An AMI Threat Detection Mechanism Based on SDN Networks

Po-Wen Chi*, Chien-Ting Kuo*†, He-Ming Ruan*, Shih-Jen Chen†, and Chin-Laung Lei*
*Department of Electrical Engineering, National Taiwan University, Taipei, Taiwan
Email: {d99921015, d98921027, d97921030, cllei}@ntu.edu.tw
†CyberTrust Technology Institute, Institute for Information Industry, Taipei, Taiwan
Email: {ctkuo, sjchen}@iii.org.tw

***Abstract*—The security of Advanced Metering Infrastructure (AMI) systems draws more and more attention nowadays. Intrusion detection systems are often deployed on the backhaul network to protect the AMI head-end system. In this paper, we proposed an efficient way to build threat detecting mechanism in AMI systems with the help of software defined networks (SDN). Moreover, we also enhance the OpenFlow architecture to provide more powerful detection mechanism to secure the AMI system. The proposed solution not only enhances the security of AMI systems, but also preserves the traffic quality of this structure.**

Keywords–AMI; SDN; Specification-based detection

I. INTRODUCTION

Recently, the AMI system, which serves as a key role in Smart Grid, became popular due to the benefits it could bring. This new infrastructure enables the exploration of the possibilities of energy utilization by providing certain communication and control functionalities. However, AMI introduces new security challenges while providing various benefits due to semi-open networks, improper security mechanisms and immature hardware design for AMI devices. There are already many researches which introduce security issues in AMI systems, such as [1][2]. The essence of AMI is a vast and distributed sensor system tethered by the backhaul network and some neighborhood networks (NANs) which can be open networks or closed ones. It implies that anyone on the backhaul might find their way to interfere with the AMI, especially the Internet service providers (ISPs) who can possibly control partial or all of the connections in an AMI system. Thus, we will focus on the security issue in the backhaul network in this paper.

Traditional approaches to protect a device in an IT system could be cryptographic tools such as mutual authentication that ensures the identities of each end in a communication, encryption and key management, which enforces the access control over specific storage media, or digital signature, which guarantees the source of a message. However, any of the cryptographic measures require relatively powerful hardware, and this implies that the cost of devices will be anything but cheap. But the extremely large scale of AMI systems limits the budget of the devices, and further constrains the capability of the devices and the available protection approaches. Under such dire condition, monitoring the security status of the AMI system becomes a practical and economical solution. With the status of the system security at hand, one can then address and react to security events more effectively while the cost will be much economical than traditional cryptographic protection measures.

Traditional IDS systems mostly take signature-based detection as their core technology, which detects malicious activities by describing these activities as signatures beforehand. Snort [3] is the most popular open-sourced project of this kind of IDS. However, this kind of detection alone is not sufficient since it is difficult to list all malicious behaviors and nothing can be done about unknown attacks. In order to provide a more secure network environment, specification-based detection was proposed [4][5]. With the specifications to describe the normal activities, the IDS can collect all events which do not meet the requirement of the AMI system. Thus, the administrator can decide if the network is under attacks by comprehensive analysis of events. Therefore, the administrator can still be aware of unknown attacks under the assistance of the specification-based technology.

In addition to the specification-based detection system, we observe that a new network trend, Software Defined Networking (SDN), is changing the network architecture. The SDN could be a proper primitive for an AMI system due to the vast and distributed nature of the AMI, which results in the need of efficient management mechanisms to secure the AMI systems. With the features of the SDN, it reveals a novel approach for the administrator to dynamically perform flow-level management over his own network. We believe that in the near future, more and more networks will be SDN, including AMI backhaul networks. So, we are motivated to build an IDS in SDN-based AMI backhaul networks.

In this paper, we integrate the SDN technology with IDS in the AMI system. First, we will show how to integrate traditional IDS, Snort, with SDN efficiently by offloading some checking rules from Snort to OpenFlow switches. Therefore, IDS will afford more throughputs than legacy architecture. Moreover, we propose an enhanced OpenFlow technology in which OpenFlow switches are improved by additional specification checking agents. By using our enhanced OpenFlow switches, the specification checking rules can be quickly deployed to each transmission path node in the AMI system from OpenFlow controller. We also modify some parts of OpenFlow protocol to support the proposed functionalities. If necessary, we can also deploy the controllers hierarchically to scale out the management capability for the future growth of the system scale.

This paper is organized as follows: we will introduce some related background knowledge, including the components of AMI system, the specification based IDS, and a brief introduction to the SDN network in Section II. In Section III, we will show how to integrate Snort with SDN in a more efficient way than legacy network. Our new OpenFlow technique which supports specification checking function on OpenFlow switches will be given in Section IV. Finally, we will have some conclusions of this proposed SDN-based AMI Detecting Mechanism.

 ISBN: 978-1-61208-376-6

II. BACKGROUND

In this section, we will introduce some background knowledge about the components of AMI architecture, the specification-based detection and the SDN network.

A. The components of AMI architecture

A generic AMI system consists of smart meters, concentrators, head-end, neighborhood area network, and backhaul network.

- **Smart meter:** A smart meter serves as an interface to end users and the user agent to actively monitor, record, and report messages to the concentrator it belongs to.
- **Concentrator:** A concentrator acts as a network gateway of a group of smart meters. It collects data from smart meters and forward messages for smart meters and AMI head-ends.
- **Head-end:** This system acts as an I/O interface of an AMI system. The major functionality is to deal with the information exchange between the AMI system and other systems, such as MDMS, which manages all the meter data in a centralized or distributed way.
- **Neighborhood area network (NAN):** An NAN takes the task to connect smart meters and concentrators. It provides routes for smart meters and collectors to transmit messages. ZigBee networks and Power Line Communication (PLC) networks are popular candidates for NAN nowadays.
- **Backhaul network:** The backhaul network provides routes for concentrators and AMI head-ends to transmit commands, records, or any other messages. The backhaul network could be the open Internet. For security concerns, the connections between AMI head-ends and concentrators are possibly established by virtual private networks (VPNs).

B. The Specification-based Detection

Berthier et al. [4][5] proposed an IDS framework and a specification-based intrusion detection system for AMI systems in 2010 and 2011 respectively. The specification-based intrusion detection was first introduced in 1997 by C. Ko [6]. Specifications define the expected behaviors of the system activities via the functionalities to perform and the security policies to be obeyed. Thus, any behavior that strays from the specifications can be regarded as a security violation. Recently, security specifications have been defined for routing protocols [7][8][9], VoIP protocols [10][11][12], control systems [6][13][14], and unmanned vehicles [15].

C. Software Defined Networking, SDN

The idea of SDN was first proposed by Nick McKeown et al. in [16]. They proposed an idea that decouples the control plane and the data plane of each network node. The data plane is still kept on each network node while the control plane is concentrated logically on one controller. The data plane handles each packet with flow entries, which are tuples of flow matching fields and actions. All flow entries are managed by the controller. OpenFlow[17] is the most common architecture and protocol of SDN. In this paper, we assume the AMI backhaul network is SDN and we will build an IDS/IPS service on the backhaul network.

III. SDN AND SNORT INTEGRATION

Snort is an open source signature-based IDS system. The traditional architecture of Snort deployment is to mirror all traffics to Snort. Snort will check all traffic by pre-defined rules. If there is any packet that matches pre-defined rules, Snort will send an alarm and may inform firewall to block the suspicious traffic. Figure 1 is a deployment example.

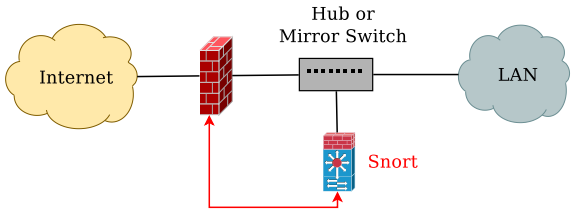

Figure 1. Traditional Snort Deployment.

When considering the SDN environment, there are two common ways to deploy the Snort service. The first way is to implement the mirror function on an OpenFlow switch, like Figure 2. To implement the mirror function on an OpenFlow switch, the OpenFlow controller will set one flow entry with two output ports: one is the regular forwarding port and the other is the port to Snort. Then, all traffics will be forwarded not only to destinations but also to Snort for analysis. Once a suspicious traffic is detected, Snort can notify the OpenFlow controller to command the OpenFlow switch to drop the specific traffic.

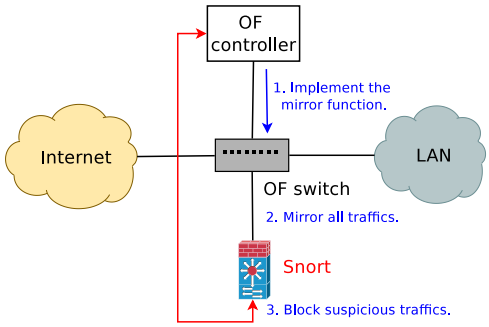

Figure 2. Snort Deployment in SDN: mirror implementation.

Most SDN frameworks use this deployment architecture, like Ryu [18]. The second way is presented in Figure 3. This approach ports Snort from a daemon to an SDN application. All traffics will be passed to the OpenFlow controller through *PACKET_IN* events of OpenFlow protocol. The OpenFlow controller then handles the received traffics by Snort SDN application. [19] uses this kind of architecture. The problem of this architecture is the unaffordable burden on the OpenFlow control channel. This is because all traffics are transmitted on both the data plane and the control plane. So, using *PACKET_IN* as a data forwarding method will possibly overwhelm the system.

Thus, we hereby propose a new integration approach. The matching field of a Snort rule is composed of Snort rule headers and some Snort rule options. We find some parts of these matching fields are L2-L4 matching rules which are also supported by OpenFlow switches, such as IP address,

 ISBN: 978-1-61208-376-6

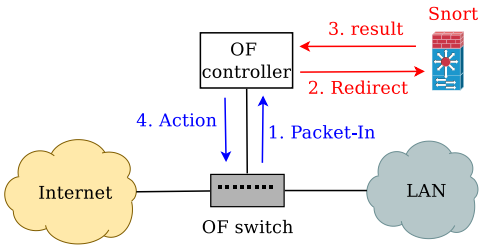

Figure 3. Snort Deployment in SDN: *PACKET_IN*.

TCP/UDP port, TOS in the IP header, ICMP code and so on. Therefore, we move these matching works from Snort to OpenFlow switches. Figure 4 illustrates the architecture proposed in this paper. First of all, we build a Snort rule parser to derive OpenFlow rules from Snort rules. Then, the OpenFlow controller sets these OpenFlow rules to OpenFlow switches and OpenFlow switches will relay only suspicious traffics to Snort for further analysis. The controller can also dispatch these suspicious traffics to multiple Snort servers when load balancing is necessary. Once a Snort alarm happens, the Snort server will inform the OpenFlow controller to block the traffic. In this architecture, traffics are relayed in a much more efficient way.

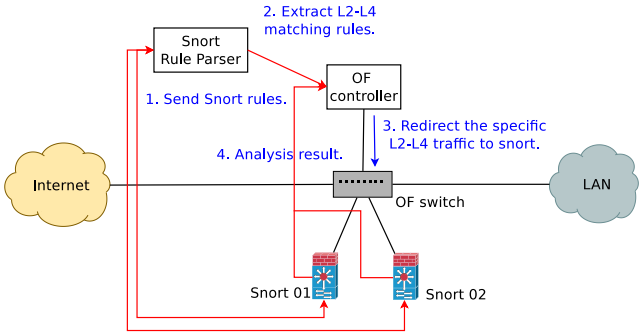

Figure 4. Our proposed integration method.

Now, we will introduce our idea about OpenFlow security enhancement. The idea is presented in Figure 5. There are two main modifications compared to the original OpenFlow. First, we add a specification management server module on the OpenFlow controller and a specification checking agent

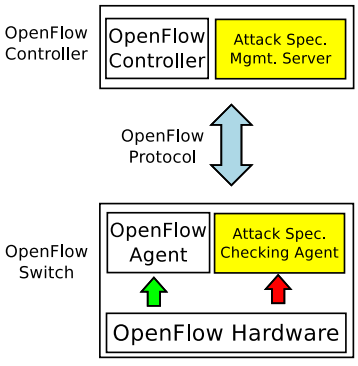

Figure 5. Security-enhanced OpenFlow Architecture.

module on the OpenFlow switch. These two modules are communicated with vendor specific elements. We can use all existing matching fields of OpenFlow as parts of specifications to filter interested traffics. The main function of the specification management server module is to dispatch specifications to agents and to receive alarms. This module will determine if an attack happens by collecting alarms. The main function of the specification checking agent is to execute specification checking procedure and to alarm the server when abnormal conditions happen. Second, we add a new output port *ATT_SPEC_CHECK* on OpenFlow switches to channel the traffics to the specification checking agent.

In this architecture, the specification-based detecting engine hosts on OpenFlow switches. However, the computation resource might vary from switch to switch, so the specification server is designed to dispatch works according to switches' ability.

Now we will introduce how to protect AMI systems with the proposed enhanced OpenFlow. The overview of an AMI system with the SDN-based attack detection architecture can also be found in Figure 6. All backhaul OpenFlow switches are improved with our enhancement. We also make concentrators support our enhanced OpenFlow switch function. The system administrator will first define proper specifications and then configure the SDN controllers with these specifications. After the configuration, the SDN controllers can dispatch these checking tasks to all OpenFlow switches, and all OpenFlow switches are responsible for checking if any pre-defined condition happens. Since concentrators are counted as OpenFlow switches and possess lesser resource, the tasks for concentrators should be lightweight, such as infrequent checking works.

Note that the whole system can observe all traffics in the flow level through these OpenFlow switches. If some condition happens, the switch which observes the condition will inform the SDN controller. The specification management server module will decide if these alarms are misbehaviors or not. If there is misbehavior in the backhaul network, the SDN controller will block the corresponding flow. Therefore, in this architecture, the misbehavior can be discovered in the backhaul network without impact on AMI-head end.

There are some advantages of the proposed architecture. First of all, the detection is distributed over all OpenFlow switches and makes it easy for the administrator to locate the real problem in the whole backhaul network. Thus, the administrator can isolate the network region where attacks come from. Besides, by using the OpenFlow technique, it is possible to trace and ease misbehaviors in the flow level. Moreover, the administrator can dynamically change forwarding paths of all traffics to protect the AMI system from attacks. So, our proposed OpenFlow enhancement with specification-based detection system can bring a more secure AMI system.

IV. Conclusions

In this paper, we proposed our idea about how to integrate IDS with SDN networks to protect the AMI systems. We made use of SDN functionalities to offload rule-based detection systems. We also enhanced the OpenFlow switches to support specification-based detection system for unknown attacks. With the proposed methods, the AMI systems will be able to provide more effective and efficient defense against security threats. This ongoing work will have a PoC system

 ISBN: 978-1-61208-376-6

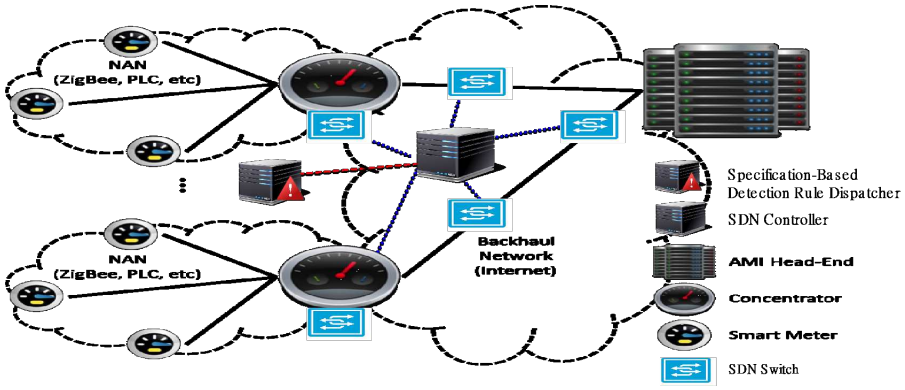

Figure 6. SDN-based AMI Attack Detection Architecture.

and related performance metrics for further evaluation in the future work.

ACKNOWLEDGEMENT

This study is conducted under the III Innovative and Prospective Technologies Project of the Institute for Information Industry which is subsidized by the Ministry of Economic Affairs of the Republic of China.

REFERENCES

[1] M. A. Faisal, Z. Aung, J. R. Williams, and A. Sanchez, "Securing advanced metering infrastructure using intrusion detection system with data stream mining," in Proceedings of the 2012 Pacific Asia conference on Intelligence and Security Informatics (PAISI'12), 2012, pp. 96–111.

[2] R. A. R. Kinney, P. Crucitti and V. Latora, "Modeling cascading failures in the north american power grid," in The European Physical Journal B – Condensed Matter and Complex Systems, 2005, pp. 101–107.

[3] Snort. [Online]. Available: https://www.snort.org/ [retrieved: Nov., 2014]

[4] R. Berthier, W. Sanders, and H. Khurana, "Intrusion detection for advanced metering infrastructures: Requirements and architectural directions," in Smart Grid Communications (SmartGridComm), 2010 First IEEE International Conference on, 2010, pp. 350 –355.

[5] R. Berthier and W. Sanders, "Specification-based intrusion detection for advanced metering infrastructures," in Dependable Computing (PRDC), 2011 IEEE 17th Pacific Rim International Symposium on, 2011, pp. 184 –193.

[6] C. Ko, M. Ruschitzka, and K. Levitt, "Execution monitoring of security-critical programs in distributed systems: a specification-based approach," in Security and Privacy, 1997. Proceedings., 1997 IEEE Symposium on, 1997, pp. 175–187.

[7] C.-Y. Tseng et al., "A specification-based intrusion detection system for aodv," in Proceedings of the 1st ACM Workshop on Security of Ad Hoc and Sensor Networks, ser. SASN '03. New York, NY, USA: ACM, 2003, pp. 125–134.

[8] C. H. Tseng, T. Song, P. Balasubramanyam, C. Ko, and K. Levitt, "A specification-based intrusion detection model for olsr," in Proceedings of the 8th International Conference on Recent Advances in Intrusion Detection, ser. RAID'05. Berlin, Heidelberg: Springer-Verlag, 2006, pp. 330–350.

[9] H. M. Hassan, M. Mahmoud, and S. El-Kassas, "Securing the aodv protocol using specification-based intrusion detection," in Proceedings of the 2Nd ACM International Workshop on Quality of Service &Amp; Security for Wireless and Mobile Networks, ser. Q2SWinet '06. New York, NY, USA: ACM, 2006, pp. 33–36.

[10] H. Sengar, D. Wijesekera, H. Wang, and S. Jajodia, "Voip intrusion detection through interacting protocol state machines," in Proceedings of the International Conference on Dependable Systems and Networks, ser. DSN '06. Washington, DC, USA: IEEE Computer Society, 2006, pp. 393–402.

[11] P. Truong, D. Nieh, and M. Moh, "Specification-based intrusion detection for h. 323-based voice over ip," in Signal Processing and Information Technology, 2005. Proceedings of the Fifth IEEE International Symposium on. IEEE, 2005, pp. 387–392.

[12] P. Thyda and A. Koki, "A protocol specification-based intrusion detection system for voip and its evaluation," IEICE transactions on communications, vol. 91, no. 12, 2008, pp. 3956–3965.

[13] H.-C. Lin, M.-K. Sun, H.-W. Huang, C.-Y. H. Tseng, and H.-T. Lin, "A specification-based intrusion detection model for wireless ad hoc networks," in Proceedings of the 2012 Third International Conference on Innovations in Bio-Inspired Computing and Applications, ser. IBICA '12. Washington, DC, USA: IEEE Computer Society, 2012, pp. 252–257.

[14] T. Roosta, D. K. Nilsson, U. Lindqvist, and A. Valdes, "An intrusion detection system for wireless process control systems," in Mobile Ad Hoc and Sensor Systems, 2008. MASS 2008. 5th IEEE International Conference on. IEEE, 2008, pp. 866–872.

[15] R. Mitchell and I.-R. Chen, "Specification based intrusion detection for unmanned aircraft systems," in Proceedings of the first ACM MobiHoc workshop on Airborne Networks and Communications. ACM, 2012, pp. 31–36.

[16] N. McKeown et al., "Openflow: Enabling innovation in campus networks," in SIGCOMM Comput. Commun. Rev., no. 2. ACM, 2008, pp. 69–74.

[17] Openflow switch specification. Open Networking Foundation. [Online]. Available: https://www.opennetworking.org/sdn-resources/onf-specifications/openflow [retrieved: Nov., 2012]

[18] Ryu sdn framework. [Online]. Available: http://osrg.github.io/ryu/ [retrieved: Nov., 2014]

[19] S. Shin et al., "Fresco: Modular composable security services for software-defined networks," in Proceedings of the 20th Annual Network and Distributed System Security Symposium (NDSS'13), 2013.

 ISBN: 978-1-61208-376-6