Capitalizing on SDN-Based SCADA Systems: An Anti-Eavesdropping Case-Study

Eduardo Germano da Silva, Luis Augusto Dias Knob, Juliano Araujo Wickboldt,
Luciano Paschoal Gaspary, Lisandro Zambenedetti Granville, Alberto Schaeffer-Filho
Institute of Informatics
Federal University of Rio Grande do Sul
Porto Alegre, Brazil
Email: {eduardo.germano, luis.knob, jwickboldt, paschoal, granville, alberto}@inf.ufrgs.br

***Abstract*—Power grids are responsible for the transmission and distribution of electricity to end-users. These systems are undergoing a modernization process through the use of Information and Communication Technology (ICT), transforming the electric system into Smart Grids. In this context, Supervisory Control and Data Acquisition (SCADA) systems are responsible for the management and monitoring of substations and field devices. In this paper, we investigate the use of SDN as an approach to assist in the modernization of SCADA systems. We discuss its possible benefits, such as simplified management of power system resources. Moreover, SDN can facilitate the creation of new network applications that previously, with traditional networks, were more complex to be implemented. To illustrate the benefits of the use of SDN in SCADA, we designed a mechanism that aims to prevent a possible eavesdropper from fully capturing communication flows between SCADA components. The mechanism was implemented as an SDN-based application for SCADA systems that uses multipath routing, which relies on SDN features to frequently modify communication routes between SCADA devices. Further, we performed an experimental evaluation to verify the impact and performance of the mechanism in the SCADA network.**

I. Introduction

Electric power grids are undergoing an intense modernization process through the use of Information and Communication Technology (ICT), transforming the electric system into Smart Grids [1]. Typically, power plants are complex environments, comprising thousands of devices that assist in the monitoring and control of resources, which rely on automated processes for the operation of the grid. In this context, Supervisory Control and Data Acquisition (SCADA) systems are widely distributed systems used in the management and monitoring of automated processes and components, *e.g.*, substations and field devices, in the electrical grid [2].

SCADA systems require technologies that facilitate resource management and allow the monitoring of the proper operation of communication networks [3]. In particular, Software-Defined Networking (SDN) is as a promising approach that can assist in the modernization of SCADA communication networks [4]. Some preliminary research efforts have advocated the use of SDN in SCADA [3], [5]. SDN offers an architecture that can facilitate the management and configuration of network devices. An SDN architecture can simplify network operation and optimize its performance compared to traditional management techniques, since network programmers are provided with a comprehensive view and direct control of the network, through a centralized controller device [6].

The purpose of this paper is twofold: *(1)* to investigate the advantages of using SDN in SCADA systems, and *(2)* to demonstrate a concrete case-study of an SDN application that can be used to increase privacy in SCADA. Initially, we discuss the possible benefits that can be achieved through the adoption of SDN into SCADA systems, such as simplified configuration of devices and better management of power system resources. Also, SDN characteristics can assist in the growth of the power system network infrastructure, facilitating the creation of new network applications that previously, with traditional architectures, were more complex to be implemented.

Furthermore, to illustrate the benefits of the use of SDN in SCADA, this paper also presents a case-study scenario describing a mechanism to enhance the privacy of information that is carried over SCADA networks. Our solution aims to prevent a possible eavesdropper in the network from fully capturing communication flows between SCADA components. To do this, we present an SDN-based network application for SCADA systems that uses *multipath routing*, which relies on SDN features to frequently modify communication routes between SCADA devices. This allows packet exchange between two end-devices in a SCADA network to be performed through more than one communication route. Further, evaluation results are presented, which measure the impact and performance of the implemented mechanism.

This paper is organized as follows: Section II presents some background about SCADA systems and SDN, and discusses the benefits of using SDN in SCADA systems. Section III describes a case-study scenario for the use of SDN in SCADA and the multipath routing strategy. Section IV presents the evaluation results and a performance analysis of our mechanism. Section V describes the related work. Finally, Section VI concludes the paper.

II. SDN-Based SCADA Systems

Smart Grids are power distribution networks that depend on an increased level of automated monitoring and control, often exchanging data over IP-based communication protocols [1]. Compared to legacy power systems, Smart Grids rely on bidirectional and high-speed communication technologies to provide more flexible and accurate energy management [7]. Supervisory Control and Data Acquisition (SCADA) systems

are considered one of the main components of the power grid, and allow the control, management and acquisition of remote data from equipment and power substations. Due to their increasing complexity, SCADA systems demand techniques to simplify the management of system equipment, to ensure performance requirements, to automate their operation and to offer support for resilience functionality [3].

A. SCADA Systems

SCADA systems are used in critical infrastructures such as power plants, water supplies, oil and gas facilities. In power plants, in specific, SCADA systems are used to control and monitor essential equipment for energy delivery. These systems comprise distributed components, which are often dispersed around thousands of kilometers and allow the continuous data acquisition that is critical to the functioning of the power grid [2]. These systems are organized in two main types of components: the control center, which includes the MTU (Master Terminal Unit), and substations geographically dispersed. The core of the SCADA system is the MTU. This component gathers information about the system operation and displays it to SCADA operators. Further, the MTU is capable of sending commands to substations to configure field devices in a remote way. Substations comprise a RTU (Remote Terminal Unit), which manages field devices such as sensors and actuators that are responsible for telemetry of automated processes and for the execution of commands sent by the MTU, and transmit data to the MTU. Figure 1 shows a typical SCADA architecture with the control center and its substations.

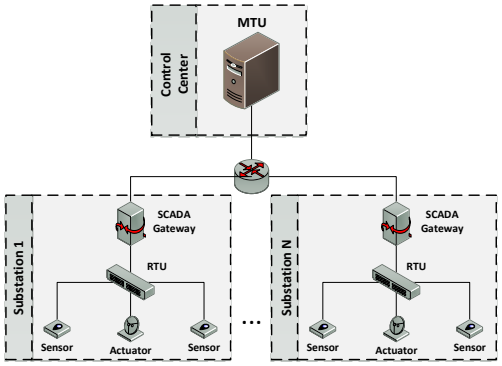

Fig. 1. Typical SCADA architecture.

Due to the increasing number of interconnected devices, sensors and actuators, and also the larger volume of information exchanged between components, SCADA systems are becoming more complex. In their majority, components of the SCADA system communicate through protocols originally developed for process automation, which have been ported to operate over the TCP/IP stack [8], *e.g.*, MODBUS TCP/IP [9], DNP3 over TCP/IP [10] and Ethernet/IP [11]. Further, modern SCADA systems are connected directly or indirectly to the Internet. Consequently, SCADA systems are susceptible to threats such as malwares and cyber-attacks. Therefore, a SCADA system must take into consideration aspects of system security, like timeliness, availability, integrity of data and components, and confidentiality [2]. Such systems require the ability to flexibly manage and configure a growing number of components and to monitor data flows across their communication networks, in order to prevent cyber-attacks, intrusions or malware from compromising the system operation, since the malfunctioning of the grid can result in major disasters. Thus, we aim to investigate the use of network management techniques in general, and SDN in particular, to assist in the management of SCADA communication networks.

B. SDN and OpenFlow

Software-Defined Networking (SDN) is an emerging architecture for managing, monitoring and controlling switching devices and network traffic [4], [6]. SDN decouples the network control and the forwarding planes. This can simplify network management, offering to network programmers a comprehensive view of the network and the ability to control network devices from a centralized controller [12]. The SDN architecture consists of the following components: *(i) switches:* data forwarding devices that use a flow table to forward packets; *(ii) flow table:* a table that contains a list of flow entries and associated actions to be applied to the respective flows; *(iii) controller:* software component that manipulates and controls the flow tables of switches; and *(iv) secure channel:* communication channel that connects each switch to a controller and allows the controller to install flow rules. Figure 2 illustrates the SDN architecture and its components.

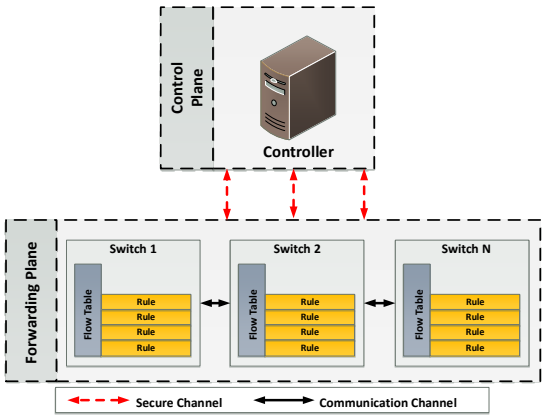

Fig. 2. SDN architecture.

To standardize the communication between the controller and the switches, the OpenFlow protocol has been proposed [13]. OpenFlow defines how applications running on the controller can program the flow table of each network switch. The communication between the controller device and the switches is performed over a secure channel, enabling the controller to manage and control all network switches, and to send and receive control messages to and from the switches.

C. Discussion: Investigating the Benefits of SDN in SCADA

In this paper we advocate the use of SDN to assist in the management of SCADA systems. SDN can enable more flexible SCADA networks, since the addition of new policies and services requires changing the controller only [5]. Arguably, the use of SDN in SCADA will support more resilient systems, as solutions to mitigate attacks and other threats can be more easily implemented in the controller.

TABLE I. BENEFITS OF SDN-BASED SCADA SYSTEMS FOR FCAPS MANAGEMENT.

Property	Description
Fault	SDN enables the implementation of mechanisms for increasing the resilience of SCADA systems. The centralized view of the controller allows more efficient fault detection, isolation of affected components, and remediation of abnormal operation in the SCADA network.
Configuration	The OpenFlow protocol provides a standard API for the correct configuration of new devices added to the SCADA network and their communication protocols. This can reduce the configuration overhead of these components.
Accounting	The measurement capabilities of the controller provides the ability to collect metrics and statistics about the network traffic. This information can be used in dimensioning the capacity of the SCADA network, to plan the growth of the power grid, or to detect abuses in resource usage.
Performance	SDN can facilitate the use of QoS policies in SCADA systems, to perform load balancing between communication links and to optimize the operation of system components.
Security	The controller also permits the implementation of applications that can add more security to the SCADA system, *e.g.*, in terms of detecting malicious activity or protecting the information exchanged in the SCADA network. To illustrate this, Section III presents an anti-eavesdropping SDN-based application for SCADA.

SCADA systems can benefit from the characteristics of SDN in several ways, such as:

- **Flexibility:** SDN enables more flexible systems [14], in which applications and protocols can be modified via a centralized controller. In SCADA systems, this will permit easily adding new field devices or upgrading existing applications in the SCADA network.
- **Centralized management:** the centralized control plane offers a global view of the network. Thus, an SDN-based SCADA control center will be able to manage not only field devices, but also monitor and control the network that interconnects system devices.
- **Standard API:** the OpenFlow protocol provides a standard API for controlling network switching devices. In SCADA networks, this standardization will permit a better integration of geographically dispersed equipment from different vendors.
- **Programmability:** via the controller it is possible to easily add new functionality to the network on demand. In SCADA, this will allow creating a range of customized services, *e.g.*, to control the reading frequency of field devices at a specific time of day.

Further, the characteristics of SDN can also enhance FCAPS (*fault*, *configuration*, *accounting*, *performance* and *security*) management in SCADA systems. Table I indicates some of the possible benefits of SDN-based SCADA systems for each FCAPS properties.

III. ANTI-EAVESDROPPING IN SDN-BASED SCADA

This section presents our *multipath routing* strategy for SDN-based SCADA systems, and how it can be used to improve privacy in these systems. Firstly we present a case study scenario as a motivation for developing network applications that improve privacy in SCADA. Then we describe our multipath routing strategy to SCADA networks, using SDN.

A. Case-Study Scenario

Consider a SCADA system responsible for controlling the electrical grid of a particular region, where a central control station monitors and manages multiple substations. The network topology of this SCADA system contains redundant communication routes, which allow, in case of a communication link breakdown, the exchange of messages between system components through an alternative path. In this paper we assume that the communication network connecting the SCADA components can be implemented using an SDN network. All components of the SCADA system, control center and substations, communicate through a high-speed wired SDN network, using a legacy communication protocol. The protocol adopted was ported to run over the TCP/IP stack and does not provide a secure communication between system devices, *i.e.*, communication is not encrypted, which allows a person without permission to eavesdrop the messages that travel in the SCADA network.

Eavesdropping is a network layer attack that consists in the interception of packets that travel over the network, with the intention of collecting confidential information. Unencrypted and weakly encrypted information exchange allow an individual attacker to intercept data transmitted over the network if he or she has access to the communication medium. In other words, an eavesdropper can obtain passwords, view the content of message exchanges and confidential information if the eavesdropper can access the local network.

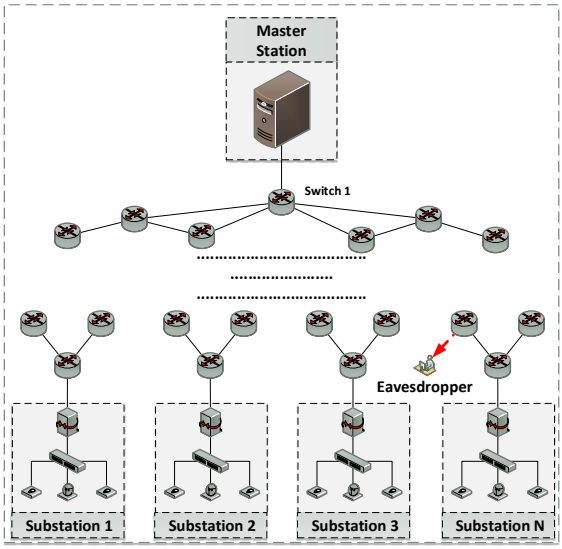

Fig. 3. Case study scenario.

SCADA systems, largely, use insecure and unencrypted communication networks [15]. In this context, through the placement of listening devices well positioned in the network, an eavesdropper can easily, for example, capture instructions forwarded from an MTU to sub-MTUs, RTUs, or even relevant information from sensors and actuators in the system [16]. Moreover, an eavesdropper can also collect the end-devices IP address and the access credentials of the SCADA system. If the IP address of the SCADA server is known by an attacker, it can be easily taken down or shutdown using a traditional

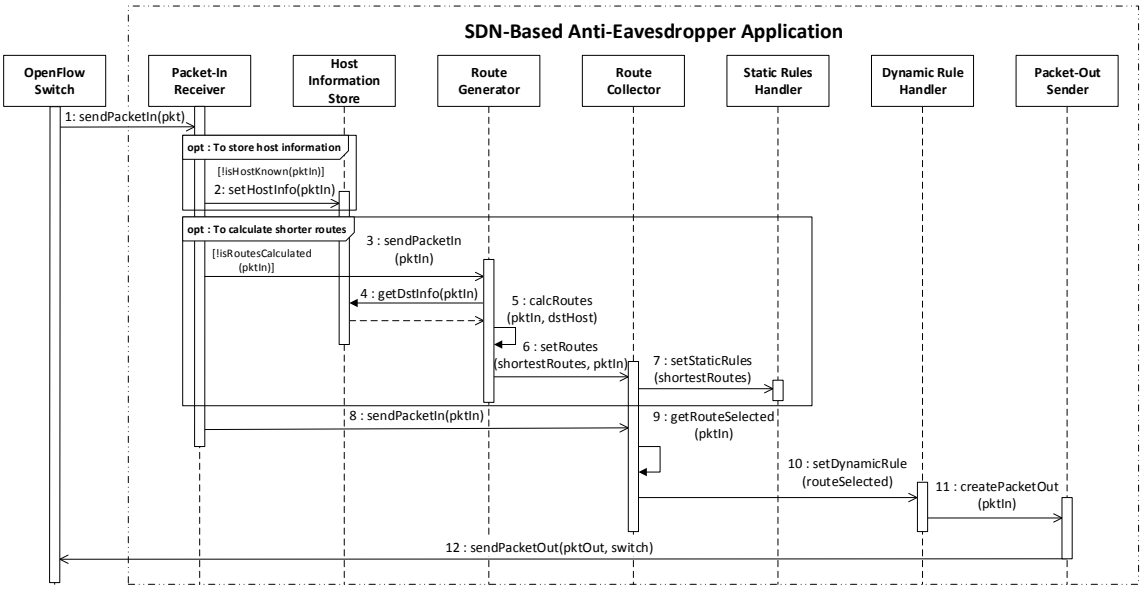

Fig. 4. Sequence diagram of the multipath routing algorithm.

Denial of Service (DoS) attack [17]. Finally, with the access credentials of a system, a person can control substations, and steal corporate data and delete system files [17]. Figure 3 gives an overview of the scenario presented in this case study.

B. SDN-Based Anti-Eavesdropping Approach

Most routing algorithms used nowadays allow communication between devices through a single path for a long period of time [18]. In case a listening device is placed in this path, a large number of messages may be intercepted. This may facilitate message decryption if cryptography has been used. Furthermore, some attacks perform traffic analysis in communication patterns over encrypted connections, which decrease the effectiveness of cryptography techniques [19]. A communication network can be more efficient and robust if it has one or more extra paths for information flows, thus increasing resilience, security, fault tolerance and load balancing [20]. The technique of *multipath routing* was first proposed in the 1970's, and since then it has been used for different purposes in different types of networks [21].

In this paper, we present an SDN-based mechanism that can thwart eavesdropping attacks. Our mechanism uses the facilities provided by SDN to aid SCADA networks in the defense against unauthorized interception of flows by dispersing traffic across multiple paths. Thus, each route transmits only a portion of the packets exchanged during communication. The SDN controller knows the switches *a priori*, but identifies the end-hosts on demand. It also takes advantage of redundant network connectivity, allowing a source device to use multiple routes to communicate with a target device.

Considering the topology illustrated in Figure 3, and that the *master station* starts a continuous communication flow with a specific *substation N*, the proposed algorithm works as follows (each step below is depicted in the diagram in Figure 4). When the first data packet of a flow is received by the first switch (switch 1, in Figure 3), the switch will send a `Packet-In` message to the controller (step 1). If the *master station* is not known to the OpenFlow controller, information about this host (*master station*) will be stored, including its IP address, MAC address and the port number of the switch in which it is connected (step 2). Next, the algorithm calculates the N shortest routes between the *master station* and the specific *substation*, if these routes have not been calculated yet (step 3). To calculate the N shortest routes, information about the destination host is retrieved (step 4). Using the information retrieved from the source and destination hosts, Dijkstra's algorithm [22] is used to calculate the N shortest routes (step 5), in N stages. Considering $N = 2$, in the *first stage*, Dijkstra's algorithm identifies the shortest route between the two network devices, and subsequently all link costs have their weight increased by a tenfold factor. Immediately after, in the *second stage* (and with the link costs increased), Dijkstra's algorithm is executed again to return the second shortest route. Finally, also in the second stage, the link costs of the first route are reestablished to the original values. As explained later, the N shortest routes will be used to deliver a communication flow using different paths and, for this reason, they are stored to be used afterwards (step 6).

Our strategy also relies on the use of *timers* specified by OpenFlow. Using the `Hard TimeOut` timer, which is represented in seconds, we define two types of rules to realize the multipath routing technique: *dynamic rules* and *static rules*. On the one hand, dynamic rules are defined with a low value for `Hard TimeOut`, allowing this kind of rule to expire often. On the other hand, *static rules* do not expire over time, thus they do not need to be reinstalled again on switches. Therefore, after storing the N shortest routes between two hosts, the algorithm will immediately install the static rules on the switches that belong to the N paths (step 7), except on the switches that splits the N shortest routes chosen for communication (which were calculated above).

After installing the static rules, the algorithm retrieves information about the N shortest routes (step 9). Route selection is performed via an internal flag, which allows the

alternation between routes. For example, considering only two paths ($N = 2$), if a flow is transmitted on the first route, when the dynamic rules expire and are reinstalled, the flow will be transmitted on the second route, and *vice versa*. To achieve this, the algorithm must install dynamic rules only on the switch that splits the N routes (step 10 – and switch 1 in Figure 3). Dynamic rules expire according to the value of the `Hard TimeOut` timer. For example, if the timer is set to 5 seconds, dynamic rules will expire and will be reinstalled every 5 seconds. Finally, with the information from the `Packet-In` message, the algorithm generates a `Packet-Out` message (step 11) and sends it to the switch that initiated the interaction with the controller (step 12).

If the controller receives again a `Packet-In` message indicating that the *master station* wants to restart the communication with the same *substation*, the controller will install only dynamic rules on the switch that splits the N routes. In this case, according to the diagram in Figure 4, after receiving a `Packet-In` message (step 1), the algorithm will only select the desired route (step 9) to install the dynamic rules on the corresponding switch (step 10), generate a `Packet-Out` message (step 11) and send it to the switch that requested the interaction (step 12). The pseudocode for the multipath routing strategy described above is illustrated in Algorithm 1. As discussed in the next section, this mechanism is able to prevent an eavesdropper from capturing entire communication flows between the master station and specific substations.

Algorithm 1 SDN-Based Anti-Eavesdropper PseudoCode

```
1:  procedure MULTIPATH(pktIn, switch)
2:
3:      if (!isHostKnown(pktIn)) then
4:          setHostInfo(pktIn)
5:
6:      if (!isRoutesCalculated(pktIn)) then
7:          dstHost ← getDstInfo(pktIn)
8:          shortestRoutes ← calcRoutes(pktIn, dstHost)
9:          setRoutes(shortestRoutes, pktIn)
10:         setStaticRules(shortestRoutes)
11:
12:     routeSelected ← getRouteSelected(pktIn)
13:     setDynamicRule(routeSelected)
14:     pktOut ← createPacketOut(pktIn)
15:     sendPacketOut(pktOut, switch)
16:
17:     return None
```

IV. Prototype and Experimental Results

In this section we outline the prototype implementation and present the experimental setup, including the topology as well as the description of each scenario used in the experiments. Then, we analyze the performance of the proposed solution.

A. Prototype Overview

A prototype for the SDN-based anti-eavesdropping application was built using the POX OpenFlow controller. Figure 5 depicts the components that comprise this application. These include: `Packet-In Receiver`: component responsible for capturing `Packet-In` messages received by the OpenFlow controller; `Host Information Store`: upon receiving a `Packet-In` message, in case there is no information about a given element in the network, this component stores relevant information for that device; `Route Generator`: component responsible for calculating the N shortest routes between two devices in the network; `Route Collector`: component that stores the routes calculated, and that selects a specific path for communication; `Static Rules Handler`: component that creates the static rules that will be installed in all switches along the N shortest routes between two devices, except in the switch that splits these paths; `Dynamic Rule Handler`: component that defines the dynamic rules that will be installed in the switch that splits the communication routes between two devices; `Packet-Out Sender`: after completing the process of route definition, this component sends a `Packet-Out` message to the switch that sent the request to the controller.

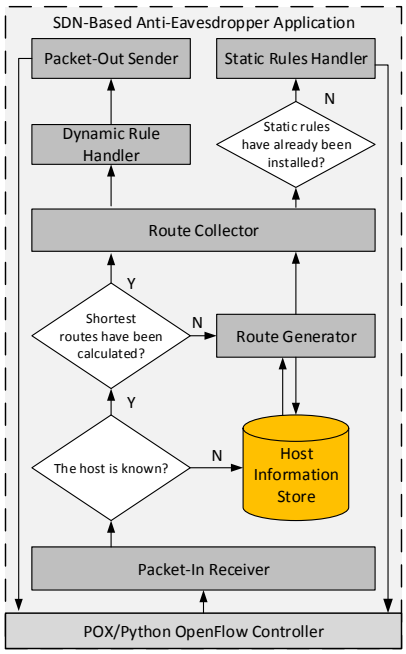

Fig. 5. Anti-eavesdropping application.

B. Experimental Setup

The scenarios used in the performance analysis of our prototype consider a network topology based on studies of the power grid in countries like USA [23] and Italy [24]. Our network topology contains redundant communication paths, *i.e.*, different paths that lead to the same destination. The network topology consists of 10 switching devices and a number of hosts, which are responsible for simulating the behavior of SCADA system components. The topology was created using Mininet [25]. Mininet is a network emulator that enables the creation of virtual SDN/OpenFlow networks, including virtual hosts, switches, controllers, and links. The switches in the topology used in our experiments were numbered from 1 to 10. Furthermore, there is a *master station* directly connected to switch 1 and one power *substation* directly connected to each one of the nine remaining switch devices. Figure 6 illustrates the configuration of the network topology used in our experiments.

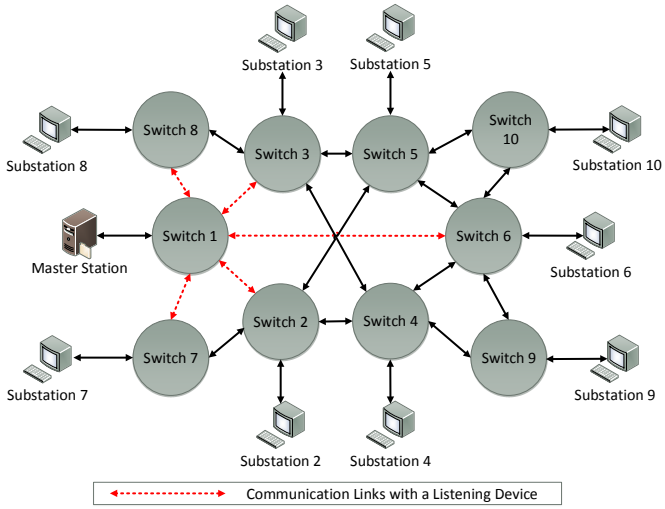

Fig. 6. Configuration of the network topology used in the experiments.

Our experiments consisted of all nine substations sending data simultaneously to the master station in the SCADA network. Each scenario runs for 600 seconds. The communication protocol chosen for message exchange was MODBUS TCP/IP [9]. The substations forward data packets (512 bytes) every 15 seconds, containing information from their respective sensors. This has been carefully chosen to simulate the behavior of a SCADA network, where the substations send periodic information to the master station. The speed of the communication links was set to 10 Gb/s. The initial value of all link costs was defined as 1, which is the default value. Finally, we introduced traffic listeners on 5 communication links that connect switch 1 (which is directly connected to the master station) to switches 2, 3, 6, 7 and 8. These listening devices simulate the behavior of an eavesdropper, and cover all possibilities of communication with the master station.

Further, we defined five scenarios (A, B, C, D and E) to evaluate the performance of our application. The first scenario (A) has an OpenFlow controller with POX default behavior, using the Spanning Tree algorithm [26] for unicast routing with only one communication path between devices. The remaining scenarios (B, C, D and E) use our multipath application, but flows are defined with different values of `Hard TimeOut` timer. This is used to determine how long a flow will follow a particular route before the dynamic rules expire. The value of `Hard TimeOut` in scenarios B, C, D and E is respectively 5, 10, 15 and 20 seconds. In these experiments, the scenarios that use multipath routing were configured to operate with two communication routes ($N = 2$).

C. Evaluation Results

Firstly, we analyzed the routes chosen by the multipath strategy when two specific SCADA components communicate, namely the substation connected to switch 10 and the master station. Figure 7 presents the two best routes selected by the application during the experiments. In scenarios B, C, D and E, the first route selected was the one with the lowest cost, containing only 3 hops, which is presented as *First Route*. Further, in all scenarios, after increasing the cost of the links used in the first route, the second route chosen had 4 hops, presented as *Second Route* in Figure 7.

In order to observe the effects of choosing a given value

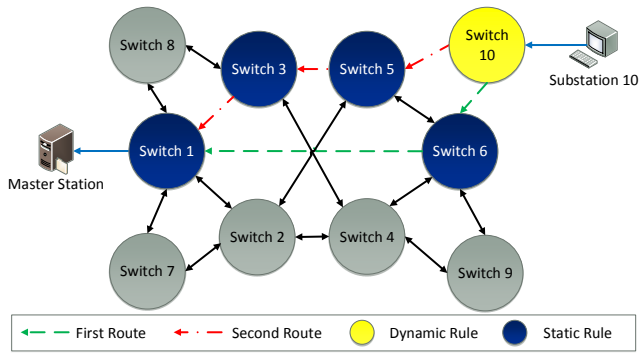

Fig. 7. Anti-eavesdropping communication between components.

for `Hard TimeOut` timer, and conduct a performance comparison between our solution and the default behavior of POX, we defined a set of *metrics*. Initially, we consider the *(i)* total number of flow rules installed at a given moment. We also measure the *(ii)* percentage of packet loss and the *(iii)* amount of `Packet-in` messages received by the controller in each scenario. Further we present the *(iv)* traffic rate in the secure channel. Furthermore, we analyzed the *(v)* amount of exposed communication among each substation and the master station in each scenario. The experiments for each scenario were performed 30 times with a confidence level of 95%.

We compared the number of rules installed in switches at a given time both using POX default behavior (scenario A) and a scenario using the proposed multipath strategy (scenario C)[1]. Figure 8 presents the number of rules necessary to accomplish the communication between *substation 10* and the *master station* in scenarios A and C. Note that POX default solution installs multiple rules simultaneously, reaching a peak of 36 rules after $20s$). However, the multipath strategy maintains a stable number of rules, ranging between 7 and 8 rules. This is due to the lifetime of dynamic rules, which expire often. By analyzing the controller default behavior we noticed that it installs a rule for each type of flow between two devices, *e.g.*, one rule for ARP flows and another for TCP. This has impacted considerably the number of rules in scenario A.

We also analyzed the TCP packet loss in each scenario, which is depicted in Figure 9. The results indicate that the default solution presented lower packet loss, on average 0,5%, thus requiring fewer retransmissions. However, scenarios B, C, D and E presented slightly higher packet loss, respectively 3.1%, 2.7%, 1.3% and 1.1%. Despite that, the measured rate of retransmissions due to packet loss is still considered acceptable. We noticed that most packet retransmissions for scenarios using our multipath application occurred after switching between communication paths.

Further, we measured the amount of `Packet-In` messages received by the controller in each scenario. Compared to the default behavior, the multipath strategy obtained better results, sending less `Packet-ins` to the controller. These results are presented in Figure 10, where 134 `Packet-ins` were received by the controller in scenario A, 95 in scenario B, 97 in scenario C, 94 in scenario D, and 95 in scenario E. This indicates that the multipath application caused less

[1]Although we performed a similar analysis with the other multipath scenarios, these are not shown here due to space constraints.

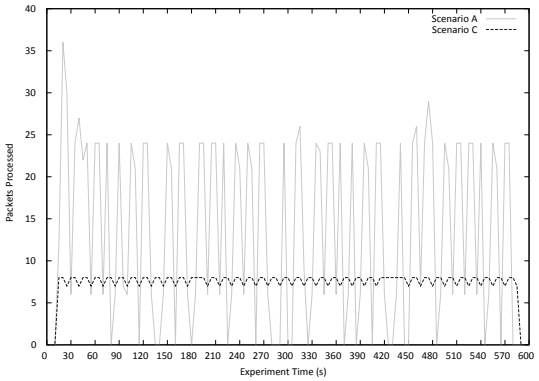

Fig. 8. Number of rules installed during the experiments.

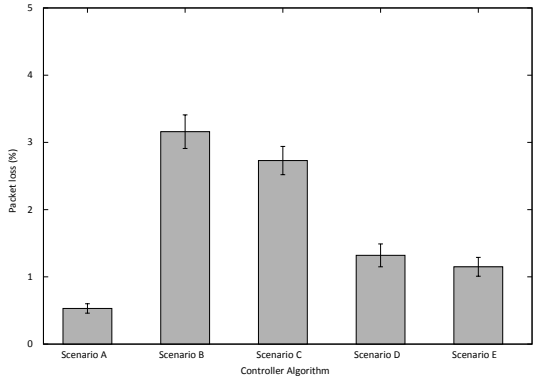

Fig. 9. Packet loss in each scenario.

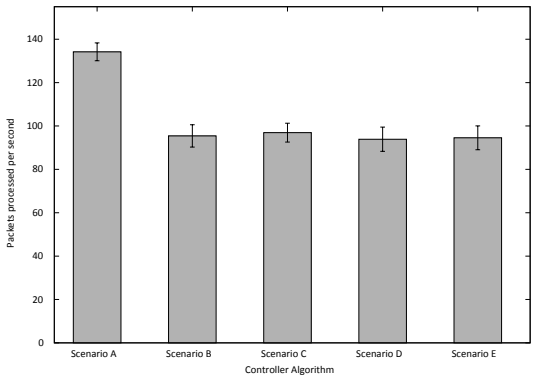

Fig. 10. Number of packets processed by the controller.

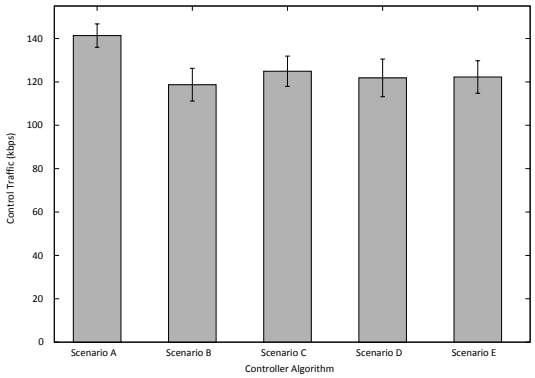

Fig. 11. Traffic generated in the secure communication channel.

processing overhead, compared to the default behavior of POX. However, no significant variation was observed if we consider the lifetime of dynamic rules.

Figure 11 depicts the amount of generated traffic in the secure channel in each scenario. With the POX default behavior, the communication rate between controller and switches was higher, generating up to 138 *kbps* in the secure communication channel. The scenarios using the multipath strategy, in general, generated less traffic between switches and the controller. In particular, the traffic generated in the secure communication channel was 115 *kbps* in scenario B, 122 *kbps* in scenario C, 119 *kbps* in scenario D, and 119 *kbps* in scenario E.

In order to evaluate the ability of preventing an eavesdropper from capturing communication flows, we instantiated listening devices in all five direct communication links to switch 1, which is directly connected to the master station. These listening devices aim to simulate the behavior of an eavesdropper, who positioned himself in privileged points of the SCADA network. We analyzed the amount of packets that were intercepted for each existing communication flow. The end result was similar for all scenarios that use the multipath application. However, using POX default solution, which relies on a single path to accomplish communication, all the information exchanged between the substation and the master station has been exposed. For example, in scenario A it was possible to capture all the information exchanged between substation 7 and the master station, by intercepting the data packets that arrived through switch 7. Instead, using the multipath strategy, an attacker positioned in the same point in the network could intercept only 25% of packets exchanged between substation 2 and the master station, and 75% of packets exchanged between substation 7 and the master station. This is because more than one route is configured to send TCP packets, however all TCP acks are received through one path, *i.e.*, the first shortest route. Table II details the amount of exposed communication in each scenario.

TABLE II. AMOUNT OF COMMUNICATION EXPOSED AMONG SUBSTATIONS AND THE MASTER STATION.

Communication Link / Communication Exposition	**POX default behavior (Spanning Tree Algorithm)**	**Multipath solution and $N = 2$**
Switch 2 to Switch 1	Substation 2 = 100% Substation 4 = 100%	Substation 2 = 75% Substation 4 = 75% Substation 5 = 25% Substation 6 = 25% Substation 7 = 25% Substation 9 = 25%
Switch 3 to Switch 1	Substation 3 = 100% Substation 5 = 100%	Substation 3 = 75% Substation 4 = 25% Substation 5 = 75% Substation 8 = 25% Substation 10 = 25%
Switch 6 to Switch 1	Substation 6 = 100% Substation 9 = 100% Substation 10 = 100%	Substation 6 = 75% Substation 9 = 75% Substation 10 = 75%
Switch 7 to Switch 1	Substation 7 = 100%	Substation 2 = 25% Substation 7 = 75%
Switch 8 to Switch 1	Substation 8 = 100%	Substation 3 = 25% Substation 8 = 75%

D. Discussion

With respect to performance, the proposed multipath application generates a lower workload to the controller when compared to the default behavior, which performs routing by a single path. The packet loss of the multipath strategy can be even lower by increasing the lifetime of dynamic rules. However, increasing the lifetime of dynamic rules allows an eavesdropper to intercept more communication.

Further, as discussed in the previous section, even if the eavesdropper is well positioned in a specific point of the network, it cannot intercept an entire communication between two devices. As shown in Table II, if the routing is done via two paths ($N = 2$), in the worst case, the eavesdropper will intercept no more than 75% of a communication flow. The maximum level of exposure can be minimized if the topology has more redundant paths. The exposure level of a communication which uses our scheme can be calculated as: $\textit{Exposure} = (50 + 50/N)/100$, where N is the number of paths. For example, if we choose 5 paths for routing ($N = 5$), the maximum level of exposure will be 60%.

V. Related work

In this section we present research efforts that are related to our work. In Section V-A we review some work that use SDN in Smart Grids. Section V-B presents studies that aim to ensure grid connectivity with multipath routing. Finally, Section V-C presents research efforts based on network traffic analysis in SCADA systems.

A. SDN in Smart Grids

Research efforts investigating the use of SDN in Smart Grid communication networks are still scarce. Cahn *et al*. [3] discuss how SDN can alleviate some of the current problems in Smart Grid communication networks. The authors present the design and development of a new architecture for communication with grid substations, allowing the network to be auto-configurable, secure and reliable against possible system misconfigurations, through the use of SDN. The SDN-based architecture was called Software-Defined Energy Communication Network (SDECN), and a prototype was developed using the Ryu OpenFlow controller and evaluated in a testbed with real IEDs (Intelligent Electronic Devices). Further, Goodney *et al*. [5] propose the use of SDN to control the communication between devices responsible for measuring electrical waves in the grid, known as PMUs (Phasor Measurement Units). The authors developed an SDN-based network application to facilitate the management of PMUs and provide support for essential features, such as multicast and multi-rate.

B. Multipath Routing in Smart Grids

Differently from our proposal, which alternates the information flow between multiple paths, Hong *et al*. [27] investigate how to transmit duplicate information using multiple communication routes in Smart Grids. In particular, the authors present two multipath routing algorithms, specifically developed for Smart Grids. These algorithms aim to solve the min-max non-disrupting k-path computation problem (M^2NKPCP), in which two routing paths share switches and a possible failure of a specific equipment can disable an entire communication flow. The algorithms calculate totally disjoint routes, and they differ by the trade-off between running time and quality of the output. Also, Vaidya *et. al* [28] focus on other part within the Smart Grid, by using multipath routing more specific in the AMI (Advanced Metering Infrastructure). AMI is responsible for the automatic measurement, management and analysis of energy consumption and distribution to end-users. The study aims to mitigate the problems of security mechanisms in routing protocols in wireless ad hoc networks, through the adoption of multipath routing in wireless mesh AMI networks.

C. Traffic Analysis in SCADA Systems

Barbosa *et al*. [29] investigate the main characteristics of network traffic in SCADA systems. The study looks into the similarity between SCADA traffic and SNMP traffic. The authors analyze nine different datasets, of which six are SNMP traces and three are SCADA traces. From the results, the study concludes that SCADA traffic and SNMP traffic are similar in the sense that devices generate information flows in a periodical fashion. Further, Cheung *et al*. [30] propose an IDS (Intrusion Detection System) based on behavioral models for SCADA networks. This IDS creates models that represent the expected network behavior of the devices that are connected to a SCADA system. The authors point out that SCADA systems have topologies that hardly change over time, and thus the behavior of the devices maintains a pattern. This facilitates the detection of possible attacks that may cause changes to the expected network behavior. Finally, Barbosa [31] presents an IDS that can detect data injection and DoS attacks. This IDS explores the traffic periodicity in SCADA systems.

VI. Conclusion and Future Work

Power grids are responsible for the transmission and distribution of electricity to end-users. However, over the recent years, power grids are becoming more sophisticated, with the aim of increasing their safety, reliability, economical and energy efficiency, and reducing their environmental impact. To assist in the modernization process of electric power grids, we are investigating the use of SDN in SCADA systems. In this context, SDN-based SCADA systems can facilitate the design and development of Smart Grid network applications, by making them more robust and flexible. Also, we presented a concrete case-study of an SDN-based application for multipath routing to increase the privacy of the information that is carried over SCADA networks, and make it more difficult for an eavesdropper to capture communication flows between SCADA devices. The multipath routing mechanism is based on the use of dynamic and static flow rules. We acknowledge that the work presented in this paper has applicability beyond the prevention of eavesdropping. Although we chose to limit the scope of the paper to a single case study, other uses could include load balancing and resilient routing.

Further, we performed an experimental evaluation to verify the impact and performance of the mechanism in the SCADA network. We found that dynamic rules with a shorter lifetime make it more difficult for an eavesdropper to intercept the communication, but a longer lifetime may be advantageous for large-scale SCADA systems, because this reduces the management overhead in the controller. As future work, in order to avoid the min-max non-disrupting k-path computation problem (M^2NKPCP) [27], we intend to refine the algorithm and permit the selection of completely disjoint routes.

Acknowledgement

This work is supported by ProSeG - Information Security, Protection and Resilience in Smart Grids, a research project funded by MCTI/CNPq/CT-ENERG # 33/2013.

REFERENCES

[1] H. Farhangi, "The path of the smart grid," *Power and Energy Magazine, IEEE*, vol. 8, no. 1, pp. 18–28, January 2010.

[2] K. Stouffer, J. Falco, and K. Scarfone, "Guide to industrial control systems (ics) security," *NIST special publication*, pp. 800–82, 2011.

[3] A. Cahn, J. Hoyos, M. Hulse, and E. Keller, "Software-defined energy communication networks: From substation automation to future smart grids," in *Smart Grid Communications (SmartGridComm), 2013 IEEE International Conference on*, Oct 2013, pp. 558–563.

[4] J. Wickboldt, W. De Jesus, P. Isolani, C. Bonato Both, J. Rochol, and L. Zambenedetti Granville, "Software-defined networking: management requirements and challenges," *Communications Magazine, IEEE*, vol. 53, no. 1, pp. 278–285, January 2015.

[5] A. Goodney, S. Kumar, A. Ravi, and Y. Cho, "Efficient pmu networking with software defined networks," in *Smart Grid Communications (SmartGridComm), 2013 IEEE International Conference on*, Oct 2013, pp. 378–383.

[6] C. Monsanto, J. Reich, N. Foster, J. Rexford, and D. Walker, "Composing software-defined networks," in *Proceedings of the 10th USENIX Conference on Networked Systems Design and Implementation*, ser. nsdi'13. Berkeley, CA, USA: USENIX Association, 2013, pp. 1–14. [Online]. Available: http://dl.acm.org/citation.cfm?id=2482626.2482629

[7] W. Wang and Z. Lu, "Survey cyber security in the smart grid: Survey and challenges," *Comput. Netw.*, vol. 57, no. 5, pp. 1344–1371, Apr. 2013. [Online]. Available: http://dx.doi.org/10.1016/j.comnet.2012.12.017

[8] V. M. Igure, S. A. Laughter, and R. D. Williams, "Security issues in scada networks," *Computers & Security*, vol. 25, no. 7, pp. 498 – 506, 2006. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0167404806000514

[9] A. Swales, "Open modbus/tcp specification," *Schneider Electric*, vol. 29, 1999.

[10] X. Lu, Z. Lu, W. Wang, and J. Ma, "On network performance evaluation toward the smart grid: A case study of dnp3 over tcp/ip," in *Global Telecommunications Conference (GLOBECOM 2011), 2011 IEEE*, Dec 2011, pp. 1–6.

[11] P. Brooks, "Ethernet/ip-industrial protocol," in *Emerging Technologies and Factory Automation, 2001. Proceedings. 2001 8th IEEE International Conference on*, vol. 2, Oct 2001, pp. 505–514 vol.2.

[12] S. Shah, J. Faiz, M. Farooq, A. Shafi, and S. Mehdi, "An architectural evaluation of sdn controllers," in *Communications (ICC), 2013 IEEE International Conference on*, June 2013, pp. 3504–3508.

[13] N. McKeown, T. Anderson, H. Balakrishnan, G. Parulkar, L. Peterson, J. Rexford, S. Shenker, and J. Turner, "Openflow: Enabling innovation in campus networks," *SIGCOMM Comput. Commun. Rev.*, vol. 38, no. 2, pp. 69–74, Mar. 2008. [Online]. Available: http://doi.acm.org/10.1145/1355734.1355746

[14] N. Feamster, J. Rexford, and E. Zegura, "The road to sdn," *Queue*, vol. 11, no. 12, pp. 20:20–20:40, Dec. 2013. [Online]. Available: http://doi.acm.org/10.1145/2559899.2560327

[15] Y. Mo, T.-H. Kim, K. Brancik, D. Dickinson, H. Lee, A. Perrig, and B. Sinopoli, "Cyber-physical security of a smart grid infrastructure," *Proceedings of the IEEE*, vol. 100, no. 1, pp. 195–209, Jan 2012.

[16] E. Chikuni and M. Dondo, "Investigating the security of electrical power systems scada," in *AFRICON 2007*, Sept 2007, pp. 1–7.

[17] J. Liu, Y. Xiao, S. Li, W. Liang, and C. L. P. Chen, "Cyber security and privacy issues in smart grids," *Communications Surveys Tutorials, IEEE*, vol. 14, no. 4, pp. 981–997, Fourth 2012.

[18] W. Lou and Y. Fang, "A multipath routing approach for secure data delivery," in *Military Communications Conference, 2001. MILCOM 2001. Communications for Network-Centric Operations: Creating the Information Force. IEEE*, vol. 2. IEEE, 2001, pp. 1467–1473.

[19] J.-F. Raymond, "Traffic analysis: Protocols, attacks, design issues, and open problems," in *Designing Privacy Enhancing Technologies*. Springer, 2001, pp. 10–29.

[20] J. He and J. Rexford, "Toward internet-wide multipath routing," *Network, IEEE*, vol. 22, no. 2, pp. 16–21, 2008.

[21] N. F. Maxemchuk, "Dispersity routing," in *Proceedings of ICC*, vol. 75. Citeseer, 1975, pp. 41–10.

[22] E. W. Dijkstra, "A note on two problems in connexion with graphs," *Numerische mathematik*, vol. 1, no. 1, pp. 269–271, 1959.

[23] T. Overbye and J. Weber, "Visualizing the electric grid," *Spectrum, IEEE*, vol. 38, no. 2, pp. 52–58, Feb 2001.

[24] A. E. Motter, S. A. Myers, M. Anghel, and T. Nishikawa, "Spontaneous synchrony in power-grid networks," *Nature Physics*, vol. 9, no. 3, pp. 191–197, 2013.

[25] B. Lantz, B. Heller, and N. McKeown, "A network in a laptop: Rapid prototyping for software-defined networks," in *Proceedings of the 9th ACM SIGCOMM Workshop on Hot Topics in Networks*, ser. Hotnets-IX. New York, NY, USA: ACM, 2010, pp. 19:1–19:6. [Online]. Available: http://doi.acm.org/10.1145/1868447.1868466

[26] J. P. Russell, D. M. Goodman, C. D. Murton, C. T. W. Ramsden, and J. Shields, "Spanning tree algorithm," Apr. 16 2002, uS Patent 6,373,826.

[27] Y. Hong, D. Kim, D. Li, L. Guo, J. Son, and A. O. Tokuta, "Two new multi-path routing algorithms for fault-tolerant communications in smart grid," *Ad Hoc Networks*, vol. 22, no. 0, pp. 3 – 12, 2014, special Issue on Routing in Smart Grid Communication Networks. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1570870514001012

[28] B. Vaidya, D. Makrakis, and H. Mouftah, "Secure multipath routing for ami network in smart grid," in *Performance Computing and Communications Conference (IPCCC), 2012 IEEE 31st International*, Dec 2012, pp. 408–415.

[29] R. Barbosa, R. Sadre, and A. Pras, "A first look into scada network traffic," in *Network Operations and Management Symposium (NOMS), 2012 IEEE*, April 2012, pp. 518–521.

[30] S. Cheung, B. Dutertre, M. Fong, U. Lindqvist, K. Skinner, and A. Valdes, "Using model-based intrusion detection for scada networks," in *Proceedings of the SCADA Security Scientific Symposium*, vol. 46, 2007, pp. 1–12.

[31] R. R. R. Barbosa, "Anomaly detection in scada systems: a network based approach," Ph.D. dissertation, University of Twente, Enschede, April 2014. [Online]. Available: http://doc.utwente.nl/90271/