On the Practicality of Detecting Anomalies with Encrypted Traffic in AMI

Robin Berthier*, David I. Urbina†, Alvaro A. Cárdenas†, Michael Guerrero†, Ulrich Herberg‡, Jorjeta G. Jetcheva‡, Daisuke Mashima‡, Jun Ho Huh§, and Rakesh B. Bobba*,
*University of Illinois at Urbana-Champaign, †University of Texas at Dallas,
‡Fujitsu Laboratories of America, and §Honeywell Labs
{rgb,rbobba}@illinois.edu, {david.urbina,alvaro.cardenas,michael.guerrero}@utdallas.edu
{uherberg,jjetcheva,dmashima}@us.fujitsu.com, junho.huh@honeywell.com

***Abstract*—Encryption is a key ingredient in the preservation of the confidentiality of network communications but can also be at odds with the mission of Intrusion Detection Systems (IDSes) to monitor traffic. This affects Advanced Metering Infrastructures (AMIs) too where the scale of the network and the sensitivity of communication make deploying IDSes along with encryption solutions mandatory. In this paper, we study four different approaches for reconciling the twin goals of confidentiality and monitoring by investigating their practical use on a set of real-world packet-level traces collected at an operational AMI network.**

I. Introduction

The use of Intrusion Detection Systems (IDSes) in Smart Grid deployments is becoming a fundamental approach in securing smart grid networks, especially as the scale and scope of such networks increases, encompassing ever more critical functionality, and as they complete their migration to standardized communication stacks. Similarly, increased use of encryption is expected to both prevent eavesdropping and better protect sensitive and private data, such as fine-grained meter readings. A side-effect of encrypting communication is the loss of visibility into network traffic, which prevents IDSes from performing packet-level inspection analysis.

While an IDS can be given encryption keys to decrypt and parse messages, in practice there are several reasons for having a hierarchical analysis where one IDS monitors encrypted traffic and another one has the ability to decrypt special messages. Having an IDS capable of analyzing encrypted communications is particularly important to prevent information leakages, as large companies keep increasing the number of analysts looking at intrusion alarms, and smaller companies outsource the analysis of logs to outside companies.

A variety of techniques have been discussed to enable IDSes to monitor encrypted traffic in AMI networks [5], including sharing keys with IDS sensors, leveraging partial encryption, or applying traffic analysis techniques. In this paper, we investigate how some of those approaches could work in practice by applying them experimentally on real traffic captured at a large operational utility AMI network.

In particular, we study four different approaches:

- Monitoring the periodicity of meter communications,
- Detecting rogue devices through passive fingerprinting,
- Tracking unknown flows by baselining the network connectivity graph, and
- Identifying traffic patterns and outliers through unsupervised clustering.

We discuss the effectiveness of each of these approaches in detecting suspicious activity in the context of the AMI traces that we were able to collect. The main contributions of this work are to offer a detailed view of the internal network communications found on a large AMI and to provide insights on the challenges and possible solutions to identify intrusions despite the deployment of encryption.

II. Related Work

Analysis of encrypted traffic has been an active research area for the past two decades, and can be classified into two broad categories: 1) traffic classification techniques that use machine learning to classify flows per application [11], [8], and 2) packet analysis techniques that attempt to identify protocol operations through packet-level inspection [9].

Our work falls in the second category, since AMI communication traffic uses a single application-level protocol, namely ANSI C12.22. The work closest to ours is [7], where the authors proposed an IDS approach that does not need to inspect packet payloads. They leveraged the periodicity of process control system communication to identify outliers that could potentially represent malicious behavior. They use packet sizes, packet directions, and the relative time position of packets within a time series to label similar packets. The main limitations of this approach is the challenge of correctly tuning or training the parameters related to the polling cycles in order to achieve a good detection accuracy. A difference with our approach is our focus on AMI traffic and the need to identify different C12.22 sessions within a single TCP session.

Ali and Al-Shaer [1] demonstrate that AMI behavior can be modeled using 'event logs' collected at access points. Specification invariants, generated from AMI device configurations, are used to verify the AMI behavior models near the access points where the logs are collected. Our approach looks at the encrypted C12.22 traffic to detect anomalous behavior, which means there is more flexibility for placing IDS sensors in different physical locations, and enabling us to detect attacks that are located within the Neighborhood Area Network (i.e., not reaching the access points).

III. Data Collection

We partnered with a large U.S. elecric utility to collect network traffic at the head-end of a 30,000-meter operational AMI network, over several time periods. An AMI deployment usually comprises a Wide Area Network (WAN) that connects collection engines at the head-end with access points in the field, and Neighborhood Area Networks (NANs), each of which is connected to one or more access points. A NAN is intended to carry smart meter communications. In our case, the WAN uses TCP/IP and consists of about 90 distinct access points. Each NAN uses a wireless mesh network that is not IP based but uses proprietary protocols. All datasets

were recorded in PCAP format on the TCP/IP portion of the network.

The AMI uses ANSI C12.22 [13] as the application-layer communication protocol. Only packets on the TCP port used by C12.22 (port 1153) were recorded. The PCAP capture files were dissected using both Wireshark and an IDS sensor called Amilyzer [6]. For each trace, TCP/IP and C12.22 [13] protocol information were extracted. The C12.22 payloads consist of an Association Control Service Element (ACSE) header and one or more Extended Protocol Specifications for Electric Metering (EPSEM) elements. ASCE headers contain control information about the association between communicating entities, such as caller and called identifiers (calling/called AP-titles). EPSEM elements have either a service request or response and hold C12.19 data values. In most of the packets recorded at the utility facility, the EPSEM elements were encrypted but the TCP/IP headers and the ACSE header were sent in the clear. Table I summarizes the main features of the two traces collected at the utility site.

Date recorded	*Duration*	*#Packets*	*#IPs*	*#ApTitles*
April 23, 2014	24 hours	1,009,982	86	25,645
May 5, 2014	24 hours	1,011,988	87	28,471

Table I
DATASETS COLLECTED IN 30,000-METER AMI NETWORK

IV. EXPERIMENTAL DATA ANALYSIS

A. Taxonomy of AMI Traffic

Our goal in this section is to understand what types of network traffic are present in real-world AMI network traces. Figure 1 and Figure 2 depict packets flowing between the meters and the collection engine in our two traces. The sampling resolution is one minute, i.e., the Y-axis counts the number of packets per minute (1,440 minutes per day).

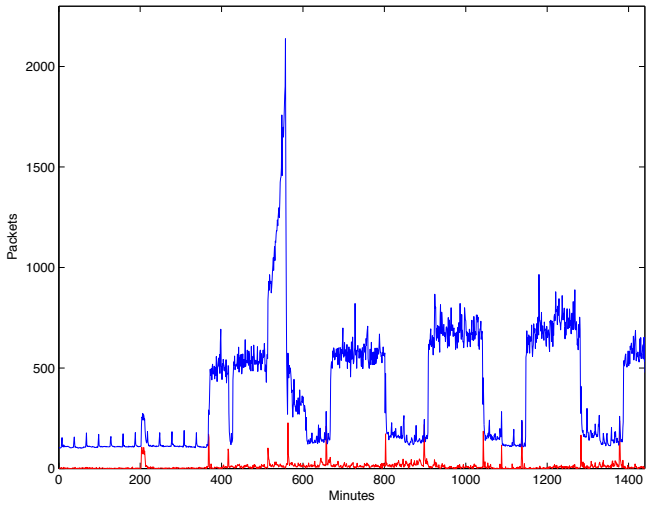

Figure 1. Number of packets per minute collected on April 23 over a 24-hour period. Overall traffic is shown in blue and TCP retransmissions are shown in red.

The red (lower) curve in both figures represents TCP retransmissions (1.6% of the total number of packets). Those retransmissions indicate possible losses and delays on the communication links used by the WAN. They are useful information to be monitored by an IDS in order to detect irregularities. For instance, a sudden loss of quality of service could be due to a denial-of-service attack.

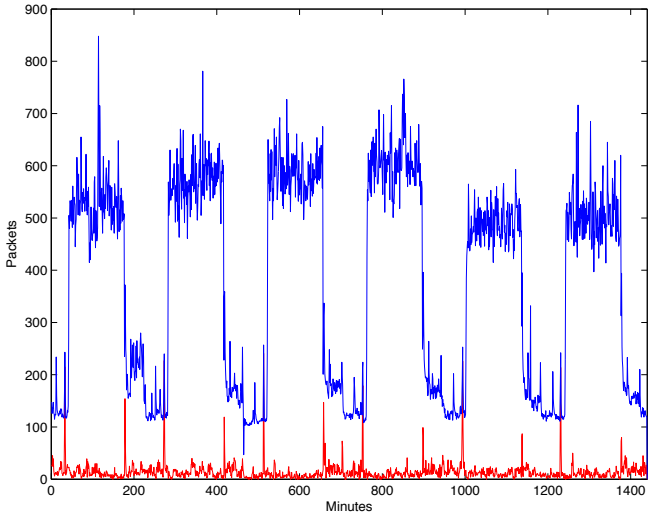

Figure 2. Number of packets per minute collected on May 5 over 24 hours on an AMI of 30,000 meters. Overall traffic is shown in blue and TCP retransmissions are shown in red.

We then identified three categories of traffic in both traces: 1) a periodic and continuous background traffic of keep-alive requests, 2) a periodic set of AMI requests/responses, and 3) aperiodic traffic.

Packets in the first category are unencrypted and consist of *identify* requests sent by cell relays every 60 seconds. They have an empty *called-Ap-title* indicating that they are broadcast messages, but they receive an acknowledgment packet from the collection engine about 0.2 seconds after being sent.

Packets in the second category are encrypted and larger in size (254 bytes compared to 92 bytes long for the keep-alive packets). They have a period of 240 minutes and are initiated by the collection engine. The period of 240 minutes was identified by applying an auto-correlation technique, as shown on Figure 3. In the figure, the interval between spikes can be considered to be the period. Meters being contacted by those requests respond after a random delay over a period of 135 minutes that starts 10 minutes after the collection engine sent the requests. This traffic, which consists of periodic meter readings, shows strong regularity on the second trace (Figure 2) but only starts after 367 minutes in the first trace (Figure 1). After checking with the utility, this initial gap of 6 hours with no periodic traffic was due to a system outage. Knowledge of the expected periodicity of the traffic enables an IDS to report such irregularities.

The third category consists of three types of unencrypted requests for which we did not identify a period. They include *registration* requests, *write* requests, and response errors to some encrypted requests. They were initiated by 114 devices in the first trace, and 159 devices in the second trace, which represents less than 0.5% of the meter population. We compared the ApTitles and found that only 14 were common across the two traces, indicating that this traffic is specific in time and in scope.

We analyzed further the registration requests and responses and found that there was about 1 registration response every 10

minutes, and that most registrations happen within 20 minutes of each other. This indicates that the network as a whole is relatively stable, and that there are occasional disconnections, typically in small clusters where one poor quality or disconnected link led to the disconnection of multiple meters and their change of affiliation to a neighboring access point's network. We believe that tracking the patterns of registration responses in an AMI network, can be helpful in identifying anomalous behavior, including attacks against the AMI mesh routing protocol, which lead to an increase in registration traffic.

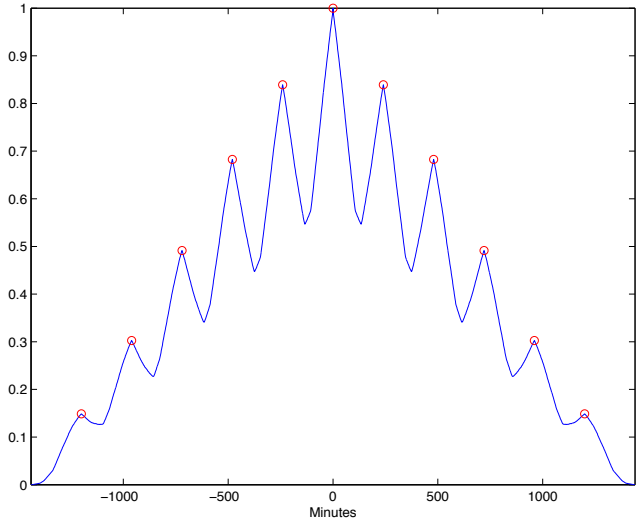

Figure 3. Auto-correlation applied to the second 24-hour trace to find the period of 240 minutes between encrypted requests initiated by the collection engine.

B. AMI Device Fingerprinting

While encryption hides payload content, it is still possible to extract useful information from the unencrypted packet headers. The idea in this section is to check if the network stack of AMI devices and OS/firmware running on them could be fingerprinted to be validated despite the encryption. This would enable us to add device fingerprint signatures to the IDS in order to flag unknown or unauthorized devices appearing on the network.

In order to explore this direction, we ran a passive fingerprinting tool called *P0f* (version 3.06b) [14] on both traces. Based on the contents of captured packets, P0f extracts fingerprints consisting of MTU signature and TCP signature [14], which vary across protocol stack implementations. We note that, in traces collected at the head-end, only device information about the collection engine and the access points is visible. Monitoring the characteristics of smart meter devices would require that packets be recorded within NANs or that a fingerprinting mechanism based on C12.22 payloads, which are visible to the collection engine even when access points are mediating the communication, is implemented.

P0f identified three distinct fingerprints in both of our traces, besides ones attributed to the collection engine. Those three fingerprints were not found in P0f's signature database and are associated with the IP addresses of access points. The exact fingerprints being sensitive information, we label them FP_1, FP_2, and FP_3. FP_1 indicates larger initial TTL and window size, as well as more TCP options set. The only difference between FP_2 and FP_3 is found in initial TTL. We observed that all access points share FP_1 when they are initiating TCP connections, while FP_2 and FP_3 divide access points into two groups when they accept incoming TCP connections. This result indicates that the pairs of fingerprints (FP_1, FP_2) and (FP_1, FP_3) can be used to identify two families of devices. We confirmed with the utility partner that thoses devices are different models. Another important result is the consistency of those pairs of fingerprints across the two traces, indicating that an attacker intruding on the network with a personal computer, impersonating meters [10], or access points running malicious/unauthorized firmware could be flagged.

C. Connectivity Graph

The concept of identifying intrusions using a network connectivity graph refers to learning the relationships among end points in order to flag when a new end point or a new link appears in the graph. Access to two 24-hour windows of traffic, 10 days apart from each other, can help us understand if such a technique could be promising.

The addresses of end points in an AMI can be extracted at the physical layer (MAC addresses), IP layer (IP addresses) and at the application layer (ApTitles). An IDS sensor can extract those addresses and keep a database updated to identify when a new end point initiates communication and when a new pair of end points is exchanging traffic.

Table II contains the unique number of IP addresses and ApTitles found in each dataset. Those results reveal important changes from one collection date to the next. In particular, we observe greater variations in the source and destination roles of end points compared to their unique count. This indicates that new nodes appeared but a larger number of nodes that were used solely as sources became destinations, and vice-versa.

End points	*Initial set*	*Added*	*Removed*
IP addresses	86	8	9
Source IP	69	21	9
Dest. IP	57	1	2
ApTitles	25,645	9,096	6,270
Calling ApTitles	21,768	11,843	5,951
Called ApTitles	8,768	946	7,328

Table II
ADDRESS INFORMATION COLLECTED AT LAYERS 3 AND 7 OVER A 24-HOUR PERIOD, AND DIFFERENCES WITH A 24-HOUR TRACE COLLECTED 10 DAYS LATER

Table III contains similar results but for relationships among end points, where each connection is identified by a source and destination IP address pair. We observe the consequence of role reversal with a large number of new relationships added and relationships removed between the initial and the most recent data set collected.

Unique connections	*Initial set*	*Added*	*Removed*
(Src IP, Dst IP) pairs	124	22	11
ApTitles	27,777	15,152	13,449

Table III
NUMBER OF UNIQUE CONNECTIONS AMONG END POINTS COLLECTED AT LAYERS 3 AND 7 OVER A 24-HOUR PERIOD, AND DIFFERENCES WITH A 24-HOUR TRACE COLLECTED 10 DAYS LATER

The results from the connectivity graph analysis indicate that the AMI under consideration was too dynamic at the time of our data collection for this approach to be effective. We got confirmation from the utility partner that a large number of meters were added and reassigned to specific cell relays between the two days during which we collected data. Once the population of end points and the communication patterns are stable, the technique of monitoring the connectivity graph can be highly effective to detect rogue end points, spoofing attacks, and anomalous changes in communication patterns.

D. Unsupervised Learning

Traffic analysis of encrypted communications has traditionally leveraged two pieces of information that are not concealed by encryption: social behavior of a node (who is talking to whom), and network traffic statistics such as packet sizes, timings, and header information from the various traffic flows.

Our goal in this section is to apply unsupervised learning algorithms to encrypted AMI network traffic in order to understand whether or not traditional feature vectors used for encrypted traffic classification can reveal an underlying structure of AMI network flows. In particular, since we know that C12.22 packets contain either requests or responses from a limited number of commands, our initial goal is to see if unsupervised classification algorithms can identify clusters of encrypted network flows of packets with potentially similar commands being sent or received and also can detect an outlier, which may be an indication of attacks, and regular AMI communications, which would be represented as clusters.

One of the main differences between the challenges we encountered and prior work on encrypted traffic classification [12], [3], [4], [2] is that in that prior research, the authors extract feature vectors from TCP flows. Because TCP is a connection-based protocol, this allows researchers the ability to collect aggregate statistics of several packets (multiple packet sizes and timing information corresponding to one session) going back and forth between sender and receiver. While C12.22 supports connections, most of the communication (in fact, all of the flows in our trace) consist of only one or two packets: either a notification update such as a meter sending its table to the server at periodic intervals, or a request and a reply.

Because all communications in our C12.22 trace are either a single packet or a request-response pair, we can only create network flows of one or two packets. We extracted C12.22 flows by focusing on the following header parameters: `called-AP-title`, `called-AP-invocation-id`, `calling-AP-title`, and `calling-AP-invocation-id`. `Calling-AP-title` and `called-AP-title` are the device identifiers of the source and destination of the packet, respectively. `Calling-AP-invocation-id` is a sequence number used to keep track of sessions and to eliminate duplicate packets. `Called-AP-invocation-id` is empty for the first packet of a session, and matches the previous `calling-AP-invocation-id` for response packets.

C12.22 headers also contain a Request Control Flag (RCF) that according to the standard specifies the following values: `0x0` to denote that the sender wants a reply to this packet, `0x1` to denote that the sender wants a reply to this packet only under error conditions, and `0x2` to instruct the receiver to not respond to this packet. We also observed a flag value of `0x3` that is manufacturer-specific. This flag helped us in identifying flows (when we see `0x2` we know we do not have to look for a matching `calling-AP-invocation-id`). We also confirmed that all replies (the second packets of our flows) have the flag RCF `0x2`, thereby confirming that our matching flows identifies a maximum of only two packets per flow. In the traces analyzed, we also found an unknown flag (not defined by the standard).

Among all unsolicited messages (first packets in a flow), the distribution of the RCF flag is shown in Table IV.

Flag	*Percentage*
Always Respond, `0x0`	44%
Respond on Exception, `0x1`	18%
Never Respond, `0x2`	19%
Manufacturer Flag, `0x3`	19%

Table IV
DISTRIBUTION OF RCF FLAG AMONG ALL UNSOLICITED MESSAGES (FIRST MESSAGE OF A SESSION)

Identifying single-packet flows (unsolicited-messages) and two-packet flows with request-reply pairs enabled us to start creating flow-based feature vectors. Those vectors consists of (1) the size (in bytes) of the payload in the first packet of the flow, (2) the size (in bytes) of the payload in the second packet of the flow (zero if there is no second packet), (3) the time (in ms) between packets (zero if there is no second packet), and (4) the direction of the flow: 1 if the flow is originated from the collection engine, and 0 if the flow originated from a device in the field.

Figure 4 shows the multidimensional space of the feature vectors. A first observation is that there is a clear outlier: one of the response messages has a payload of 1,044 bytes, which makes it much larger than the rest. Looking at the packet that generated this outlier we found that it is a packet composed of three C12.22 layers. Similarly we found that the largest unsolicited packet (a request of 2,089 bytes) consisted of four C12.22 layers concatenated together. Upon further inspection, we found 82 packets in the first trace with multiple C12.22 payloads; furthermore those extra layers have the same source and destination ID (Called and Calling AP titles). They represent 0.0147% of all packets and are due to packet aggregation performed by the access points.

Other observations from Figure 4 include the following: (1) unsolicited messages have a wide range of packet payload lengths, while replies are usually small and tend to be less than 200 bytes; (2) unsolicited messages that receive a reply have small payloads (less than 900 bytes) while unsolicited messages do not ask for replies can be up to 2000 bytes; (3) there is a wide range of packet sizes for unsolicited messages originating from the AMI, while packet sizes of unsolicited messages starting at the server side are small (up to 500 bytes). These traffic patterns can be used to develop anomaly detection rules such as: (1) a reply larger than 200 bytes is anomalous, (2) if an unsolicited message is larger than 800 bytes, then it will not receive a reply (if it does, then this is an anomaly), and (3) an unsolicited message from the server to the AMI larger than 500 bytes is an anomaly.

We then applied unsupervised learning algorithms on the feature vectors to identify patterns in the dataset. In order to make sure that one dimension is not dominating the others, we normalized each dimension using the following formula: $x' = (x - x_{min})/(x_{max} - x_{min})$.

We used the K-means algorithm to identify clusters of packets. While there are several automatic suggestions recommending values for K there is no clear consensus on the selection of K, and the best approaches rely on attempting multiple cluster numbers and evaluating them with exploratory data analysis. We experimented with small values of K from

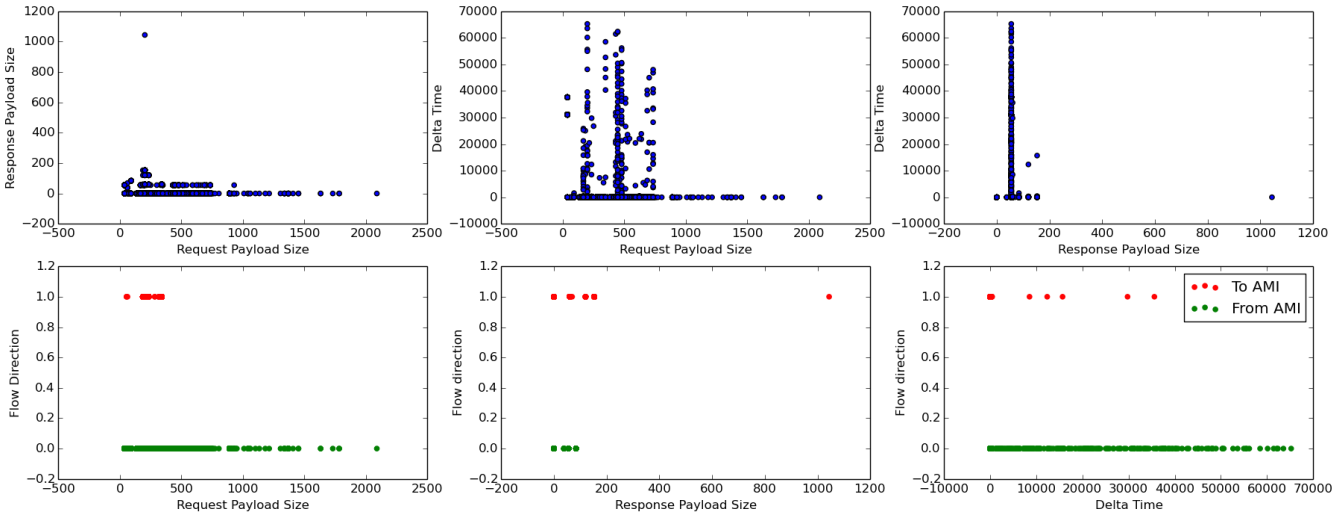

Figure 4. Feature vector space representing each feature against the payload size in bytes and the time difference in seconds.

2 clusters to 6 in order to identify a small number of clusters from which we could do a manual analysis of the clusters, and in particular of the packets belonging to each cluster. We focus on 6 clusters in this paper, with cluster labels of 0 through 5.

From our clusters, we were able to identify that cluster 5 has 98% of the identify requests (from a total of 75,563), which are unencrypted and recognizable. Although, we could not extend our analysis to identify other specific commands sent and received due to a lack of corresponding plaintext packets, these results support our hypothesis that clusters of encrypted C12.22 communications can identify specific commands exchanged in the network. Our goal is to extend this work to identify all commands being exchanged in the network (e.g., with the help of a testbed) and then instruct an IDS to allow communications within traditional clusters (commands) and flag as suspicious communication flows that do not fall into any traditional commands. The other side of this interpretation is that encrypted communications can leak the type of commands being shared in a network, and this is something a designer should take into account when doing vulnerability assessments.

Clusters can also be used to identify faults or misconfigurations. Looking at clusters 2 and 4, we found several packets with missing elements such as calling invocation IDs. Overall, clusters 2 and 4 contain 100% (50.1%, and 49.9% respectively) of malformed packets (from a total of 174,607). This clustering of malformed packets likely occurs because of different types of errors. As part of the next steps to implement this approach in an IDS, we plan to add root cause information to those clusters in order to inform operators about their criticality. Finally, we also identified clusters 3 and 5 as being composed by packets with responses, while the other clusters have an overwhelming percentage of unsolicited messages without replies.

As a final part of our analysis, we now visualize the clusters. While the feature vector lies in a four dimensional space, we can make a projection that maximizes the variance of the feature space. This projection of a 4-D vector to a 2-D vector can be achieved with Principal Component Analysis (PCA). Table V shows that keeping only the two most significant components retains 98% of the variance of the data.

Figure 5 depicts the 2-Dimensional space resulting after applying PCA with 2 components to the feature vectors;

# Comp.	*Ret. Var.*	*# Comp.*	*Ret. Var.*
1	78%	2	98%
3	99%	4	100%

Table V
COMPONENTS VS. RETAINED VARIANCE

each cluster is assigned a different color. While in our initial analysis of the clusters we did not identify this clear separation of the network flows, upon looking at this PCA projection we noticed that the cluster on the right is cluster 0, and upon further inspection we noticed that this cluster is composed of over 90% of unsolicited messages (unidirectional flows) sent to the meters; therefore, the connections in cluster 0 stand out from the ones in other clusters by having a 1 in the flow dimension, and 0 for both, the timing and the size of the reply. This behavior is a clear discriminant feature of the network flows as only 7% of connections are initiated by the server (93% of connections are initiated by the smart meters); furthermore, only 9% of connections are followed by a response flow.

It is clear that the last feature, the direction of the flow, is the feature that dominates in this projection to two dimensions. To illustrate better the variety of clusters we decided to apply PCA to only the first three dimensions of the feature vector, the results can be seen in Figure 6. The remaining clusters are not as easily separable, but it is clear that they lie on different parts of the 2-D space.

V. SUMMARY & CONCLUSIONS

This paper offers an experimental study of different approaches that a network-based IDS for AMI could use to cope with encrypted traffic. An AMI has unique characteristics related to the portion of traffic and packets being encrypted, the number of service requests available, and the way session flows are handled that make this study an important step towards achieving the twin goals of confidentiality and security monitoring. In particular, we found that fingerprinting devices and watching for the periodicity of meter requests were good candidates to alert on suspicious activity. We also found that the 30,000-meter operational AMI used in our study was too dynamic to train a connectivity graph baseline over 2

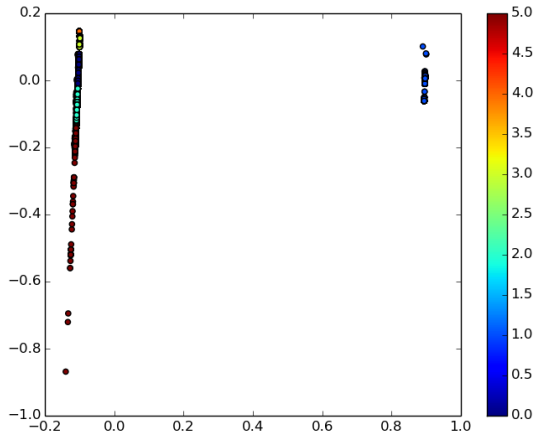

Figure 5. PCA with 2 components retaining 98% of variance

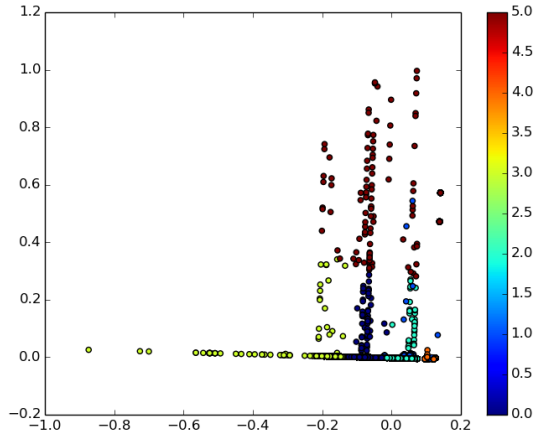

Figure 6. PCA of clustered packets with k=6

days of traffic. By extracting feature vectors representing connection flows we were able to identify several anomalies (e.g., packets with multiple C12.22 payloads), and rules of "normal" behavior, such as the size of reply packets being lower than 200 bytes, or that bidirectional communications have request packets with sizes smaller than 800 bytes.

Table VI provides a summarized mapping between the different approaches investigated in this paper and the types of malicious behavior they can detect.

As the next step, we plan to increase the length of our data collection to confirm our exploratory data analysis, and to measure the security of IDS rules for encrypted packets against adversaries that will try to evade them.

ACKNOWLEDGMENTS

This material is based upon work supported in part by the Department of Energy under Award Number DE-OE0000097 and by Fujitsu Laboratories of America. The opinions expressed are those of the authors alone.

Attacks	*Period.*	*Fingerpr.*	*Graph*	*Clustering*
Traffic tampering	✓			✓
Traffic injection	✓		✓	✓
Replay attack	✓			✓
Authentication abuse				✓
Spoofing		✓	✓	
Rogue device		✓	✓	
Compromised meter	✓		✓	✓
Resource exhaustion	✓		✓	✓

Table VI
MAPPING BETWEEN INTRUSION DETECTION TECHNIQUES AND DETECTABLE ATTACKS DESPITE ENCRYPTION

REFERENCES

[1] M. Q. Ali and E. Al-Shaer. Configuration-based ids for advanced metering infrastructure. In *Proceedings of the 2013 ACM SIGSAC Conference on Computer & Communications Security*, CCS '13, pages 451–462, New York, NY, USA, 2013. ACM.

[2] L. Bernaille and R. Teixeira. Early recognition of encrypted applications. In *Passive and Active Measurement Conference (PAM), Proc.*, pages 165–175, Louvain-la-Neuve, Belgium, April 2007.

[3] L. Bernaille, R. Teixeira, I. Akodkenou, A. Soule, and K. Salamatian. Traffic classification on the fly. *SIGCOMM Comput. Commun. Rev.*, 36(2):23–26, 2006.

[4] L. Bernaille, R. Teixeira, and K. Salamatian. Early application identification. In *Conference on Future Networking Technologies (CONEXT 2006), Proc.*, page 6, 2006.

[5] R. Berthier, J. G. Jetcheva, D. Mashima, J. H. Huh, D. Grochocki, R. B. Bobba, A. A. Cárdenas, and W. H. Sanders. Reconciling security protection and monitoring requirements in advanced metering infrastructures. In *Smart Grid Communications (SmartGridComm), 2013 IEEE International Conference on*, pages 450–455. IEEE, 2013.

[6] R. Berthier and W. H. Sanders. Monitoring advanced metering infrastructures with amilyzer. In *Cyber-security of SCADA & industrial control systems*. C&ESAR, 2013.

[7] M. Hoeve. Detecting intrusions in encrypted control traffic. In *Proceedings of the first ACM workshop on Smart energy grid security*, pages 23–28. ACM, 2013.

[8] T. Karagiannis, K. Papagiannaki, and M. Faloutsos. BLINC: Multilevel traffic classification in the dark. *SIGCOMM Comput. Commun. Rev.*, 35(4):229–240, 2005.

[9] R. Koch and G. D. Rodosek. Command evaluation in encrypted remote sessions. In *Network and System Security (NSS), 2010 4th International Conference on*, pages 299–305. IEEE, 2010.

[10] S. McLaughlin, D. Podkuiko, S. Miadzvezhanka, A. Delozier, and P. McDaniel. Multi-vendor penetration testing in the advanced metering infrastructure. In *26th Annual Computer Security Applications Conference (ACSAC), Proc.*, pages 107–116, New York, NY, USA, 2010. ACM.

[11] A. D. Montigny-Leboeuf. Flow attributes for use in traffic characterization. Technical Report CRC-TN-2005-003, Communications Research Centre, Canada, December 2005.

[12] A. W. Moore and D. Zuev. Internet traffic classification using Bayesian analysis techniques. In *SIGMETRICS '05: Proceedings of the 2005 ACM SIGMETRICS International Conference on Measurement and Modeling of Computer Systems*, pages 50–60, New York, NY, USA, 2005. ACM Press.

[13] A. Snyder and M. Stuber. The ANSI C12 protocol suite - updated and now with network capabilities. In *Power Systems Conference: Advanced Metering, Protection, Control, Communication, and Distributed Resources, 2007. PSC 2007*, pages 117–122, 2007.

[14] M. Zalewski. p0f v3: passive fingerprinter, 2012. http://lcamtuf.coredump.cx/p0f3/README.