ARES: An Autonomic and Resilient Framework for Smart Grids

Yona Lopes[1], Natalia Castro Fernandes[1], Débora Christina Muchaluat-Saade[1], Katia Obraczka[2]
[1]MídiaCom Lab – Universidade Federal Fluminense (UFF)
Niterói, RJ – Brazil
[2]University of California, Santa Cruz
Santa Cruz, California, USA
{yona,natalia,debora}@midiacom.uff.br, katia@soe.ucsc.edu

***Abstract*—In smart grids, the broad use of Distributed Energy Resources (DERs) in distribution networks introduces the need for protective relaying schemes similar to those used in high-voltage networks. The introduction of intermittent DERs, as solar panels, also requires more autonomic and dynamic SCADA networks. Hence, power systems experience an increased demand for resilience in the distribution communication network. However, resilience methods currently in use still cannot meet those protection requirements. This work proposes ARES, a framework for autonomic and resilient communication for smart grids. ARES provides resilient, robust, and flexible communication for smart grids with Software Defined Network (SDN). Our proposal also provides autonomic services for SCADA that can improve smart grid application performance and efficiency. ARES fault resilience module is implemented and tested using Mininet 2.2.1 and RYU controller and presents maximum recovery time of 610 microseconds, which is an important advance compared to other proposals. In addition, ARES is transparent to end devices, keeping compatibility with legacy measurement and actuation devices.**

I. Introduction

Smart grids integrate the electrical power grid with information and communication technologies, deeply changing the system from generation to consumers. With smart metering deployment the utility distributed control system (DCS) reaches residential consumers. Thus, smart communication among all devices becomes a requirement, as the amount of network devices and the communication network traffic increases.

This scenario requires an intelligent distributed system for management and monitoring of automated processes and components, as substations, field devices, and smart meters. The broadly used system is called SCADA (Supervisory Control and Data Acquisition). However, as detailed in [1], high cost and low interoperability make traditional SCADA based solutions impractical for large scale installations, effectively limiting their adoption to the monitoring and control of large plant and mission critical machinery. Nevertheless, as mentioned in [2], the modernization of SCADA communication networks could be a solution. We agree on this issue since assumptions outlined in [1] are ensured, as interoperability, security [3], easy device discovery, easy network configuration, etc. Although these are very important points, Silva et al [2] only address security and easy network configuration.

Moreover, aiming at sustainability, smart grids proposal comprises the deployment of Distributed Energy Resources (DER), as solar panels and wind generators. DERs can be located in residential consumers and can use smart meters to send/receive information. This kind of generation is intermittent, resulting in a dynamic system that only requests network resources in some periods of day. Also, the broad acceptance of smart grids and DER in electrical distribution networks claims for protective relaying schemes, called protection and control systems. They should be similar to those used in high-voltage networks [4], which are strongly based on communication networks. For instance, as detailed by Ali et al. [5], real-time energy management automation in microgrids is needed to improve power quality but is a challenging issue as it requires a robust and fast communication architecture. IEC 61850 [6] based communication architecture for microgrid automation has been addressed as a solution by Ali et. al [5] and many authors [7], [8], [9], likewise requires fast communication for protection and control systems. These proposed papers have very innovative and motivating solutions, which advanced the state of the art for electrical networks, but authors consider that there will be a robust network (failure-free) and do not address communication challenges.

For a correct implementation of protection and control systems, electrical protection messages must be sent within milliseconds time constraints [10], which demand the provision of Quality of Service (QoS) in smart grid communication networks [11]. Current failure recovery protocols, however, usually do not meet these delay requirements [4], [11], [12], [13]. Furthermore, other methods provide instant fault-tolerance at the cost of flooding the network with redundant packets or having rigid deployment assumptions that do not scale [12].

This paper proposes a new autonomic and resilient framework for smart grids, called ARES. The proposed solution presents an architecture based on Software Defined Network (SDN) with OpenFlow [14] that meets the rigid electric power grid protection requirements from generation to consumers and improves traditional SCADA. ARES introduces a new API that allows supervisory and control services to dynamically modify the network. Hence, new power grid control services are available in the ARES management plane, creating a new

978-3-901882-89-0 @2017 IFIP

generation of SCADA, which we call SCADA-NG. ARES performs a real-time failure recovery and reports failure events to SCADA-NG, showing the recovery path and the failure link/device. Besides, it automatically identifies connected and disconnected devices, notifying SCADA-NG. This provides scalability and simplicity, because the network becomes autonomic by enabling the connection of end devices without a manual configuration. ARES implementation improves the traditional SCADA system and SCADA networks. Another important point is that ARES calculates and sets layer-2 multicast trees in order to reduce the impact of layer-2 flooding [10][1].

One of the main ARES contributions is to reduce failure recovery time to microseconds, a recovery time much smaller than the one provided by Rapid Spanning Tree Protocol (RSTP), which is the most broadly used protocol at the link layer for failure recovery. Likewise the network delay during a failure meets the rigid time constraint specified by IEC 61850 [10]. Therefore, proposals such as presented in [5], [7], [8], [9] can actually be implemented. ARES also quickly reports failure situation to the SCADA-NG. Moreover, ARES is transparent to end devices, which do not require any hardware or software modification, keeping compatibility with legacy measurement and actuation devices.

ARES failure recovery algorithms were implemented and tested using the RYU controller and Mininet 2.2.1 network emulator [15]. The main objective of the performed tests was to analyze recovery time, network delay during a failure, and time to report a failure, using the proposed framework.

The remaining sections of the paper are organized as follows. Section II presents smart grid time constrains and recovery time protocols. Related works are presented in Section III. Our proposal is presented in Section IV. Section V shows experimental results. Finally, Section VI presents our conclusions.

II. Smart Grid Quality of Service

Smart grid is an innovative solution for electrical networks that consists in an integrated architecture for generation, transmission, and distribution systems. This architecture is strongly based on communication networks [13], replacing traditional control cables. Thus, electrical power system endpoints, such as meters, DERs, Electric Vehicle (EV), and batteries, are interconnected by communication networks.

This new electrical network design brings many advantages, such as data information dissemination that enables greater control, management, and electrical system protection. This is possible due to all device interconnection. Therewith, new smart grid energy applications emerge such as Demand Response (DR); home energy management systems; systems for design and management of microgrids and Virtual Power Plants (VPPs); systems for real time load shedding; etc. Many smart grid energy applications have rigid time constraints in terms of communication availability and delay [10]. Therefore, specific characteristics of this new energy-delivery concept have driven several research projects aimed at designing an adequate communication infrastructure to meet the expected QoS for smart grids [4]. For instance, IEC 61850 standard has addressed the problem of DERs insertion (IEC 61850-7-420 [10]), recommending the same time threshold established for substation protection and control. IEC 61850 recommends delays from 3ms to 100ms for protection messages according to the message type[2]. Moreover, in 2010 the United States Department of Energy analyzed communication requirements for smart functions (e.g., Demand Response and DER). They defined reliability levels for each service and maximum delays in the order of millisecond for smart grid protection and control [11].

In addition, since 2005, rigid restrictions have been described by the IEEE 1646 standard [16]. IEEE 1646 addresses delay requirements for some substation operations at as little as 4ms and 5ms, for 60Hz and 50Hz AC frequencies. For applications requiring communication between substations, delay requirements are more relaxed. Thus, remote activation of a protection scheme at a substation must occur within 8ms to 10ms after a fault at that substation has been remotely detected at an adjoining substation.

A. Smart Grid Resilience

A failure in the power electric system produces a big amount of protection and control messages to isolate the fault, which are important to prevent further failures. However, if the network is not properly designed, a big amount of messages may overload the communication network. That would generate a cascading failure, where power grid failures result in communication network failures, which could result in power grid failures again. If protection and control messages are not delivered due to a communication failure, protection equipment cannot work properly. It results in an unprotected system, which may lead to electrical failures of enormous proportions. Even though there were many technological advances in power systems to avoid this issue, resilience guarantee is still a challenge. The broadly used Rapid Spanning Tree Protocol (RSTP) [17] has recovery times up to a few seconds, which are two orders of magnitude far from the demands posed by smart grids [4]. Although there are newer versions of this protocol that feature much better recovery times, they are usually proprietary solutions [4] or they are defined for a specific kind of topology and configuration [18].

Other protocols have been proposed for creating a resilient network that apply to QoS requirements in power systems, such as Parallel Redundancy Protocol (PRP) and High-availability Seamless Redundancy (HSR). Both of them are recommended in [19]. Those protocols ensure packet delivery by duplicating messages and/or duplicating network devices. PRP consists of duplicating messages by two totally

[1]Automation protocols that use destination MAC multicast addresses flood communication networks as layer-2 switches usually do not recognize a MAC multicast address.

[2]Those values are related to GOOSE and SV messages in IEC 61850 standard.

separate and similar networks. If one network is down, the other network that is still up to deliver the message. HSR works similarly, but in a ring topology. A source device sends the same frame over both ring ports. A destination receives two identical frames in the fault-free state. Thus, it is considered that PRP and HSR provide zero recovery time in case of a single component failure.

Nevertheless, RSTP, HSR, and PRP do not scale, as they are unable to face the massive introduction of DER and protection systems in electrical distribution networks [4]. RSTP can only be implemented in a ring topology, with few amount of switches to not slow even more time recomposition. HSR increases network traffic, possibly causing overloads. In addition, with PRP and HSR, end device processing also increases due to the fact that each message has to be processed twice. Further, PRP and HSR require specific network topologies and demand specific implementation in end devices [13]. Furthermore, implementing two similar and independent networks between all devices is a very expensive solution economically viable only for substations. We also highlight that despite the zero recovery time, PRP and HSR only work for a single network failure. Therefore, PRP, HSR, and RSTP would not be able to ensure resilient communications solutions required by [5], [7], [8], [9], becoming unfeasible the deployment of solutions presented by authors.

III. Related Work

A low recovery time is a very important point that has been analyzed in recent works [4], [18]. Selga et al. [4] propose an approach that extends the principles of the TRILL (Transparent Interconnect of Lots of Links, a specification that enables multipathing in data centers) and SPB (Shortest Path Bridging) to specific conditions posed by smart grids. Nevertheless, the authors show in tests that their proposal cannot cope with the most stringent requirements defined in smart grids. Thereby, Selga et al. encourage researchers to solve it by taking into account some principles of the PRP protocol. However, such solutions would result in the aforementioned problems of PRP and HSR. Pustylnik et al. [18] provide an in depth analysis of RSTP performance along with simple equations for estimating network failover and recovery times. The authors also performed tests in a real environment. RSTP as defined by IEEE 802.1D-2004 has better performance than its older version with measured failover time of 89.71ms for 25 switches. Unfortunately, despite improvement, it still cannot achieve a good recovery time. Unfortunately, to achieve 20ms with RSTP, a very specific and proprietary (Siemens) solution is required [18]. Moreover, we are working with 3ms as a threshold because it is the IEC 61850 more rigid delay (GOOSE message for trip command) what leaves 20ms off the threshold needed.

Other important related works proposed the use of SDN for smart grids. Goodney et al. [20] demonstrate that SDN can be used as the underlying technology for power grid networking. They implemented a multicast network for phasor measurement unit (PMU) data and compared the SDN solution to the conventional approach as well as to two other advanced PMU networking methods. They showed that SDN achieves lower latency and optimal network utilization. Sydney et al. [21] propose the use of OpenFlow to provide resources in MPLS WANs. Cahn et al. [22] propose the SDECN (Software-Defined Energy Communication Network), which uses SDN as a solution for substations networks. Also in a substation scenario with OpenFlow, Lopes et al. [13] propose SMARTFlow, which was able to meet the requirements of IEC 61850 by using applications developed as a proof of concept. However, all of them focused on message delay time and not in network recovery time. Pfeiffenberger et al. [12] address the use of OpenFlow 1.3 and fast failover group for power systems. They highlight the advantages of fast failover mechanism for recovery time in power systems. However, few details are presented about algorithms and control logic and authors have not implemented the proposal. Reitblatt et al. [23] propose a solution called Fattire for recovering failures in generic networks, but they do not present a motivational analysis for recovery time, because they performed just a single test that took seconds to recover from a failure. Silva et al. [2] investigate the use of SDN in SCADA systems. Authors affirm that SDN-based SCADA systems can facilitate the design and development of smart grid network applications, by making them more robust and flexible. The authors focuses on an application for multipath routing to increase the privacy of the information that is carried over SCADA networks. Authors do not perform tests targeting at SCADA systems, as the time to report an event to supervisory, neither performance test, as recovery time and network delay. Moreover, the communication protocol chosen for message exchange was MODBUS, which was one of the first protocol for SCADA networks. MODBUS is the simplest and cheapest SCADA protocol, and because of that vendors usually modify their operation in order to implement more functionalities that are not comprised in the protocol. Thus, MODBUS have being increasingly replaced by Distributed Network Protocol 3 (DNP3) and IEC 60870-5, both from the 90s [3], and more recently, by MMS (Manufacturing Message Specification) - IEC 61850. Besides that, IEC 61850 based communication architecture has being indicated for smart grid communication. To achieve interoperability in SCADA networks, a solution has to deal with a standard, modern, and robust protocol.

Our proposal named ARES meets smart grids requirements and achieves an excellent recovery time. Moreover, we dynamically reconfigure all network data according to network state changes if needed. ARES is not topology-dependent and it is able to recover from a variable number of network failures in a timely manner, as long as there is at least one available physical communication path between source and destination.

IV. ARES Proposal

We propose a framework called Autonomic and Resilient Environment for Smart Grids (ARES). ARES main objective is to provide resilient, robust, and flexible communication for smart grids with SDN. Besides, ARES allows an intelligent

interaction between distributed energy resources, loads, and management systems as SCADA. In order to ensure interoperability between smart grid devices and to unify procedures, ARES has been designed according to the IEC 61850 model.

A. ARES and IEC 61850

Manufacturers of DER devices are facing the age-old issues of the choice of communication standards and protocols. In the past, DER manufacturers developed their own proprietary communication technology. However, utilities, aggregators, and other energy service providers start to manage DER devices, which are interconnected with the utility power system. Thereby, coping with these different communication technologies presents major technical difficulties, implementation costs, and maintenance costs. Therefore, utilities and DER manufacturers recognize the growing need to have one international standard that defines communication and control interfaces for all DER devices [10]. Another important point is that we claim that the implementation of IEC 61850 model from a smart meter to the supervisory system will make solutions simpler and interoperable.

B. ARES Architecture

ARES architecture is illustrated in Figure 1. The idea is to segment smart grids in autonomous systems with a responsible controller. Therefore, the number of switches corresponds to one autonomous system, and Figure 1 represents one autonomous systems. ARES is designed with five planes that exchange information with each other. The main goal of our proposal is to provide important information for energy applications, allowing the implementation of an efficient control and supervisory system that may automatically map and group distributed energy resources and loads in the system.

The ARES top plane is called `ARES Management Plane`, where the energy supervisory applications are located. We propose a system located on the top plane, called SCADA Next Generation (SCADA-NG). SCADA-NG extends the current Supervisory Control and Data Acquisition (SCADA) to enclose smart grid energy applications requiring automatic interaction with the core network. In addition, to remote monitoring and substation control, SCADA-NG is responsible for controlling, configuring, and monitoring all smart grid information interacting with systems such as DERs, EVs and smart meters. SCADA-NG is configurable, so the energy utility will only have specific useful energy applications in its supervisory. `ARES Management Plane` exchanges information with the lower plane. As a result, SCADA-NG energy applications can automatically show its components, such as distributed energy resources and loads. This makes energy applications more scalable and flexible allowing SCADA to run smart grid real time applications.

The next plane, called `ARES Control Plane`, is responsible for controlling the communication network. ARES components (detailed in Section IV-C) map, calculate, and configure all network switches. Also, this plane provides the ARES Application Programming Interface (API), responsible for the interaction between SCADA-NG energy applications and ARES components. ARES API and ARES components can be implemented in any OpenFlow controller. They can also be implemented in one or more controllers. Another important highlight is that ARES aims at resilience. Communication faults cannot interfere in electrical power systems. It is necessary that, in case of communication network failure, ARES transparently recovers communications for devices and energy applications.

ARES API provides network services to SCADA-NG such as routing paths; link configuration; load and generation detection such as DERs, EVs, batteries, and meters; definition and configuration of security policies and access control. We designed ARES API to allow the development of new energy applications that rely on the network support. This is a main innovation, since current SCADA applications rely on a network that only interconnects devices but without automatic configuration possibility. ARES provides autonomic services that can improve smart grid application performance and efficiency.

Next to ARES Control Plane, there is the `OpenFlow Control Plane`. This plane presents the core control functions of an OpenFlow network and communicates, using the OpenFlow protocol, with the `Data Plane`. The Data Plane comprises the communication between all physical devices, which are automatically and efficiently configured by the OpenFlow Control Plane. ARES aims at facilitating the communication network design and configuration for the electrical sector with high quality of service.

The lowest plane is where power grid endpoints are located. In that plane, called `Energy Plane`, smart buildings, smart meters, DERs, EVs, and every device that is considered a load, a generator, or a storage item are located. SCADA-NG is designed to efficiently operate and manage energy devices with the MMS protocol (according IEC 61850) and OpenFlow networks.

C. ARES Components and Network Resilience

ARES components work on reactive, proactive, and hybrid configuration using OpenFlow. In proactive configuration, the controller automatically sets rules as soon as a switch is turned on, based on a component algorithm. In reactive configuration, the controller responds to a particular event such as a DER connect or disconnect. Hence, an event automatically triggers reactive actions. However, aiming at providing greater configuration speed, a hybrid configuration may also be used.

`Topology Discovery` is an ARES component used for building a network topology view, which is used for ARES algorithms. For a safe network deployment, we propose `Firewall`, `Access Control`, and `Secure Communication Channel` components, which define access rules and configure network devices through SCADA-NG and ARES API information. `Association` ARES component automatically identifies connected and disconnected devices, notifying SCADA-NG. `Association` and `Topology Discovery` allow a completely smart grid

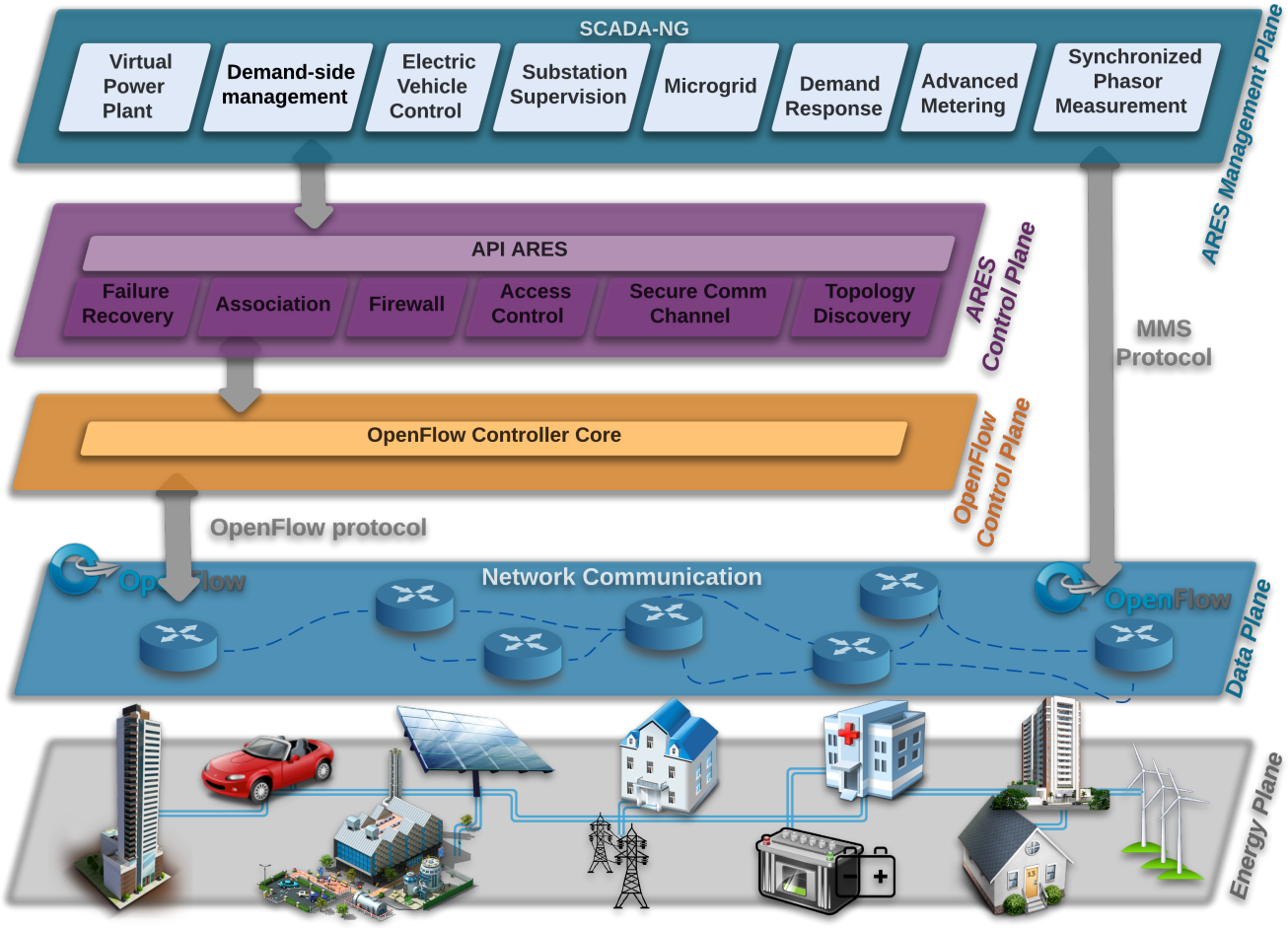


Fig. 1. ARES architecture for smart grids.

network view comprising loads, smart meters, DERs, EV, switches, and so on.

For a resilient communication, we propose the `Failure Recovery` ARES component that is responsible for building and recovering network unicast/multicast links. `Failure Recovery` provides services to: ARES API in order to allow SCADA-NG resources to configure its energy application or its network resources; other ARES components in order to build links; and in the network initialization. This component finds the shortest path from source to destination and finds backup paths that could be used for failure recovery. The `Failure Recovery` component aims at allowing an efficient and transparent failure recovery. To make it possible, the component is based on OpenFlow table groups and uses a specific group called fast failover. Fast failover groups are set with a number of paths with different priorities, where best paths are prioritized. A switch always executes an alive path with higher priority first, thus, in case of failure in the first option, the switch forwards packets to the next alive option (backup path). The idea of ARES is to discover a main path and a backup path using a different output port for every switch of every source-destination pair. Whenever a failure happens, the backup path is automatically chosen and ARES is called to create new backup paths. The `Failure Recovery` component implements Algorithm 1. The inputs of Algorithm 1 are: OpenFlow protocol events, multicast groups, and unicast source and destination addresses. These inputs are provided by ARES API, OpenFlow API, or also by other ARES components. In the end of that algorithm, all flow entries and failure recovery paths are added to switches.

First of all, the path list is empty. Algorithm 1 calculates the shortest path to all possible source/destination pairs or groups defined in multicast and unicast variables. In the network initialization, these variables contain all communication pairs/groups defined in the configuration files of power grids. After network initialization, the algorithm is called, setting new communication pairs/groups defined by SCADA-NG applications. Unpredicted communication requests are evaluated

Algorithm 1: Calculation and configuration of unicast path, multicast trees and failure recovery

```
   Input: ofp_event, topo_net, multicast, unicast
1  paths = []
2  list_flows_temp =
   calc_paths(topo_net, multicast, unicast)
3  list_flows = []
4  for entry in list_flows_temp do
5      topo_net.remove_port(entry)
6      multicast = []
7      unicast = [entry.sw, entry.dst]
8      flows_failover =
       calc_paths(topo_net, multicast, unicast)
9      if len(flows_failover) > 0 then
10         group_entry =
           create_group(entry, flows_failover[0])
11         list_flows.append(group_entry)
12         for i in (1..len(flows_failover) − 1) do
13             list_flows.append(flows_failover[i])
14         end
15     end
16     list_flows.append(entry)
17     topo_net.add_port(entry)
18 end
19 install_paths(list_flows)
```

on the run. If accepted, they are also treated as input for this algorithm.

After calculating all main paths between sources and destinations, backup paths are calculated. As shown in line 4, for each switch of each main path, the algorithm removes the alive output port (line 5), simulating a failure. From line 6 to 8, it calculates the new path from that switch until the destination. If there is a path (line 9), it defines a fast failover group using the original flow entry of the main path and the new flow entry assuming the failure in the switch port (line 10). The fast failover group is then included in a flow list (line 11) as well as the entire path configuration to failure recovery (line 12-14). Lastly, the original entry (main path) is added (line 16), the original topology is restored (line 17), and the calculation continues until all backup paths have been established.

It is worth noticing that the described behavior of this component depends on the OpenFlow version. Indeed, the best results for failure recover depends on the use of group tables, which are available since OpenFlow 1.1. As many OpenFlow facilities are still based on OpenFlow 1.0, we developed an auxiliary algorithm to cope with older versions of OpenFlow, described in Algorithm 2.

Algorithm 2 is called whenever a link failure event is received in the OpenFlow controller. Hence, it is called on demand, while Algorithm 1 is proactive. Moreover, using OpenFlow 1.0, the network initialization only sets the main paths and Algorithm 2 is called only for fault recovery. Hence, the inputs for this recovery algorithm also include a list of the current flow entries in all switches, which we call $paths$. First, the faulty links are detected (line 1) and function $find\_paths$ selects the paths affected by the fault (line 2). Hence, backup paths are calculated (line 3). After that, the flow entries of the affected paths are removed (line 4) and the backup paths are installed (line 5) .

Algorithm 2: Failure Recovery with OpenFlow 1.0.

```
   Input: ofp_event, topo_net, multicast, unicast, paths
1  faulty_links = find_fault(ofp_event, topo_net)
2  faulty_paths =
   find_paths(faulty_links, paths, unicast, multicast)
3  new_paths = calc_paths_1.0(topo_net, faulty_paths)
4  remove_paths(faulty_paths)
5  install_paths(new_paths)
```

V. ARES EVALUATION

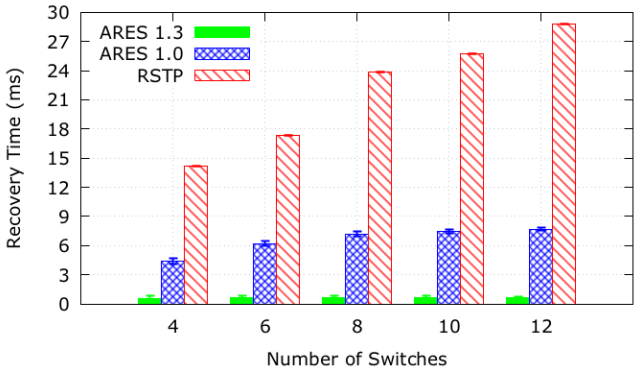
In order to evaluate the failure recovery performance, we developed the `Failure Recovery` ARES component using RYU controller in OpenFlow 1.3 (Algorithm 1) and OpenFlow 1.0 (Algorithm 2). Experiments were emulated using Mininet[3] [15] version 2.2.1. We developed a module in Mininet that builds topologies with end devices uniformly distributed in the network. To emulate traffic from end devices we used GEESE [24], an IEC 61850 traffic generator. We developed the ARES API in Python, running over the RYU controller.

The framework was emulated through virtualization on a computer with Intel Core i5-3210M processor, and 4GB of RAM. Tests were carried simultaneously with three virtual machines, each with one virtual CPU, 1024 MB of memory and running Ubuntu 11.10 operating system. All results have a confidence interval of 95%. The experiments were repeated 30 times and lasted 100 seconds each, including network stabilization, flow table configuration, message exchange, and simulation time. GEESE traffic generator was started after network stabilization. Network failures were generated 60 seconds after the emulation had started. We vary parameters such as number of network end devices, number of switches, topology type, and number of end devices per multicast group.

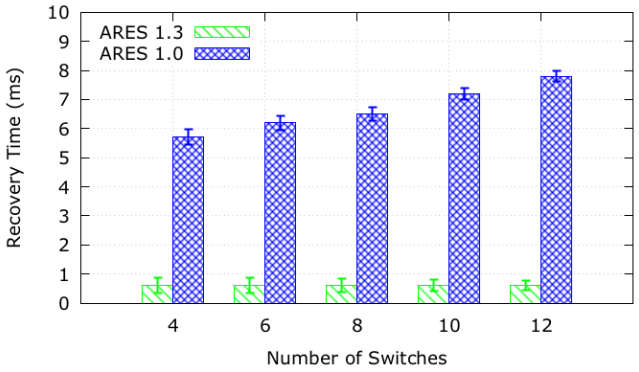
All experiments were tested in ring and mesh topologies. We emulate the communication of a microgrid using IEC 61850 assuming the usage of 4 to 12 switches. The scenario consists of five different multicast groups and 10 end devices distributed uniformly by switches. GEESE was instantiated per end device to generate GOOSE traffic, which is real protection and control traffic. The idea was to create small and large microgrid scenarios.

The recovery time is calculated by subtracting T_{fault} of T_{NC}, where T_{fault} is the delay to transmit a message from

[3]Mininet is a network emulator that creates a realistic virtual network, running real kernel, switch and application code, on a single machine (http://mininet.org/)



(a) Recovery Time in a ring topology.



(b) ARES Recovery Time in a mesh topology.

Fig. 2. Failure Recovery ARES Results

A to B during a failure and T_{NC} is the same delay in normal network conditions.

For comparison purposes, Figure 2(a) also illustrates the improved-RSTP recovery time when using a network ring topology presented by Pustylnik et al. [18]. As Figure 2(b) presents recovery time when using a mesh topology, it is not possible to compare with improved-RSTP that was developed for ring topologies. HSR and PRP are redundancy and not recovery methods. We observe in Figures 2(a) and 2(b) that the recovery time did not exceed 8 ms in the network controlled by ARES with OpenFlow 1.0. This shows that ARES, even with an older version of OpenFlow, meets the rigid requirements of smart grid recovery time showing better times than RSTP, which presents about 15 ms as its fastest time. The RSTP delay is explained due to the characteristic of traditional networks and its distributed control plane, in which each failure results in an exchange of control messages in the network before establishing a new path. The whole process requires time. The value of 8 ms for ARES with OpenFlow 1.0 is due to reactive OpenFlow switch behavior for failures. After a network failure, switches send warning messages to the controller for further new calculation path. Thus, after the controller receives the failure event, it calculates the new path and configures all switches in the path. As the backup path is not pre-configured on the switch, the recovery time is longer.

We can observe in Figures 2(b) and 2(a) that ARES with OpenFlow 1.3 shows excellent recovery times that did not exceed 0.6 ms. We emphasize that these results are very important because they show that ARES based on OpenFlow 1.3 does not depend on the number of switches, confirming that ARES is a scalable solution. This is due to the proactive nature of group tables associated to the ARES failure recovery algorithm. Thus, recovery times depend only on the switch processing time plus the time for it to realize that the main option is inactive.

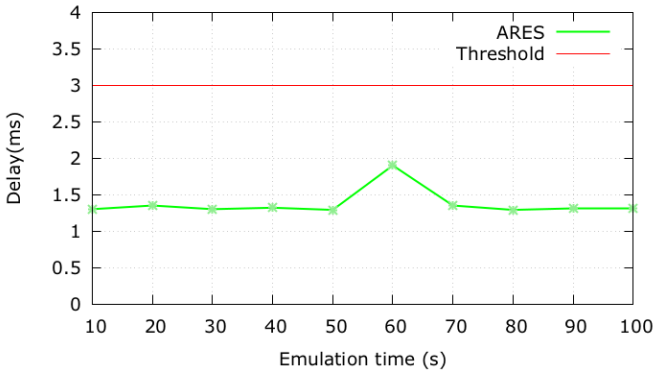


Fig. 3. Network delay during a failure, assuming a ring topology with 12 switches.

Another very important evaluation is shown in Figure 3. Figure 3 shows the average delay to deliver a GOOSE message to all destinations of the multicast group in a ring topology with 12 switches (scenario with greater delay) in stable condition and during a failure. About 60 seconds after the emulation starts, one link is randomly disconnected using the Linux ifdown command. We observe that the average delay to deliver the message from source to all destinations did not exceed 2ms in the network controlled by ARES. This shows that ARES meets the rigid time constraint specified by [10], [11] (3ms for GOOSE messages) even when a network failure happens.

VI. Conclusion and Future Works

Smart grids bring countless benefits such as environmental preservation, reducing human errors, automation and implementation of new capacity, among others. However, there are still problems to be solved. Institutes, companies, and committees have attempted to define communication requirements related to electric power system protection and control and smart grid QoS. Thus, millisecond time constraints have been standardized [11], [10]. However, as cited by [4], the methods currently used for failure recovery, as RSTP and PRP, do not meet these rigid requirements.

Our proposal, called ARES, was implemented and tested. The emulation results showed that ARES reduced the recovery time to microseconds, a much smaller time than RSTP. The proposed system achieves less than 610 microseconds recovery time in the evaluated scenario. Besides ARES is transparent to the end devices and it does not increase device processing

as PRP and HSR. Also, ARES neither duplicates the amount of switches in the network as PRP protocol, nor duplicates the network traffic as HSR, and ARES is a scalable solution.

Besides presenting ARES, this paper showed that our proposal is able to meet smart grid requirements by using applications developed as a proof of concept. We identified and discussed issues related to smart grid QoS, methods used for resilience and their advantages and disadvantages. Even with the use of older OpenFlow versions, ARES showed good results. Moreover, the tests showed advantages of using OpenFlow 1.3 and fast failover groups. Using ARES, it is now possible to pre-configure switches for network failure. In this way, recovery times depend only on the device because they do not need to exchange information with the controller for discovering a backup path.

Our paper focuses on failure recovery components, aiming at presenting the overall architecture. Considering SCADA-NG applications, security components (Firewall, Access Control, Secure Communication Channel, Association), Quality of Service (QoS) aspects, and multiple controllers will be subject of future work. Currently, we are developing SCADA-NG applications and security components.

Also as future work, we intend to test our proposal in a real OpenFlow network. We believe that without machine virtualization, recovery times obtained with ARES can be even better. Also, we aim at evaluating ARES security modules against different types of network attacks, and implement a new generation of SCADA applications using ARES API.

ACKNOWLEDGMENT

The authors would like to thank CNPq, FAPERJ, and CAPES.

REFERENCES

[1] R. Lazzarini, C. Stefanelli, and M. Tortonesi, "Large-scale e-maintenance: A new frontier for management?" in *2013 IFIP/IEEE International Symposium on Integrated Network Management (IM 2013)*, May 2013, pp. 732–735.

[2] E. G. da Silva, L. A. D. Knob, J. A. Wickboldt, L. P. Gaspary, L. Z. Granville, and A. Schaeffer-Filho, "Capitalizing on sdn-based scada systems: An anti-eavesdropping case-study," in *2015 IFIP/IEEE International Symposium on Integrated Network Management (IM)*, 2015, pp. 165–173.

[3] Y. Lopes, N. Fernandes, T. de Castro, V. Farias, J. Noce, J. Marques, and D. Muchaluat-Saade, *Security Solutions and Applied Cryptography in Smart Grid Communications*. Idea Group Inc: IGI Global, 2016, ch. Vulnerabilities and Threats in Smart Grid Communication Networks.

[4] J. M. Selga, A. Zaballos, and J. Navarro, "Solutions to the computer networking challenges of the distribution smart grid," *IEEE Communications Letters*, vol. 17, no. 3, pp. 588–591, 2013.

[5] I. Ali and s. hussain, "Communication design for energy management automation in microgrid," *IEEE Transactions on Smart Grid*, vol. PP, no. 99, 2016.

[6] International Electrotechnical Commission, "IEC 61850: Communication networks and systems for power utility automation," IEC, Tech. Rep. IEC 61850, 2002- 2013.

[7] A. Ruiz-Alvarez, A. Colet-Subirachs, F. A.-C. Figuerola, O. Gomis-Bellmunt, and A. Sudria-Andreu, "Operation of a utility connected microgrid using an IEC 61850-based multi-level management system," *IEEE Transactions on Smart Grid*, vol. 3, no. 2, pp. 858–865, 2012.

[8] T. S. Ustun, R. H. Khan, A. Hadbah, and A. Kalam, "An adaptive microgrid protection scheme based on a wide-area smart grid communications network," in *2013 IEEE Latin-America Conference on Communications*, 2013, pp. 1–5.

[9] S. M. Manson, A. Upreti, and M. J. Thompson, "Case study: Smart automatic synchronization in islanded power systems," *IEEE Transactions on Industry Applications*, vol. 52, no. 2, pp. 1241–1249, 2016.

[10] International Electrotechnical Commission, "IEC 61850-7-420: Basic communication structure - DERs logical nodes," IEC, Tech. Rep., 2009.

[11] U.S. Department of Energy, "Communication requirements of smart grid technologies," Tech. Rep., Oct. 2010.

[12] T. Pfeiffenberger, J. L. Du, P. B. Arruda, and A. Anzaloni, "Reliable and flexible communications for power systems: Fault-tolerant multicast with SDN/OpenFlow," in *International Conference on New Technologies, Mobility and Security (NTMS)*, July 2015, pp. 1–6.

[13] Y. Lopes, N. C. Fernandes, C. A. M. Bastos, and D. C. Muchaluat-Saade, "SMARTFlow: A Solution for Autonomic Management and Control of Communication Networks for Smart Grids," in *30th ACM Symposium on Applied Computing (SAC)*, 2015.

[14] N. McKeown, T. Anderson, H. Balakrishnan, G. Parulkar, L. Peterson, J. Rexford, S. Shenker, and J. Turner, "Openflow: enabling innovation in campus networks," *ACM Special Interest Group on Data Communication (SIGCOMM)*, vol. 38, no. 2, pp. 69–74, 2008.

[15] B. Lantz, B. Heller, and N. McKeown, "A network in a laptop," in *ACM Special Interest Group on Data Communication (SIGCOMM) - Hotnets'10*, 2010.

[16] Institute of Electrical and Electronics Engineers, "Communication Delivery Time Performance Requirements for Electric Power Substation Automation," Tech. Rep., 2005.

[17] ——, "801.1D: IEEE Std for Local and metropolitan area networks - MAC Bridges," IEEE, Tech. Rep., 2004.

[18] M. Pustylnik, M. Zafirovic-Vukotic, and R. Moore, "Performance of the RSTP in Ring Network Topology," Siemens, White Paper, 2007.

[19] International Electrotechnical Commission, "IEC 62439-3: Parallel Redundancy Protocol (PRP) and High-availability Seamless Redundancy (HSR)," IEC, Tech. Rep. 62439, 2010.

[20] A. Goodney, S. Kumar, A. Ravi, and Y. H. Cho, "Efficient PMU networking with software defined networks," in *SmartGridComm*, 2013.

[21] A. Sydney, D. S. Ochs, C. Scoglio, D. Gruenbacher, and R. Miller, "Using GENI for experimental evaluation of Software Defined Networking in smart grids," in *Computer Networks*, 2014.

[22] A. Cahn, J. Hoyos, M. Hulse, and E. Keller, "Software-defined energy communication networks: From substation automation to future smart grids," in *IEEE SmartGridComm*, 2013.

[23] M. Reitblatt, M. Canini, A. Guha, and N. Foster, "FatTire: Declarative Fault Tolerance for SDN," *HotSDN'13*, pp. 109–114, 2013.

[24] Y. Lopes, D. C. Muchaluat-Saade, N. C. Fernandes, and M. Z. Fortes, "Geese: A traffic generator for performance and security evaluation of IEC 61850 networks," in *IEEE 24th ISIE International Symposium on Industrial Electronics*, June 2015, pp. 687–692.