An Approach to Online Network Monitoring Using Clustered Patterns

Jinoh Kim*†, Alex Sim†, Sang C. Suh*, Ikkyun Kim‡
Texas A&M University, Commerce, TX 75428 *
Lawrence Berkeley National Laboratory, Berkeley, CA 94720 †
ETRI, Daejeon, Korea ‡
Email: jinoh.kim@tamuc.edu, asim@lbl.gov, sang.suh@tamuc.edu, ikkim21@etri.re.kr

***Abstract*—Network traffic monitoring is a core element in network operations and management for various purposes such as anomaly detection, change detection, and fault/failure detection. In this paper, we introduce a new approach to online monitoring using a pattern-based representation of the network traffic. Unlike the past online techniques limited to a single variable to summarize (e.g., sketch), the focus of this study is on capturing the network state from the multivariate attributes under consideration. To this end, we employ clustering with its benefit of the aggregation of multidimensional variables. The clustered result represents the state of the network with regard to the monitored variables, which can also be compared with the previously observed patterns visually and quantitatively. We demonstrate the proposed method with two popular use cases, one for estimating state changes and the other for identifying anomalous states, to confirm its feasibility.**

1. Introduction

Monitoring network traffic is an integral part in network operations and management. The basic requirement for network monitoring is to effectively capture the traffic dynamics in a timely manner. For example, some anomalies are the indication of performance bottlenecks with a huge number of simultaneous connections, which may be caused by several reasons such as flash crowds, denial of service attacks, or network component failures. The monitored results are then used for reconfiguring the network to optimize performance or to reinforce security with the historical information.

A crucial challenge for traffic monitoring is the exponential increase of the volume of data [1], [4]. This should be more critical to online monitoring with a large number of incoming/outgoing packets to be processed within a time interval. For example, it can be over tens of millions of packets per second for a single 10 Gbps link. For this reason, the in-depth analysis (e.g., on a per-flow basis) would not be feasible for online monitoring, particularly in a large-scale network (e.g., ISP or enterprise networks). In response to this, data streaming computation has been widely studied to provide a summary of network traffic within a time interval. For instance, sketch is a statistical summary technique based on counting with hashing to analyze the network traffic [10], [12], [17]. While decent for streaming computation without keeping extravagant per-flow data, a sketch is limited to give the statistics for a single traffic variable. The probabilistic density information was also considered as the snapshot of the network traffic [5]. However, this class of techniques have the same problem and are restricted to one dimensional variable to analyze. As a result, the individual variables should be analyzed independently with this scheme and how to combine them is left to the administrator.

In this paper, we propose a new approach that offers a high-level summary of the network traffic in regard to the multivariate attributes under consideration. With this approach, what the administrator sees is an abstract *pattern* compiled from the traffic variables being monitored, rather than observing a bunch of the independent statistical information for the variables. By doing so, the detection problems in traffic analysis (e.g., change detection and anomaly detection) can be reduced to the pattern comparison problem. For ease of exposition, we use a term "network state" defined as the recap of the network traffic with respect to the monitored variables. The observed pattern represents the network state for the associated time interval, and it can be compared with the patterns in the archive to interpret the traffic dynamics.

To represent the network state, we employ *clustering* with its benefit of the aggregation of multidimensional attributes. The clustered result is taken as a pattern that tells the network state in question. In addition, the connected cluster information (e.g., centroid positions) is used to compare patterns in a quantitative manner. Note that the focus of this paper is fundamentally different from the past studies employed clustering for anomaly detection [8], [9], [11], [15], the primary concern of which is to test individual data elements to classify. The main interest of our research is to develop a method for the high-level online monitoring without much details.

In this paper, we present the new approach along with two use cases. The first use case is the estimation of traffic changes between two different time intervals. The network state is captured with a static number of clusters, and the change is measured with the "degree of changes" established based on the movement of the centroid coordinates. The second use case we demon-

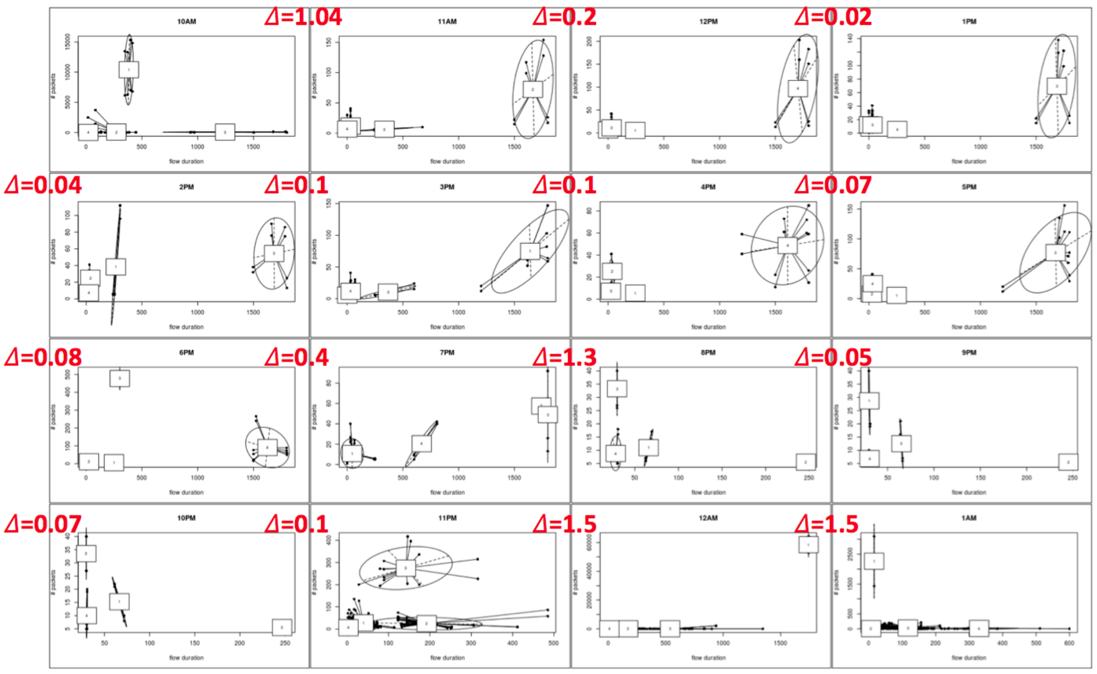

Figure 1. Clustering results over 16 consecutive hourly time windows on UNIBS data trace for flow duration on x- axis and average number of packets in flow on y-axis, between 10AM on Sep. 30, 2009 and 2AM on Oct. 1, 2009. The number of clusters is set to 4, based on the sum of squares in groups. Note that cluster IDs were randomly selected by the clustering tool.

strate is anomaly detection. In this use case, we assume a flexible number of clusters for a time window, and the abnormality of a cluster is estimated with the patterns of attack clusters using a new measure of "likelihood" constructed based on the centroid positions and the sum of squares information. We will show the feasibility of the proposed approach with the evaluation conducted with public traffic traces.

2. Detection of Network State Changes

In this section, we present how the clustered patterns can be employed to estimate the changes of network states over time. In addition, we introduce a new measure to gauge the change between two patterns in a quantitative manner.

In this study, we use a 16-hour trace excerpted from the UNIBS traffic trace, between 10AM on September 30, 2009 and 2AM on October 1, 2009 [6]. The data set contains the information for network flows[1] with timing, and the ground-truth data with the associated application for each connection is provided [16]. As statistics, the average number of flows is 789 flows/hour with a high degree of variance (min=20, max=7052).

The key proposed idea is to capture the network state from the monitored variables and represent it with a clustered pattern. We chose a conventional partitioning-based clustering (K-means) with its manageable complexity and scalability. We set the number of clusters to 4 ($K = 4$) for a single time window, which is derived from the total sum of squares. We basically considered two attributes, flow duration and the number of packets per flow, to represent the state of the network. Thus, the clustered pattern is compiled with the above two traffic variables.

Figure 1 demonstrates the clustering results over 16 time windows (for 16 hours). Note that the cluster IDs in the plots were randomly assigned. As shown in the figure, the clustered results show the captured network states, as well as the correlated patterns (Δ in the figure will be explained shortly). For example, the pattern for 10AM time window is quite different from the one for 11AM time window. In contrast, the clustered patterns from 11AM to 5PM are visually similar. The three patterns for 8PM–10PM time windows are also resembling, whereas the last three time windows (11PM–1AM) have somewhat distinctive patterns.

To understand the traffic characteristics, we referred to the composition of applications for each window using the ground-truth information, shown in Figure 2. From the two figures of 1 and 2, we see a strong correlation. For example, the breakdown graph (Figure 2) shows a high degree of similarity from 11AM to 5PM and from 8PM to 10PM, respectively, which agrees with similarity of the clustered patterns in Figure 1. On the other hand, there is a high degree of difference in the breakdown graph between 10AM and 11AM. Similarly, we can see huge differences from the windows

1. A flow is identified with five tuples of source IP address, source port number, destination IP address, destination port number, and protocol in TCP/IP header

of 11PM–1AM in Figure 2.

We see that the visual patterns are helpful to catch the traffic dynamics. We next introduce a new measure to quantitative estimate the difference between two patterns.

Degree of changes (Δ):

We introduce a new measure of "degree of changes" (Δ) that computes the difference of two patterns based on the movement of the centroid positions of the clusters. The basic intuition behind this is that the centroid coordinates of two patterns would be close without a heavy change if the network states in comparison are tightly related. That is why we fixed the number of clusters in this use case.

Suppose two time windows W_i and W_j, and the associated cluster sets $C_i = \{c_i^0, c_i^1, ..., c_i^k\}$ and $C_j = \{c_j^0, c_j^1, ..., c_j^k\}$, respectively, where k is the number of clusters. Each cluster c_x^y has its centroid p_x^y. Without knowing which cluster in W_i is mapped with one in W_j, we find a set of pairs showing the minimal move. Suppose a distance function $D : C_i \times C_j \to \mathbb{R}$. Then the problem is reduced to the assignment problem that finds a bijection $f : C_i \to C_j$ with the minimal distance function:

$$\Delta_{i,j} = \sum_{l \in C_i} D(l, f(l))$$

Hungarian algorithm is a well-known method for this type of problem with $O(k^3)$ of the computational complexity [13].

The Δ values in Figure 1 represent the degree of changes for two adjacent windows. From the Δ's, we see relatively small Δ's from 11AM to 6PM and from 8PM to 10PM ($\Delta \leq 0.2$). On the other hand, a high degree of changes were observed for the adjacent windows of (10AM, 11AM), (7PM, 8PM), (11PM, 12AM) and (12AM, 1AM) ($\Delta > 1.0$). While strong correlations are observed overall, the Δ(10PM, 11PM) is the only exception with a small value (Δ=0.1) although the two windows are slightly different visually. This indicates that the visual patterns and quantitative measures are complementary and need to be considered together to properly read the state of the network.

In summary, we observed a strong correlation between the clustered patterns and traffic composition that confirms the feasibility of the proposed method. The clustered patterns with the established measure would be helpful to follow up the changes of network states in both qualitative and quantitative manners.

3. Detection of Anomalous States

We next explore the effectiveness of the use of clustered patterns for network anomaly detection. For this use case, we consider the KDDCup 1999 data ("kddcup.data_10_percent_corrected") [3] that has been widely used in the anomaly detection study. A data instance in the data set is a record of a single TCP connection. The individual record also contains a label

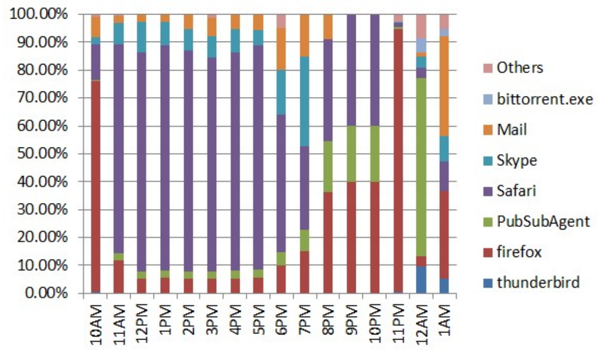

Figure 2. Breakdown of applications for time windows (10AM–1AM), compiled from the ground-truth data

for: a normal connection ("NORMAL"), a denial of service attack ("DOS"), an unauthorized access from a remote host ("R2L"), an unauthorized access to root functions ("'U2R"), and one employed for probing for vulnerabilities ("Probe"). Since no timing information is provided in the data set, we formed 16 partition windows ("AA" – "AP") serially from the beginning of the data file, each of which contains 1,000 connections.

Two connection-related attributes were considered in this case study: "src_bytes" is the number of bytes from the source to destination host and "dst_bytes" is the number of bytes from the destination to source. Due to a high degree of skewness, we normalized the data using a log function. For example, the summary of src_byte *before* normalizing is: 0 (min), 45 (1st quartile), 520 (median), 3026 (mean), 1032 (3rd quartile), 693400000 (max); the summary *after* the normalization is: 0.00 (min), 1.65 (1st quartile), 2.72 (median), 2.16 (mean), 3.01 (3rd quartile), and 8.84 (max). The normalization makes sense since a 10-byte difference is still critical to the connections in the 1st quartile group ($\leq$ 46 bytes), while it is insignificant to the right-skewed connections such as one with 693,400,000 bytes (max).

As in the previous section, we employ the K-means clustering to obtain the patterns. One difference is that the number of clusters is dynamically determined for each time window, while it was fixed in the first use case that focuses on the degree of changes. The number of clusters for a time window is estimated by the optimum average silhouette width. The procedure for clustering is as follows: For each window W_i, the number of clusters (k_i) is estimated with the data instances in that window (D_i and $|D_i|$ = 1,000). Then the clustering algorithm is executed with an input of k_i to obtain a clustered pattern for W_i.

Figure 3 shows the clustering result and Table 1 provides a summary of the windows with labels. From Table 1, we can see the windows from AH to AL have a lot of DOS connections and three windows of AA, AE and AP include some malicious connections. From the figure, we can see noticeable similarity from many clean windows (AB–AD, AF, AM–AO). The windows

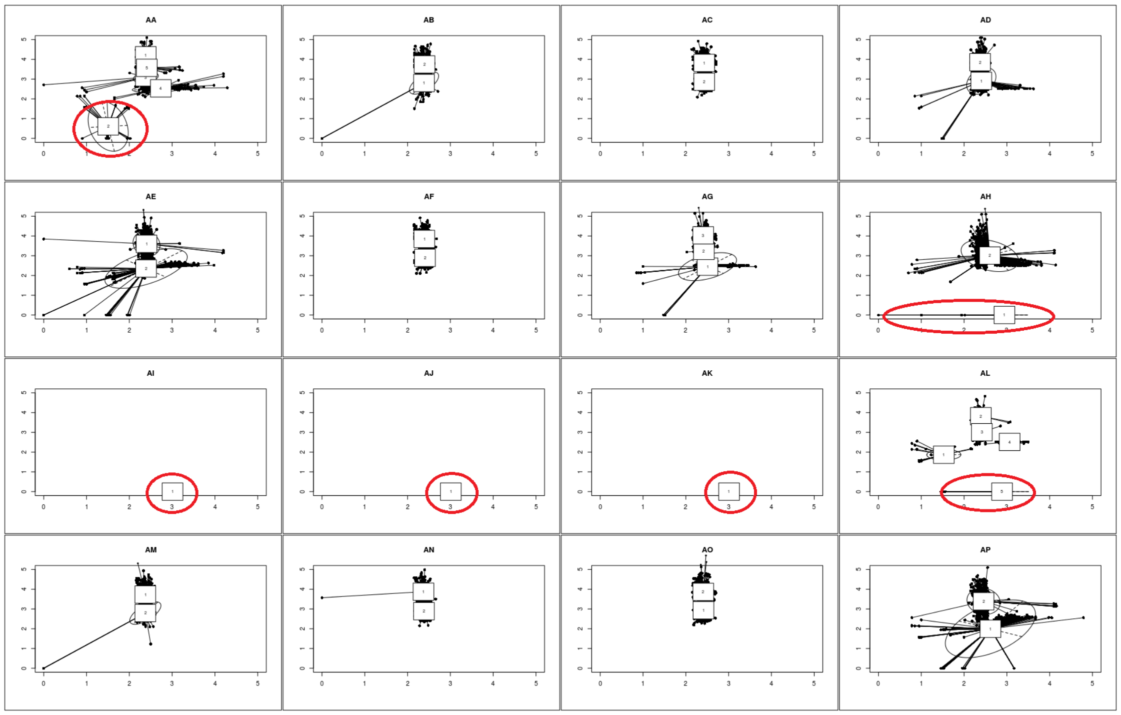

Figure 3. Clustering results for dataset A: Each window contains 1,000 connections without overlapping. The K-means algorithms is used and the number of clusters is estimated using the partitioning around medoids technique. The clusters in a thick circle are not found from a cluster with normal connections only.

containing malicious events (AA, AE, AH–AL, and AP) yield quite different shapes from the ones observed in the clean windows. We can see that DOS makes a specific pattern as shown in the windows of AH–AL.

Measuring likelihood of attacks:

We next present a metric of "likelihood" as a quantitative tool for operators to test the potential of a cluster as an attack class. One simple way to compare two clusters would be to use the Euclidean distance $d(c_1, c_2)$, where c_i is the centroid position of cluster C_i in a multi-dimensional space; hence, $d \in \mathbb{R}$ and $d \geq 0$. Then we can assume a greater likelihood for two clusters with a smaller distance. When $d = 0$, it implies that the given clusters have the same centroid position, and it will result in the greatest likelihood by definition. While simple, the limitation of this method is that none of d other than zero may not be adequately interpreted to estimate the likelihood since it can be any positive real number. For instance, it could be $d = 1$ or $d = 100$; what do they mean in terms of the likelihood? To establish a metric of the likelihood, we want to scale it between zero (the least likely) and one (the most likely) for the relative comparison.

We assume that a cluster is represented with two parameters of $<c, v>$, where c is the centroid position and v is a degree of variation in the cluster. A larger v thus indicates that the elements in the cluster are further dispersed in the space. Obtaining the centroid position is straightforward with the K-means clustering as any cluster has a centered point that minimizes the total squared sum of the elements ("within-SS" or simply wss). We utilize this parameter (wss) to characterize a degree of variation, which is defined as $v = \frac{wss}{n}$, where n is the number of elements in the cluster. With this representation of a cluster, we establish two measures for the likelihood defined as follows.

To define the first measure for the likelihood, we borrow a concept of the statistical percentile in a Gaussian distribution using *p-value*. The first measure (l_{pv}) is derived from the statistical percentile, by assuming that the centroid position of the compared cluster as the mean and the $\sqrt{v}$ as the standard deviation. Suppose a cluster $C' = <c', v'>$ and an attack cluster $C = <c, v>$, and we would like to determine the likelihood of an attack of C for C'. The normal score (z-score) is calculated by $\frac{c-c'}{\sqrt{v}}$, where $(c - c')$ is the distance in the Euclidean space. With the calculated normal score, l_{pv} is defined as the two-fold percentile using the associated *p-value*. Note that we assumed $v = \min(v, 0.01)$ to prevent the divide-by-zero error.

We next introduce the second measure l_{dr} based on the centroid positions and the radius of the cluster. While l_{pv} tests how close the center of the cluster is to the attack cluster's center, we consider a cluster as a circle with a radius that is assumed equal to the deviation of the cluster to establish l_{dr}. Thus, cluster $C' = <c', v'>$ is represented as a circle with the center

TABLE 1. Traffic composition & likelihoods of attacks

Window	AA	AB	AC	AD	AE	AF	AG	AH	AI	AJ	AK	AL	AM	AN	AO	AP
Normal	997	1000	1000	1000	998	1000	1000	792	0	0	0	511	1000	1000	1000	979
DOS	0	0	0	0	0	0	0	208	1000	1000	1000	489	0	0	0	20
U2R	2	0	0	0	2	0	0	0	0	0	0	0	0	0	0	0
R2L	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
Probe	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

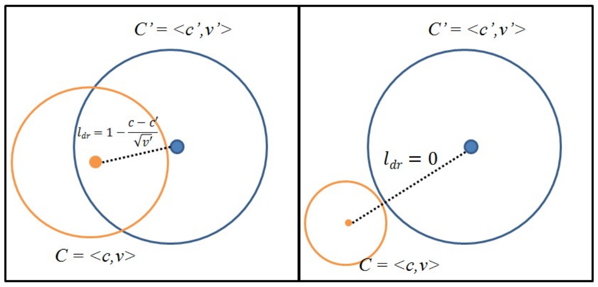

Figure 4. Estimating likelihood based on the radius of the circle (l_{dr}): The blue circle is a cluster in question and the orange circle is an attack cluster. In the left figure, the center of the attack cluster is located within the blue circle and the likelihood is defined as the relative distance compared to the radius of the blue circle ($l_{dr} = 1 - \frac{c-c'}{\sqrt{v'}}$). The right figure show an example that the two circles have no overlap since the center of the attack cluster is located outside the blue circle ($l_{dr} = 0$).

of c' and the radius of $r' = \sqrt{v'}$. To test the likelihood of an attack represented with $C =<c, v>$, we compare the two center points c and c' with the radius r'. In the comparison step, we compute l_{pv} as follows:

$$l_{dr} = \max\left(1 - \frac{c - c'}{\sqrt{v'}}, 0\right)$$

Figure 4 demonstrates two examples for l_{dr}. In the figure, the blue circle is a cluster in question, and the orange circle is an attack cluster. As shown in the left figure, we compute the likelihood by $l_{dr} = 1 - \frac{c-c'}{\sqrt{v'}}$, since the center of the attack cluster is located within the blue circle. In contrast, it is a zero ($l_{dr} = 0$) in the right figure as the two circles have no overlap, and the center of the attack cluster is located outside the blue circle.

To evaluate the defined measures, we precalculated the patterns of the attack clusters for each attack class. Using the patterns for individual attacks, we compute likelihood using l_{pv} and l_{dr} for each window. As there can be multiple clusters for a single attack type and a window can have more than a single cluster, we define a collective measure L that takes the max from the set of the likelihood values. In detail, suppose a window with a set of clusters $\{C'_1, C'_2, ..., C'_m\}$, and an attack class (e.g., DOS) with another set of clusters $\{C_1, C_2, ..., C_n\}$. Then the collective likelihood L_α for attack type α is defined as:

$$L_\alpha = \max_{i,j} l(C'_i, C_j), \text{ where } 0 < i \leq m \text{ and } 0 < j \leq n$$

TABLE 2. Evaluation of measures (number of cases and percentage)

Measure	$L \geq 0.5$		$L \geq 0.7$	
	FP	FN	FP	FN
L_{pv} (2D)	8 (50%)	3 (19%)	6 (38%)	4 (25%)
L_{dr} (2D)	1 (6%)	2 (13%)	0 (0%)	3 (19%)
L_{pv} (3D)	0 (0%)	4 (25%)	0 (0%)	5 (31%)
L_{dr} (3D)	0 (0%)	3 (19%)	0 (0%)	3 (19%)

The following illustrates the steps to evaluate likelihood:

1) If the window has no attack and none of L is greater than or equal to the threshold, it is a *true negative* (TN);
2) If the window has any types of attacks and any of the associated L is greater than or equal to the threshold, it is a *true positive* (TP);
3) If the window has any types of attacks and any of the associated L is less than the threshold, it is a *false negative* (FN);
4) Otherwise, it is a *false positive* (FP).

We examined with two threshold values, $L \geq 0.5$ and $L \geq 0.7$, using two attributes (src_bytes and dst_bytes) and using three attributes (plus "connection duration"). We refer to the former as "2D" and the latter as "3D". Table 2 shows the calculated FPs and FNs. From the result, we can see using three features reduces the number of the false decisions but with a slight increase of false negatives. Also we can see that L_{dr} works conservatively compared to L_{pv}.

4. Related work

With the heavy increase of traffic volumes, flow-level traffic analysis would not be feasible for high-speed network monitoring. To identify significant changes or abnormality, the technique of sketch has been extensively explored [10], [12], [17]. The sketch basically makes use of a set of hash functions, and the incoming data points are counted using the key in question and the hash functions. For example, the key would be 64 bits with source and destination IP addresses. The statistics of the hashed results, such as *min, mean, quartile*, can then be referenced for the detection. Probabilistic models and samplings have also been considered to summarize network traffic. The work in [5] proposed a dynamic sampling technique to reduce the size of streaming data based on the difference of the probabilistic density. The authors employed the Kolmogorov-Smirnov test to measure the distance of two summaries observed in different time intervals. The past techniques are limited to capture a single traffic variable only, and individual variables should be analyzed separately. The key difference of the proposed approach is the ability to capture the multivariate traffic variables to provide a comprehensive view of the network state.

A large body of studies employed clustering for intrusion/anomaly detection [8], [9], [11], [15]. The main focus of many of such studies was on identifying intrusive events individually with a trained model consisting of normal or anomalous behaviors. For instance, the authors in [15] made an assumption that anomalous events would be much smaller than the normal events in quantity. The clustering information is then used to test whether the input is anomalous or not, based on this assumption. Similarly, a recent work [14] proposed a method using co-clustering to improve the detection performance. The main focus of our work is fundamentally different from the past work and our interest is rather to identify the network states deviated from the normal, than detecting individual anomalous connections based on in-depth analysis.

5. Conclusion

This paper proposes a new approach to the high-level online network monitoring using clustered patterns. The main goal of this study is to enable the comprehensive analysis with multivariate attributes. We demonstrated the feasibility of the proposed technique with two popular network monitoring applications. For the detection of state changes, we demonstrated our analysis on the clustering results with the associated groundtruth information, with the established measure of “degree of changes”. We also presented the network anomaly detection as the second use case. We defined a new metric to estimate the likelihood of a cluster to be an attack class and set up two measures with the clustering information, one based on percentile in the Gaussian distribution and the other based on the radius of a cluster. The evaluation results support the effectiveness of the new approach with the pattern-based representation of network states and the quantitative measures.

This research is in the initial stage and we showed the potential of our approach in this paper. We plan to further examine with other large-scale traffic traces, such as MAWILab [7] and ESnet [2], to elaborate the proposed method.

6. Acknowledgment

This work was supported in part by the Office of Advanced Scientific Computing Research, Office of Science, of the U.S. Department of Energy under Contract No. DE-AC02-05CH11231 and by Institute for Information & communications Technology Promotion(IITP) grant funded by the Korea government(MSIP) (No.B0101-15-1293).

References

[1] Cisco white paper: Cisco vni forecast and methodology, 2015-2020, http://www.cisco.com/c/dam/en/us/solutions/collateral/service-provider/visual-networking-index-vni/complete-white-paper-c11-481360.pdf.

[2] ESnet. https://www.es.net/.

[3] KDD Cup 1999 Data. http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html.

[4] K. Cho, K. Fukuda, H. Esaki, and A. Kato. Observing slow crustal movement in residential user traffic. In *Proceedings of the 2008 ACM Conference on Emerging Network Experiment and Technology, CoNEXT 2008, Madrid, Spain, December 9-12, 2008*, page 12, 2008.

[5] J. Choi, K. Hu, and A. Sim. Relational dynamic bayesian networks with locally exchangeable measures. *LBNL Technical Report, LBNL-6341E*, 2013.

[6] M. Dusi, A. Este, F. Gringoli, and L. Salgarelli. Using GMM and svm-based techniques for the classification of ssh-encrypted traffic. In *Proceedings of IEEE International Conference on Communications, ICC*, pages 1–6, 2009.

[7] R. Fontugne, P. Borgnat, P. Abry, and K. Fukuda. Mawilab: Combining diverse anomaly detectors for automated anomaly labeling and performance benchmarking. In *Proceedings of the 6th International COnference*, Co-NEXT '10, pages 8:1–8:12, 2010.

[8] P. Garcia-Teodoro, J. E. Díaz-Verdejo, G. Maciá-Fernández, and E. Vázquez. Anomaly-based network intrusion detection: Techniques, systems and challenges. *Computers & Security*, 28(1-2):18–28, 2009.

[9] L. Khan, M. Awad, and B. Thuraisingham. A new intrusion detection system using support vector machines and hierarchical clustering. *The VLDB Journal*, 16(4):507–521, Oct. 2007.

[10] B. Krishnamurthy, S. Sen, Y. Zhang, and Y. Chen. Sketch-based change detection: Methods, evaluation, and applications. In *Proceedings of the 3rd ACM SIGCOMM Conference on Internet Measurement*, IMC '03, pages 234–247, 2003.

[11] K. Leung and C. Leckie. Unsupervised anomaly detection in network intrusion detection using clusters. In *Proceedings of the Twenty-eighth Australasian Conference on Computer Science - Volume 38*, ACSC '05, pages 333–342, Darlinghurst, Australia, Australia, 2005. Australian Computer Society, Inc.

[12] Z. Liu, A. Manousis, G. Vorsanger, V. Sekar, and V. Braverman. One sketch to rule them all: Rethinking network flow monitoring with univmon. In *Proceedings of the 2016 conference on ACM SIGCOMM 2016 Conference, Florianopolis, Brazil, August 22-26, 2016*, pages 101–114, 2016.

[13] G. A. Mills-Tettey, A. Stentz, and S. B. Dias. The dynamic hungarian algorithm for the assignment problem with changing costs. Technical report, Carnegie Mellon University, East Lansing, Michigan, July 2007.

[14] E. E. Papalexakis, A. Beutel, and P. Steenkiste. Network anomaly detection using co-clustering. In *Proceedings of the 2012 International Conference on Advances in Social Networks Analysis and Mining (ASONAM 2012)*, ASONAM '12, pages 403–410, 2012.

[15] L. Portnoy, E. Eskin, and S. Stolfo. Intrusion detection with unlabeled data using clustering. In *In Proceedings of ACM CSS Workshop on Data Mining Applied to Security (DMSA-2001*, pages 5–8, 2001.

[16] F. Rgringoli, L. Salgarelli, M. Dusa, N. Cascarano, F. Risso, and k claffy. Gt: picking up the truth from the ground for internet traffic. *ACM SIGCOMM Computer Communication Review*, 39(5), October 2009.

[17] M. Yu, L. Jose, and R. Miao. Software defined traffic measurement with opensketch. In *Proceedings of the 10th USENIX Conference on Networked Systems Design and Implementation*, nsdi'13, pages 29–42, 2013.