Denial of Service Detection & Mitigation Scheme using Responsive Autonomic Virtual Networks (RAvN)

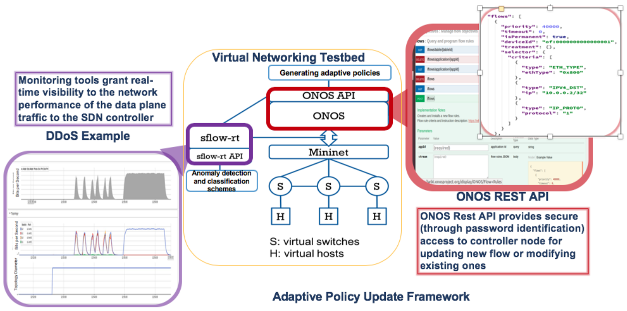
Allen Starke, Zixiang Nie, Morgan Hodges, Corey Baker, and Janise McNair

Abstract—**In this paper, we propose a responsive autonomic and data-driven adaptive virtual networking framework (RAvN) that integrates the adaptive reconfigurable features of a popular SDN platform called open networking operating system (ONOS), the network performance statistics provided by traffic monitoring tools such as T-shark or sflow-RT, and analytics and decision making skills provided from new and current machine learning techniques to detect and mitigate anomalous behavior. For this paper, we focus on the development of novel detection schemes using a developed Centroid-based clustering technique and the Intragroup variance of data features within network traffic (C.Intra), with a multivariate gaussian distribution model fitted to the constant changes in the IP addresses of the network to accurately assist in the detection of low rate and high rate denial of service (DoS) attacks. We briefly discuss our ideas on the development of the decision-making, and execution component using the concept of generating adaptive policy updates (i.e. anomalous mitigation solutions) on-the-fly to the ONOS SDN controller for updating network configurations and flows. In addition, we provide the analysis on anomaly detection schemes used for detecting low rate and high rate DoS attacks versus a commonly used unsupervised machine learning technique Kmeans. The proposed schemes outperformed Kmeans significantly. The multivariate clustering method and the intragroup variance recorded 80.54% and 96.13% accuracy respectively, while Kmeans recorded 72.38% accuracy.**

Index Terms—**machine learning, software-defined networks**

I. Introduction

THE interconnected world is evolving exponentially. Devices connected to the internet are becoming pervasive and/or intelligent. Serving a wide range of devices requires the interconnected system to maintain heterogeneous network resource constraints (i.e networking protocols, consistent available bandwidth, low latency requirements, etc.) to function properly. On the other hand, the increased interconnected state of the world brings an increased opportunity for malicious users to manipulate network resources to deny service to or from various interconnected nodes or networks. Successful management for large-scale heterogeneous networks requires that the network be robust and adaptive to maintain a productive and reliable networking environment, especially during faults or cyber attacks. In recent past to present day, manual reconfiguration was a response to cyber attacks, but it incurs a significant delay between the detection and healing (reconfiguration) of the network. Software-oriented networks provide the potential of realizing automated networks that can detect and mitigate faults or cyber attacks within the system.


Monitoring tools grant real-time visibility to the network performance of the data plane traffic to the SDN controller
DDoS Example
Bits per Second
Bits per Second
Topology Diameter
Virtual Networking Testbed
Generating adaptive policies
ONOS API
ONOS
sflow-rt
sflow-rt API
Anomaly detection and classification schemes
Mininet
S
S
S
H
H
H
S: virtual switches
H: virtual hosts
ONOS REST API
ONOS Rest API provides secure (through password identification) access to controller node for updating new flow or modifying existing ones
Adaptive Policy Update Framework

```
"flows": [
  {
    "priority": 40000,
    "timeout": 0,
    "isPermanent": true,
    "deviceId": "of:0000000000000001",
    "treatment": {},
    "selector": {
      "criteria": [
        {
          "type": "ETH_TYPE",
          "ethType": "0x800"
        },
        {
          "type": "IPV4_DST",
          "ip": "10.0.0.2/32"
        },
        {
          "type": "IP_PROTO",
          "protocol": "1"
        }
      ]
    }
  },
```

Fig. 1. Responsive Autonomic Virtual Network (RAvN) Framework

Software-defined networking (SDN) is the evolving option for managing the future standard of dynamically evolving networks versus traditional static networking systems. Recent research has focused on integrating machine intelligence and machine learning (ML) into SDN networking architectures. Data-driven network (DDN) are proposed in [1] to automatically tune routing algorithms and protocols based on decisions deduced from the data collected in real time. A knowledge plane is proposed in [1] to process data from network performance statistics and make intelligent actions to manage the network.. Knowledge-centric networking is proposed in [2] where ML can extract useful information from sensors at the edge of an Internet-of-Things (IoT) network to reduce the burden in the core of the network. The authors of proposed cognitive networks that explicitly rely on learning and data gathering to adapt the configurable parameters of a network. As discussed in recent research, e.g., [3], a crucial component needed in realizing the vision of the intelligent adaptive network is a feedback signal. In the case of SDN, a signal tied to the SDN controller's northbound interface that relays QoS statistics of the connected host's applications.

In general, developing an adaptive network involves three main components including detection and efficiently categorizing traffic between significant anomalous changes in the network environment and normal fluctuations considered to be anomalous;using decision-making to develop optimal solutions to mitigate anomalous behavior without effecting performance in the rest of the network; and a northbound feedback interface

Starke, Nie, Hodges, and McNair are in the Department of Electrical and Computer Engineering at the University of Florida, Gainesville, FL USA. E-mail: {allen1.starke,znie,morganliam,}@ufl.edu and mcnair@ece.ufl.edu

Baker is in the Department of Computer Science at the University of Kentucky, Lexington, KY USA. e-mail: baker@cs.uky.edu

to trigger execution of adaptive update policies [4], [5], [6], [7].

In this paper, we propose a responsive autonomic data-driven adaptive virtual networking framework (RAvN) that integrates the adaptive reconfigurable features of a popular SDN platform called open networking operating system (ONOS), network performance statistics provided by traffic monitoring tools, and analytics and decision making skills provided from machine learning techniques. The rest of the paper is structured as follows: section II provides related works, and section III dives into detail of our proposed anomaly detection scheme and description of our RAvN architecture. Section IV and V include the experimental setup and results with discussions. Section VI and VII are the background information and conclusions respectively.

II. Related Work

In the literature, ML methods have been used within a cross-layer security framework to monitor for malicious behavior. Methods for monitoring traffic in SDNs have included Deep Packet Inspection, Support Vector Machines, Neural Networks, and Decision Trees [8], [9]. In [9], behavior was classified using a Maximum Likelihood (MLE) approach. However, this approach, and most supervised machine learning methods in general assume that every possible class and the distribution of possible samples for each of these classes are appropriately characterized by training data.

The author of [10], [11] utilized supervised machine learning approaches, using SVM and neural network classifiers for classifying and preventing network security attacks in an SDN architecture. Authors of [12] use machine learning to predict the most vulnerable host that could be attacked in the SDN environment. Using supervised machine learning methods to provide security in SDN can be shunned as not a realistic approach since network traffic will not have labels for training the supervised mode and bad data in the training sample will have a negative impact in detection and classification performance. The authors of [13] utilize security policies on the SDN controller for segment routing in the presence of anomalies. These security policies are initialized during the network setup phase and do not change while the network is operating.

Recently, application-specific QoS management has become more significant. Authors of [14] designed a method to link IP addresses in DNS responses to application names derived from the OS, to classify the applications running on the host machines. Authors imply integrating this with QoS systems in SDNs to control bandwidth allocation for services at the application level. The work [3] translate network metrics into an application metric for the idea of having a structured and extensible connection between applications and the SDN controller. Most application-aware networking research, such as [3], [14], only hint at how information can be shared with an SDN from the data plane for network re-configuration leaving the realization as future work.

Our contributions in this paper are as follows:

- Most supervised machine learning assume that every possible class and the distribution of possible samples for each of these classes are appropriately characterized by training data. We remove this assumption and implement a system that can adapt to changes in communication behavior based on cross-feature information and can robustly detect and classify anomalous communication packets in real time. The intragroup variance method works well with any number of selected features from the network traffic (i.e. meaning it does not require implementing methods such as PCA for feature selection).
- Recent research on application-aware networking focus on developing methods to detecting and classify flows of greedy bandwidth applications for the purpose of QoS management and only hint at how the information can be shared with an SDN for re-configuration. Our work establishes the necessary feedback loop successfully updating new policies to ONOS SDN controller northbound interface for network re-configuration (i.e. flow updates, re-routing, network QoS management, etc.) based on analysis of gathered network performance statistics.
- Unlike previous works, we generate adaptive policies based on results from our proposed clustering and intragroup variance anomaly detection scheme, and utilize the connection to ONOS rest API for dynamic mitigations to the anomalous behavior.

III. Anomaly Detection & Mitigation Schemes

Currently, unsupervised machine learning techniques are used to detect changes in unlabelled real-time network traffic. (Unsupervised machine learning, denial of service and other background definitions are provided in Section VI.) However, DoS attacks are becoming more intelligent and are able to disguise themselves as normal traffic. Networks can no longer rely on unsupervised machine learning techniques which use distance calculations, such as clustering, to identify anomalous changes within the network.

A. Slow Rate DoS Detection Scheme

Through observational analysis, attacked network traffic demonstrates similar patterns across all of it's network traffic features. Selected feature sets in the attacked network traffic demonstrated the same or similar values as previous packets transmitted in succession. Taking advantage of this, detection schemes that are able to determine how closely related or sparse the feature values of the current packet under investigation is compared to past packets transmitted can detect slow-rate DoS attacks. For this reason, a sliding window-based anomaly detection scheme utilizing the optimal number of cluster groups within a sample dataset and the intragroup variance between the selected network traffic features,C.Intra, is explored for identifying sparse or closely related network traffic.

1) Clustering Technique: The simple clustering technique compares the variance and mean of a sample data set to optimally cluster single array data. If the variance of the sample is greater than the mean, then we split the sample into two parts using the mean as the point of separation; providing two new groups of elements. We continue this process for each

Algorithm 1: C.Intra method - Finding the intragroup variance and average cluster number of each window

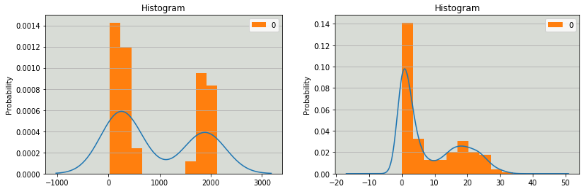
```
windowCluster = []
windowVariance = []
testData = ”input network traffic data”
normData = normalize(testData)
for i in range(length(testData)) do
    tempList = testData[:,i]
    clusters = []
    tempChunks = split(tempList,4)
    for j in range(length(tempChunks)) do
        x = tempChunks[j]
        clusNum = cluster(x)
        clusters.append(clusNum)
    end for
end for
for i in range(length(normData)) do
    tempList = normData[:,i]
    windowV = []
    tempChunks = split(tempList,4)
    for j in range(length(tempChunks)) do
        x = tempChunks[j]
        var = variance(x)
        windowV.append(var)
    end for
    windowCluster.append(clusters)
    windowVariance.append(windowV)
end for
windowClusterT = transpose(windowCluster)
windowVarianceT = transpose(windowVariance)
averageWindowCluster = mean(windowClusterT)
IntraV = mean(windowVarianceT)
Result: average window cluster, intragroup variance
```

new group, and stop when the variance of each cluster group is lower than the mean of the respective group. The scheme provides the optimal number of clusters within the sample.

2) Intragroup Variance: The intragroup variance is calculated using equation 4 and the process is shown in Algorithm 1. First steps include, normalizing the original data set and separating the network traffic into groups using the sliding window. For each network feature (i.e the columns in the matrix data set) the variance is calculated and stored.. Taking the average between the stored variances of each network feature results in the intragroup variance of the network traffic within the sliding window. Sample data sets with low intragroup variances demonstrate closeness of the data within the sliding window.

B. High Rate DoS Detection Scheme

Most high-rate DoS attacks involve flooding network packets to the victim node in-order to reduce available bandwidth and disconnect them from the rest of the network. During high-rate DoS attacks, there is a halt to alternating source and destination IP addresses (i.e. IP addresss for both source and destination repeat for a huge batch of successive packets captured).



(c) (d)

Fig. 2. (a) Source IP (window size = 2000) (b) Source IP (window size = 40). The histogram demonstrates the probability of change in subsequent network packet IP addresses for different window sizes.

1) Transforming IPs to Binary Options: For the purpose of detecting high-rate DoS, we adopt a technique used in [15] to transform the source and destination IP addresses into binary options.

$$I_{n,p} = \begin{cases} 0, & \text{if } P_n = P_n - 1. \\ 1, & \text{if } P_n \neq P_n - 1. \end{cases} \tag{1}$$

As states in equation 1, if the IP address is repeated in succession then we append a ’0’. When the IP address of the current packet changes from the previous packet then we append a ’1’).

2) Multivariate Gaussian Distribution Model: Normal gaussian distribution is the most commonly recognized distribution observed in most processes. Single and multivariate gaussian distribution is defined in the equations below [16], [17]:

$$N(x;\mu,\sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-(x-\mu)^2/2\sigma^2} \tag{2}$$

where μ represents the mean and σ^2 represents the variance. Adding to this equation for multivariate Gaussian distributions we have [17]:

$$p(x) = \sum_{j=1}^{k} \phi_j N(x;\mu_j,\Sigma_j) \tag{3}$$

where ϕ_j is the assigned weight of the respective Gaussian (i.e. strength of Gaussian curve) and Σ_j is the covariance matrix. The next step is splitting the array into user specified sliding-window sizes. Throughout our experiments the sliding-window size was varied to find the optimal window size that provided the best results. Recording the occurrence of the ’0’ value for each window segment and plotting the distribution resulted in the histogram, shown in figure 2 for the source IP (destination IP distribution model followed similar trend as source IP model). Increasing the window size assisted in creating a cleaner separation for the multivariate Gaussian distribution, while decreasing the window size resulted in the two peaks mixing making it harder to distinguish. Fitting these histograms to multimodal gaussian distribution provided the parameters shown in table I.

TABLE I
MULTIVARIATE GAUSSIAN DISTRIBUTION PARAMETERS (WINDOW = 40)

Gaussian Multivariate Parameters	Mean	Variance	Weight
Source IP	1.281	2.303	0.5504
	17.09	44.3	0.4496
Destination IP	1.477	3.102	0.5784
	23.61	54.77	0.4216

TABLE II
ANOMALY DETECTION PERFORMANCE EVALUATION

Anomaly Detection Schemes	TPR	TNR	PPV	ACC
C.Intra Method	33.49%	98.89%	96.79%	96.12%
Multivariate Gaussian Clustering	43.42%	88.91%	79.66%	80.54%
Kmeans+norm+PCA	34.29%	95.47%	73.97%	76.28%
Kmeans	35.78%	88.41%	75.54%	73.38%

C. Responsive Autonomic Virtual Networking Framework

To develop an autonomous adaptive SDN the three major components (i.e detection, decision-making, and execution) must be realized and put in a coherent system. The detection component with some of the novel clustering detection schemes is presented in the previous sub-sections. To develop the other two components; monitoring of the data plane network, analysis of the network performance statistics, and a reliable feedback loop to the SDN controller must be in-place.

1) Decision-making: It is proven in recent work [18] that supervised deep learning is the best option in accurately categorizing network traffic based on which type of cyber attack or fault is occurring in the network environment. Raw telemetry data from communication networks is usually not labeled [19], therefore, it is often necessary to use unsupervised algorithms in networking applications. Implementing a hybrid unsupervised and supervised machine learning solution grants the opportunity of generating optimal mitigations, created by automation of javascripts and python shell scripts, to attacks. In our case, adaptive policy updates are generated with the primary task of disconnecting the attack nodes from the victim. Other secondary task can be implemented to re-route the traffic of the victim node to its destination using segment routing. Network performance statistics can be monitored from the data plane using the ONOS SDN internal packet processors, or third party entities such as T-shark and sflow-RT [20].

2) Execution: The feedback loop to the SDN controller is the most crucial piece of the puzzle. Without it there are no adaptive updates sent to the controller autonomously, resulting in the state of the network remaining the same as the initial configuration. Adaptive policies can be sent to the ONOS northbound interface using javascript to connect to the rest api provided from the open-source networking platform [21]. Devices connected to the SDN southbound interface (i.e. data plane) are open networking technologies, such as open virtual switch (OVS), that support utilizing virtual networking components and grants SDN controller entities access to reprogramming packet processing, and handling capabilities using a network programming language ”programming protocol-independent packet processors” (P4).

IV. PERFORMANCE ANALYSIS

The proposed RAvN environment was simulated in Mininet. Developers included classes that support the inclusion of wireless devices in the Mininet environment [22], so a heterogeneous wired and wireless networking architecture can be studied. In addition, we deployed an instant virtual network on a stand-alone computer and were able to expand this network by allowing the connection of multiple external nodes and other computation resources, including other PCs, mobile devices, VMs, etc. The SDN controller used was the open-source network operating system ONOS [21]. For experimentation purposes, the cluster algorithms were trained and tested using the Intrusion Detection Evaluation Dataset (CICIDS2017) developed by the Canadian Institute for Cybersecurity. This dataset contains realistic network cyber attacks generated from the top automated cyber attack tools available [18]. The IDS data set uses CICFlowMeter, a network traffic flow generator and analyzer used to generate bidirectional flows. More than 80 statistical network traffic features such as Duration, Number of packets, Number of bytes, Length of packets, etc. are recorded for the forward and backward packet transmissions [23], [24]. Network traffic and performance statistics collected by third party monitoring tools are stored in a database (InfluxDB, AWS, etc.) for analysis. For comparison the popular unsupervised machine learning technique, Kmeans, is implemented. In previous works [25], Kmeans has proven to be reliable on old datasets emulating intrusions on military networks [26].

A. Evaluation Methods

The size of the data set was n = 172,785 and the size of the training and testing set was n = 86,568, and 86,199 respectively. The network traffic consisted of 4 types of traffic including benign, DoS slowloris, DoS slowhttp, and DoS hulk. We recorded true positive (TP), true negative (TN), false positive (FP), and false negative (FN) results. Using these values we calculated true positive rate: $TPR = \frac{TP}{(TN+FP)}$(sensitivity), true negative rate: $TNR = \frac{TN}{(TN+FP)}$(specificity), positive predictive value: $PPV = \frac{TP}{(TP+FP)}$(precision), and overall accuracy: $ACC = \frac{TP+TN}{(TP+FP+FN+TN)}$.

V. NUMERICAL RESULTS

For this work, we recorded the accuracy of the proposed algorithms, and compared them to the most commonly used unsupervised machine learning algorithm k-means. We also demonstrate the impact of our anomaly detection and mitigation framework.

A. Clustering Schemes Performance Comparison

As can be seen in table II, the accuracy's of the proposed algorithms outperformed the popular k-means algorithm. Both k-means clustering algorithms (i.e. k-means, k-means+norm+PCA) achieved an accuracy of 73.37% and

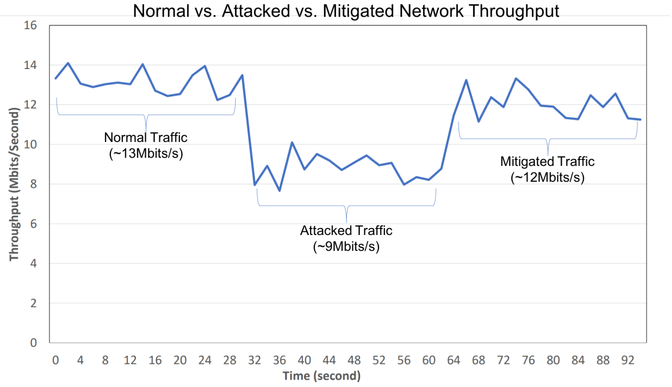


Fig. 3. Network throughput during normal, DDoS attack, and mitigated network traffic

76.28% respectively. The k-means algorithm clusters data points by alternatively assigning data points to clusters and updating cluster representatives. Data points are assigned to the cluster in which they have the minimum Euclidean distance from the cluster centers. The networking data set produced by replicating state-of-the-art DoS attacks on normal traffic proves to be a challenge for the popular unsupervised learning approach of implementing kmeans+normalization+PCA. Another downfall of k-means algorithm is information on the dataset must be known prior to execution (i.e. the optimal number of clusters in the data is needed).

The multivariate clustering method recorded 80.54% accuracy rating. This method is very good at detecting consistencies in IP addresses (i.e. no changes in source or destination IP addresses from previous packets). For this reason, this method has a high percent chance of detecting high rate DoS attacks such as ”DoS hulk” which floods large amounts of packets to victim devices. This method is also good for detecting greedy-bandwidth applications which portray similar consistent characteristics in the source and destination IP address, thus resulting in false alarms being triggered.

The clustering intragroup variance method, C.Intra, performed the best out of the three with 96.13% accuracy due to the fact that DoS traffic portrays similar or identical feature characteristics in successive packet transmissions. C.Intra is good at detecting these window segments of successive network features that tend to have a small deviation or repeat themselves from previous packets. Signs of DoS included low average cluster numbers accompanied by a very low or consistent intragroup variance. In our case a sequence of values < 2.0 with a slight deviation of $\pm$ 0.175 between values, whilst their respective intragroup variance values $<$ 0.01. Even though this method performed the best, it still has a challenging time with detecting attacked network traffic that is not successive and evenly distributed within normal traffic (i.e. single packet attacks aimed at disrupting the application layer). From our knowledge and through reviewing this data set, only injection-based cyber attacks demonstrate the characteristics of the challenging network traffic, and will be addressed in future work.

B. *Impact of attack detection & mitigation framework*

In this section, we analyze the impact of the attack detection strategy proposed in this paper on network performance, using the Mininet-WiFi network simulator. In this simulation, the controller monitors the node and flows within the network, extracts the features needed by the *RavN* to detect attacks, generate the mitigation solution and enforce the mitigation mechanism by refreshing the flow tables in switches. The simulation environment consists of $N = 100$ nodes randomly placed in an area of 100×100 meters with each node having a range of $R = 10$ meters.

We simulate a targeted cyber attack on 5 nodes in the network by sending large number of TCP SYN packets to the attacked nodes to drain their resources. By using the *RavN* framework we were able to detect this TCP cyber attack and request the controller to take an action by starting the mitigation mechanism. The controller limits the flow of TCP SYN packets to the attacked nodes, which improves the network performance considerably. During the attack, the average throughput of the network initially decreased by 32% but due to the *RavN*-initiated response, it was increased by 24%. It is noted the impact on the network traffic from the cyber attack spanned 30-50 seconds before the mitigation process took effect, indicating a quick response to the anomalous behaviour.

VI. Background Information

A. *Network Cyber Attacks*

Today, there are many automated tools developed to carry out a plethora of cyber attacks on target servers, such as Low Orbit Ion Canon (LOIC), High Orbit Ion Canon (HOIC), Hulk, GoldenEye, Slowloris, Slowhttptest, and Damn Vulnerable Web App (DVWA) [18]. There are two common types of DoS attacks including low-rate DoS and high-rate DoS aimed at depleting different resources of the victim. In low-rate DoS, attackers attempt to exhaust memory resources of victim devices by sending few packets to attack the application level directly. In high-rate DoS, attackers attempt to exhaust available bandwidth by flooding multiple packets from ”bots” to the victim [27].

B. *Unsupervised Machine Learning & Clustering Techniques*

In implementation, it is infeasible to assume that all possible behaviors can be identified and characterized in training data prior to implementation of a system malicious attacks and their associated behaviors on the communication grid can and will be re-imagined and re-implemented. Thus, a system is needed that can adapt to changes in communication behavior based on cross-layer information and can robustly detect and classify anomalous communication packets in real time.

Clustering algorithms are catergorized as one of the four types including connectivity, centroid, distribution, and density based-models. For the purpose of this work, we adopt the centroid and distribution model of clustering for DoS detection, and we compare our new scheme to the centroid-based K-means approach computed using the equation: $d(\mathbf{p}_n, \mathbf{c}_k) =$

$\left(\sum_{d=1}^{D}(\mathbf{p}_n - \mathbf{c}_k)^2\right)^{1/2}$ where $\mathbf{p}_n = [p_1, p_2, \ldots, p_D] \in R^D$ is a D dimensional vector containing the the features associated with the n^{th} data point (in our case, a network packet) and $\mathbf{c}_k = [c_1, c_2, \ldots, d_D] \in R^D$ is a D dimensional vector of the k^{th} cluster representative.

C. Variance & IntraGroup Variance

The variance (σ^2) is the expectation of the squared deviation of a random variable from its mean (μ): $\sigma^2 = \frac{\Sigma(X-\mu)^2}{N}$. The variance is proportional to the scatter of the data, i.e., it is small when the data set is clustered together, and large when the data set is widely scattered. Intragroup variance, also known as within-group variance or intracluster variance, refers to variations caused by differences within individual cluster groups. The intragroup variance, S_p^2 or σ_p^2, calculates the variance of each individual cluster/group then finds the average, as shown in Equation (4) [28].

$$S_p^2 = \frac{\Sigma(y_{i1} - \mu_1)^2 + \Sigma(y_{i2} - \mu_2)^2 + ... + \Sigma(y_{ig} - \mu_g)^2}{N - g} \tag{4}$$

where y_{i1} represents each observation in a group, μ_i represents the mean for that group, N is the sample size and g is the number of groups.

VII. Conclusion

In conclusion, the benefits of implementing our proposed responsive autonomic virtual network (RAvN) architecture in place of the common networking architectures or in place of traditional SDN strategies has been discussed. It was discussed how implementing the RAvN architecture can provide secure communications, detection and mitigation of cyber attacks, and additional resilience in the presence of faulty or attacked data or communication nodes within the network. The paper also evaluated the use of various clustering methods. We've experimented with the popular unsupervised machine learning k-means algorithm for detecting DoS anomaly within realistic network traffic. We've experimented with clustering using a multivariate Gaussian distribution model fitted to the constant changes in the IP addresses of the network, and we also introduce a new method of detecting DoS attacks within network traffic using a simple clustering algorithm and intragroup variance (C.Intra). The purposed intragroup variance and multivariate gaussian methods out perform the common k-means method by 19.84% and 4.26% respectively.

Acknowledgment

This work was supported by the National Science Foundation under Grant Number 1809739 and by the University of Florida/Harris Corporation Excellence in Research Fellowship.

References

[1] C. R. David Clark, Craig Partridge and J. Wroclawski, “a knowledge plane for the internet”,” in *Proc. ACM SIGCOMM, 2003*, 2003, pp. 3–10.

[2] D. W. et. al, “vision and challenges for knowledge centric networking (kcn)”,” in *[Online]. Available: https://arxiv.org/abs/1707.00805*, 2017.

[3] H. Jahromi and D. Delaney, “An application awareness framework based on sdn and machine learning: Defining the roadmap and challenges,” in *2018 10th International Conference on Communication Software and Networks (ICCSN)*, July 2018, pp. 411–416.

[4] A. Leinwand and K. Conroy, “”network management: A practical perspective”,” in *Addison-Wesley*, 1996.

[5] W. Stallings, “”snmp, snmpv2, and rmon: Practical network management”,” in *Addison-Wesley*, 1996.

[6] M. Subramanian, “”network management: Principles and practice”,” in *Pearson Education*, 2010.

[7] W. K. et. al, “”adaptable and data-driven softwarized networks: Review, opportunities, and challenges”,” in *IEEE Journal proceedings*, 2018.

[8] P. A. et. al., “Machine learning in software defined networks: Data collection and traffic classification,” in *2016 IEEE 24th International Conference on Network Protocols (ICNP)*, 2016.

[9] F. et. al., “Cross-layer security framework for smart grid: physical security layer.”

[10] N. Meti, D. G. Narayan, and V. P. Baligar, “Detection of distributed denial of service attacks using machine learning algorithms in software defined networks,” in *2017 International Conference on Advances in Computing, Communications and Informatics (ICACCI)*, Sept 2017, pp. 1366–1371.

[11] siddharth Gangadhar and J. Sterbenz, “Machine learning aided traffic tolerance to improve resilience for software defined networks,” in *2017 9th International Workshop on Resilient Networks Design and Modeling (RNDM)*, 2017.

[12] S. Nanda, F. Zafari, C. DeCusatis, E. Wedaa, and B. Yang, “Predicting network attack patterns in sdn using machine learning approach,” in *2016 IEEE Conference on Network Function Virtualization and Software Defined Networks (NFV-SDN)*, Nov 2016, pp. 167–172.

[13] e. a. Varadharajan, Vijay, “A policy based security architecture for software defined networks..” Cornell University, 2018.

[14] N. Huang, C. Li, C. Li, C. Chen, C. Chen, and I. Hsu, “Application identification system for sdn qos based on machine learning and dns responses,” in *2017 19th Asia-Pacific Network Operations and Management Symposium (APNOMS)*, Sept 2017, pp. 407–410.

[15] A. Krasnov, “”detecting ddos attacks using the analysis of network traffic as dynamical system”,” 2018.

[16] Y. L. Liu, R., “”kernel estimation of multivariate cumulative distribution function”,” in *Journal of Nonparametric Statistics*, 2008.

[17] M. Deshpande, “”clustering with gaussian mixture models”,” in *https://pythonmachinelearning.pro/clustering-with-gaussian-mixture-models/*.

[18] I. S. et. al, “”toward generating a new intrusion detection dataset and intrusion traffic characterization”,” in *4th International Conference on Information Systems Security and Privacy*, 2018, pp. 108–116.

[19] D. Cote, “”using machine learning in communication networks [invited]”,” in *Journal for Communication Networks*, 2018.

[20] (2018, April) sFlow-RT. [Online]. Available: https://sflow-rt.com/

[21] “ONOS-Open Network Operating System,” https://wiki.onosproject.org/, March 2018, last accessed: March 4, 2018.

[22] R. R. Fontes, S. Afzal, S. H. Brito, M. A. Santos, and C. E. Rothenberg, “Mininet-wifi: Emulating software-defined wireless networks,” in *11th International Conference on Network and Service Management (CNSM)*. IEEE, 2015, pp. 384–389.

[23] A. L. et. al, “”characterization of tor traffic using time based features”,” in *3rd International Conference on Information System Security and Privacy, SCITEPRESS*, 2017.

[24] G. G. et. al, “”characterization of encrypted and vpn traffic using time-related features”,” in *nd International Conference on Information Systems Security and Privacy (ICISSP 2016)*, 2016, pp. 407–414.

[25] A. Starke, J. McNair, R. Trevizan, A. Bretas, J. Peeples, and A. Zare, “Toward resilient smart grid communications using distributed sdn with ml-based anomaly detection,” in *IFIP Conference on Wired/Wireless Internet Communications*, vol. 1. IFIP, 2018, pp. 1–12.

[26] “KDD Cup 1999 Data,” http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html, March 1999, last accessed March 4, 2018.

[27] K. J. S. . T. De, “”mathematical modelling of ddos attack and detection using correlation”,” in *Journal of Cyber Security Technology*, 2017, pp. 175–186.

[28] G. Singh, “”a simple introduction to anova”,” in *https://www.analyticsvidhya.com/blog/2018/01/anova-analysis-of-variance/*, 2018.