



LAWRENCE
LIVERMORE
NATIONAL
LABORATORY

LLNL-TR-410901

IP Profiling via Service Cluster Membership Vectors

A. Bartoletti

February 27, 2009

Disclaimer

This document was prepared as an account of work sponsored by an agency of the United States government. Neither the United States government nor Lawrence Livermore National Security, LLC, nor any of their employees makes any warranty, expressed or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States government or Lawrence Livermore National Security, LLC. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States government or Lawrence Livermore National Security, LLC, and shall not be used for advertising or product endorsement purposes.

Auspices Statement

This work performed under the auspices of the U.S. Department of Energy by Lawrence Livermore National Laboratory under Contract DE-AC52-07NA27344. This work was funded by the Laboratory Directed Research and Development Program at LLNL under project tracking code 08-FS-007.

FY08 LDRD Final Report
IP Profiling via Service Cluster Membership Vectors
LDRD Project Tracking Code: 08-FS-007
Tony Bartoletti, Principal Investigator

Abstract

This study investigates the feasibility of establishing and maintaining a system of compact IP behavioral profiles as a robust means of computer anomaly definition and detection. These profiles are based upon the degree to which a system's (IP's) network traffic is distributed among stable characteristic clusters derived of the aggregate session traffic generated by each of the major network services. In short, an IP's profile represents its degree of membership in these derived service clusters. The goal is to quantify and rank behaviors that are outside of the statistical norm for the services in question, or present significant deviation from profile for individual client IPs. Herein, we establish stable clusters for accessible features of common session traffic, migrate these clusters over time, define IP behavior profiles with respect to these clusters, migrate individual IP profiles over time, and demonstrate the detection of IP behavioral changes in terms of deviation from profile.

Introduction/Background

Contemporary means of computer network intrusion detection rely upon specific "signatures" (traffic in known bad-content, connections to known bad destinations, use of unknown services on unusual ports) or make other brittle assumptions regarding expected bad behavior. Adversaries have adapted to these detection means by encrypting content, employing large proxy servers as destinations (rendering most all destinations both good and bad in order to frustrate IP-blocking) and have adapted to the modern network security practice of disallowing services on unusual ports by "tunneling" malicious command traffic and data exfiltration within the standard service protocols, and targeting attacks to the application layer.

For these reasons, anomaly methods have become increasingly critical for detecting compromised systems and related abuse such as illicit data-exfiltration in the absence of prior knowledge of specific attacks.

Anomaly detection methods place an additional burden upon the adversary, who must now appear both syntactically correct and statistically normal. By defining IP profiles in terms orthogonal to the range of characteristic service clusters, a uniform means to broad and generalized anomaly detection is supported.

An additional motivator for the proposed method is the relative ubiquity of network flow summary data in large enterprise network monitoring, and in support of feature generation. In contrast to signatures derived over specifics of session content, which quickly become stale as network content evolves, fundamental measures such as per-service session packet and byte volumes, durations and invocation times are expected to remain meaningful over time. While clearly less informative than full packet capture in detection of known threat content, these session summary

features provide a straightforward and durable foundation for statistical treatment of traffic behaviors.

This work is an extension of a study conducted by the author and a colleague, where cluster set distance is derived and employed to distinguish service traffic, especially when services may be set up illicitly on unusual ports[1].

Available Data

The data subject to this study is "session connection summary" data, representative of a large corporate enterprise. Each event record summarizes a network connection in terms of endpoints (initiating and responding IP addresses), time of connection, session duration, protocol and service, total bytes sent, total bytes received, total packets sent and total packets received. Although minimally informative (no packet payload content is investigated) this form of summary is commonly produced for gigabit-per-second traffic due to its brevity. It is reasonable to assume that results might improve for those cases where greater detail is available for profile consideration.

Such session summary data is usually captured at a site's outer boundary router, a common traffic "choke point" minimizing the need to manage multiple sensors. Such sensor position has a serious drawback when outbound client requests are internally "proxied" – the sensor sees only a handful of proxy systems issuing thousands of session requests, rather than the actual thousands of IPs issuing sessions on their own.

We limit our interest to profiling the behavior of internal systems, with respect to boundary-crossing activities they have initiated. Hence, we filter the traffic to sessions that have been initiated in the outbound direction and evidence bi-directional data flow. In this regard, we limit ourselves to protocol 6 (TCP) and protocol 17 (UDP) traffic, and for TCP insist that the flow record indicate the "full session handshake" flags are present. This helps to ensure we are not double-counting sessions.

The flow records are provided in files containing 15 minutes of boundary-crossing traffic, 96 files per day, each file about 50 MB uncompressed and representing over 100,000 session (flow) records. In the daily accumulations of 96 15-minute files, filtered to TCP and UDP protocol and two-way data flow, the following protocol-port combinations are seen to represent nearly all of the traffic:

Service Description	Protocol-Port	Sessions/Day	Percent	Proxied?
HTTP (web) traffic	6-80	3,500,000	61.9	Yes
DNS (Domain Name Service)	17-53	1,400,000	24.8	Yes
HTTPS (Secure web)	6-443	560,000	9.9	Yes
NTP (Network Time Protocol)	17-123	130,000	2.3	No
SMTP (Simple Mail Transfer Protocol)	6-25	30,000	0.5	Yes
SSH (Secure Shell)	6-22	28,000	0.5	No
FTP (File Transfer Protocol)	6-21	5,000	0.1	No
TOTAL		5,653,000	100.0	

Figure 1 – Daily Session Counts by Service (average over 4 consecutive workdays)

Any uniform profiling of session activity will need to be made in terms of these services. Unfortunately, it turned out that several of these services were "proxied" - hundreds of individual IPs conducting (for instance) HTTP sessions have their requests funneled through a handful of systems that conduct the outbound sessions on behalf of the actual client IPs. The border sensor sees only the proxies contributing to highly-aggregated behavior in such cases. To further complicate matters, there are hours or days when the proxies fail, and then the number of IPs seen conducting HTTP sessions will jump from (eg) 10 to over 1000.

The following table (figure 2) reveals the number of internal IPs seen issuing border-crossing sessions on the major services. Where non-proxied, these are typically "desktop" systems.

Protocol-Port	Approximate Unique IPs Active per Day (at boundary)	Proxied?
6-80	10/1900 (proxied/non-proxied)	Yes (inconsistent)
17-53	10	Yes
6-443	10/400 (proxied/non-proxied)	Yes (inconsistent)
17-123	1300	No
6-25	3	Yes
6-22	240	No
6-21	200	No

Figure 2 – Daily IP Counts by Service

Based upon this data assessment, we chose to conduct IP profiling on all 7 indicated protocol-port services, but for ports 80 and 443 (web traffic) limiting ourselves to only those days when proxies were in effect, as these days far outnumbered the occasions of proxy-failure. This means our ability to profile the "end user desktop" IPs is limited to SSH (6-22), FTP (6-21) and NTP (17-123) service traffic.

Producing Feature Vectors

In preparing the data for service clustering, we transform each session flow record into a characteristic "feature vector" presenting sufficient component range and variation for cluster support. In this regard, we have chosen to produce feature vectors based upon the following 6 components:

(Proximity-to-midnight,duration,bytes-sent,bytes-received,packets-sent,packets-received)

We partition these derived records according to protocol-port, and separately retain the identity of the initiating IP for each record. Such elements will not contribute to the clustering calculations.

The justification for tracking both byte and packet counts is that services often display characteristic packet sizes, or bytes per packet, and unusual deviations in these numbers can be evidence of abuse. We transform the session timestamp to "proximity to midnight" (PTM), in order that very late evening activity (11 PM) and very early morning activity (1 AM) not be artificially and widely separated. Scaling this measure to the range [0.0,1.0] means that sessions occurring during regular work hours are assigned PTM values above 0.5, while off-hour activities will have PTM < 0.5.

Preliminary assessment of the data revealed all features other than PTM to be highly skewed toward zero, with a relatively sparse population of large outliers. In order to better condition upon fine structure in the dense region near zero, these skewed components were transformed as follows:

$$x' = 0 \text{ if } x = 0$$

$$x' = 0.05 + 0.95 * (\ln(x) - \min(\ln(x)) / (\max(\ln(x)) - \min(\ln(x)))) \text{ otherwise.}$$

The before and after result of these logarithmic normalizations is depicted in figure 3, where the white background is used to indicate the presence of data too small to produce visible histogram columns. One can clearly see structure amenable to clustering in the log-normalized representations.

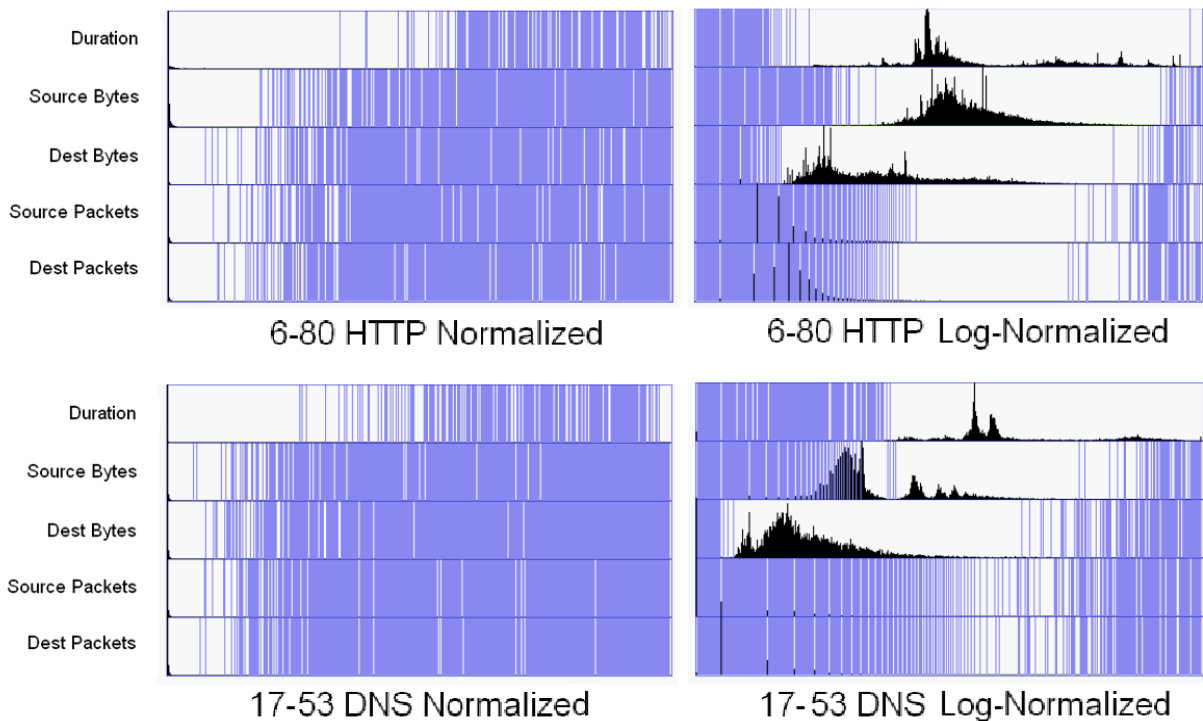


Figure 3 – Feature Vector Component Distributions

Data Clustering and Stability

Data clustering exists to take points from a continuous (or discrete) and often high-dimensional space, and assign them categorically to a lesser collection of groups. This is desired foremost as a means to quickly understand the structure (if any) present in the data - which values of which features tend to occur together, which "quadrants" of the space appear empty, etc. Put most simply, clustering algorithms seek to identify regions of high density bounded or separated by regions of relatively lower density, and then to assign each point to the "region to which it belongs", accordingly.

Clustering algorithms can be broadly classed into "top-down" methods (generally involving k-means and repeated bisections) and bottom-up methods (various agglomerative techniques), all of which have variants according to the distance

functions and similarity criteria one chooses to apply. Commonly, these algorithms expect "desired number of clusters" to be supplied as an input parameter, and then they divide (or agglomerate) until the desired number of clusters is obtained. This can be frustrating, when the efficacy must be tested over a variety of clustering solutions.

A range of techniques have been developed that attempt to identify the "correct" number of data clusters evident in a data set. This will always be a bit of a black art, as one can always create pathological data sets for which any fixed cluster assessment logic will fail to arrive at a consistent or "reasonable" number. Consider for example figure 4, where one might argue either that 3, or 7, is a better measure of the number of clusters presented.

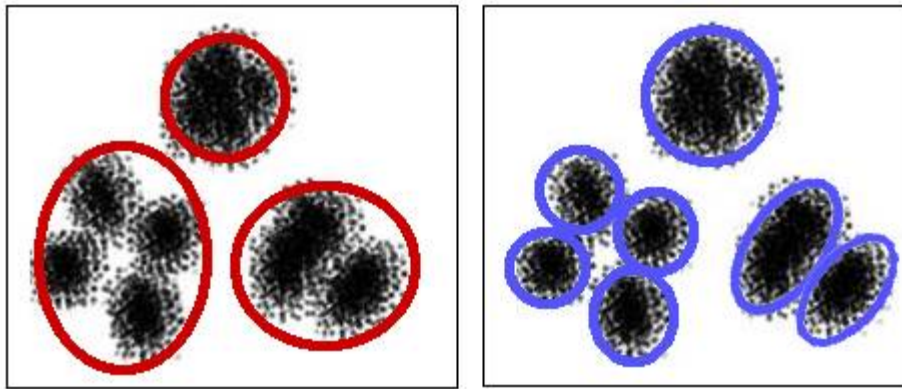


Figure 4 – Cluster Count Ambiguity

We have chosen to employ a stability-based cluster number determination, inspired by the work of Heer and Chi [2]. Stability-based methods proceed by repeatedly taking many random (say) 80% samplings of the data points and applying a fixed k-way clustering to each sample. Each pair of samples will have about 64% of the base population data in their mutual intersection, and we consider whether pairs of points taken from the intersection do, or do not appear consistently in a common cluster under both sample cluster solutions. A high fraction of point-pairs that are either members of a common cluster in both samples (or of different clusters in both samples) is taken as evidence that the samples presented a similar cluster structure (the clusters correspond to "true" and consistent regions of relative density, unaffected by the 80% sampling). The Rand statistic, Fowlkes-Mallows statistic, and Jaccard coefficient provide various normed similarity measures in comparing cluster solutions. By repeating the experiment many times (e.g. conducting a round-robin of similarity measures among the sample cluster solutions) one produces a PDF, and then CDF of the statistic in question, the particular curve indexed by "k", the number of requested clusters.

This method of testing for stability requires that each pair of sample datasets have most of their elements in common, owing to the nature of the Rand, Fowlkes-Mallows and Jaccard formulations as similarity measures (the Jaccard formulation requires a specific a-priori matching exist between the clusters of two k-way solutions). A small fraction, say 20% of the points, differ between the samples and can be considered as "noise" that seeks to obscure the clustering result. In the presence of such noise, the ability of a clustering procedure to cluster the "commonly held" points similarly is then seen as a testament to the stability of the solution, and points especially to having chosen a "correct" cluster count.

We considered the work of Meila[3] in promoting the notion of “Information-Variation” distance, specifically for assessing the similarity of competing clustering solutions. Every clustering induces a random variable X , representing the cluster (index) to which a randomly selected point has been assigned. Recall that the entropy of any random variable X is given by

$$H(X) = -\sum (P_X(i)\log(P_X(i))), \text{ where } i \text{ ranges over the possible values of } X,$$

and for any two random variables X and Y , one can define their mutual information by

$$I(X,Y) = \sum (P_{XY}(i,j)\log(P_{XY}(i,j)/P_X(i)P_Y(j))), \text{ } i \text{ and } j \text{ ranging over the values of } X \text{ and } Y, \text{ respectively.}$$

The mutual information between two random variables X , Y indicates the degree to which knowledge of the outcome of one provides information regarding the outcome of the other. If they are independent, then $I(X,Y) = 0$. One can then go on to define variation of information to represent the sum of information gained and lost, when assessing one variable as opposed to another. This sum is depicted in green in figure 5 below.

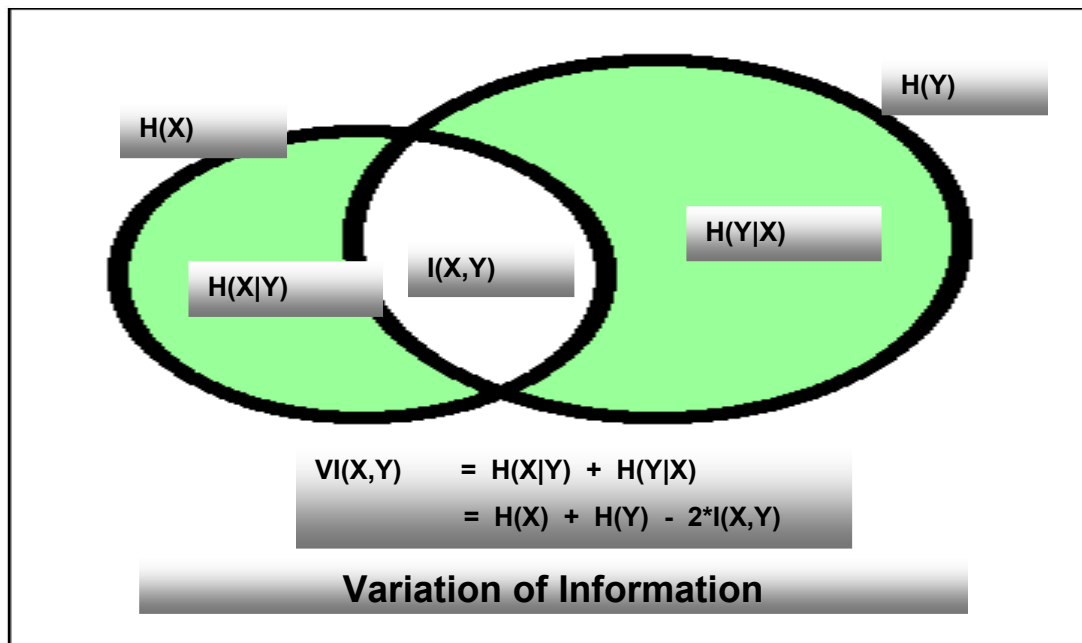


Figure 5 – VI-Distance

Applying VI-distance as a metric in the cluster-stability tests outlined by Heer and Chi provides a strong indicator of the “correct” number of clusters of a data set. We exercised this claim in a test consisting of 1000 points, arranged in the form of 6 distinct Gaussian clusters. Of this set, 20 independent random 80% selections were made, each clustered into k clusters, and a round-robin VI-distance comparison of the 190 pairs of solutions was made. The following graph of CDFs of the VI-distances for cluster counts $k=2$ through $k=15$ is obtained (see figure 6). Note how

the desired number of clusters (6) is distinguished as the first (rightmost) member of the regular curves on the left side of the graphic.

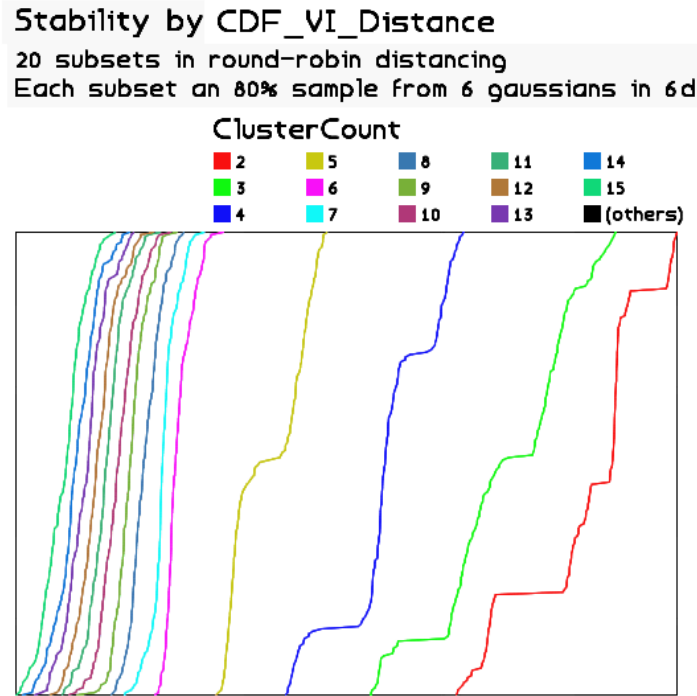


Figure 6 – Cluster Count Estimation by Stability of VI-Distance

A facility of VI-Distance is that it can be used to compare random variables with different structure, such as might be imposed by comparing clusterings with differing cluster counts. However, the resulting graph turned out to be far more difficult to interpret when the data was noisier or more ambiguous than the example with well-behaved Gaussians. Our desire to employ consistent clustering as a basis for characterizing the contributing generators (individual IP behaviors) requires a test of stability that will always be applied between pairs of clusterings of identical cluster count. From one day to the next, our datasets will have no points in common. Rather, they will have been produced by (approximately) the same generators.

Instead, we conducted our k-way clustering comparison by considering the centroids of the k clusters from each set, and matching them up so as to produce a minimal sum of vector distances. Explicitly, we **define a cluster set distance D** as follows.

If $\{a_1, a_2, \dots, a_k\}$ and $\{b_1, b_2, \dots, b_k\}$ are the centroids for clusterings A and B, we seek a set of k pairs $\{(a_i, b_j)\}$ with $1 \leq i \leq k$, $1 \leq j \leq k$, each value of i and j occurring exactly once, such that the sum of vector distances $D = \sum \text{dist}(a_i, b_j)$ is minimal. We take "dist" to be the ordinary Euclidean distance on points, although any desired distance function on components is allowed. We take this minimal D to be the distance between the centroid sets A and B.

Note that one cannot rely upon a greedy algorithm to obtain a minimum here. Consider $a_1=0$, $a_2=3$, $b_1=2$, $b_2=6$. The greedy algorithm would dictate using $\text{dist}(a_2,b_1) = 1$, forcing us to then use $\text{dist}(a_1,b_2) = 6$, for a sum of 7. In contrast, the minimal matching is obtained by $\text{dist}(a_1,b_1) + \text{dist}(a_2,b_2) = 2+3 = 5$. It might then seem one would have to measure all $k!$ permutations in matchings from $A \times B$ to determine the minimal matching. However the Hungarian Matching algorithm[4] can be employed to obtain the desired minimal pairing of distance-weighted edges in polynomial time (optimally $O(k^3)$). This involves a particular reduction procedure over the $K \times K$ matrix $[\text{dist}(a_i, b_j)]$.

We should justify calling D an actual "distance" function. Clearly, for any two sets of k centroids A and B , with $D = \min \sum \text{dist}(a_i, b_j)$, we will have $D \geq 0$, and $D = 0$ IFF the two sets of centroids are identical. It should also be clear that we must have $D(A,B) = D(B,A)$. To establish the triangle inequality, we must demonstrate that for sets A, B, C we must have $D(A,C) \leq D(A,B) + D(B,C)$. Let the centroids of each set be fixed in index. Then $D(A,B)$ is obtained by a particular permutation P_{AB} of indices, and $D(B,C)$ is obtained by another permutation P_{BC} of indices. Our definition demands that some permutation P_{AC} of indices give the minimal "distance" $D(A,C)$. Consider instead an alternate distance $D'(A,C)$ given by permutation $P'_{AC} = P_{BC} * P_{AB}$. This permutation induces a matching between sets A and C by composition of the individual mappings used to obtain $D(A,B)$ and $D(B,C)$. As the triangle inequality is thus satisfied component-wise, we must have $D'(A,C) \leq D(A,B) + D(B,C)$. As we must also have $D(A,C) \leq D'(A,C)$, the triangle inequality is satisfied by D .

Armed with this measure of distance between k -clusterings of disjoint sets, we can apply it as well to a stability-based method for cluster-count determination. Figure 7 demonstrates the result of this method, applied to the very same 6-gaussians data that was the basis of figure 6. Here, the suggested number of clusters can be found by seeking global (or local) minima, with 6 producing a clearly distinguished minima.

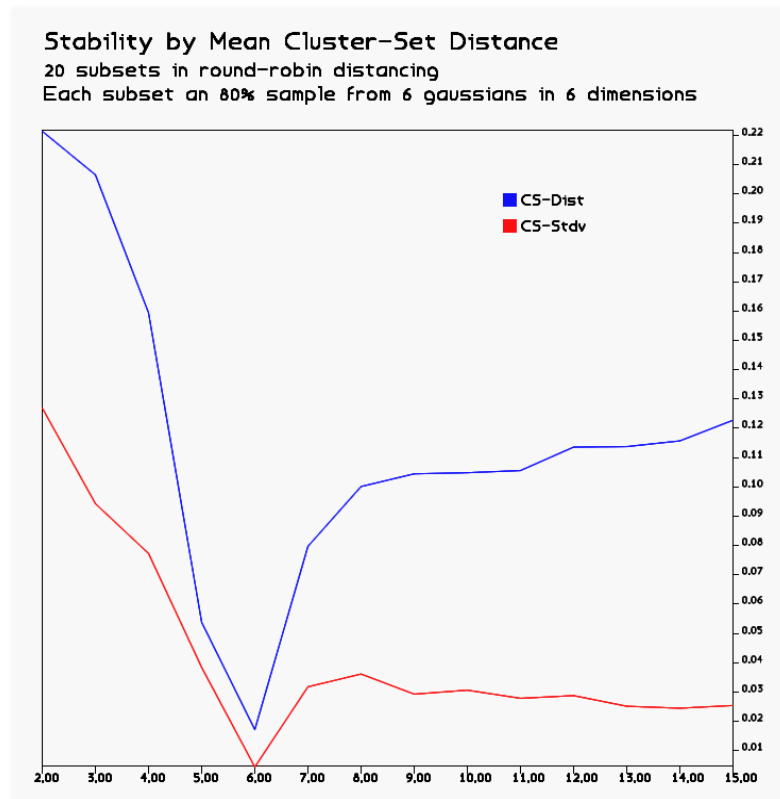


Figure 7 – Cluster Count Estimation by Mean Cluster-Set Distance

When we apply this method of cluster count determination to the actual session data, the results are not quite as unambiguous. However, they were sufficient to allow a reasonable selection of cluster count for the various services under study. This is exemplified by figure 8, for SMTP and DNS traffic respectively. For each service, the records for each of 4 separate days were sampled in 20 subsets at 80%, and a k-way clustering was applied. The minimal matching distance was determined for the 190 round-robin pairings of cluster solutions, and the mean distance and standard deviation recorded. The plot of these means, for $2 \leq k \leq 15$, provides a single blue line in each figure (red for stdv) and the entire process is repeated for 4 consecutive days. The heavy blue and red lines indicate the mean values for the 4 days. The evident consistency of the results suggests it is reasonable to assign an “optimal” cluster count employing this method.

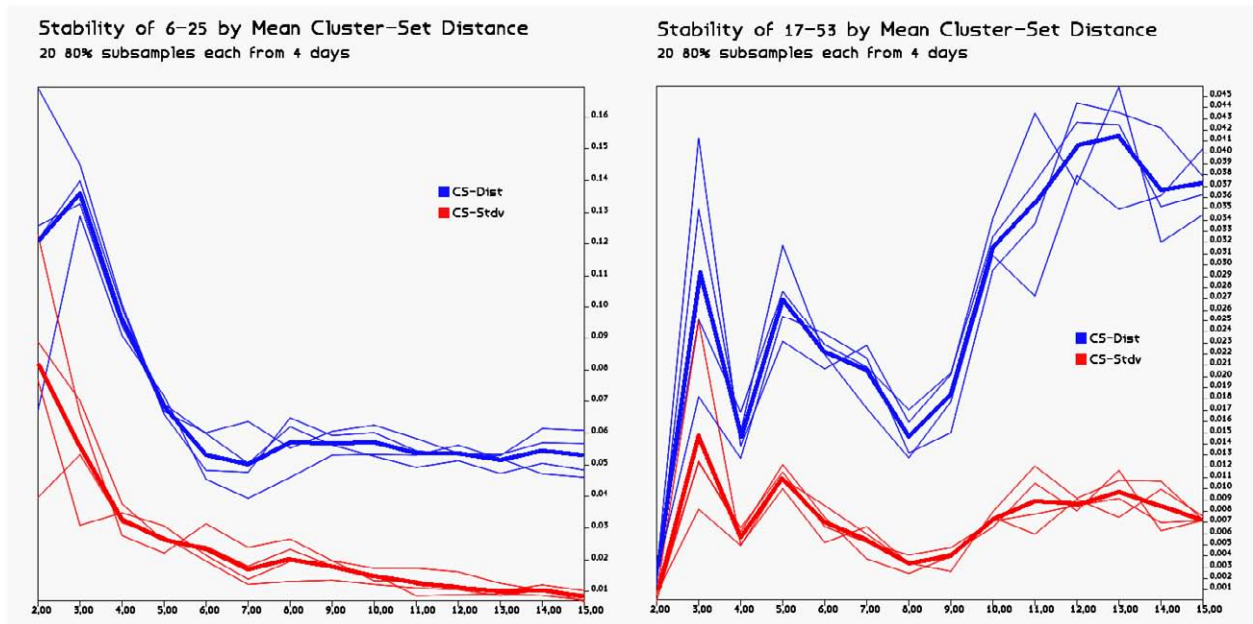


Figure 8 – Cluster Count Determination

Based upon the results obtained by this method, the following selection of cluster counts was decided for each of the 7 services subject to profiling:

Service Description	Protocol-Port	Cluster Count
HTTP (web) traffic	6-80	11
DNS (Domain Name Service)	17-53	8
HTTPS (Secure web)	6-443	11
NTP (Network Time Protocol)	17-123	7
SMTP (Simple Mail Transfer Protocol)	6-25	7
SSH (Secure Shell)	6-22	7
FTP (File Transfer Protocol)	6-21	11

Figure 9 – Selected Cluster Counts by Service

As an aid to visualizing the longitudinal stability of the service clustering, a clustering solution was obtained for each of 15 consecutive workdays. The centroids obtained for the first day were treated as the “root” cluster centroids for the given service, and each is given a unique color in the parallel coordinates graph in figure 10 below. On each subsequent day, an independent clustering is obtained. The new centroids are matched to the root centroids according to minimal matching distance, and colored accordingly. For each cluster, the black-edged line indicates the mean of the corresponding centroids.

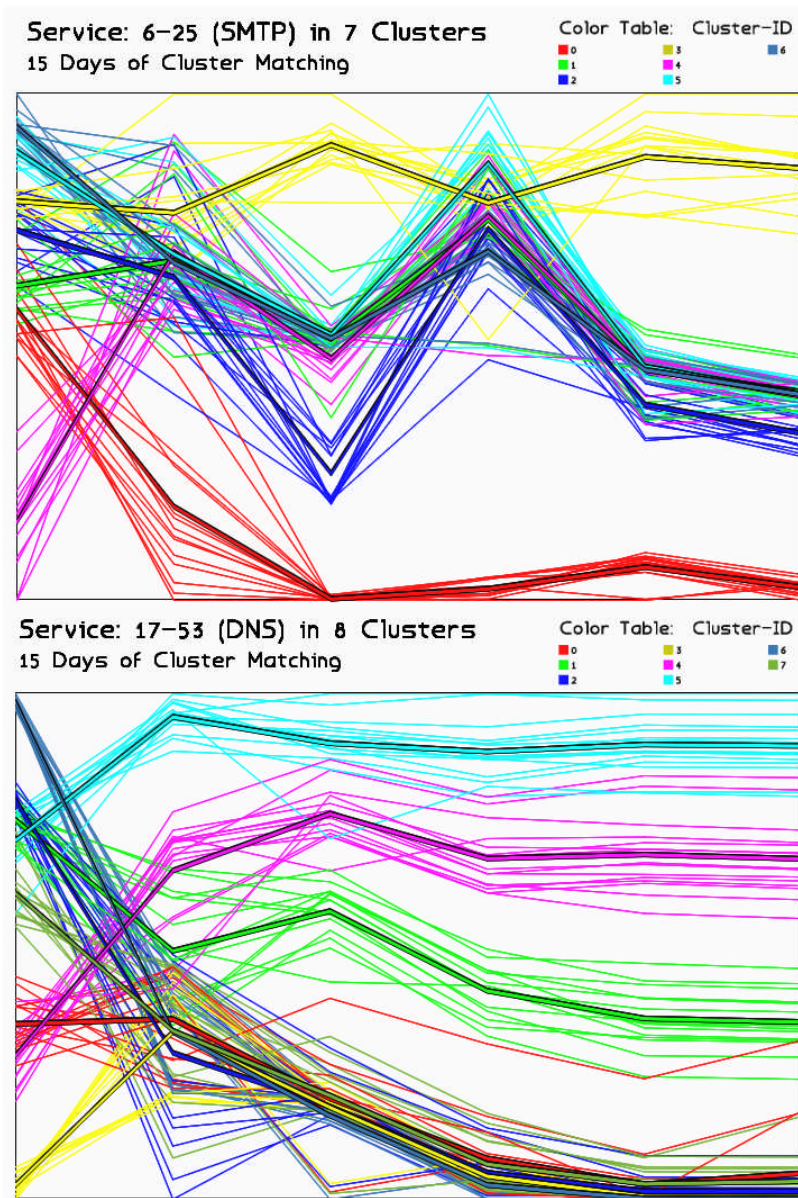


Figure 10 – Parallel Coordinate Visuals of Service Cluster Histories

A Regime for Service and IP Profile Updates

In our routine IP-profiling activity, data is processed on a daily basis. Each day, the session summary data is filtered down to the selected services and then further restricted according to our other limiting criteria (“outbound” initiated sessions, full sessions only, two-way packet exchanges) and the feature vectors produced. The resulting collection of records is then processed in support of two separate activities:

1. Creation or update of **canonical service clusters** and centroids
2. Creation or update of **IP profiles** and daily anomaly ranking

To describe these activities in detail, we provide the following definitions:

Def. Canonical Service Clusters

For an agreed-upon structure of service feature vectors, and a predetermined clustering method and cluster count K , let an initial set of such vectors be clustered and declare the resulting set of cluster centroids to be the initial **canonical service clusters** (CSC) for the service in question, and write

$$CSC = \{csc_1, \dots, csc_K\}$$

to denote the set of centroid vectors, ordered (initially) in an arbitrary but fixed manner. For each subsequent day contributing to the history of the service, that day’s feature vectors are clustered by the same methods, producing a new set of centroids $SC = \{sc_1, \dots, sc_K\}$, where the ordering is now performed in accord to the minimal sum (Euclidean) matching distance to the centroids of the CSC. For a desired decay factor α , we migrate the CSC toward the current day’s centroids to produce the updated CSC’ by

$$CSC' = \{ \alpha csc_i + (1 - \alpha)sc_i \}, \text{ for } i = 1, \dots, K.$$

In our study, we chose $\alpha = 0.90$ as the weight to apply to the previous mean on each update. Note that in calculating the initial centroids csc_i of the clusters, one could also maintain and migrate the covariance matrix (CV) of the component features. Where μ_{1i} and μ_{2i} are the historical and recent values of an i -th centroid component, and CV_{1ij} and CV_{2ij} are the historical and recent values of a cell in the covariance matrix, one can produce the updated components by

$$\begin{aligned} \mu_i &= \alpha \mu_{1i} + (1 - \alpha) \mu_{2i} \\ CV_{ij} &= \alpha (CV_{1ij} + \mu_{1i} \mu_{1j}) + (1 - \alpha) (CV_{2ij} + \mu_{2i} \mu_{2j}) - \mu_i \mu_j \end{aligned}$$

If the covariance matrix is maintained (and is invertible), a more accurate measure of the association of arbitrary feature vectors to the canonical service centroids can be obtained by using Mahalanobis distance [5].

Armed with the service CSC’s, we produce and update daily IP profiles by considering the degree to which each IP’s activities on a given service align to the corresponding service clusters.

Def. Service Cluster Membership Vector (SCMV)

Let $V(IP, S) = \{ v_t \}$, $t = 1, \dots, T$ be the feature vectors corresponding to the activities of a particular client IP on a selected service S , those activities comprised of T

sessions for a given day. Let $CSC = \{C_1, \dots, C_K\}$ be the K canonical service clusters representing the basis by which the vectors $V(IP, S)$ are to be assessed.

We define this IP's **service cluster membership vector (SCMV)**, for the day in question, to be the normalized K -dimensional vector $u = \langle u_1, \dots, u_K \rangle$, where the value u_i represents the degree to which the feature vectors of $V(IP, S)$ are associated with service cluster C_i . In particular, for each v_t in $V(IP, S)$ we may consider the vector distance from v_t to each canonical service cluster centroid as

$$d_{i,t} = \text{dist}(v_t - C_i), i = 1, \dots, K$$

and take the contribution of v_t to each cluster to be inversely proportional to this distance, raised to a power of our choosing. (For our study, we chose $p=2$). This leads each v_t to contribute to the i -th component u_i of the normalized membership vector u in the amount

$$1 / ((d_{i,t})^p \sum_j^K (1 / d_{j,t})^p)$$

Summing the (normalized) contributions from all v in $V(IP, S)$, we obtain

$$\begin{aligned} u &= \langle u_i \rangle, i = 1, \dots, K \\ &= \langle (1/T) \sum_t [1 / ((d_{i,t})^p \sum_j^K (1 / d_{j,t})^p)] \rangle \end{aligned}$$

Note that if we let the exponent $p \rightarrow \infty$, then each vector v merely contributes the amount $1/T$ to the component whose centroid is closest to v (or contributes $1/(nT)$ to the closest n components in the case of an n -way tie).

As with the canonical service cluster centroids themselves, we establish for each IP and service an initial SCMV to represent that IP's "profile", and for subsequent days we calculate a new value from that day's activities and migrate the profile accordingly, maintaining a "running mean" SCMV $\mathbf{U}$ for the IP by

$$\mathbf{U}' = \langle u'_i \rangle = \langle \alpha \mathbf{U}_i + (1 - \alpha) u_i \rangle, \text{ for } i = 1, \dots, K$$

In summary, each day of processing involves

1. IP Behavior Snapshot

Production of that day's SCMV's, for each IP and each service, conducted by assessing the feature vectors of each IP and service with respect to the current canonical service cluster centroids (CSCs).

2. IP Anomaly Ranking

Comparisons of each IP's SCMV's to their own historical mean SCMV's, in order to generate an anomaly score for ranking purposes.

3. IP Profile Maintenance

Migration of each IP's historic profile SCMV's using the current day snapshot SCMV's.

4. Canonical Service Cluster Maintenance

Independent clustering of all behaviors evidenced for each given service to produce the current day's service clusters, and the migration of the historical canonical service clusters (CSCs) accordingly.

Results/Technical Outcome

Performance

For the typical daily volumes (5.5M sessions, 10-2000 IPs active), the aforementioned processing requires approximately 15 minutes on a commodity Dell blade (i686 processor). This is roughly 100 times faster than “real time”, and suggests ample leeway exists to extend the profiling activity.

This measure does not include the cluster stability testing employed at the outset to select the cluster count for each service.

Measures of Effectiveness

Ideally, we would have a history of systems for which some subset was known to be subject to compromise (evidenced by beaconing or unusual data exfiltration) during the period of study. This would provide a degree of “ground truth” in the separation of compromise-derived anomaly from the background of (assumed) innocuous anomaly.

In lieu of such data, we can examine the sensitivity of the subject profiling measures in terms of the distributions of deviation from profile and also the temporal patterns of high anomaly. Two such methods are depicted here.

In the first method, we consider how well these IP profiles serve to isolate individual IPs. In particular, we consider the daily membership vectors (SCMVs) generated for each IP (and for each service), and determine how well proximity to the IP’s service profile serves to classify the membership vectors.

For each service, we limit our investigation to the profiles of those IPs whose behaviors were evidenced in each of the 43 “consecutive” workdays of the study (excluding weekends and holidays). For each qualifying IP, we define a service-neighborhood to be the (presently spherical) region centered on the IP’s service profile, whose radius is given by the IP’s most distant membership vector. Hence, the region encompasses every membership vector evidenced by the IP. We then conducted a random sample of days to obtain the membership vectors attributed to IPs other than the profiled IP, sorted these together with the IP’s own vectors in terms of distance to profile, and mapped the rates of true and false positives as distance from profile increased to the neighborhood radius. The resulting ROC curves, and effectiveness in terms of area under curve (AUC) are depicted in figure 11.

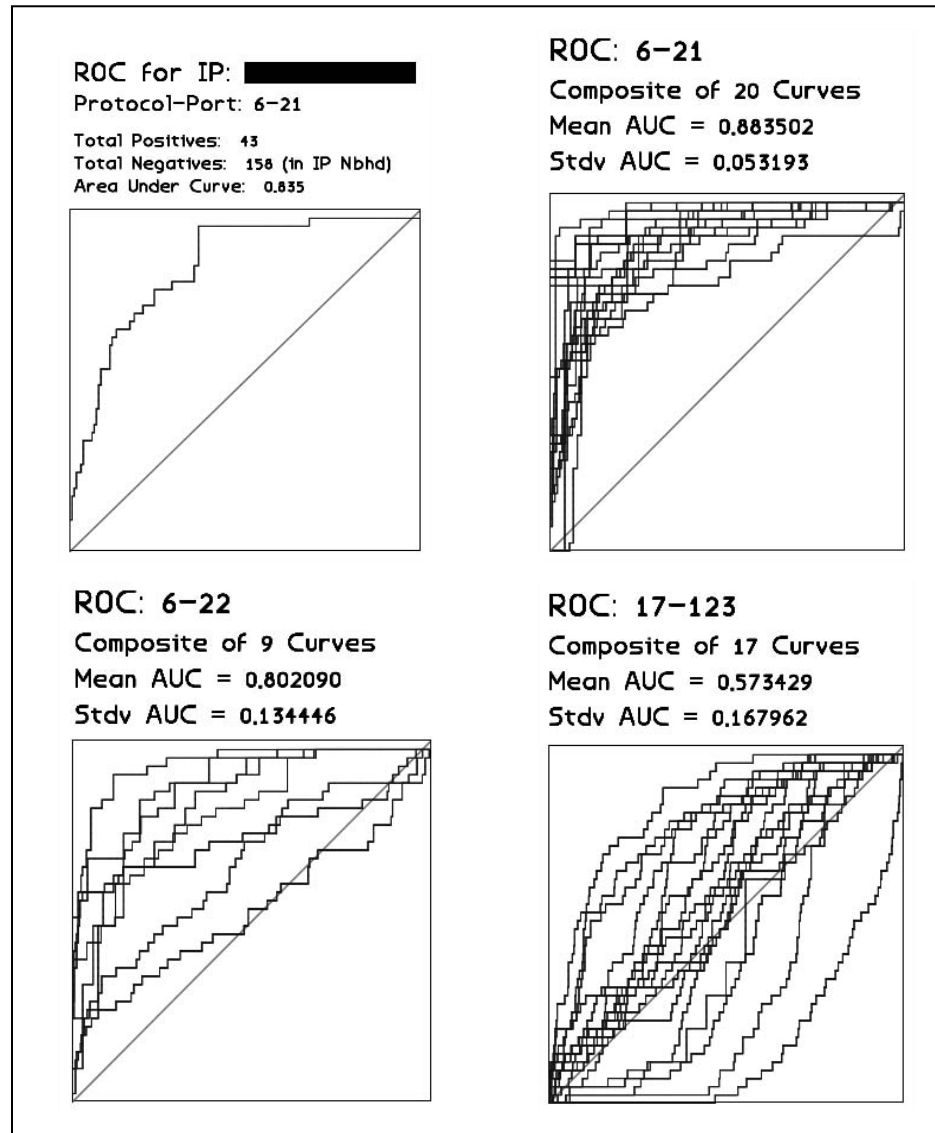


Figure 11 – Selected ROC Curves for Non-Proxied Services

These ROC curves are for comparative purposes only, as they are heavily affected by outliers of each IP, which determines the population of false positives. Even so, one can see that the FTP service (6-21) allows much better discrimination of IP behavior than either SSH (6-22) or NTP (17-123). One can surmise the dismal ability to discriminate based upon NTP is due to the lack of variations able to be exercised within that protocol. To obtain a more accurate assessment of the effectiveness in IP discrimination, one could measure profile neighborhoods in terms of Mahalanobis Radii. We also expect that we would obtain strong results for web traffic (HTTP, HTTPS) if we could access the individual IP behaviors "behind" the proxy servers.

In the second method, we compare each IP's daily membership vector to the history of its moving average. By displaying these differences in a triangular array, one can immediately visualize the relative stability or volatility in the behavior of each IP, and easily identify those points in time when a significant departure from historical behaviors has occurred (figure 12).

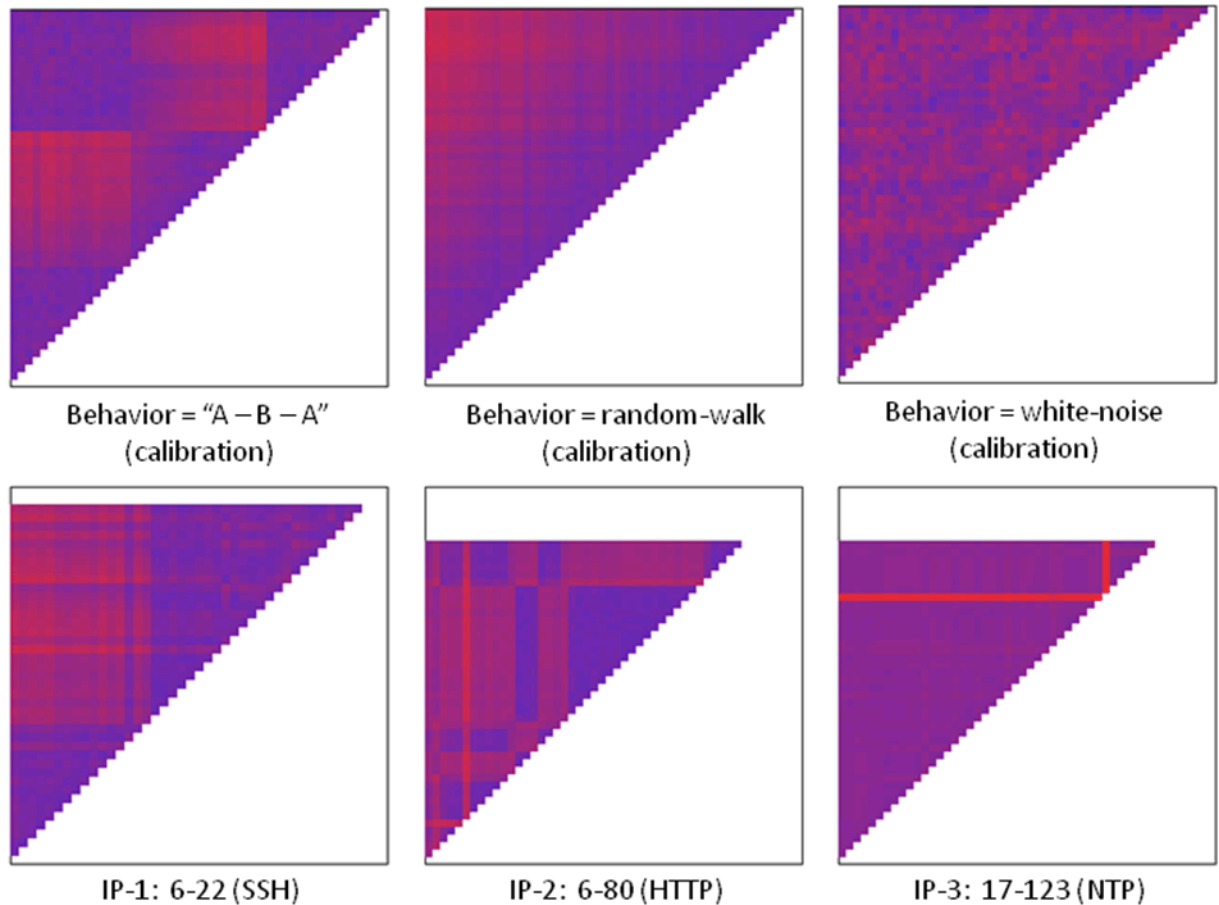


Figure 12 – Visualizing IP Histories

One interprets a graph in figure 12 by proceeding from the lower-left to upper-right corner along the "white diagonal" (cells directly adjacent the colored region). Each cell on that diagonal represents another day (its corresponding daily membership vector), and to its left lies a row of colored cells moving back through past days. Those cells are colored toward blue when the current day's vector is close to that past day's running mean membership vector U , and tends toward red when it is correspondingly distant. For example, in the graph for "IP-3: 17-123 (NTP)", the horizontal cells left of almost every diagonal cell are uniformly blue-purple, indicating that each day was relatively similar to all past days, until one arrives at the 7th from last day. On that day, a significant departure is evident – the entire row is bright red. Thereafter, the behavior returns to its previous quality, similar again to all past days except for the one anomalous day. Likewise, one can interpret IP-2 (a web proxy) to alternate somewhat irregularly between at least two distinctive behaviors. Lastly, IP-1 is seen to have one behavior for the lower 1/3 of its diagonal, and then to switch to a new behavior for the remaining 2/3, being thereafter similar to its recent past, but dissimilar from its distant past.

The values along the first super-diagonal are thus an indication of the "present anomaly" of the corresponding present day (white cell to the immediate right).

Summary

We have demonstrated the practicality, in terms of performance, of maintaining and migrating daily canonical service clusters for a range of IP services, of maintaining and migrating individual IP mean service cluster membership vectors (SCMV) for thousands of IPs (systems) representing the behavioral profiles for those IPs, and for assessing each IP's present SCMV to that of its running mean, in providing an anomaly measure.

In an operational environment, one could seek IP SCMV's that are proximal to those of a known compromise, or one could take highly anomalous SCMV, identify the component(s) responsible for the divergence, and then locate actual session summary records in recent store that correspond to the identified clusters for further analysis.

Future work could consider optimizations in the clustering methods employed in seeking more distinct clustering solutions (more definitive cluster counts) and improved cluster stability over time. Additionally, sensitivity testing is needed, wherein either known compromise behavior or simulated departures from norm are applied in order to calibrate various detection thresholds.

References

- [1] **Characterizing Network Services through Cluster-Set Variations**, Anthony Bartoletti, Nu Ai Tang, Lawrence Livermore National Laboratory, UCRL-TR-211020, <http://www.osti.gov/bridge/servlets/purl/15016408-OWMNng/native/15016408.pdf>
- [2] **Mining the Structure of User Activity using Cluster Stability**, Jeffrey Heer, Ed H. Chi, PARC (Palo Alto Research Center) {jheer, echi} @parc.com
- [3] **Comparing clusterings --- an information based distance**, [Marina Meilă](#), Journal of Multivariate Analysis, Volume 98 , Issue 5 (May 2007) ISSN:0047-259X
- [4] http://en.wikipedia.org/wiki/Hungarian_algorithm
- [5] http://en.wikipedia.org/wiki/Mahalanobis_distance