Utility of User Roles in Comparing Network Flow Behaviors

Jeffrey S. Dean
United States Air Force
jeffrey.dean.1@us.af.mil

***Abstract*—We compared the practice of grouping users based on roles to define normal network behaviors with grouping users based on behavioral similarities. For this comparison we utilized Netflow-derived features to characterize the network traffic of users on a small campus, and labeled the feature vectors based on user roles. Using the same flow-feature sets we also grouped users based on shared network behaviors, using K-means++ to cluster user data sets. Intra-group similarities under the grouping strategies were tested in two ways. First, by using a linear (Nearest Centroid) classifier to measure how separable group data sets were. Second, by averaging each user's feature vector values by week of activity, and measuring pairwise distances between the centroid vectors to determine intra-group and inter-group distance distributions. Our tests showed that creating user groups via clustering user data sets created tighter ranges of user behavior as compared to grouping based on user roles.**

***Index Terms*—Netflow, machine-learning, behavior**

I. Introduction

Network administrators face a lot of challenges, but perhaps the most potentially damaging and difficult to detect challenge is the malicious insider. To detect network users behaving badly, a number of tools have been developed to identify anomalous user behaviors. In many cases detection is performed by comparing user profiles, either by change detection (comparing a user's current and past profiles) or deviation detection (comparing a user's profile against those of a group) [6]. Measured changes or deviations beyond some set threshold can be flagged, spurring further investigation.

While anomaly detection could be addressed by comparing past and present user profiles, such an approach has limitations. User behaviors could be anomalous during profile creation. Classifiers trained on individual profiles could overfit the data, resulting in false positives when user taskings change [11]. Defining normality based on aggregate activities of groups provides broader ranges of behaviors, reducing false positives. Ideally users within a group would behave similarly; grouping users that behave dissimilarly could result in broader thresholds leading to high false-negative rates. Identifying an approach to group users with similar behaviors would therefore be useful in creating meaningful detection thresholds.

One approach to creating groups of similar users is to do so based on organizational roles. Some network behavior analysis tools such as IBM's Identity Risk and Investigation Solution (IRIS) [1] enable grouping peers as a means of defining normal network behaviors. The assumption is that, for example, an engineer could be expected to behave more like other engineers than administrators on the network.

Another approach to grouping similar users is to employ observed behavioral similarities. This approach is used by banks to detect financial fraud [7]. Based on selected measures of activity, user-data sets can be clustered. Using this strategy should result in detection thresholds broader than those obtained with single user profiles, yet tighter than bounds obtained through role-based grouping.

We examined the utility of using organizational roles in defining the bounds of normal network traffic, as measured using Netflow. This was performed by collecting five weeks of Netflow records from a /21 subnet on a small campus, associating flow records with the roles of users on systems generating the traffic, and generating vectors of Netflow-derived features representing user activities over set periods. The role-labeled feature vectors were used to train a linear (Nearest Centroid) classifier. Data sets were then re-labeled, using K-means++ to cluster users based on centroid vectors (vectors of mean-feature values). The classifier was then trained to differentiate data sets labeled by cluster number. For each strategy (role-based or clustering) used to group user data sets an additional user group was created by drawing user data sets proportionally from each user group. This pseudo-user group provided a contrast in classification performance as compared to role or cluster-defined groups.

We also computed pairwise distances between user centroid vectors. For each user grouping strategy we plotted intra-group and inter-group distance distributions, as well as distance distributions comparing user activities over time. Through these experiments we determined that user roles were not closely coupled with user network behaviors, and grouping users based on behavioral similarities produces tighter ranges of measured behaviors as compared to role-based grouping.

II. Related Work

Improving behavioral anomaly detection through role-based grouping of user profiles has been widely examined and implemented. Legg et al. [9] extracted features from log data, creating user profiles that were compared to role-based groups. Feature sets were normalized and projected into a two-dimensional space using PCA decomposition. Point distances from the space origin defined an anomaly score, with high values flagged.

The Identity Risk and Investigation Solution (IRIS) created by IBM defined normal activities based on peer behaviors [1]. Features used for detection include access frequencies for applications, access intervals, and login times, among others. Feature values for each user were compared against peer group value distributions to determine if the values were anomalous.

Mathew et al. [10] tested different classifiers (Support Vector Machines, Naive Bayes, K-means clustering, Decision Trees) on statistical summaries of SQL query results. Results were labeled by the role (Faculty, Staff, Framework, Chair) of the user that made the query. Classifiers were trained one-on-one (Chair vs. Faculty, etc.). They found the best performance (91% - 100% detection) using the K-means classifier.

Users can also be grouped based on observed behavioral similarities. Frias-Martinez [4]–[6] defined a network access-control system in which user groups were defined by clustering (via K-means++) flow-based feature vectors, to identify similar behaviors. To detect anomalies, new feature vectors were compared against previously defined clusters. If too dissimilar, the new vector is declared anomalous. Feature vectors were based on port usage, including values such as total flows, average bytes per flow, average milliseconds per flow, total packets, average packets per flow, among others.

Dean [3] extensively compared role-based versus behavior-based approaches to grouping user data sets. This paper presents a subset of the analyses performed in [3].

III. Data Sets

To generate our Netflow data set, we collected five weeks of network traffic from an IPv4 /21 subnet at the Naval Postgraduate School (NPS). Large components of the collected flow records had to be removed, based on their lack of utility in assessing user behavior patterns. During two days of the collection period, network managers performed a port scan across all systems on the network, greatly increasing the flow counts recorded. In addition, over a three week period a significant fraction of the observed traffic was generated by a number of computers making frequent but unsuccessful attempts to connect over port 5223 (Apple Push Notification) to Apple servers. We also removed any traffic transiting the router without an endpoint on the /21 network. Finally, we dropped all DHCP and NTP flows, due to lower relevance to user network activities. The captured data was converted into Netflow v5 records using the SiLK [2] tool suite.

A. User Role Assignment

With the SafeConnect [8] network access database in use at NPS, we were able to associate much of the collected flow data with individual user names. User names were then correlated with each user's respective role and department, by matching them with data extracted from the campus Lightweight Directory Access Protocol (LDAP) server. After user roles were determined, numbers were used to represent user names in order to anonymize the data. Users were grouped into three categories (Faculty, Staff. Student), with each category composed of subgroups (see Table I). The number of users per role group varied widely, leading to the generation of unbalanced data classes.

Table I
Counts of the Role Groups in Data Set

Categories	Role Groups	Count
Staff	Administration	29
	Admin	30
	Class management	8
	Funding/acquisition	12
	IT support	35
Faculty	Lecturer	42
	Research Assistant	84
	Tenure	151
Student	Distance Learning Student	16
	Masters	954
	PhD Student	12

While the roles of individuals were straightforward to identify, there was a great deal of overlap between the different role group categories. Tenured faculty members often performed research and lectured. PhD and Masters students often shared the same classes. The same problem exists in many organizations. Engineers may perform management-like duties for contracted technical personnel, including the tracking and reporting of project status. Although some roles may be distinctive based on access requirements to resources (e.g. SSH connections to key servers), most roles we observed showed tremendous overlap in terms of tasks shared.

B. Group Refactoring

Because data set sizes for the role-based groups were highly unbalanced, the feature vectors for the two largest classes (Masters Student and Tenured Faculty) were downsized (85% and 50 % respectively) through random selection and removal.

IV. Feature Vector Development

To determine what port-activity features to derive from the Netflow data for our analysis, we evaluated port usage for each selected role group. As we wanted to focus on flows generated by user activities, we tried to remove flow records resulting from automated (non-user initiated) processes.

A. Automatic Flows Detection

Our process for identifying automated flows was based on patterns we found to be closely associated with non-user generated traffic. These patterns were:

- Repeated bi-directional flows (shared server port, protocol, outgoing bytes, packets, flags and incoming bytes, packets, flags) between two systems
- Repeated intervals between flow start times (nearest second, for intervals > 2 seconds), for flows between two systems over same server port and protocol
- Repeated web page loads, with approximately the same interval between page load start times.

Repeated bi-directional flows, intervals between flow starts and web-page reloads were identified based on the instances exhibiting outlier counts. We defined outlier counts in the same way outliers are defined for Tukey [12] boxplots, where the inter-quartile range in a set of count values (C) is $IQR = 3^{rd}\ quartile\ value - 1^{st}\ quartile\ value$. The outlier counts we identified were on the high end of the count range, i.e. $C_o = \{c_j \geq 3^{rd}\ quartile\ value + 1.5 * IQR \mid c_j \in C\}$.

For the first two patterns, flow sets between two systems (e.g. a user client and distant server) sharing a common server port and protocol were tested. Flow records were deemed automatic if counts of bidirectional flows (or flow intervals) were outliers compared to other flow pairs/intervals observed in the flow data set. For flows tagged as automatic due to outlier interval values, flows equally spaced in time were flagged except for the initial flow, based on the assumption that a user action initiated the first flow of the series.

Identifying repeated web page loads as automatic activity was a decision we made due to the large number of flows spawned in loading many web pages. A web browser left open on a site that automatically refreshes content can add a significant number of flow records to a user's profile. Conversely, users leaving browsers open might be a behavior an organization may wish to include in comparing user behaviors.

Web page loads were identified as flow sets where:

- Intervals between flow starts are < 4 seconds (flow burst)
- More than 90% of bytes transferred in the set were via ports 80 or 443 (HTTP or HTTPS)
- Total flow count in a set, excluding DNS and no-payload TCP packets, was > 20

Interval values (I) between page-load start times were rounded off, as reloads did not recur with second-level precision. Longer intervals were rounded using a factor $d = 10 * \lfloor (It_{delta} + 5)/10 \rfloor$, itself rounded to the nearest 10 seconds. Intervals were rounded to the nearest multiple of d, i.e. the intervals counted were $I' = d * \lfloor (I + 0.5d)/d \rfloor$. For our analyses, we used a $t_{delta} = 0.08$. For each user system the rounded intervals between similar web page loads were counted, and if the count for one interval value was an outlier compared to other intervals, repeated web page loads in sequences of three or more were flagged.

Because web pages like CNN and Fox News performed automatic reloads when part of the content (stories, advertisers, graphics) changed, identifying repeated loads from the same web site required some measurement of similarity. Similarity of web-page loads was determined by comparing:

- Total flows (less than 25% difference in flow counts)
- The distributions of byte transfers associated with server ports and with server IP addresses in the two data sets

If two page loads were approximately the same size in flow counts, similarity between two page-load flow sets (F_1, F_2) was measured based on bytes passed per IP address connected to and bytes passed per server port used. For a distant IP address a_i in flow set F_x,we define bytes passed to/from a_i as $b(F_x[a_i])$. We define a distance measure between F_1 and F_2 in terms of bytes passed per distant IP address to be $d_{ip} = ((\sum_{i=1}^{m}(\frac{b(F_1[a_i])}{m_{ip}} - \frac{b(F_2[a_i])}{m_{ip}})^2)^{1/2})/m$, where $m_{ip} = max(b(F_1[a_i]), b(F_2[a_i]))$. We also define the bytes

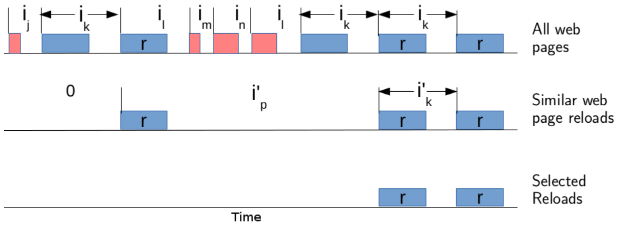

Figure 1. Web Page Reload Identification

passed to/from distant servers over port p_j as $b(F_x[p_j])$, and a corresponding Euclidean distance as $d_p = ((\sum_{j=1}^{n}(\frac{b(F_1[p_j])}{m_p} - \frac{b(F_2[p_j])}{m_p})^2)^{1/2})/n$. Using both measures, we define a total distance between the two flow sets as $d = (d_{ip} + d_p)/2$. Flow sets were deemed similar if d fell below a threshold, or $d < \theta_d$. We used a θ_d =0.9 for our analysis.

Figure 1 depicts the process of identifying repeating web page loads. Repeating sets of page-load flows are labeled with "r" in Figure 1 . The intervals between web page loads are captured, with one interval value (i_k) occurring most frequently. The web page loads with start times separated by the identified interval are selected, tested for similarity, and repeated sequences (three or more) of similar web-page loads were annotated as automatic after the first load.

B. Feature Definitions

Based on these algorithms, flows within user data sets were evaluated and automated flows flagged. Flow sets minus the flagged records were designated as "cleaned", both cleaned and non-cleaned data sets were tested to determine if cleaning enhanced classifier performance. The ports used for each user group were ranked by total flow counts, based on the cleaned flow records generated. Table II shows the ports we selected as potentially useful in characterizing user network activities.

Port	Protocol	Often used for:
22	TCP	Secure shell
80	TCP	HTTP
88	TCP	Kerberos
137	UDP	NETBIOS Name Service
138	UDP	NETBIOS Datagram Service
389	UDP	LDAP
443	TCP	HTTPS
445	TCP	Microsoft Directory Services (SMB)
5222	TCP	Jabber/GoogleTalk Client Connection
5353	UDP	Multicast DNS
8080	TCP	HTTP-alt
8055	TCP	Senomix Timesheets Server
9443	TCP	VMware HTTPS, SSL
60000	TCP	MS Exchange RPC Client Access Service
60001	TCP	MS Exchange Address Book

Table II
SELECTED PORTS AND PROTOCOLS FOR FEATURES

For each port X listed in Table II, we defined the following features:

- Total inbound bytes for a user system over port X, divided by total bytes passed during a sampling interval

Feature Name	Type	Description
port_X_in	Port Behavior	Total port X bytes inbound/total bytes all ports
port_X_out		Total port X bytes outbound/total bytes all ports
port_X_std		Port X standard deviation/ mean of byte values
port_entropy		entropy of distant ports
bytes_out	Volume & Flow Density	Total bytes outbound/total bytes passed
packets_out		Total packets outbound/total packets passed
bpp		Average bytes per packet
tcp_frac	Protocol	TCP fraction of total flows
udp_frac		UDP fraction of total flows
igmp_frac		IGMP fraction of total flows
multicast		Fraction of multicast IP address flows (224.0.0.0/4)
flag_entropy	Handshaking	Entropy of TCP flag counts
duration_std	Temporal Behavior	Standard deviation of flow duration values
interval_mean		Average interval between flow start times
interval_std		Standard deviation of flow start time intervals
ip_distance_std	Address Related Features	Standard deviation of src/dst IP address distance/2^{32}
ip_distance_mean		Mean of src/dst IP address distance/2^{32}
addr_entropy		Entropy of the IP addresses connected to
direction		Fraction of flows outgoing

Table III
STATISTICAL AND INFORMATION-THEORY-DERIVED FEATURES

- Total outbound bytes for a user system over port X, divided by total bytes passed during a sampling interval
- The coefficient of variation (mean/standard deviation, or μ/σ) for the byte values observed passed over port X (inbound and outbound), during a set sampling period

The entropy of server port-protocol flow counts observed during a set sampling period was also determined and added as a behavioral feature. Given the flow counts of n server port, protocol combinations observed in a data set C, the normalized entropy of the port-protocol combinations can be computed as $H(X) = -\sum_{c_i \in C} p(c_i)log(p(c_i))/log(n)$.

Besides the port behavior features, we identified features relating to flow volume and density, protocols, TCP handshaking, temporal behaviors and IP address metrics (Table III).

The flow volume and density features were selected to characterize traffic flows relative to direction (fraction of bytes outbound, fraction of packets outbound) and packet sizes (bytes per packet). The protocol features reflect protocol usage (fraction of TCP/UDP/IGMP/multicast flows), while the entropy of TCP flag occurrences is a TCP handshaking feature. Temporal behaviors refer to measurements of flow durations (standard deviation) and intervals between flow starts (mean and standard deviation). The address related features relate to distances between source and destination IP addresses (expressed as 32 bit numbers), normalized entropy of IP addresses in the data set, and the fraction of flows outbound. Combined, the different feature groups add up to 61 features. For our experiments, we applied PCA to reduce the number of dimensions to 31 (retaining 95% of data-set variance).

C. *Data Slicing Intervals*

One factor that can affect the analysis of network behaviors is the decision criteria for sampling network-flow data. Longer sampling intervals (e.g. one day) provide gross measurements of overall activities, where measurements on activities of potential interest can be averaged out and lost. Short-duration sampling intervals (e.g. 15 minutes) can be used to characterize activities at the task level, but flow bursts associated with specific tasks are more likely to be split between different sampling periods and feature measures will show greater variability (i.e. appear noisy) as users switch between tasks.

To observe the effects of different data sampling intervals on the classifier's ability to differentiate user-group data sets, we sampled each user's Netflow records using four intervals. Expressed in minutes, the intervals chosen were 15, 30, 60 and 1440 (one day). For each user IP address and interval, records of flows to and from the system were aggregated. Features of flows (bytes, packets, duration) extending across intervals were segmented proportionally, based on the fraction of the flows within each inclusive interval. After flow set segmentation based on the interval used value vectors of the chosen features were created, each labeled with the user ID number and role subgroup associated with the flow data.

V. USER GROUPING

In addition to grouping user-data sets based on roles, we also grouped users by clustering data sets. For each set of user feature vectors (cleaned/not cleaned, 15, 30, 60, and 1440 minute intervals), the vectors were aggregated by the week the traffic occurred. For each week, centroid vectors (vectors of mean values for each feature) were created and clustered using K-means++. For our experiments we set k = 11, to create the same number of groups as those created based on user roles. Feature-vector data sets were labeled based on the cluster ID number of their respective users.

A. *Group-Neutral Test Sets*

One approach to measure the strength of a data-class definition strategy is to create a hybrid class, composed of data drawn proportionally from the other classes. If the original classes contained data elements enabling differentiation between classes, a multi-class classifier should perform poorly on the hybrid class relative to the others. This provides a reference point, demonstrating the effectiveness of a class definition strategy in grouping similar things together.

For each grouping strategy evaluated (role-based, user-clustering) and feature-vector data set, we created a hybrid class by extracting user-data sets proportionally from the defined groups. From each defined group user-data sets were randomly selected, such that between 13 and 17 percent of the total vectors for that group were identified. These vectors were extracted and incorporated into a hybrid class.

VI. Experiments

Two different methodologies were tested to capture the differences between the user grouping strategies, in terms of supporting the bounding of behaviors for anomaly detection. In the first approach, a Nearest Centroid classifier was used to differentiate between the role-based and hybrid-class data sets, for each combination of flow-data cleaning (clean vs. not cleaned) and sampling interval. The experiment was repeated after clustering user centroid vectors for each week of data, and using cluster numbers as user group labels.

In the second approach we determined pairwise Euclidean distances between centroid vectors, each representing a week of network traffic for one user. For each user in group G, pairwise distance distributions were computed for individual users (comparing across weeks of activity), between users within group G (intra-group distances), and between users in group G and the other user groups (inter-group distances). For clustering-based user groups the inter- and intra-group distances were determined for each week.

For each experiment the number of clusters (k) was set to the number of user roles, to enable more direct comparisons with the role-group results; the value of k can be varied to tune group sizes and minimize point-to-point distances within clusters.

A. Classifier Testing

For each combination of sampling interval and flow filtering (clean/unclean) status, the feature-vector data sets for each user were used to train and test a Nearest Centroid classifier. To perform multi-class classification, the classifier was trained using a "one versus the rest" approach. The training/testing data split used was 70:30 for each experiment.

Training and testing was performed twice; once for feature vectors labeled based on user roles and again using feature vectors labeled based on the clustering-defined user groups.

B. Self-Similarity Analysis

To determine whether clustering-defined user groups exhibit greater self-similarity in measured network behaviors relative to role-based groups, we performed pairwise distance measurements between user centroid-vector data sets for each grouping strategy. Individual variations in behavior (self-similarity) were measured by determining pairwise distances between user centroid vectors for each week of observed activity. Intra-group behavioral differences were measured per week, by determining the pairwise distances between each user's centroid vector and the centroid vectors of other users within the group.

VII. Results

Figure 2 shows the precision and recall scores achieved by the Nearest Centroid classifier on the role-based labeled data sets. Role groups on the X axis are listed in order of decreasing data-set sizes. Because of the proportional data extraction used to create the pseudo group, its size was between that of the Research Assistant and Tenured Faculty role groups.

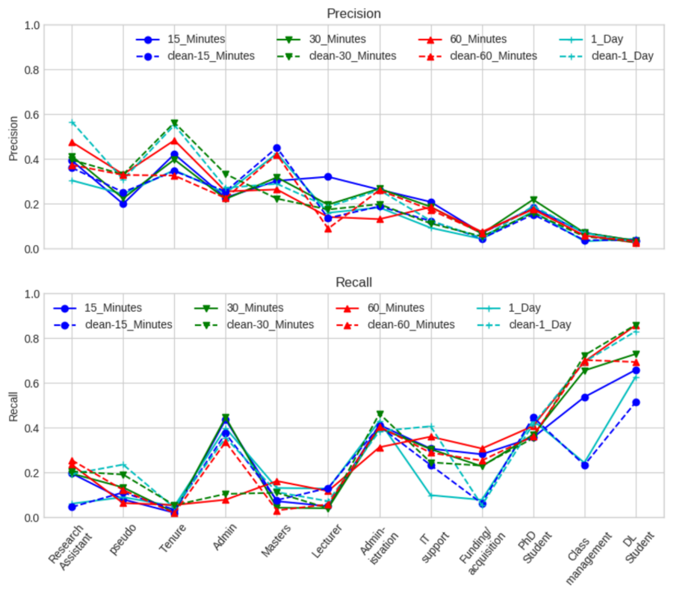

Figure 2. Classifier Precision and Recall on Role-Based User Groups

As can be seen, the precision scores follow an almost linear pattern in relation to the decreasing data set sizes, and the pseudo-role group did not score significantly worse than comparably sized role groups. This pattern was evident across all feature vector variants tested. These results indicate a weak relationship between the roles used to group user data sets and the feature-vector data sets associated with each role.

Figure 3 shows precision and recall scores achieved for data sets labeled via user data-set clustering. Only the results relating to feature vectors derived from the non-filtered (unclean) flow records are shown to enhance readability. Each feature-vector variant clustered differently, resulting in no consistency in precision and recall scores across the cluster labels. Two patterns are notable; the feature vectors derived from one-day sampling intervals showed more consistent precision and recall scores across the cluster labels, and the pseudo-group precision and recall scores were in most cases substantially lower than what was observed for the cluster-defined user groups.

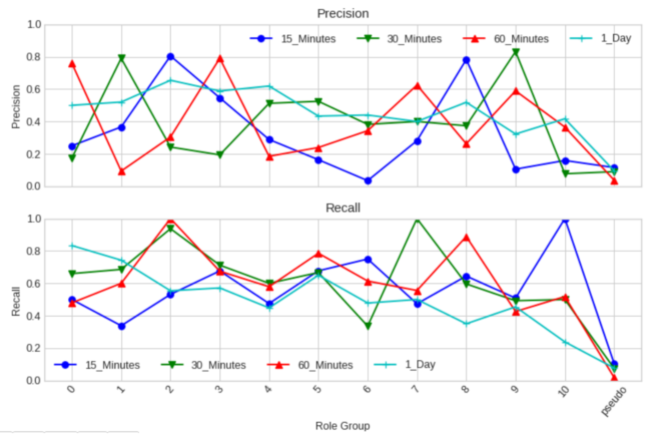

Figure 3. Classifier Precision and Recall on Clustered User Groups

Based on these observations we can infer that the cluster-defined user groups exhibited more self-similarity than found in the role-based user groups.

The intra- and inter-group distance comparisons were also noticeably different for the role-based vs. cluster-based data sets (Figures 4 and 5). While the intra-group distance distributions for role-based groups were generally on a par with the inter-group distributions, intra-group distances for the cluster-based user groups were generally either visibly lower than the intra-group ranges or about the same as the lowest inter-group range. The figures show the ranges obtained by comparing centroid vectors derived from the one-day sampling interval data sets. Similar results were obtained with the other sampling intervals and data cleaning state.

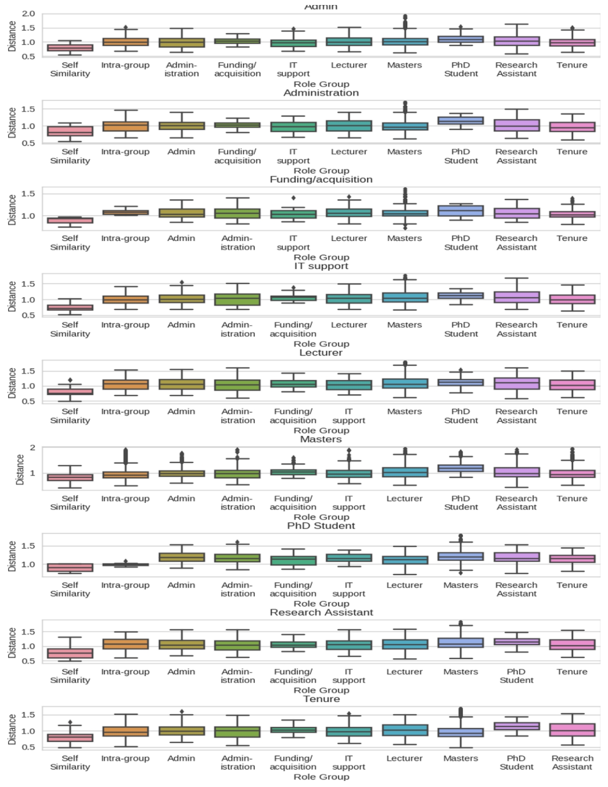

Figure 4. Pairwise Distances for Role-Based User Groups

Based on these experiments we conclude that clustering user-data sets is a better strategy for creating groups of users with similar behaviors than using role-based labels. By minimizing intra-group distances between feature vectors, tighter bounds on user behaviors can be set. With cluster-size tuning through varying k, it should also be possible to optimize anomaly detection thresholds.

Acknowledgment

I'd like to give my deep thanks to my wife as she endured my long hours and diverted attention while documenting this research. I'd also like the thank Dr. Neil Rowe, who provided much needed guidance and patience while I completed my dissertation.

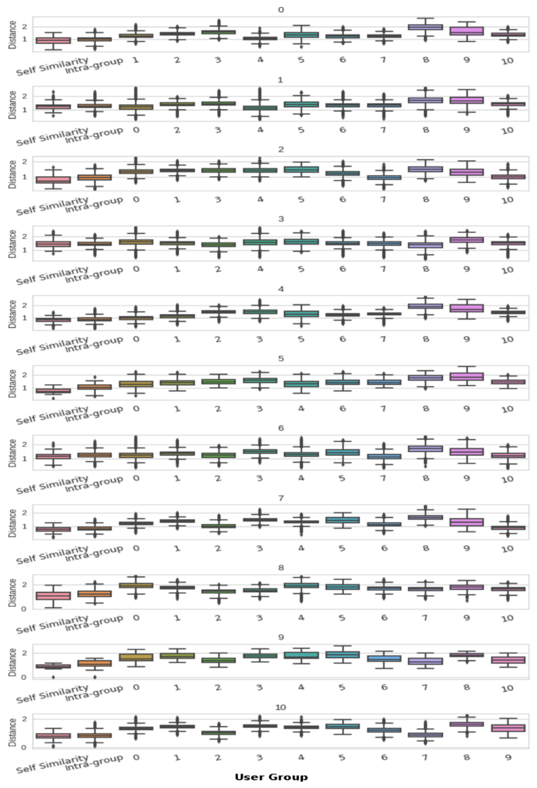

Figure 5. Pairwise Distances for Behavior-Based User Groups

References

[1] G. F. Anderson, D. A. Selby, and M. Ramsey. Insider attack and real-time data mining of user behavior. *IBM Journal of Research and Development*, 51:465–475, May 2007.

[2] CERT/NetSA at Carnegie Mellon University. SiLK (System for Internet-Level Knowledge). [Online]. Available: http://tools.netsa.cert.org/silk. [Accessed: July 13, 2011].

[3] Jeffrey S. Dean. *Systematic assessment of the impact of user roles on network flow patterns*. PhD thesis, Naval Postgraduate School, 2017.

[4] V. Frias-Martinez, J. Sherrick, S.J. Stolfo, and A.D. Keromytis. A network access control mechanism based on behavior profiles. In *Computer Security Applications Conference, 2009. ACSAC'09. Annual*, pages 3–12. IEEE, 2009.

[5] V. Frias-Martinez, S.J. Stolfo, and A.D. Keromytis. Behavior-profile clustering for false alert reduction in anomaly detection sensors. In *Computer Security Applications Conference, 2008. ACSAC 2008. Annual*, pages 367–376. IEEE, 2008.

[6] Vanessa Frias-Martinez. *Behavior-based Admission and Access Control for Network Security*. PhD thesis, 2008. AAI3333480.

[7] David J Hand and David J Weston. Statistical techniques for fraud detection, prevention, and assessment. *Mining massive data sets for security*, pages 257–270, 2008.

[8] Impulse Point. SafeConnect: Network Access Control. [Online]. Available: http://www.impulse.com/. [Accessed: July 23, 2016].

[9] Philip A Legg, Oliver Buckley, Michael Goldsmith, and Sadie Creese. Automated insider threat detection system using user and role-based profile assessment. *IEEE Systems Journal*, 11(2):503–512, 2017.

[10] S. Mathew, M. Petropoulos, H. Ngo, and S. Upadhyaya. A data-centric approach to insider attack detection in database systems. In *Recent Advances in Intrusion Detection*, pages 382–401. Springer, 2010.

[11] Karen Scarfone and Peter Mell. Guide to intrusion detection and prevention systems (IDPS). *NIST Special Publication*, 800(2007):94, 2007.

[12] John W Tukey. Box-and-whisker plots. *Exploratory Data Analysis*, pages 39–43, 1977.