Toward Resilient Smart Grid Communications Using Distributed SDN with ML-Based Anomaly Detection

Allen Starke(✉), Janise McNair, Rodrigo Trevizan, Arturo Bretas, Joshua Peeples, and Alina Zare

University of Florida, Gainesville, FL 32611, USA
{allen1.starke,rodtrevizan,jpeeples}@ufl.edu,
{mcnair,arturo,azare}@ece.ufl.edu

Abstract. Next generation "Smart" systems, including cyber-physical systems like smart grid and Internet-of-Things, integrate control, communication and computation to achieve stability, efficiency and robustness of physical processes. While a great amount of research has gone towards building these systems, security in the form of resilient and fault-tolerant communications for smart grid systems is still immature. In this paper, we propose a hybrid, distributed and decentralized (HDD) SDN architecture for resilient Smart Systems. It provides a redundant controller design for fault-tolerance and fail-over operation, as well as parallel execution of multiple anomaly detection algorithms. Using the k-means clustering algorithm from the machine learning literature, it is shown that k-means can be used to produce a high accuracy (96.9%) of identifying anomalies within normal traffic. Furthermore, incremental k-means produces a slightly lower accuracy (95.6%) but demonstrated an increased speed with respect to k-means and fewer CPU and memory resources needed, indicating a possibility for scaling the system to much larger networks.

Keywords: Software defined networks · Anomaly detection
Machine learning · Security · Resilience

1 Introduction

The next-generation power grid, named the Smart Grid, has drawn the attention of academia, industry and government agencies due to the great impact of such systems on the distribution of power within and between various regions. These next generation systems integrate control, communication and computation to achieve stability, efficiency and robustness of the physical processes. While a great amount of research has addressed these objectives, science and technology related to secure SG communications is still relatively immature. Additionally,

K. R. Chowdhury et al. (Eds.): WWIC 2018, LNCS 10866, pp. 83–94, 2018.
https://doi.org/10.1007/978-3-030-02931-9_7

many critical cyber-physical infrastructures are increasing dependency of control of physical processes on communication networks, thus becoming exposed to various cyber-threats or faulty operation. An example is the network of smart grid subsystems shown in Fig. 1. The smart grid subsystems are local agents, composed of Distributed Energy Storage Systems (DESS), such as flywheels and grid-connected batteries, a Synchronous Generator, a Phasor Measurement Unit (PMU) and a Distributed State Estimator (DSE). The PMUs in the smart grid form a communication network, where smart grid data is regularly exchanged with other PMUs to coordinate and analyze energy performance measures. In the attack model considered, a PMU subsystem may be attacked in order to disable it or to inject faulty data; or a PMU element may fail, generating faulty data as a result of the failure. If a PMU subsystem is attacked or fails, the communication of invalid data may introduce substantial errors when exchanged with the other connected PMU subsystems, creating an avalanche effect. We propose to secure the PMU subsystem network by introducing a hybrid distributed and decentralized (HDD) software-defined network (SDN). The HDD-SDN architecture will leverage data and statistics from PMU communications to gain situational awareness and will provide machine intelligence to detect and protect against abnormal network behavior within the distributed PMU communication network. SDN is a networking paradigm in which the forwarding hardware is decoupled from the control decisions. The network intelligence is logically centralized in software-based controllers (the control plane), and the network devices become simple packet forwarding devices (the data plane) that can be programmed via an open interface [1]. The controller is the only source responsible for determining routing paths, developing policies, partitioning the network, as well as other network administrative functionality. The traditional SDN operation is a centralized, global controller. While the centralized approach does strengthen the capability of the controller to manage the entire network, it is well known that a centralized approach also creates a vulnerability for a single point of failure. This is a significant barrier to using SDN for large-scale cyber-physical networks. New approaches must ensure that SDN controllers are fault tolerant on a larger scale and retain the advantages of the centralized perspective (global view) even while the implementation takes a distributed approach.

1.1 Related Work and Contributions of This Work

To this point, there has been limited research on SDN for monitoring of smart grid communications. Previously, SDN for Smart Grid has been proposed using centralized, non-real-time network monitoring and control. For example, the work of [2], presents a self-healing PMU smart grid network using SDN. When a cyber-attack takes place on a PMU then that node is isolated and reconfigured to a previous stable state. These works do not consider cyber-attacks on the centralized SDN architecture itself nor the restoration and reconfiguration of the SDN controllers in a distributed SDN design. Very recently, distributed SDN (D-SDN) solutions have been proposed for specific categories of Smart Grid, including (PEVs) [3]. A few works proposed a distributed architecture for cyber-

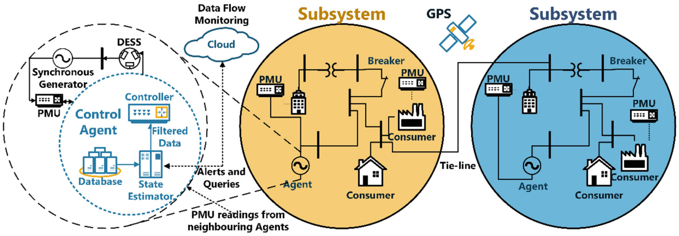


Fig. 1. Distributed PMUs in the Smart Grid. The distributed PMUs form a communication network, where smart grid data is regularly exchanged with other PMUs to coordinate and analyze energy performance measures. If a PMU subsystem is attacked or fails, the communication of invalid data may introduce substantial errors when exchanged with the other connected PMU subsystems.

security [4]. The work in [5], provides detailed insights on various approaches for developing a distributed SDN architecture.

Our proposed HDD-SDN is a real-time, distributed approach that uses current information available at the PMUs and can respond in real-time to failures and attacks. HDD-SDN analyzes traffic flow as well as smart grid measurement data and use machine learning techniques to identify, fix, and then attempt to prevent similar cyber-attacks, while continuing to detect anomalous behavior in the future. Our work will leverage the *physically distributed controller approach* described in [5] and employs machine intelligence-based anomalous flow detection to increase resilience of smart grid systems as well as the underlying SDN communication network.

The rest of the paper is divided as follows. Section 2 introduces the hybrid distributed and decentralized controller software-defined network architecture for smart grid systems. Section 3 describes the execution of multiple anomaly detection algorithms to be able to maintain reliability while reducing latency and CPU/memory use. Section 4 provides the experimental setup, a combination of Smart Grid test bed and SDN network simulation. Section 4 also provides results on the machine learning clustering algorithms that were evaluated for anomaly detection in network traffic and generated PMU data. Finally, Sect. 5 provides conclusions and future work.

2 Network Architecture

As mentioned previously, Fig. 1 shows an example application for the HDD-SDN architecture, namely the control and monitoring of PMU communications in a smart grid system. The problem of protecting and controlling the power grid is reduced into simpler, more tractable engineering problems by subdividing the power system into small regions or zones. For example, a fault in a transmission line or a fault in the control of the power output of a generator are problems solved locally by monitoring the variables measured by local sensors. The proliferation of the fault comes when data exchange begins between the local PMU and other PMUs.

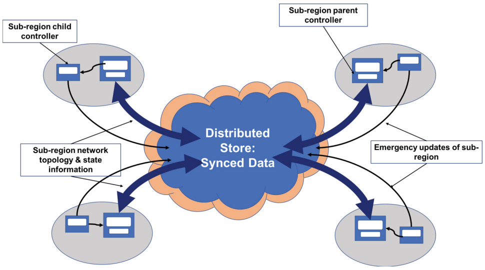

Sub-region child controller
Sub-region parent controller
Distributed Store: Synced Data
Sub-region network topology & state information
Emergency updates of sub-region

Fig. 2. Distributed SDN architecture and store

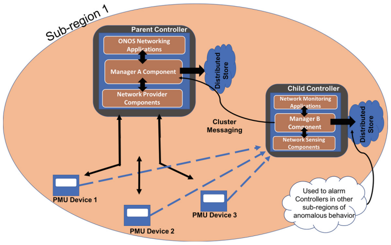

Sub-region 1
Parent Controller
ONOS Networking Applications
Manager A Component
Network Provider Components
Distributed Store
Child Controller
Network Monitoring Applications
Manager B Component
Network Sensing Components
Distributed Store
Cluster Messaging
PMU Device 1
PMU Device 2
PMU Device 3
Used to alarm Controllers in other sub-regions of anomalous behavior

Fig. 3. Sub-region operations - decentralized controller architecture

The proposed HDD-SDN architecture is shown in Fig. 2. The network consists of a collection of sub-regions, which can represent the PMU sub-systems within the smart grid architecture. In the hierarchical architecture, multiple controllers share their sub-region network topology and state information with their neighbors in the distributed system. This is accomplished using the distributed store function of the open-source network operating system ONOS [6]. The distributed store provides a global view of the entire networking system, and ensures sub-systems are performing as efficiently as possible. Each controller is responsible for managing the nodes under its sub-region/domain, and for updating important information from their sub-region to the distributed system. In this sense, the controllers are connected in a mesh using a specific TCP port for interactions and using keep-alive messages to monitor each controller's status. Consistency levels among the users can be "strongly consistent", which in ONOS indicates frequent updates of network topology state to the distributed stores using the RAFT protocol, or "eventually consistent", which implies less frequent updates using the Anti-Entropy protocol [7]. In the case of this project, we use an "eventually consistent" model for updating the distributed store of anomalous behavior taking place in a region. More information about the types of stores used in ONOS can be found here: [7]. Multiple controller support, introduced in OpenFlow 1.2 [8], allows a switch to connect to multiple controllers simultaneously. Each controller has either a MASTER role, EQUAL role, or a SLAVE role to each switch in the network. Next, we describe the decentralized approach that provides the SDN and Smart Grid network with additional benefits compared to a centralized strategy.

2.1 Decentralized SDN Architecture

The sub-region operation for the proposed HDD-SDN architecture is shown in Fig. 3. Child controllers are placed in each sub-region to monitor data and control planes. This provides a redundant, decentralized SDN architecture for each sub-region. The primary tasks of the child controller are to intercept: (1) network data packets relayed in the data plane between PMU devices, and (2) network control packets transmitted from the parent controller to the PMU's switching devices. The child controller cannot make changes to the network it is monitoring. It is only allowed to communicate with the distributed store (hierarchical distributed controller cluster) if its parent controller demonstrates anomalous behavior to the network. If this condition is detected, the child controller can flag the parent's anomalous behavior and relay the indication to its neighboring parent controllers. A neighbor parent controller can then take control of the affected sub-region, and the faulty controller can be reconfigured.

The proposed approach also allows the parent controller to offload security features, such as firewalls, deep packet inspection (DPI), or anomaly detection techniques to the child controller, since the logical decentralized child controller can be aware of all statistics in the network. Delegating jobs to separate controllers releases the burden placed on one controller, and allows room for numerous possibilities for managing or reconfiguring the network. Further details about the parent and child controller roles and operation are shown in Fig. 4 and described below. We depend on network statistics from the entire system to identify anomalous behavior, including (1) anomalous behavior in the network traffic and payload content, i.e. control data sent between the SDN controller and PMU devices, network data with current and voltages reading exchanged between PMU devices and a state estimator, (2) anomalous behavior in topology changes, i.e. additional node changes within the network, such as new addition of external nodes and unwanted changes within network configurations; and (3) a variety of network performance changes, i.e. broken links, lost connections, and performance decreases.

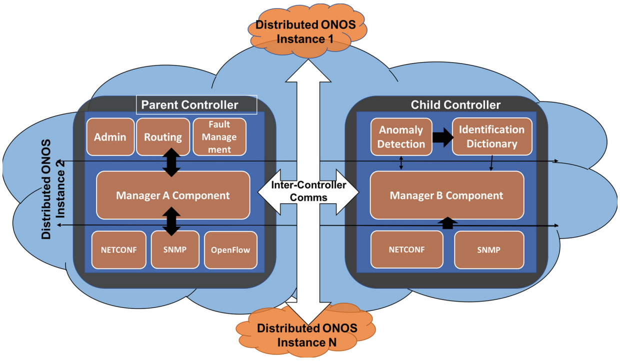
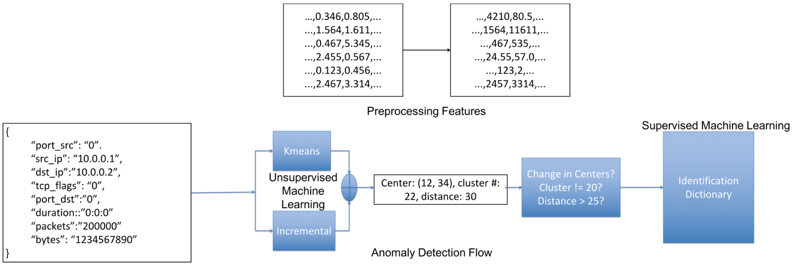


Fig. 4. Decentralized sub-regional architecture details

2.2 Parent Controller

Efficient resource allocation and management takes place on the parent controller to provide congestion control, load balancing, and traffic engineering for the smart grid and to distribute the computation load among the PMUs. The parent controller is also responsible for reconfiguring the network when the child controller demonstrates anomalous behavior in the system. The parent controller interacts with the devices in the region using the Openflow protocol for updating flows, while simple network management protocol (SNMP) and NETCONF are used for gaining statistical information and reconfiguring devices when necessary. Each parent controller of each sub-region allocates enough memory to store the network states of their neighboring regions, allowing for each controller to monitor the network performance of the neighboring regions.



```
...,0.346,0.805,...
...,1.564,1.611,...
...,0.467,5.345,...
...,2.455,0.567,...
...,0.123,0.456,...
...,2.467,3.314,...
```

```
...,4210,80.5,...
...,1564,11611,...
...,467,535,...
...,24.55,57.0,...
...,123,2,...
...,2457,3314,...
```

```
{
     “port_src”: “0”.
     “src_ip”: “10.0.0.1”,
     “dst_ip”:”10.0.0.2”,
     “tcp_flags”: “0”,
     “port_dst”:”0”,
     “duration::”0:0:0”
     “packets”:”200000”
     “bytes”: “1234567890”
}
```

Center: (12, 34), cluster #: 22, distance: 30

Preprocessing Features
Supervised Machine Learning
Kmeans
Unsupervised Machine Learning
Incremental
Change in Centers? Cluster != 20? Distance > 25?
Identification Dictionary
Anomaly Detection Flow

Fig. 5. Example flow diagram of applications in child controller

2.3 Child Controller

As stated previously, the child controller has no ability to change topological information of the region. Its sole purpose is monitoring the status and performance of the parent controller and the devices that form the sub-regions network using management protocols like SNMP, and NETCONF for retrieving information of the system. As indicated in Fig. 2, the child controller contains a few security applications specifically for determining anomalies of the packets sent in the control or data plane. The anomaly detection (AD) module uses a combination of machine learning algorithms for clustering different types of network traffic and payload data, and an identification dictionary (IDD) module. The decentralized architecture grants use of multiple anomaly detection algorithms in parallel, without greatly increasing the CPU and memory usage, as it would if management and anomaly detection were executed on a centralized controller.

Feature Scaling and Preprocessing. The application flow of the child controller is described in Fig. 5. Preprocessing features is a standard step for implementing machine learning algorithms for anomaly detection [9]. However, while most techniques attempt to normalized a wide range of different features into a standard range of 0 to 1 [9], in this approach, we increased the scaling factor by 1000 while incrementing through the columns. This granted a substantial change in the calculation of centers when certain features are present and allowed the clustering algorithm to better distinguish between different cluster groups.

3 Anomaly Detection in the HDD-SDN Child Controller

Various machine learning techniques can be implemented in parallel to detect anomalous behavior. A commonly used clustering algorithm, *k-means*, is implemented, along side a sparse clustering extension called *incremental k-means* [10]. 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 to cluster centers computed using Eq. 1:

$$d(\mathbf{p}_n, \mathbf{c}_k) = \left(\sum_{d=1}^{D} (\mathbf{p}_n - \mathbf{c}_k)^2 \right)^{1/2} \tag{1}$$

where $\mathbf{p}_n = [p_1, p_2, \ldots, p_D] \in R^D$ is a D dimensional vector containing 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. Cluster representatives are using the vector mean of the data points assigned to that cluster. Since the k-means algorithm requires repeated pair-wise distance computations between each data point and each cluster representative, the computational load is significant for large data sets. Incremental k-means is a k-means approximation that is applicable to large scale sparse dynamic datasets (such as network traffic). During each iteration, the standard k-means algorithm uses the entire dataset for recalculating cluster centers. In contrast, Incremental k-means updates the previous centers with only newly input data [10].

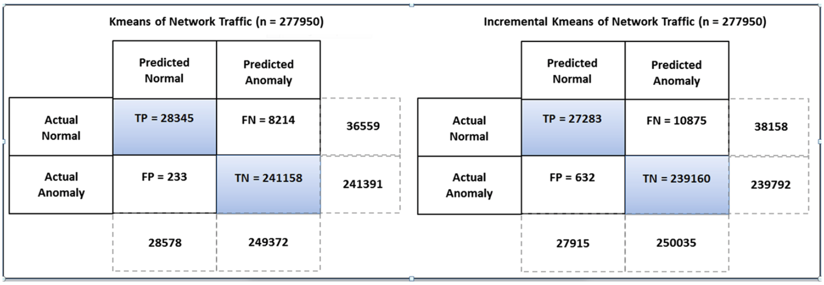
The k-means approach and its extensions have an obvious limitation in that the number of clusters must be known before executing the algorithm. However, the number of clusters are rarely known in advance, particularly for dynamic datasets containing anomalies. Cluster validity metrics are one mechanism to address this issue. Cluster validity metrics provide a quantitative measure of clustering effectiveness for a particular data set. Thus, a data set can be repeatedly clustered with a different number of clusters and, then, each result evaluated using a validity metric to determine the appropriate number of clusters. Validity metrics appropriate for the k-means algorithm include Dunn's index and the Davies-Bouldin index [11,12]. A draw-back of this potential approach is the need to repeatedly cluster the data using a different number of clusters each time. Alternatively, the number of clusters can be adapted in real time with incoming data by generating new clusters when data points appear that are far from all current cluster representatives (similar to approaches used to generate new clusters in Dirichlet Process Mixture Modeling) [13]. These new clusters could then be evaluated and tagged as corresponding to anomalous behaviors when appropriate. When anomalous behavior is identified, short-term mitigation techniques can take place for a quick response to the problem. For example, a firewall can be activated on an infected IP address.

Identification and Detection Dictionary in the HDD-SDN Child Controller. Figure 4 also shows the Identification and Detection Dictionary (IDD) process in the Child Controller. The IDD uses a simple logical algorithm for comparing extracted features from the anomalous traffic to features associated with different types of cyber-attacks as identified during a training phase. Different types of cyber-attacks affect different features of network traffic. If the anomalous behavior is completely different from what is stored and cannot be identified, then the system has learned a new type of cyber-attack or a new type of fault in the system. This information will be recorded and classified as a new anomaly. Then it is sent to the distributed store for the other regions to be able to take action in mitigating that type of attack or fault, or tagging reported data with the current attack or fault. After an anomaly is identified, the ONOS system can now produce long-term changes in the network configuration to prevent future attacks of this type,such as changing flows, reconfiguring the attacked device, rerouting around the offending sub-region, etc.

The extracted features from the anomalous traffic in a sub-region sometimes cannot be directly used in another sub-region. Different regions may have different network topology and configurations. An abnormal traffic pattern in one region may be normal in another region. We envision the child controller to be able to develop mitigation schemes for their respective sub-domains in which they reside. For instance if anomalous behavior is detected in sub-region 1, the child controller (monitor controller) in that region will develop a mitigation scheme for its sub-region only. This information is transmitted to it's respective parent controller to take action. The child controller will then relay information about the type of attac, what network features were affected, feature threshold values to be used by other child controllers in separate sub-regions. It is up to the other sub-regions to monitor their network features and determine if they are being changed in a similar manner. If their thresholds aren't met then no change will take place for the other regions. If their thresholds are met then the child controllers of those regions will develop their own mitigation scheme to correct the anomalous behavior.

4 Experimental Results

The proposed HDD-SDN environment was simulated in Mininet using the wireless devices environment. The SDN controller used was the open-source network operating system ONOS [6]. ONOS provides a large API for users to develop their own networking applications to meet requirements of custom networking scenarios.



Kmeans of Network Traffic (n = 277950)

	Predicted Normal	Predicted Anomaly	
Actual Normal	TP = 28345	FN = 8214	36559
Actual Anomaly	FP = 233	TN = 241158	241391
	28578	249372	

Incremental Kmeans of Network Traffic (n = 277950)

	Predicted Normal	Predicted Anomaly	
Actual Normal	TP = 27283	FN = 10875	38158
Actual Anomaly	FP = 632	TN = 239160	239792
	27915	250035	

Fig. 6. Kmeans and incremental kmeans confusion matrix for network traffic

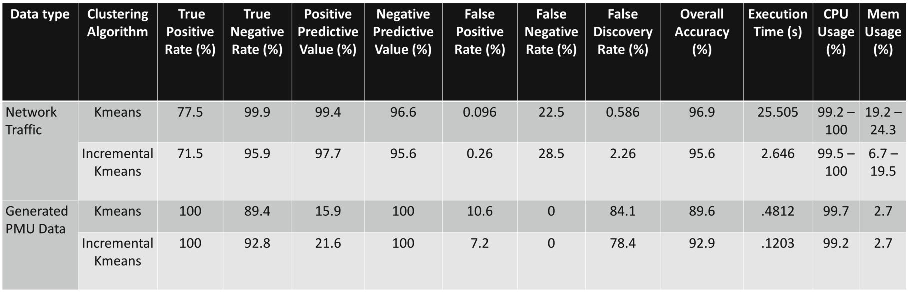
4.1 Simulation Setup

Using Mininet, we developed networking scenarios and topologies similar to realistic smart grid environments. 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. In this environment the state estimator was the client node connected to multiple PMUs setup as servers or client nodes.

We are developing our testbed to integrate a smart grid network emulator called OPAL-RT to generate the smart grid information. Virtual measurements of the smart grid emulator are taken from OPAL-RT, streamed from the server to the clients, and analyzed by the HDD-SDN application in ONOS. Network state information (i.e. number of devices, number of links, flow information, network traffic, etc.) collected by the external SDN parent and child controllers were stored in a database (InfluxDB, AWS, etc.) to be analyzed by an instance AD module operating in the child controller. Any necessary network reconfigurations were sent back to the parent SDN controller for the proper actions to take place. For experimentation purposes, the cluster algorithms were trained using the KDD CUP 1999 dataset that contains a standard set of data, which includes a wide variety of classified intrusions emulated in a military network environment [14]. With this data we extracted key features of network traffic identical to features which can be obtained using ONOS controller during network monitoring to compare anomaly detection techniques. We simulated measurements of the recorded and exchanged measurements by PMUs using MATLAB.

4.2 Performance Evaluation

For this experiment, we executed both k-means and incremental k-means for clustering and, subsequently, anomaly detection on network traffic data and PMU measurement data. In order to evaluate how well the network traffic was classified, the data was separated into training and testing sets. The training set is used to generate cluster centers for the k-means and incremental k-means



Data type	Clustering Algorithm	True Positive Rate (%)	True Negative Rate (%)	Positive Predictive Value (%)	Negative Predictive Value (%)	False Positive Rate (%)	False Negative Rate (%)	False Discovery Rate (%)	Overall Accuracy (%)	Execution Time (s)	CPU Usage (%)	Mem Usage (%)
Network Traffic	Kmeans	77.5	99.9	99.4	96.6	0.096	22.5	0.586	96.9	25.505	99.2 – 100	19.2 – 24.3
	Incremental Kmeans	71.5	95.9	97.7	95.6	0.26	28.5	2.26	95.6	2.646	99.5 – 100	6.7 – 19.5
Generated PMU Data	Kmeans	100	89.4	15.9	100	10.6	0	84.1	89.6	.4812	99.7	2.7
	Incremental Kmeans	100	92.8	21.6	100	7.2	0	78.4	92.9	.1203	99.2	2.7

Fig. 7. Table showing accuracy of anomaly detection algorithms for network traffic and PMU data

algorithms. Once the cluster centers are calculated, a label (normal or anomaly) is manually assigned to each cluster based on the IDD approach described above. The testing set is then added to the dataset and assigned the label of the cluster center closest to that data point. The size of the training set was n = 194,565 (70%) and the size of the testing set was n = 83385 (30%). The results of the classification of the network traffic are shown in Fig. 6. 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 (sensitivity), true negative rate, TNR (specificity), positive predictive value PPV (precision), negative predictive value, NPR, false positive rate FPR (fallout), false negative rate, FNR, false discovery rate FDR, and overall accuracy, ACC, with the inclusion of execution time, CPU and memory usage as follows:

$$\begin{aligned}
TPR &= TP/(TN+FP)\\
TNR &= TN/(TN+FP)\\
PPV &= TP/(TP+FP)\\
NPV &= TN/(TN+FN)\\
FPR &= FP/(FP+TN)\\
FNR &= FN/(TP+FN)\\
FDR &= FP/(TP+FP)\\
ACC &= (TP+TN)/(TP+FP+FN+TN)
\end{aligned} \tag{2}$$

As shown in Fig. 6, we observed that k-means produces a high accuracy (96.9%) of identifying anomalies within network traffic. Incremental k-means produces a slightly lower accuracy (95.6%), but demonstrated an increased speed with respect to the k-means by approximately 22 s. For the CPU and memory usage, k-means used more resources than incremental kmeans. This is likely due to the fact that kmeans is a batch algorithm, using the entire dataset during every iteration to recalculate cluster centers.

On the other hand, it was observed that the accuracy of anomaly detection for the OPAL-RT generated PMU data is lower than the accuracies provided when clustering network traffic. This is possibly because features of the PMU data are not as readily distinguishable from each other as expected, causing a higher false positive rating than expected. The proposed anomaly detection technique using clustering still preserves approximately 90% accuracy, mostly due to the true negative value exceeding others. It can be seen that CPU and memory usage are much lower for this data type, due to use of fewer features when clustering. This is a positive indicator for scaling the system to larger PMU networks (Fig. 7).

For performance overhead, we implemented a single ONOS controller with network management applications running on a device with the following specs: Ubuntu OS 14.04 (64-bit), AMD A6-6310 APU processor, 3.3 GiB memory, Java version 1.7.0 to manage a simple network consisting of 5 switches and 10 hosts. CPU usage for this system ranged from 1.3–17.3% of the device. We predict adding an additional controller on the same device to manage the same network will double this usage. With the ONOS system we can deploy separate controllers as VMs operating on different servers if need be. The research in [7] highlights the amount of increased bandwidth between decentralized controllers communication as the number of nodes controlled is increased.

5 Conclusion and Future Work

In conclusion, the benefits of implementing an adaptive distributed and decentralized SDN in place of the common networking or in place of traditional SDN strategy has been discussed. It was shown that implementing the HDD-SDN architecture can provide safe fail-over using redundant systems and additional resilience in the presence of faulty or attacked data or communication nodes in the smart grid system. The paper also evaluated the use of a combination of machine learning clustering algorithms for parallel processing of anomaly detection and discussed potential approaches for automated determination of the number of clusters needed. K-means produced a high accuracy for identifying anomalies within normal traffic and incremental k-means produced a slightly lower accuracy with increased speed and fewer CPU and memory resources, indicating a possibility for scaling the system to much larger networks. In future work, the process for preprocessing of the features before executing the machine learning will be examined. In addition, we continue to develop the SDN architecture integration with the OPAL-RT Smart Grid test bed.

Acknowledgment. The authors would like to thank the Harris Corporation Excellence in Research program for providing funding for this research.

References

1. Monsanto, C., Reich, J., Foster, N., Rexford, J., Walker, D.: Composing software defined networks. In: The 10th USENIX Symposium on Networked Systems Design and Implementation (NSDI 2013), pp. 1–13. IEEE (2013)

2. Lin, H., Chen, C., Wang, J., Qi, J., Jin, D.: Self-healing attack-resilient PMU network for power system operation. IEEE Trans. Smart Grid 1 (2016)
3. Chen, N., Wang, M., Zhang, N., Shen, X.: SDN-based framework for the PEV integrated smart grid. IEEE Netw. **31**(2), 14–21 (2014)
4. Pisharody, S., Natarajan, J., Chowdhary, A., Alshalan, A., Huang, D.: Brew: a security policy analysis framework for distributed sdn-based cloud environments. IEEE Trans. Dependable Secur. Comput. **PP**(99), 87–93 (2017). https://doi.org/10.1109/TDSC.2017.2726066
5. Nkosi, M., Lysko, A., Ravhuanzwo, L., Nandeni, T., Engelberencht, A.: Classification of SDN distributed controller approaches: a brief overview. In: 2016 International Conference on Advances in Computing and Communication Engineering (ICACCE), ICACCE, pp. 342–344 (2016)
6. ONOS-Open Network Operating System, March 2018. https://wiki.onosproject.org/. Accessed 4 Mar 2018
7. Muqaddas, A., Giaccone, P., Bianco, A., Maier, G.: Inter-controller traffic to support consistency in ONOS clusters. IEEE Trans. Netw. Serv. Manag. **14**(11), 126–133 (2017)
8. Kopeikin, A., Ponda, S.S., Johnson, L.B., How, J.P.: Multi-UAV network control through dynamic task allocation: ensuring data-rate and bit-error-rate support. In: 2012 IEEE Globecom Workshops, pp. 1579–1584. IEEE (2012)
9. Limthong, K.: Real-time computer network anomaly detection using machine learning techniques. J. Adv. Comput. Netw. **1**(1), 126–133 (2013)
10. Yadav, A.: Incremental k-means clustering algorithms: a review. Int. J. Latest Trends Eng. Technol. (IJLTET) **5**(4), 126–133 (2015)
11. Dunn, J.C.: Well-seperated clusters and optimal fuzzy partitions. Cybernetics **4**(1), 95–104 (1974)
12. Davies, D.L., Bouldin, D.W.: A cluster seperation measure. IEEE Trans. Pattern Anal. Mach. Intell. **1**(2), 224–227 (1979)
13. Neal, R.M.: Markov chain sampling methods for Dirichlet process mixture models. Comput. Graph. Stat. **9**(2), 249–265 (2000)
14. KDD Cup 1999 Data, March 1999. http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html. Accessed 4 Mar 2018