A Distributed Middleware Architecture for Attack-Resilient Communications in Smart Grids

Preprint

Yifu Wu and Jin Wei
University of Akron

Bri-Mathias Hodge
National Renewable Energy Laboratory

Presented at the IEEE International Conference on Communications 2017 SAC Symposium Communications for the Smart Grid Track
Paris, France
May 21–25, 2017

NREL is a national laboratory of the U.S. Department of Energy
Office of Energy Efficiency & Renewable Energy
Operated by the Alliance for Sustainable Energy, LLC

This report is available at no cost from the National Renewable Energy Laboratory (NREL) at www.nrel.gov/publications.

Conference Paper
NREL/CP-5D00-68118
May 2017

Contract No. DE-AC36-08GO28308

NOTICE

This report is available at no cost from the National Renewable Energy Laboratory (NREL) at www.nrel.gov/publications.

Available electronically at SciTech Connect http:/www.osti.gov/scitech

Available for a processing fee to U.S. Department of Energy and its contractors, in paper, from:

U.S. Department of Energy
Office of Scientific and Technical Information
P.O. Box 62
Oak Ridge, TN 37831-0062
OSTI http://www.osti.gov
Phone: 865.576.8401
Fax: 865.576.5728
Email: reports@osti.gov

Available for sale to the public, in paper, from:

U.S. Department of Commerce
National Technical Information Service
5301 Shawnee Road
Alexandria, VA 22312
NTIS http://www.ntis.gov
Phone: 800.553.6847 or 703.605.6000
Fax: 703.605.6900
Email: orders@ntis.gov

Cover Photos by Dennis Schroeder: (left to right) NREL 26173, NREL 18302, NREL 19758, NREL 29642, NREL 19795.

NREL prints on paper that contains recycled content.

A Distributed Middleware Architecture for Attack-Resilient Communications in Smart Grids

Yifu Wu*, Jin Wei*, and Bri-Mathias Hodge†
* The University of Akron, Akron, OH 44325, USA
†National Renewable Energy Laboratory, Golden, CO 80401, USA

***Abstract*—Distributed Energy Resources (DERs) are being increasingly accepted as an excellent complement to traditional energy sources in smart grids. As most of these generators are geographically dispersed, dedicated communications investments for every generator are capital cost prohibitive. Real-time distributed communications middleware, which supervises, organizes and schedules tremendous amounts of data traffic in smart grids with high penetrations of DERs, allows for the use of existing network infrastructure. In this paper, we propose a distributed attack-resilient middleware architecture that detects and mitigates the congestion attacks by exploiting the Quality of Experience (QoE) measures to complement the conventional Quality of Service (QoS) information to detect and mitigate the congestion attacks effectively. The simulation results illustrate the efficiency of our proposed communications middleware architecture.**

I. Introduction

As Distributed Energy Resources (DERs) technologies become increasingly popular, power resources also tend to be scattered [1]. The high DER penetrations comprise a larger portion of the generation and consumption fleets, information from these generators becomes much more critical. Therefore, it is necessary to design efficient data acquisition systems and communications infrastructures to maintain power system economic efficiency and reliability. As most of these generators are geographically dispersed, dedicated communications investments for every generator are capital cost prohibitive. Therefore, compared with investing in new network infrastructures, it is more efficient to leverage the existing communication infrastructures such as the Internet, Wireless Sensor Network (WSN), and mobile communication network to transfer sensing data from DERs to locations in the electricity system where this information can be effectively utilized [2].

However, the high penetration of DERs significantly raises the complexity of network infrastructures and thus reduces the effectiveness of the conventional network management in detecting the unusual network events. Therefore, the communication networks are more vulnerable to potential cyber attacks, such as congestion attacks, which could potentially impede the progress of DER expansion [3]. This essential challenge inspires the development of autonomous network systems with adaptive control solutions. Considering the fact that not all power system operators are familiar with network programming and configuration [4], middleware has emerged to abstract network topology and infrastructure, with which end users and network programmers can configure the whole network by only calling a local middleware objects attributes and methods [5]. Gridstat, an objected-oriented-broker middleware providing QoS information, was designed for smart grid applications [6]. A middleware structure was proposed in [7] to optimize a TCP congestion window of gateway in Network Simulator 3 (NS3). Zaballos *et al.* proposed a heterogeneous communication paradigm for smart grids that intelligently manages the interaction between Ubiquitous sensor network (USN) and next-generation network (NGN) at the link and the physical layers [8]. Dipippo *et al.* designed a priority-mapping middleware architecture for the static real-time distributed application [9]. Schantz *et al.* presented how the middleware schedules the priority for application services [10]. Li *et al.* proposed a the distributed probe method to monitor the QoS performance of the application services [11]. For middleware implementation, various research work has been done by using the co-simulation of VTB or network controller and OPNET [12], [13]. Furthermore, to address the security issues, Mostafa *et al.* developed a high-performance symmetric key cryptography scheme to secure the peer-to-peer computational grid middleware [14]. Kim *et al.* proposed a middleware architecture using encryption methods to enhance the security of smart grids [15]. All of these cryptographic approaches were developed to detect external attacks rather than insider attacks. To address this problem, we propose a distributed middleware architecture that effectively detects the insider attacks, such as congestion attack, by monitoring and analyzing the network traffic in real time using both the conventional QoS criteria as well as QoE information. QoS criteria provide objective requirements of the network performance to ensure the stability of the power system, while QoE information refers to subjective metrics that address user perception and can vary due to the expectation of the users who can be the operators or the consumers of the power systems [16]. According to the heterogeneous control structures of DERs, it is appropriate to leverage QoE information for designing the communication middleware architecture for the power systems with high penetration of DERs [17]. QoE can be assessed via the users' satisfaction in terms of Mean Opinion Score (MOS) [18]. Since cognitive ability of the individual users may vary a lot, both objective QoS criteria and the subjective QoE information are exploited for designing the communication middleware architecture in our work. The contribution of our work is two-fold. Firstly, we design a real-time middleware architecture that is capable of detecting and defending against congestion attacks. To achieve this goal, our proposed middleware architecture monitors and analyzes the quality of the network traffic by exploiting both QoE and QoS information. Secondly, we implement our middleware architecture using NS3 and evaluate its performance in a co-simulation environment of MATLAB/Simulink and NS3 [19].

In the next section, we illustrate the problem settings of our work. Section III introduces our proposed distributed real-time middleware architecture. Simulation results and conclusions are presented in Sections IV and V, respectively.

II. Problem Settings

We propose a distributed real-time middleware architecture to enable leveraging the existing communication infrastructure to address the challenges caused by the high penetration of DERs. Our middleware architecture is designed to be an additional control layer between the application layer and the transport layer defined by the TCP/IP model. To simplify the analysis, we abstract the transport layer and the layers below as one network infrastructure layer. Therefore, in our work, the communication infrastructure consists

of application layer for power systems, control layer, and network infrastructure layer.

A. Power-System Application Layer

The use of distributed energy resources is increasingly being pursued as a supplement and an alternative to large conventional central power generations. A large amount of distributed energy systems have been connected with the conventional power distribution system, such as community-scale microgrid as illustrated in Fig. 1. which significantly increases the traffic load in the communication for power system operation. To address the challenge on the network capacity, intermediate processing units, such as aggregators, are applied for data compression. As shown in Fig. 1, the readings of the solar panel sensors, such as current and voltage, are transmitted to the local aggregator. After gathering the data, each local aggregator compresses data by extracting the features and acts as a gateway to send the compressed data to the remote servers.

In our work, we adopt the industrial communication protocol, IEC 61850 [20], for the power-system application layer. IEC 61850 is being extended to support the communication for DER management systems, although this standard was originally designed for the communication of intelligent electronic devices within substations. Current mappings in IEC 61850 are mainly to Manufacturing Message Specification (MMS), Generic Object Oriented Substation Events (GOOSE), and Sampled values (SV) protocols. We focus on IEC 61850 MMS protocol in this paper.

B. Control Layer

Due to the high penetration of various DERs, the communication network topologies are becoming more complex and heterogeneous. To optimally utilize the limited communication bandwidth, we develop a real-time distributed middleware as a control layer to allocate network resources, protect network infrastructure, simplify network configurations, and secure the data transmission for the power systems. Our proposed middleware analyzes the various objective QoS criteria defined by IEC 61850 for individual applications and integrates the information with the subjective QoE criteria provided by the operators. The detail of middleware's functionality will be illustrated in the next section.

C. Network Infrastructure Layer

The network infrastructure layer consists of network interface layer, internet layer, and transport layer defined in the conventional TCP/IP model [21]. Protocols and devices in the network infrastructure layer may vary widely according to the diverse application environments of the power systems and different requirements of operators and consumers. Our proposed middleware efficiently addresses this diversity by abstracting this layer as an object that contains feasible library functions of these various protocols and network devices. By doing so, the middleware is able to effectively configure the network infrastructure layer in real time to achieve efficient data flow management.

III. Distributed Attack-Resilient Middleware Architecture

Our middleware is developed to achieve resilient communications for the power system operation, even in the presence of congestion attacks, by exploiting the objective QoS criteria and subjective QoE information. To achieve the balance among computational power consumption, operating efficiency and accuracy, and congested channel bandwidth, the middleware instances are only installed in the application-service end hosts, including aggregators and remote servers, and gateway nodes of the mesh network between the aggregators and the remote servers as illustrated in Fig. 2. The readers should note that in Fig. 2 the link between each pair of gateway nodes represents the abstraction of a small-scale subnet. Furthermore, the middleware instances belonging to different network devices are able to communicate with their neighboring nodes. Figure 3 illustrates the interactions between the middleware instances, the power-system application layer, and the network infrastructure layer for organizing, coordinating and supervising diverse application services in smart grids. As illustrated in Fig. 3, the middleware instances are allocated between the power-system application layer and the network infrastructure layer. In Fig. 3, the application module in dashed lines indicates the fact that only the end hosts in our framework have the power-system application layers. Therefore, only the middleware instances installed on the end hosts have Application Program Interfaces (APIs) that can send control commands to the middleware instances installed on the individual gateway nodes in real time. Figure 4 presents an overview of the mechanism of our proposed middleware instances. As shown in Fig. 4, the QoS monitoring system probes the real-time QoS information of each data flow in the network infrastructure layer, evaluates the observed QoS information based on our proposed QoS assessment scheme, and stores the QoS information in a buffer for potential usage in the control algorithm module. In our mechanism, the observed QoS information is evaluated by using the QoS criteria and QoE information provided by QoS-QoE database that can be updated by either the local users or remote users via the APIs in the middleware instances. If the quality of delivery for some observed data flow, that is represented by its associated QoS information, does not satisfy the QoS criteria, an alert is sent out to trigger the attack-resilient control algorithm for further data flow management. Our proposed control algorithm achieves the intelligent data flow management by exploiting the information from the network infrastructure layer and the power-system application layer, such as QoS measurements, routing information, and QoE specifications. Additionally, as shown in Fig. 4, besides being triggered by the local QoS alert, the control algorithm in our proposed middleware instance can also be activated by the requests sent from the middleware instances on the neighboring nodes.

A. Distributed Congestion Attack Detection

The authors would like to clarify that we assume that the network system has been initially well designed to fulfill the QoS criteria and the network is always in good condition if there is no attack. Furthermore, the middleware instance on each host node is designed to monitor all the network devices on that node. As shown in Fig. 4, each middleware instance has a QoS-QoE database that specifies the priorities and requirements of individual application services, which is detailed in Table I. The middleware instance probes the QoS information of different data flows from the network infrastructure layer, such as throughput, latency, jitter, source IP address, and destination IP address in real time, and converts the obtained QoS information to a standard format consistent with the QoS criteria stored in the QoS-QoE database. The observed QoS information consists of four measurements as listed in Table II. By using the observed QoS information in Table II, our proposed QoS assessment scheme evaluates the quality of the delivery of Data Flow n in Interface i as follows [22]:

$$Q_{n.i} = (Q^p_{n.i} \geq 0) \wedge (Q^r_{n.i} \geq 0) \wedge \left(Q^d_{n.i} \geq 0\right) \wedge \left(Q^j_{n.i} \geq 0\right),$$

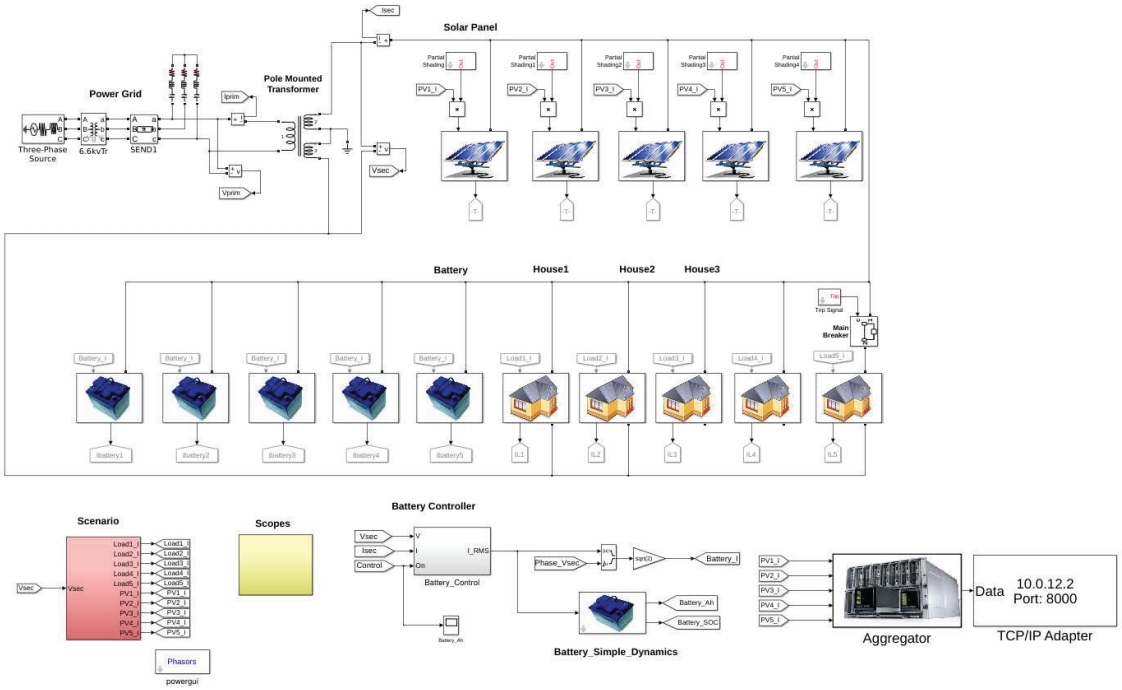

Fig. 1. The power-system application layer: a simplified power system with household loads, community-scale PV plants, and battery storage

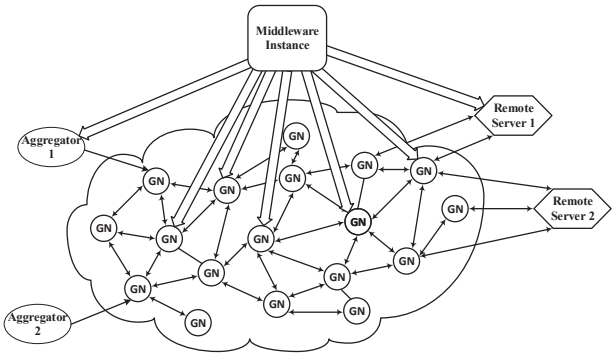

Fig. 2. Structure of our proposed middleware architecture

where

$$\begin{cases} Q^{p}_{n.i} = \frac{p_n - p^{o}_{n,i}}{p_{n_o}}, \\ Q^{r}_{n.i} = \frac{r_n - r^{o}_{n,i}}{r_{n_o}}, \\ Q^{d}_{n.i} = \frac{d_n - d^{o}_{n,i}}{d_{n_o}}, \\ Q^{j}_{n.i} = \frac{j_n - j^{o}_{n,i}}{j_n}, \end{cases} \tag{1}$$

and $p^{o}_{n,i}$, $r^{o}_{n,i}$, $d^{o}_{n,i}$, $j^{o}_{n,i}$ are observed QoS parameters of data flow n in interface i stated in Table II. If the assessment $Q_{n.i} = 0$, an alert indicating the failure of the evaluated data flow is sent to the module of control algorithm to activate the further data flow management. Additionally, the assessments of different flows in different interfaces are executed in parallel at a rate ν.

B. QoE Specification

As shown in Table I, QoE specification consists of the updated QoS criteria and the weight settings used for the network control and management. With the API in the middleware instance, the authorized

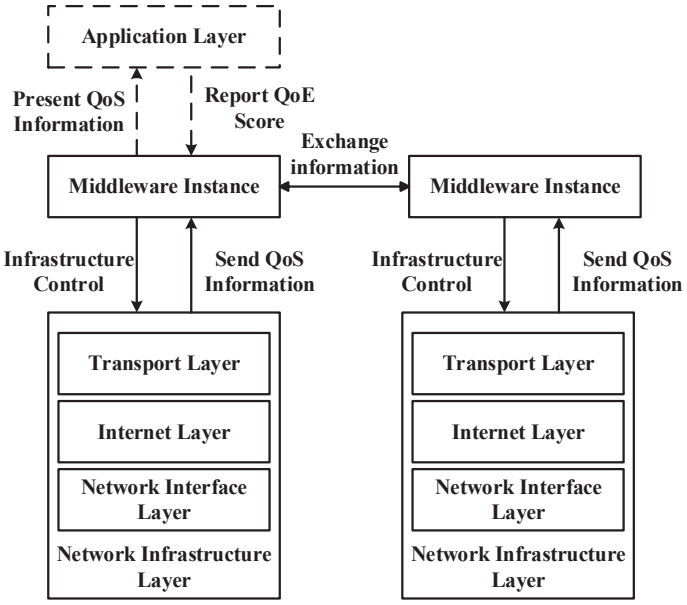

Fig. 3. Interaction between the proposed middleware instances, power-system application layers, and network infrastructure layers

users are able to report the QoE specification of certain application data flow, which results in the modification of the corresponding QoS criteria and weight settings in QoS-QoE database. This modification is further synchronized among the middleware instances of the neighboring nodes via network-layer broadcast. Once the middleware instances of neighboring nodes receive the modification message, they continue to broadcast the message to their neighboring nodes. If the middleware instance of one node has received the same message

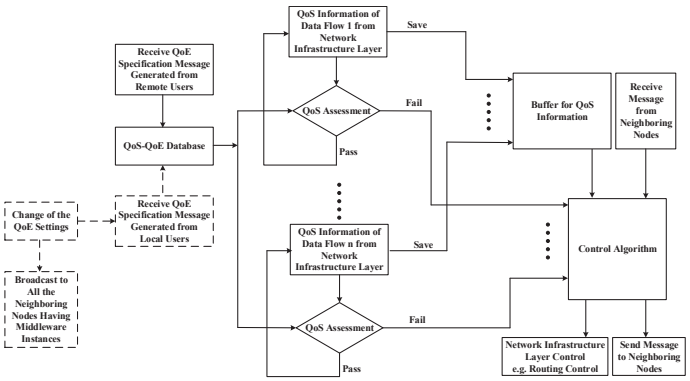

Fig. 4. Overview of our proposed middleware mechanism

TABLE I
INFORMATION PROVIDED BY QOS-QOE DATABASE

Attribute	Description
i	Interface ID of network devices
n	ID of data flow
$\mathcal{L}_n$	Priority level of data flow n
$\mathcal{S}_n$	Source IP address of data flow n
$\mathcal{D}_n$	Destination IP address of data flow n
p_n	Maximum packet loss rate of data flow n
r_n	Minimum data rate of data flow n
d_n	Maximum delay of data flow n
j_n	Maximum jitter of data flow n
w_n^p	Weight of maximum packet loss rate of data flow n
w_n^r	Weight of data rate of data flow n
w_n^d	Weight of maximum delay of data flow n
w_n^j	Weight of maximum jitter of data flow n

TABLE II
PARAMETERS OF THE OBSERVED QOS INFORMATION

Parameters	Description
$p_{n,i}^o$	Observed packet loss rate of Data Flow n transmitted through Interface i
$r_{n,i}^o$	Observed throughput of Data Flow n transmitted through Interface i
$d_{n,i}^o$	Observed maximum delay of Data Flow n transmitted through Interface i
$j_{n,i}^o$	Observed maximum jitter of Data Flow n transmitted through Interface i

twice, it updates the QoS standard database according to the message.

C. Congestion-Resilient Control Algorithm

As stated in Section III-A, the QoS monitoring system in the middleware instance evaluates the quality of delivery for each data flow n at each interface i in the associated nodes and achieves the assessment $Q_{n,i}$. If $Q_{n,i} = 0$, a local QoS alert is sent out to activate the control algorithm for further data flow management and network infrastructure control. Since the default routing table is calculated based on the link-state routing protocol, there may exist more than one shortest path between the current host node and the destination. Each path refers to one network device interface on the host node and each of these interfaces is assigned with a distinct index. In our work, we predetermine the default routing path so that the data flow is forwarded via the interface with the smallest index value.

After receiving a local QoS alert, the control mechanism in hop Gateway Node (GN) p checks the priority of each data flow that causes the alert and then recovers the QoS of the data flows according to the descending order of their priorities. The control mechanism firstly verifies whether there are available alternative routing paths based on the local routing data. If the result is positive, the control mechanism evaluates the performance of each local Interface i associated with each alternative route. Let ℓ_i be the link associated with Interface i, $\mathcal{D}_i$ denotes the set of the indices of data flows transmitted through Interface i, and $\mathcal{F}_i$ denotes the set of the data flows transmitted through Interface i. We predict the Quality of Experience (QoE) that will be achieved by transmitting data through Interface i by calculating the performance score as follows:

$$S_i = \begin{cases} 3, & \text{if } \phi_i = 0 \text{ and } \ell_i \text{ is idle;} \\ 2, & \text{if } \phi_i = 0,\ \Phi_i = 1, \text{ and } \forall f \in \mathcal{F}_i \text{ has lower priority compared with the target data flow;} \\ 1, & \text{if } \phi_i = 0,\ \Phi_i = 1 \text{ and } \exists f \in \mathcal{F}_i \text{ has higher or equal priority compared with the target data flow;} \\ 0, & \text{if } \{\phi_i = 0\} \vee \{\Phi_i = 0\}. \end{cases} \tag{2}$$

where τ is a constant parameter and

$$\phi_i = \begin{cases} 0, & \text{if there is no QoS alert sent from Interface } i; \\ 1, & \text{otherwise,} \end{cases} \tag{3}$$

$$\Phi_i = \wedge_{n \in \mathcal{D}_i} \{S_{n.i} \geq \tau\}, \tag{4}$$

$$S_{n,i} = w_n^p \times Q_{n,i}^p + w_n^d \times Q_{n,i}^d + w_n^j \times Q_{n,i}^j, \tag{5}$$

where w_n^p, w_n^d, w_n^j are weight parameters that are determined based on the feedback of the volunteer users. The volunteer users are asked to experience different network conditions with different settings of the weight parameters and evaluate their satisfactions using the scores from 1 (worst) to 5 (best). Then the settings having the arithmetic mean of all the individual scores, called Mean Opinion Score (MOS) [18], larger than 3.5 for all the network conditions are selected.

Let $\mathcal{K}$ be the set of interfaces that have positive performance score. If $|\mathcal{K}| = 1$, which means that there is only one interface having the positive performance score, the control mechanism changes the route to the only interface $k \in \mathcal{K}$. Otherwise, the control mechanism multicasts the request message to the middleware instance on the nodes $m \in \mathcal{M}$, where $\mathcal{M}$ is the set of gateway nodes that have interfaces connecting with some interface $k \in \mathcal{K}$. Each middleware instance that receives the request messages returns a feedback message containing the performance score of the associated GN m for being a hop node to transmit the target data flow. To achieve the performance score, the middleware instance on GN m first identifies the interfaces included in alternative routes and then calculates the performance score for each identified interface based on Eqs. (2) to (5). Letting $\mathcal{U}_m$ be the set of the indices of the identified interfaces, the performance score $\tilde{S}_m$ for GN m can be obtained as follows:

$$\tilde{S}_m = \max_{u \in \mathcal{U}_m} \{S_u\}, \tag{6}$$

where S_u is achieved by using Eqs. (2) to (5). After collecting the feedback messages, the control mechanism updates the performance score of each Interface i in its associated hop node, GN p, by using Eq. (7) and routes the target data flow to the interface having the highest performance score. If more than one interface have a joint highest performance score, the target data flow is transmitted through the interface with the smallest index value.

$$S_i^* = \min\left\{S_i, \tilde{S}_m\right\}, \tag{7}$$

where we assume that Interface i connects with an interface in GN m. If there is no alternative route for the target data flow having a positive performance score, the associated middleware instance hands over the routing task to the previous hop node in the original routing path by sending a handover message. Furthermore, the control algorithm can also be activated by the requests sent from the middleware instances on the neighboring gateway nodes that ask for support in managing certain data flow. After receiving a request message from the middleware instance of a neighboring gateway node, the control algorithm firstly identifies the type of the request message. If the message is a handover message that includes the target data flow ID and destination IP, the control mechanism begins to identify a proper route for the target data flow. Otherwise, if the message is a request for assessing the performance of the interface of the potential alternative route, the control algorithm checks the relative interface used for forwarding the target data flow and sends the feedback message to the middleware instance on the gateway node from which the request is sent.

IV. Simulation Results

In this section, we demonstrate and evaluate the performance of our proposed distributed middleware architecture in two cases using a real-time co-simulation test system of NS3 and MATLAB/Simulink. Our test system consists of the sensing devices that monitor the states of the renewable energy resources as shown in Fig. 1, smart meter aggregators, mesh network and the remote servers. The mesh network is simulated by using NS3 and our proposed middleware architecture is also developed as a NS3-based module for the simulation. The power system is simulated by using MATLAB/Simulink. Data generated from the power system is fed into the simulated network via the tap bridge module. Congestion attack flows are generated based on the protocol stack of Linux Ubuntu 14.04 LTS [23].

A. Simulation Settings

The topology of the mesh network used in our simulation is shown in Fig. 5. Elliptical nodes are the aggregators which send the sensing

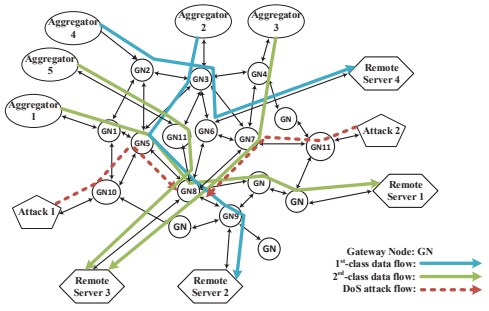

Fig. 5. Network topology used in the simulation

data to the remote servers. Circle nodes in Fig. 5 denote the gateway nodes that are used for routing and forwarding the data flow between two different subnet domains. All the links between these nodes are CSMA links which represent a complex subnet. To simplify the explanation, we use $\text{Link}_{i,j}$ to denote the CSMA link between GNs i and j. In our test system, the links between $\text{Link}_{5,8}$, $\text{Link}_{6,8}$, $\text{Link}_{7,8}$, and $\text{Link}_{8,11}$ are set to be the bottleneck links with the bandwidth set to 10 Mbps and the propagation delay set to 10 ms. Furthermore, for all the other links in our system, the bandwidth and propagation delay are set to be 10-Gbps and 0 ms, respectively.

Furthermore, we adopt IEC 61850 MMS protocol for the communications in the power systems. As stated in [20], MMS service, that is usually run over TCP/IP-based Ethernet, can only use Types 2, 3 and 5 of IEC 61850 messages. We assume that there are simultaneously five different MMS message data flows, some of which share the same network link. As shown in Table III, IEC 61850 MMS protocol specifies the critical QoS criteria of different messages by defining the upper bounds of the one-way delay. Furthermore, the QoE information provided by the authorized users, such as power operators, modifies the QoS criteria by specifying additional requirements on the communication quality, such as data rate and jitter. As shown in Table III, Data Flows 2 and 4, that are used for critical contingency analysis, have the highest priority, and Flows 1, 3, and 5, that generated by the advanced metering infrastructure, have the second priority. In the following case studies, we focus on demonstrating the effectiveness of our proposed distributed middleware architecture in protecting the critical Data Flow 2 against potential congestion attacks in the system.

TABLE III
QoS and QoE Metrics of the Data Flows in the Simulations

Application ID	NO.1	NO.2	NO.3	NO.4	NO.5
Message type	3	2	2	3	2
Constant data rate(kbps)	90	90	256	1280	640
One-way delay(ms)	<500	<100	<100	<10000	<100
Jitter(ms)	<50	<50	<50	N/A	<50
Packet loss rate	<5%	<1%	<3%	<10%	<3%
Packet size(Byte)	128	128	512	65536	3072
w^p	0.40	0.45	0.40	0.20	0.40
w^r	0.01	0.01	0.06	0.80	0.06
w^d	0.40	0.45	0.45	0.00	0.45
w^j	0.19	0.09	0.09	0.00	0.09

B. Case Study I

In this case, we assume that there is one UDP flooding attack generated from Source 1. As shown in Fig. 5, this attack targets at GN8 by exhausting the bandwidth of $\text{Link}_{5,8}$ and consequentially compromising $\text{Link}_{5,10}$. In the simulation, five power-grid data flows are transmitted beginning from $t = 0$s and the attack data flow starts to compromise the network at $t = 49$s. Our proposed distributed middleware instance on each node monitors QoS information of each data flow handled by each net device interface. Before the attack starts, Data Flows 1 and 2 work well even though they have different QoS priorities. After the attack starts at $t = 49$s, these two data flows cannot survive because their routing paths include $Link_{5,8}$ that is compromised by the attack.

By monitoring the QoS information of the data flows transmitted through the interfaces of GN 5 and implementing the QoS assessment defined in Eqs. (2) and (1), the middleware instance on GN 5 is able

to detect that there potentially exists cyber attacks and send the QoS alert for Data Flow 1 and 2 to the local control mechanism in the same middleware instance. Because Flow 2 has higher priority, the local control mechanism recovers the QoS of Flow 2 first. In our proposed mechanism, the control algorithm first checks if there are any alternative routes leading to the destination of Flow 2. After verifying that there is no alternative route for Flow 2 available based on the information in the routing table of GN 5, the associated middleware instance hands over the routing task to the previous hop node, GN 3, in the original routing path. After receiving the handover message, the control mechanism of the middleware instance on GN 3 identifies three available alternative routes for Flow 2 including $\text{Link}_{3,6}$, $\text{Link}_{3,7}$, and $\text{Link}_{3,11}$ by checking the local routing table. The control mechanism next calculates the performance score for the interfaces corresponding to these three links, that are indexed with 6, 7, and 11, by using Eqs. (2) to (5). Since Data Flow 4, that has the same priority as Flow 2, is transmitted via Interface 6, and $\text{Link}_{3,7}$ and $\text{Link}_{3,11}$ are idle, the performance scores for Interfaces 6, 7, and 11 are 1, 3, and 3, respectively. Therefore, the control mechanism in GN 3 multicasts the request message to GNs 6, 7, and 11 and collects the feedback signal stating the performance scores 3, 2, and 2 for GNs 6, 7, and 11, respectively. Based on Eq. (7), the performance scores of Interfaces 6, 7, and 11 in GN 3 are updated as 1, 2, and 2, respectively. Therefore, Interfaces 7 and 11 have the highest final performance score and the target Data Flow 2 is routed to Interface 7 that has a smaller index value.

Figure 6 shows the throughput of Data Flow 2 without employing our proposed middleware architecture. From Fig. 6, it is clear that the throughput of Data Flow 2 is constantly 90 kbps before $t = 49$ s and decreases dramatically to 0 kbps after the UDP flooding attack occurs beginning from $t = 49$ s. This implies that, without our proposed middleware architecture, Data Flow 2 cannot survive within 10 s after the attack occurs. By using our proposed middleware architecture, the throughput of Data Flow 2 is presented in Fig. 7. By observing Fig. 7, it is clear that, in the presence of the UDP flooding attack, our proposed middleware architecture effectively recovers the transmission of Data Flow 2 within about 10 s. Figure 8 shows the end-to-end latency of Data Flow 2 achieved without using our middleware architecture. It can be observed from Fig. 8 that without implementing our proposed middleware architecture the end-to-end latency of Data Flow 2 dramatically increases after the attack occurs. The delay of Data Flow 2 achieved by using our middleware architecture is shown in Fig. 9. It can be get from Fig. 9 that the end-to-end latency of Data Flow 2 temporarily increases within the first 7 s after the attack occurs and reduces to about 17 ms, that satisfies the QoS requirement of this data flow, within another 5 s. From Figs. 9 and 9, we can see that the proposed middleware architecture is effective in protecting the critical data flow from the UDP flooding attack that is one typical congestion attack.

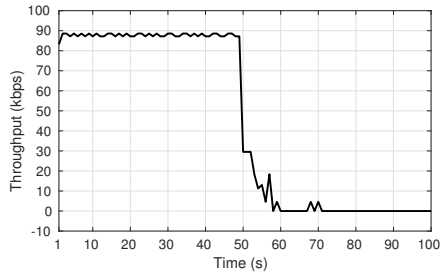

Fig. 6. Throughput of Data Flow 2 without using our proposed middleware architecture.

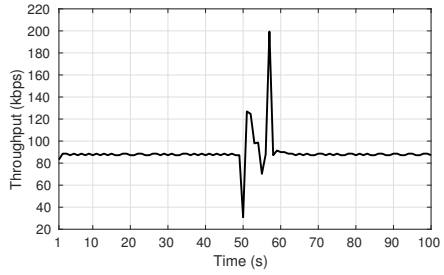

Fig. 7. Throughput of Data Flow 2 when our proposed middleware architecture is employed.

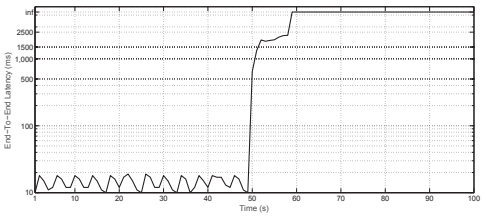

Fig. 8. End-to-end latency of Data Flow 2 without using our proposed middleware architecture.

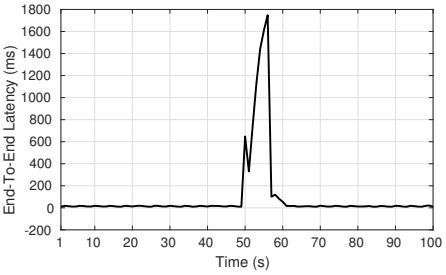

Fig. 9. End-to-end latency of Data Flow 2 when our proposed middleware architecture is employed.

C. Case Study II

In the second case study, we consider one additional UDP flooding attack, called Attack 2, that compromises $\text{Link}_{7,8}$ which is identified as the best alternative route in Case I. We assume that these two UDP flooding attacks begin simultaneously at $t = 49$ s. The cooperation procedure of the distributed middleware instances is similar to that in Case I. In this case, the middleware instance on GN 7 detects Attack 2 by monitoring the QoS information of the data flows transmitted through the interfaces in GN 7 and implementing the QoS assessment defined in Eqs. (2) and (1). Therefore, there is a QoS alert for Data Flow 3 on the interface of GN 7 that is associated with an alternative route. Because of the QoS alert, the feedback signals received by the middleware instance on GN 5 indicate that the performance score of GN 7 is 0, and thus Data Flow 2 is routed to the interface on GN 11. The throughput performance by using our middleware architecture is shown in Fig. 10, from which we can observe that our proposed middleware architecture is able to recover the throughput performance of Data Flow 2 within 15 s after these two attacks occur. Figure 11 shows the end-to-end latency of Data Flow 2 by using our middleware architecture. From Fig. 11, we can observe that the end-to-end latency is reduced to the specified QoS requirement within 10 s after the

attacks occur. Comparing the performances between Figs. 7 and 10 and the performances between Figs. 9 and 11. we can observe that throughput and end-to-end latency of Data Flow 2 have considerable oscillations after being recovered to meet the QoS requirement in Case II. This metastable behavior of QoS performance implies the sensitivity of the route of Data Flow 2 to potential cyber or physical disturbances in the future.

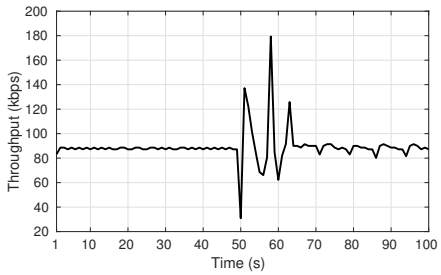

Fig. 10. End-to-end latency of Data Flow 2 is recovered by avoidance strategy in case study 2

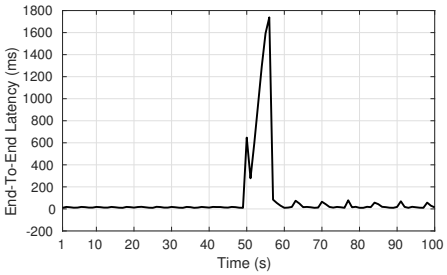

Fig. 11. End-to-end latency of Data Flow 2 is recovered by avoidance strategy in case study 2

V. Conclusions

In this paper, we propose a real-time distributed middleware architecture that effectively leverages the existing network infrastructures to achieve resilient communications for the power systems with high penetration of DERs. Our proposed middleware architecture uses the subjective QoE evaluation from the end users, who can be either operators or consumers of the power systems, as an efficient complement to the conventional objective QoS criteria to detect and defend against potential congestion attacks. As far as we know, we are the first to design and implement a middleware architecture considering QoE evaluation. The simulation results illustrate the efficiency of our proposed architecture. We assert that our work will potentially promote the largerscale DER integration by providing a resilient communication environment. Future work will evaluate the performance of our middleware architecture implemented in more complex network infrastructures to support much larger scale power systems with high penetration of DERs.

References

[1] S. Karnouskos, "The cooperative internet of things enabled smart grid," in *Proceedings of the 14th IEEE international symposium on consumer electronics (ISCE2010), June*, pp. 07–10, 2010.

[2] O. Vermesan, P. Friess, P. Guillemin, S. Gusmeroli, H. Sundmaeker, A. Bassi, I. S. Jubert, M. Mazura, M. Harrison, M. Eisenhauer, *et al.*, "Internet of things strategic research roadmap," *Internet of Things-Global Technological and Societal Trends*, pp. 9–52, 2011.

[3] J. P. Lopes, N. Hatziargyriou, J. Mutale, P. Djapic, and N. Jenkins, "Integrating distributed generation into electric power systems: A review of drivers, challenges and opportunities," *Electric power systems research*, vol. 77, no. 9, pp. 1189–1203, 2007.

[4] S. Bandyopadhyay, M. Sengupta, S. Maiti, and S. Dutta, "A survey of middleware for internet of things," in *Recent Trends in Wireless and Mobile Networks*, pp. 288–296, Springer, 2011.

[5] A. T. Campbell, G. Coulson, and M. E. Kounavis, "Managing complexity: Middleware explained," *IT professional*, vol. 1, no. 5, pp. 22–28, 1999.

[6] H. Gjermundrod, D. E. Bakken, C. H. Hauser, and A. Bose, "Gridstat: A flexible qos-managed data dissemination framework for the power grid," *Power Delivery, IEEE Transactions on*, vol. 24, no. 1, pp. 136–143, 2009.

[7] M. Casoni, C. A. Grazia, M. Klapez, and N. Patriciello, "A congestion control middleware layer with dynamic bandwidth management for satellite communications," *International Journal of Satellite Communications and Networking*, 2015.

[8] A. Zaballos, A. Vallejo, and J. M. Selga, "Heterogeneous communication architecture for the smart grid," *Network, IEEE*, vol. 25, no. 5, pp. 30–37, 2011.

[9] L. C. Dipippo, V. F. Wolfe, L. Esibov, G. Cooper, R. Bethmangalkar, R. Johnston, B. Thuraisingham, and J. Mauer, "Scheduling and priority mapping for static real-time middleware," in *Challenges in Design and Implementation of Middlewares for Real-Time Systems*, pp. 41–68, Springer, 2001.

[10] R. E. Schantz, J. P. Loyall, C. Rodrigues, D. C. Schmidt, Y. Krishnamurthy, and I. Pyarali, "Flexible and adaptive qos control for distributed real-time and embedded middleware," in *Proceedings of the ACM/IFIP/USENIX 2003 international Conference on Middleware*, pp. 374–393, Springer-Verlag New York, Inc., 2003.

[11] B. Li and K. Nahrstedt, "Qualprobes: middleware qos profiling services for configuring adaptive applications," in *Middleware 2000*, pp. 256–272, Springer, 2000.

[12] W. Li and A. Monti, "Integrated simulation with vtb and opnet for networked control and protection in power systems," in *Proceedings of the 2010 Conference on Grand Challenges in Modeling & Simulation*, pp. 386–391, Society for Modeling & Simulation International, 2010.

[13] M. Bartl, K. Molnar, and J. Hosek, "Control of network operation generator from opnet modeler environment," *IJCSNS*, vol. 11, no. 6, p. 17, 2011.

[14] M.-S. M. Mostafa, S. H. Deif, and H. A. E. I. Kholidy, "Ultra gridsec: Peer-to-peer computational grid middleware security using high performance symmetric key cryptography," in *Information Technology: New Generations, 2008. ITNG 2008. Fifth International Conference on*, pp. 137–142, IEEE, 2008.

[15] Y.-J. Kim, M. Thottan, V. Kolesnikov, and W. Lee, "A secure decentralized data-centric information infrastructure for smart grid," *Communications Magazine, IEEE*, vol. 48, no. 11, pp. 58–65, 2010.

[16] S. Möller and A. Raake, *Quality of Experience: Advanced Concepts, Applications and Methods (T-Labs Series in Telecommunication Services)*. Springer, 2014.

[17] L. Zhou, J. J. Rodrigues, and L. M. Oliveira, "Qoe-driven power scheduling in smart grid: Architecture, strategy, and methodology," *IEEE Communications Magazine*, vol. 50, no. 5, pp. 136–141, 2012.

[18] P. Brooks and B. Hestnes, "User measures of quality of experience: why being objective and quantitative is important," *IEEE network*, vol. 24, no. 2, pp. 8–13, 2010.

[19] T. R. Henderson, M. Lacage, G. F. Riley, C. Dowell, and J. Kopena, "Network simulations with the ns-3 simulator," *SIGCOMM demonstration*, vol. 14, 2008.

[20] C. Ozansoy, *Modelling and Object Oriented Implementation of IEC 61850: The New International Standard on Substation Communications and Automation*. LAP LAMBERT Academic Publishing, 2010.

[21] C. Hunt, *TCP/IP Network Administration*. O'Reilly Media, 2002.

[22] J. Mirkovic, A. Hussain, S. Fahmy, P. Reiher, and R. K. Thomas, "Accurately measuring denial of service in simulation and testbed experiments," *Dependable and Secure Computing, IEEE Transactions on*, vol. 6, no. 2, pp. 81–95, 2009.

[23] P. Shankdhar, "Dos attacks and free dos attacking tools — infosecinstitute,," 2013. [Online; accessed 1-May-2016].