitab.com/en-us/minitab/18/help-and-how-to/modeling-statistics/multivariate/how-to/principal-components/interpret-the-results/key-results
NumPy. 2017. Retrieved March 8 2018 from http://www.numpy.org/.
Palo Alto Networks. 2018. Syslog field descriptions. Retrieved March 8 2018 from https://www.paloaltonetworks.com/documentation/61/pan-os/pan-os/reports-and-logging/syslog-field-descriptions#_41809.
Pandas. 2018 Python data analysis library. Retrieved March 8 2018 from https://pandas.pydata.org/.
Pang J, Greenstein B, Gummadi R, Seshan S, Wetherall D. 802.11 user fingerprinting. In: Proceedings of the 13th annual ACM international conference on Mobile computing and networking (MobiCom '07). New York, NY, USA: ACM; 2007. p. 99–110. doi:10.1145/1287853.1287866.
scikit-learn developers. 2017. Decomposing signals in components (matrix factorization problems. Retrieved March 8 2018 from http://scikit-learn.org/stable/modules/decomposition.html#pca
scikit-learn developers. 2017. sklearn.neighbors.KNeighbors Classifier. Retrieved March 8 2018 from http://scikit-learn.org/stable/modules/generated/sklearn.neighbors.KNeighborsClassifierhtml.
scikit-learn developers. 2017. sklearn.linear_model.Logistic Regression. Retrieved March 8 2018 from http://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LogisticRegression.html
scikit-learn developers. 2017. sklearn.ensemble.RandomForest Classifier. Retrieved March 8 2018 from http://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html
scikit-learn developers. 2017. sklearn.ensemble.AdaBoost Classifier. Retrieved March 8 2018 from http://scikit-learn.org/stable/modules/generated/sklearn.ensemble.AdaBoostClassifier.html
scikit-learn developers. 2017. sklearn.ensemble.GradientBoosting Classifier. Retrieved March 8 2018 from http://scikit-learn.org/stable/modules/generated/sklearn.ensemble.GradientBoostingClassifier.html
scikit-learn developers. 2017. sklearn.tree.ExtraTreeClassifier. Retrieved March 8 2018 from http://scikit-learn.org/stable/modules/generated/sklearn.tree.ExtraTreeClassifier.html
scikit-learn developers. 2017. sklearn.ensemble.BaggingClassifier. Retrieved March 8 2018 from http://scikit-learn.org/stable/modules/generated/sklearn.ensemble.BaggingClassifier.html
scikit-learn developers. 2017. sklearn.ensemble.VotingClassifier. Retrieved March 8 2018 from http://scikit-learn.org/stable/modules/generated/sklearn.ensemble.VotingClassifier.html
scikit-learn developers. 2017. sklearn.model_selection.Grid SearchCV. Retrieved March 8 2018 from http://scikit-learn.org/stable/modules/generated/sklearn.model_selection.GridSearchCV.html
scikit-learn developers. 2017. sklearn.model_selection. StratifiedKFold. Retrieved March 8 2018 from http://scikit-learn.org/stable/modules/generated/sklearn.model_selection.StratifiedKFold.html
Stanford University. 2018 Error sum of squares (SSE). Retrieved March 8 2018 from https://hlab.stanford.edu/brian/error_sum_of_squares.html.
The Pennsylvania State University. 2018. Alternative: standardize the variables. Retrieved March 8 2018 from https://onlinecourses.science.psu.edu/stat505/node/55.
Tjhai GC, Furnell SM, Papadaki M, Clarke NL. A preliminary two-stage alarm correlation and filtering system using SOM neural network and K-means algorithm. Comput Secur 2010;29(6):712–23. doi:10.1016/j.cose.2010.02.001.
Tomšů R, Marchal S, Asokan N. Profiling users by modeling web transactions. In: Proceedings of the 37th IEEE international conference on distributed computing systems (ICDCS'17). IEEE; 2017. p. 2899–904. doi:10.1109/ICDCS.2017.164.
Verde NV, Ateniese G, Gabrielli E, Mancini LV, Spognardi A. No NAT'd user left behind: fingerprinting users behind NAT from NetFlow records alone. In: Proceedings of the 34th IEEE international conference on distributed computing systems (ICDCS '14). IEEE; 2014. p. 218–27. doi:10.1109/ICDCS.2014.30.

Margaret Gratian is a researcher at the Department of Defense and a Ph.D. candidate in Reliability Engineering at the University of Maryland, College Park. She has a Bachelor's degree in Mathematics and Computer Science from the University of Maryland. Her research interests include evaluating and quantifying user cyber security behaviors and user susceptibility to cyber crime.

Darshan Bhansali is a Research Assistant at the Maryland Cybersecurity Center at the University of Maryland. He holds a Master's degree in Information Systems and a Bachelor's degree in Computer Science. He is a former Data Analyst and his primary interests lie in Data Science and Machine Learning. His current project centers around examining self-reported instances of cyber-victimization by users on social media platform.

Michel Cukier is an associate professor of reliability engineering with a joint appointment in the Department of Mechanical Engineering at the University of Maryland, College Park. He is also the director for the Advanced Cybersecurity Experience for Students (ACES). His research covers dependability and security issues. His latest research focuses on the empirical quantification of cyber security. Dr. Cukier has published more than 70 papers in journals and refereed conference proceedings in those areas.

Josiah Dykstra holds the Ph.D. degree in Computer Science from the University of Maryland, Baltimore County. Dr. Dykstra is a Technical Director in Cybersecurity Operations in the Department of Defense. His research interests include network security, digital forensics, cloud computing, and human resilience in cyber security including augmented reality. He is a Fellow of the American Academy of Forensic Sciences and member of ACM.