international journal of critical infrastructure protection 42 (2023) 100618

Contents lists available at ScienceDirect

International Journal of Critical Infrastructure Protection

journal homepage: www.elsevier.com/locate/ijcip

A flexible OT testbed for evaluating on-device implementations of IEC-61850 GOOSE

Matthew Boeding [a], Michael Hempel [a], Hamid Sharif [a,*], Juan Lopez Jr [b], Kalyan Perumalla [b,1]

[a] Department of Electrical and Computer Engineering, University of Nebraska-Lincoln, Lincoln, 68588, NE, USA
[b] Oak Ridge National Laboratory, Oak Ridge, 37831, TN, USA

ARTICLE INFO

Keywords:
Cyber-physical systems
Smart grid
Industrial control systems
Operational technology
IEC-61850
GOOSE
Vulnerability

ABSTRACT

The growing convergence of Information Technology and Operational Technology has enhanced communication and visibility across power grids. This, coupled with the growing use of Distributed Energy Resources in power grids, has enhanced the grid capabilities while also creating a larger attack surface for malicious actors. A common protocol vulnerable to these attacks is the IEC-61850 GOOSE protocol due to its low-latency requirements, multicast packet delivery method, and lack of encryption. In this paper, we evaluate the security implications of different hardware implementations of this protocol by contrasting device response and recovery of two commercial off-the-shelf Intelligent Electronic Devices from separate manufacturers. The cyberattacks utilized in this paper are research-established GOOSE attacks with results measured in device latency and GOOSE endpoint response success.

1. Introduction

The Operational Technology (OT) sector has begun to adopt networking technologies previously only utilized within the Information Technology (IT) space. With the convergence of these two areas, many application-specific protocols were developed to accommodate the time-sensitive nature of OT processes. As networking in OT systems grows, many standards and regulations have been introduced to mitigate the risk of cyberattacks [1]. Specific to the power grid, the IEC-61850 protocol suite [2] was introduced to allow for seamless inter- and intra-substation communication. The Manufacturing Message Specifications (MMS), Sampled Values (SV), and Generic Object Oriented Substation Event (GOOSE) protocols are introduced within this standard to allow for data visibility throughout a grid network. While these protocols can automate processes that previously required manual intervention, there are security vulnerabilities to each of these protocols [3–5]. Due to its timing requirements coupled with the high message priority and its multicast transmission method [6], the GOOSE protocol is considered to be at the highest risk of attack and for data privacy concerns [7–10]. The vulnerabilities of the GOOSE protocol have been extensively studied in previous works. However, our extensive review found that the effects and consequences of device-specific implementations for these protocol vulnerabilities have not been widely studied to date.

In this paper, we measure the results of two commercially available Intelligent Electronic Devices (IEDs) from reputable manufacturers and measure their respective responses to Denial of Service (DoS), Replay, and False Data Injection Attacks (FDIA). A flexible OT testbed established and introduced in our previous work [11] is used to initiate the attack and measure results. The remainder of this paper is organized as follows. Section 2 introduces the GOOSE protocol, Section 3 outlines related research works for the evaluation of the GOOSE protocol, while Section 4 introduces the proposed evaluation methodology and result metrics, and Section 5 shows the obtained result of the research. In Section 6, we discuss our future efforts in continuation of this work, and in Section 7 we present our concluding remarks.

2. GOOSE protocol

The IEC-61850 standard introduced multiple protocols to increase data visibility across various power grid systems. The GOOSE protocol, in particular, is designed for bay- and station-level communications, distributing high-priority messages through the use of IEDs. The GOOSE messaging protocol's security vulnerabilities are largely caused by the multicast packet delivery format and the use of unencrypted data [12]. The protocol utilizes IEEE 802.1Q VLAN tagging and a pre-defined Ethernet type field, along with a publisher/subscriber framework. The

* Corresponding author.
E-mail address: hamidsharif@unl.edu (H. Sharif).

[1] This manuscript has been authored by UT-Battelle, LLC, under contract DE-AC05-00OR22725 with the US Department of Energy (DOE). The publisher acknowledges the US government license to provide public access under the DOE Public Access Plan (http://energy.gov/downloads/doe-public-access-plan).

https://doi.org/10.1016/j.ijcip.2023.100618
Received 6 March 2023; Received in revised form 5 June 2023; Accepted 20 June 2023
Available online 21 June 2023

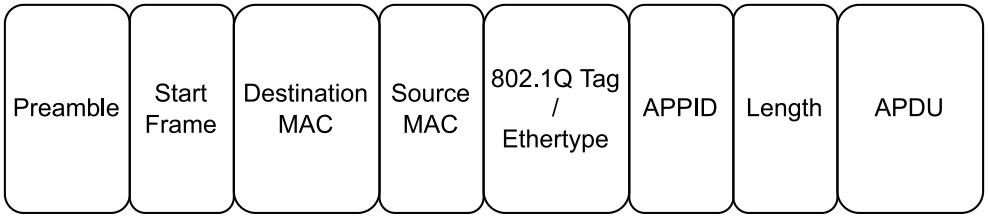

Fig. 1. High-level view of the GOOSE packet structure.

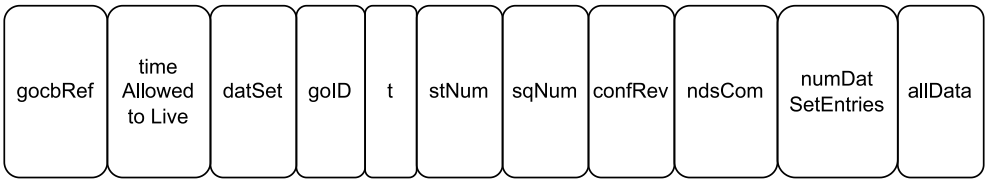

Fig. 2. GOOSE Application Protocol Data Unit (APDU) data fields.

GOOSE packet format can be seen in Fig. 1. Each packet field has a defined length, except the Length and Application Protocol Data Unit (APDU) fields, which utilize Abstract Syntax Notation One (ASN.1) for data encoding.

Each individual field of the APDU is shown in Fig. 2. Of these fields, ***gocbRef*** and ***datSet*** refer to the logical node of the control block and publication, respectively, while the goID field identifies the individual message being sent. Each IED should be synchronized with an external clock source, and the ***timeAllowedtoLive*** and ***t*** fields are used to evaluate if a publication is timed out. The ***state number*** and ***sequence number*** are both 32-bit integers that identify the order of a packet. The state increases as the contained data changes, and the sequence field value increments with each packet transmission, resetting to zero on state changes.

3. Related works

The work surrounding the GOOSE protocol can widely be categorized as either investigating protocol vulnerabilities or creating testbeds for IEDs implementing the GOOSE protocol. The protocol vulnerability research investigates the protocol itself through a theoretical view of its specified operations and examines possible implications of cyberattacks. The work surrounding testbeds, by contrast, takes this information and evaluates physical devices against these vulnerabilities.

3.1. Vulnerabilities

The GOOSE protocol's vulnerabilities explored in the literature primarily stem from the multicast messaging using a publisher/subscriber approach for each packet without the use of encryption. The protocol utilizes a subset of specifically allocated MAC addresses for defining the source and destinations of messages. This allows engineers to customize the connections for their specific network. However, the lack of authentication of the publisher, and the use of unencrypted packets, allows anyone with access to the network to see the packets. Within the literature, this vulnerability is exploited through DoS, Replay, and FDIA attacks.

Denial of Service attacks are the most simplistic form of exploit for this vulnerability. They do not require any specific format of packets, and each packet does not necessarily need to be a valid packet for the network. It only needs to present a valid MAC endpoint to which an IED is subscribed. These attacks have the goal of increasing the response time of the IED beyond the maximum allowed latency defined in the standard, disrupting its service, or entirely disabling the IED. In [12], the authors outline possible Denial of Service (DoS) attacks that can affect the IEC-61850 protocol suite. DoS attacks are implemented by flooding the network with a large number of packets, including the use of oversized packets.

Replay attacks have an increased complexity when compared to DoS attacks, as they require increased knowledge of the network to be successfully executed. The focus of this style of attack, as demonstrated in [13], is to retransmit a packet that was previously sent on the network. The packet will then be recognized as valid by a subscriber that is then forced to process that message. A replay attack can be used to resend a previously valid state, such as an open breaker, to an IED and cause the device to enter an unexpected state. The implementation can vary from updating ***timeAllowedtoLive*** and ***state number*** fields, to re-transmitting a packet without an updated ***timestamp*** or ***state number***. Without updated packet fields, these attacks can be limited in their effectiveness, due to the ***timestamp*** and allotted ***Time To Live*** (TTL) of each packet in the GOOSE protocol. However, these attacks can nonetheless cause the IED's processing time to increase, which could result in the IED violating the protocol's latency constraints. For our implementation, the replay attack will not update any packet fields, giving clear distinction between the replay and FDIA attack patterns.

The final prominent attack against the GOOSE protocol is FDIA. These attacks require more complexity but have a higher chance of seriously affecting the operation of an IED and the network as a whole. These attacks send valid network packets, synchronized to the network time source, to force IEDs into an unintended state. In [14], the authors sent an FDIA that caused incorrect breaker tripping. The authors of [15] performed a state number attack, which is a form of FDIA that increments the ***state number*** field's value, which could force an IED to become unresponsive for 2^{32} states. These demonstrated attacks show that the GOOSE protocol has multiple vulnerabilities with different attack vectors that can seriously impede and disrupt the operations of critical infrastructure.

3.2. Testbeds

The scientific literature contains other works that describe testbeds for GOOSE, with different evaluation methods. In [16], the authors introduce GEESE 2.0, a GOOSE evaluation tool that provides a GUI and GOOSE packet generation capabilities for identifying vulnerabilities present in device implementations. The ability to alter the type of generated traffic allows for versatility in packet generation. This method evaluates only the communication link for attacks on the IED, however.

Additional works, presented in [17,18], test the operation of the GOOSE protocol in response to valid input signals on the IED. For this, the authors utilize a Hardware-in-the-Loop (HIL) simulator, the

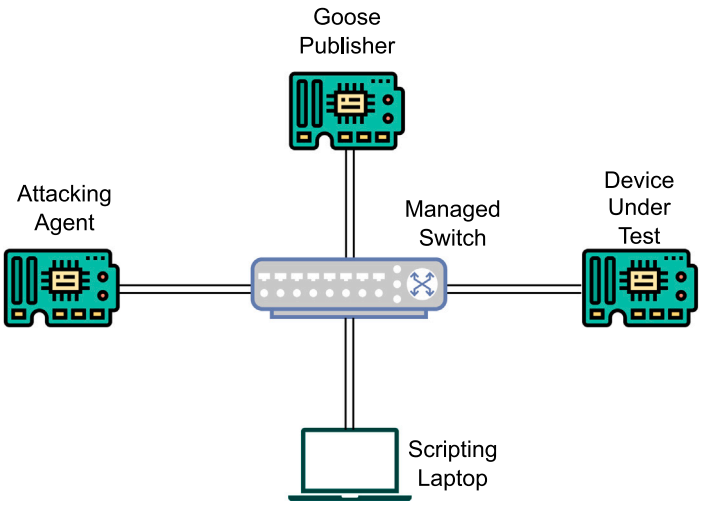

Fig. 3. High-level topology of our developed OT vulnerability discovery testbed [11].

OPAL-RT, to generate normal operating conditions for the IED. In these testbeds, the IEDs are evaluated through their response to different input signals. Additionally, the authors of [19] expand the HIL testbed to evaluate IEC-62531 encryption on IEDs. However, this method is not always shown to meet the required network latency of less than 4 ms [6], as shown in [20]. In [21], the authors created a simulated substation utilizing only commercially available IEDs to evaluate the GOOSE protocol with HIL simulation. The results of specific protocol configurations and their cybersecurity implications were outlined in [22].

Software-based testbeds have also been introduced to evaluate the communication protocol with limited or no hardware investment. The authors of [23] emulate network communications using GNS3 in parallel with physical IEDs, while [24] introduce a testbed created entirely with open source packages. There are also testbeds that utilize predefined, representative ICS network traffic to include GOOSE traffic. In [25], the authors introduce PowerDuck, a data set created from an emulated substation utilizing the GOOSE protocol. This dataset contains both standard traffic and 16 different attacks with varying durations. A synthetic dataset was also created in [26], in which the authors created traffic for 8 attack traces for an emulated 4-bus, 18-IED substation.

From these works, we can observe that the GOOSE protocol itself has been extensively investigated. It highlights, however, a gap in hardware testbeds that evaluate device-specific protocol implementations, including an IED's response and recovery during and after cyberattacks. Our work aims to address this vital oversight.

4. Methodology

In our previous work [11], a flexible OT testbed was introduced that can produce varied cyber attacks, including DoS, Replay, and FDIA attacks, with user-defined data rates and attack durations. The results from that prior work showed the reliability and accuracy of the testbed. In this paper, we build and significantly expand upon this effort and present the results of our investigation of several GOOSE implementations. The aim of this work is to demonstrate that a testbed with dynamic packet generation capabilities can accurately generate traffic and measure device responses while maintaining the ability to generate diverse attacks against a device-under-test. Our resulting testbed can execute three different types of attacks and produce attack traffic volume ranging from 1 to 90 Mbps of attack traffic based on user requirements and configurable at runtime. With these requirements, an individual device can be evaluated for any implementation flaw with replicable tests and removing any external factors from impacting the results. The testbed is designed to run standalone by emulating larger implementations, but can also operate in conjunction with other deployed devices as part of an existing installation, and can further be expanded in further iterations.

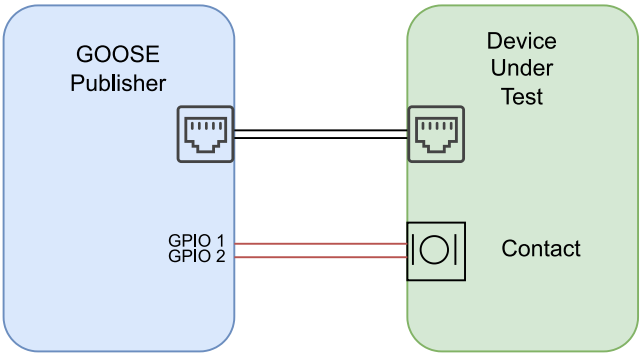

Fig. 4. Response measurement physical connections [11].

4.1. Testbed

The testbed utilized in this paper contains three main elements: the GOOSE Publisher, the Attacking Agent, and the Scripting Node. All three nodes are implemented as Linux services and are written in the C programming language. The topology of the testbed is outlined in Fig. 3. Each element has distinct tasks when evaluating the Device Under Test (DUT), and performs each task in a manner that is replicable for vulnerability confirmation.

The first node is the GOOSE Publisher and is created with the open source libiec61850 [27] running on a Raspberry Pi 4 with Raspberry Pi OS [28] installed, which is a Linux-based OS. The GOOSE Publisher can create various combinations of GOOSE publications and subscriptions to vary the network load in order to mimic a production environment. To evaluate the effect of attacks on the DUT, the GOOSE Publisher has GPIO exposed to detect the state of the contact, shown in Fig. 4. For the testbed to operate correctly, the DUT will be configured to interpret a boolean value sent in a publication by the GOOSE publisher to close a contact. The updated state and boolean value of a publication will change the state of the contact, which is measured by the GOOSE Publisher's GPIO.

In the case of an attack causing the DUT to fail in closing or opening the contact, a timeout for measured response time is initiated. The

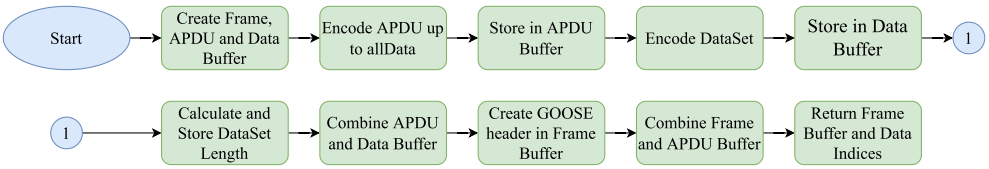

Fig. 5. GOOSE packet generation flowchart utilized in our testbed.

Table 1
Attacking agent's attack implementation.

Attack Type	Valid MAC	Valid Data Set	Valid Timestamp	Valid State
Denial of Service	☐	☐	☐	☐
Replay Attack	■	■	☐	☐
FDIA	■	■	■	■

■ - Implemented ☐ - Not Implemented.

publisher will then continue to send the state and expected boolean value until a response is received or the next event is triggered. If the next event is triggered, the state number will increase, but the boolean value will stay the same, to eliminate the indication of false positive responses. The response time of a DUT is measured utilizing a Linux nanosecond timer. While this timer may not be accurate to within 1 nanosecond, this timer is sufficient in measuring timing within microsecond accuracy, which provides sufficient resolution for our purposes.

The Attacking Agent receives parameters for the type of attack to be generated and handles packet generation and packet transmission. The packet generation flow can be seen in Fig. 5. During packet generation, the Data Set parameters and MAC endpoint are optionally sent to the attacking agent for implementation. Without these parameters, a predefined data set and MAC address are used to generate the attack traffic. Regardless of parameter inputs, the indices for each packet field are stored, in order to edit packets in case of more complex attacks such as FDIA. The implementation of different attacks in this testbed can be seen in Table 1.

Since the packet creation is dynamic, the required packets sent to achieve the desired attack rate must also be dynamic. For this reason, existing open-source libraries were not utilized, and custom code was written. The traffic generation method, outlined in Algorithm 1, utilizes a Linux nanosecond timer to track the elapsed time since the previously sent single-packet or packet burst. Our testbed implementation favors a burst approach to optimally utilize the asynchronous nature of raw sockets available to C code. This algorithm utilizes set burst sizes for different data rates, with optimal values determined empirically. The burst approach also allows us to recover attack timing synchronization in case of unforeseen system stalls in message generation that would result in lower-than-desired attack traffic being generated. Since Linux is not a real-time operating system, traffic throughput is not guaranteed. Thus our system utilizes dynamic bursts and a feedback system to adapt traffic generation to system capabilities automatically.

To coordinate the attacks, the Scripting Node utilizes access to the GOOSE Publisher and the Attacking Agent. The Scripting Node establishes a secure connection to both nodes before requesting the generation of the regular GOOSE traffic and the attack traffic. The Scripting Node passes the mentioned parameters to both devices and creates customizable, replicable tests for the DUT. To validate that attacks operate as intended, the Scripting Node utilizes TShark [29] to monitor average traffic sent throughout this test infrastructure.

4.2. Devices

The devices evaluated within this work are two commercially available production-grade IEDs from reputable manufacturers within the energy OT device sector. For due diligence and to prevent malicious abuse of our findings, we opt not to disclose the names of vendors or products utilized in our evaluations. The first tested device is a protection system with a wide range of customizations available to its functionality. Regarding the GOOSE protocol, the number of configurable publishers and subscribers can vary greatly. In our utilized configuration for this device, we opted for a single subscription with one bit being tracked for measured response time. This configuration is the least complex possible for the device, allowing for prioritization of only the measured subscription. The virtual bit measured for this configuration expresses the current value of a boolean value in the GOOSE subscription. For the rest of this paper, this device will be referred to as Device 1.

The second device we tested is a protective relay. The device is advertised as a general-purpose device and allows for a set number of GOOSE publications and subscriptions. Similar to the configuration of Device 1, this device was configured with a single subscription with a single monitored bit for output, as outlined earlier in this section. We will refer to this device as Device 2.

Algorithm 1 Traffic Generation Timing

```
Require: load, duration, packets, bursts
  t_n ← current time nanoseconds
  t_s ← current time seconds
  burst size = packets / bursts
  burst duration = 1000000000 / bursts
  temp burst = burst size
  elapsed = 0
  while elapsed < duration do
    sent = 0
    slept = 0
    current burst = t_n
    while sent < packets do
      next burst = current burst + burst duration
      if slept then
        burst size = 2 * burst size
        next burst = current burst + burst duration
        slept = 0
      end if
      for i in burst size do
        send packet
      end for
      burst size = temp burst
      sent += burst size
      current burst = t_n
      if current burst > next burst then
        slept = 1
      else
        while current burst < next burst do
          current burst = t_n
        end while
      end if
    end while
    elapsed = t_s
  end while
```

Table 2
Individual device response success rate vs. average response time.

Traffic	Device 1				Device 2			
(Mbps)	DoS		Replay		DoS		Replay	
	Success %	Average Response	Success %	Average Response	Success %	Average Response	Success %	Average Response
0	100	6.403	100	6.403	100	14.823	100	14.823
10	100	5.707	65.5	5.316	100	14.274	16	464.952
20	100	5.697	64.7	5.635	100	14.347	15.3	465.014
30	100	5.691	20.4	4.259	100	14.456	2.9	468.654
40	100	5.781	0	–	100	14.410	1.5	467.383
50	100	5.813	0	–	100	14.114	1.1	464.275
60	100	5.697	0	–	100	14.375	0.6	344.841
70	100	5.785	0	–	100	14.179	0.4	477.667
80	100	6.014	0	–	100	14.208	0.3	477.720
90	100	6.211	0	–	100	14.145	0.4	485.214

5. Results

Each device is evaluated through DoS, Replay, and FDIA attacks, with attack data rates selected from the range of 10–90 Mbps, in 10 Mbps increments. Our experiments showed that there was no need to generate data rates in excess of 90 Mbps, since both devices utilize a 100BASE-TX connection with a link saturation shown to be reached at 84 Mbps [11]. The tests were performed for 1000 events per test point, controlled by the GOOSE Publisher, which was selected in order to provide statistical reliability of the collected data. Each test utilized a network with a single publisher to MAC 01:0C:CD:01:00:03. Each packet was 125 bytes in length, with a data set containing 3 items. The items were a boolean, a float-string, and a bit-string totaling 15 bytes. The packet structure was chosen to illustrate the testbed's ability to utilize all different lengths of publications. With smaller publications, the attacking agent must perform more transmissions and packet updates to meet a specified data rate. The results for each device against the DoS and Replay attacks can be seen in Table 2. These results will be examined in further detail in the corresponding subsections for each device. The FDIA attack will be investigated in a separate subsection, as neither device was able to respond correctly under these attacks.

5.1. DoS and Replay attach results for Device 1

The first device tested shows a reliable success rate against all attack traffic during our DoS attacks, and also a reasonably high success during up to 20 Mbps of Replay attack traffic. Furthermore, the response time of this device is stable against all traffic sent, with the average response never deviating above 6.5 ms. However, our tests revealed abnormal device behavior when executing 1000 attack traffic cycles at or above 40 Mbps, indicating a significant vulnerability in the device's implementation. The traffic sent against this device for the tests can be seen in Fig. 6. The traffic sent against Device 1 was measured in 30-s intervals throughout the testing period of 30 min. While there are fluctuations within the attack traffic, the average over the test cycle closely approximates the desired data rate.

A closer inspection of the response time is shown in the boxplot of Fig. 7. This figure shows the response time variance does not change significantly between different DoS attack traffic rates. Accounting for the possible performance impact that closing a mechanical contact exhibits, there does not appear to be any significant processing issues resulting from varying attack traffic. The Replay attack results observed for Device 1 appear to confirm this observation, with the results shown in Fig. 8.

For the Replay attack, Device 1 exhibited behavior of significant interest, however. As can be observed in the results for the Replay attack, shown in Fig. 8, we failed to collected results at or above 30 Mbps of attack traffic, as the device did not reliably respond during those tests. More information on that observation will be provided below. For the tests where we could collect test results, the responses

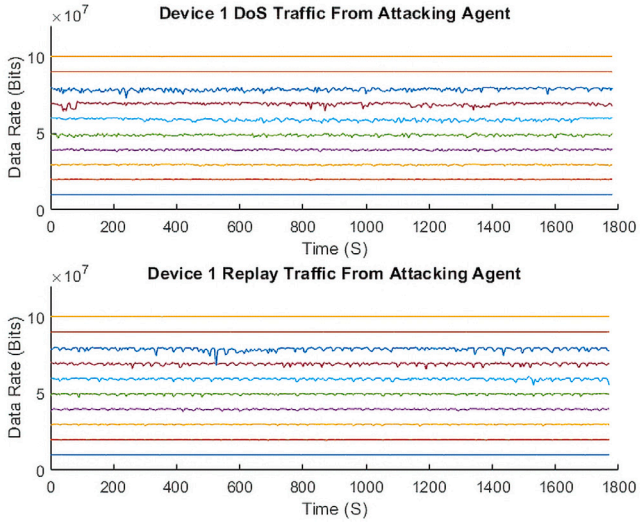

Fig. 6. Attack traffic against Device 1, in increments of 10 Mbps.

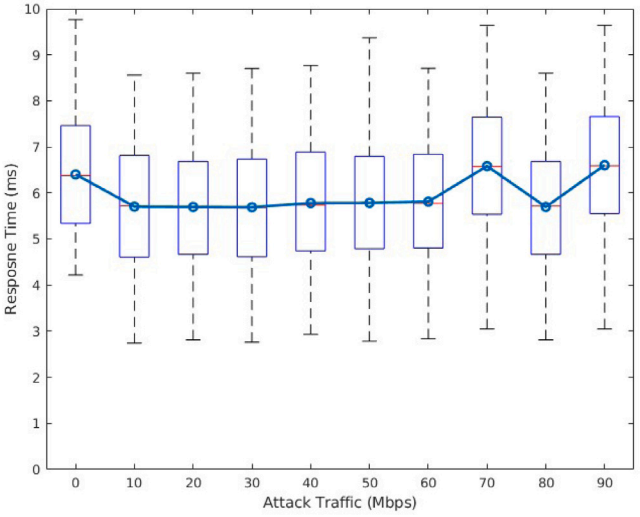

Fig. 7. Device 1 DoS response timing results.

appear to follow a trend of faster response times than those observed for the DoS attack. At 30 Mbps, the minimum response is the same as other tests, but the maximum response is much lower. This is likely due to the number of responses, as only a 20% success rate was achieved compared to the above 60% success rate achievable at lower

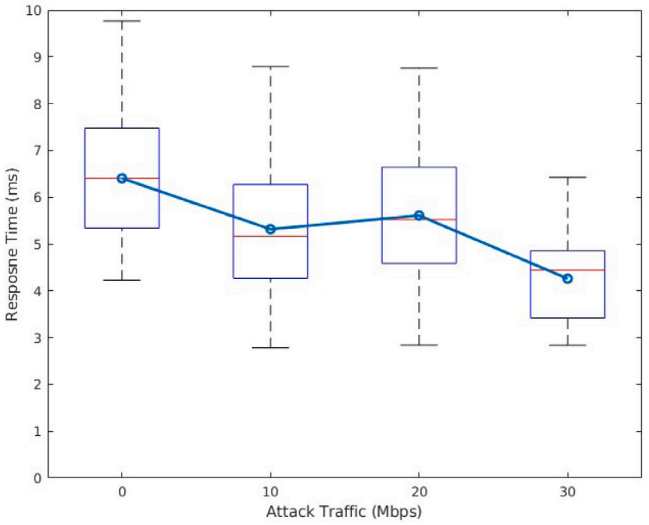

Fig. 8. Device 1 replay response.

Fig. 9. Device 1 disabled display.

traffic rates during our Replay attack testing. An additional important takeaway from this study is the lack of outliers within the data. While there is still a variance within the measured responses, the device performs without large jumps in response rates.

With regard to the failing tests during the Replay attack behavior evaluation conducted for 40 Mbps and above, we observed that during those instances the display of Device 1 depicted the message shown in Fig. 9. When this message appeared, Device 1 would immediately become unresponsive to any external communication. Previous methods to connect through SSH, or even simply pinging the IP of the device were unresponsive as well. USB communication also failed to read device information and the USB host did not recognize the device anymore. Even stopping the attack traffic and attempting to resume normal operations yielded no response either. When we reached this device state, it became clear that Device 1 was fully disabled. Further investigation showed that the only form for recovering normal operations was a complete physical power cycle of the device.

Only after power cycling the device was the protection system able to resume normal operation. At this point, further investigation was conducted to identify the root cause of disabling Device 1. To validate the requirements of disabling the device, the initial DoS attacks were repeated with augmented ***gocbref***, ***goid***, ***logical node***, and ***dataset*** fields. By individually changing each of these packet fields, we were able to identify which GOOSE packets could disable the device. The attacks were only successful in disabling the device with a GOOSE packet containing the correct ***gocbref***, ***goid***, ***logical node***, and ***dataset***, which is identical to the Replay attacks we executed. From this result, we can conclude that the device is processing each packet regardless of valid timestamp, so the testbed is able to overwhelm the device with traffic, causing a device fault. The results collected from our experimentation determined the thresholds for this device behavior occurring as shown in Fig. 10. While we did observe the behavior for attack traffic as low as 40 Mbps, we could reliably replicate this behavior for the threshold of 58 Mbps for 30 s for Replay attacks, and for 53 Mbps for 30 s for the FDIA attacks to be discussed further below.

The higher traffic rate required for this behavior to occur in Replay attacks is likely due to the timestamp being outdated. Comparing the two attacks that disabled Device 1, the timestamp is outdated within the Replay packet while the FDIA packet has a correct timestamp within the TTL field. This points to the internal state machine checking the timestamp of the message last, which introduces a processing risk by decoding and interpreting each received GOOSE packet. This indicates that there is theoretically also a path for disabling Device 1 based on a DoS attack with correct MAC addresses being used. However, given the limits of the 100BASE-TX connection utilized by this device, only packets where virtually all fields contain valid values can disable the device due to the increased message validation processing latency. Under normal operating conditions, this would not cause an issue for the IED, but this is an example of a protocol vulnerability causing unanticipated hardware impacts. Since our discovery of this device vulnerability, we have been in contact with the manufacturer to report the issue and could confirm that a subsequent firmware update resolves the issue.

5.2. DoS and Replay attach results for Device 2

The second device evaluated has a far different response to the attacks than Device 1. The initial responses without any attack traffic show a higher response time at 14.8 ms latency from sending the GOOSE packet to measuring the contact closing in direct response to receiving and processing this GOOSE packet. The traffic generated while evaluating this device can be seen in Fig. 11. Similar to Device 1, the DoS attacks have little effect on the device's responses, as there is less than 1 ms variance between all average responses. The Replay attack, however, has a greater effect on the response time, which grows to nearly 500 ms.

The variance of responses for each attack traffic for the DoS attack is shown in Fig. 12. This variance shows a greater range than Device 1, but the range of variance is not adversely affected by the DoS attack traffic. Any outliers that appear within the responses are few, and do not exceed a 25 ms response time.

The Replay attack responses shown in Fig. 13, exhibit a very different behavior than the response timing for the DoS attacks. While Device 1 was able to maintain comparable performance up to an attack traffic rate of 30 Mbps, Device 2's operation was severely degraded by the Replay attack. As soon as the attack traffic began, the response time deteriorated from 14.8 ms to 464.9 ms on average, which is an increase in latency be a factor of 31. The response rate also decreased dramatically to 16% at 10 Mbps attack traffic, and decreasing even further to only 2.9% successful responses at 30 Mbps. However, the response rates after this initial jump remained similar, with responses never climbing past 500 ms. An unexpected dip was observed at 60 Mbps, likely due to the significantly reduced number of successful responses from the device.

Examining Fig. 14, we can see the significant discrepancy in the success rates between the two devices. This shows the variance of operation between two different devices that supposedly have the same functionality. Both devices were significantly affected by the Replay attack, but Device 1 appeared to be faring better until the attack traffic reached 40 Mbps. While Device 1 was able to maintain fast responses in the presence of this attack traffic, by exceeding a certain threshold in attack traffic rate sent during Replay or FDIA attacks, we could disable the device entirely. To recover from the attack, the device required manual intervention through power cycling to resume normal operation. By contrast, Device 2's performance was severely impacted by the Replay attack, causing a 31x increase in latency at just 10 Mbps. However, Device 2 could always return to normal operation without manual intervention once the attack subsided.

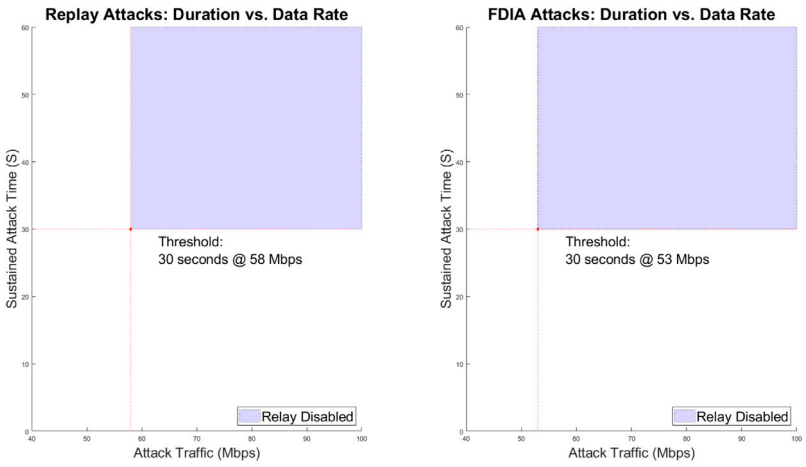

Fig. 10. Attacks disabling Device 1 and their traffic rate and duration after which the device could be disabled.

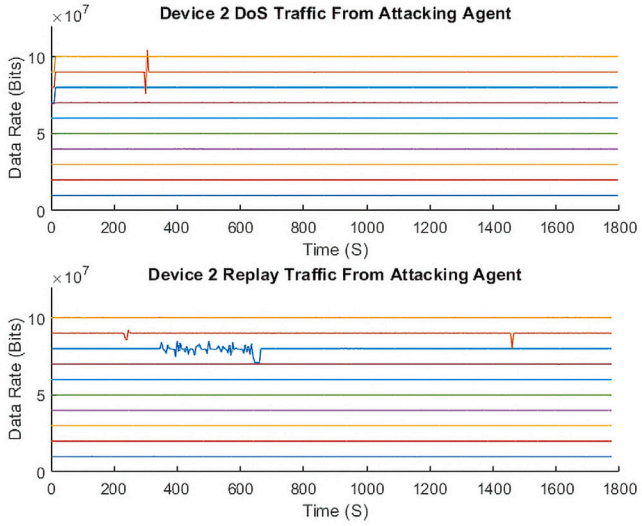

Fig. 11. Attack traffic against Device 2, in increments of 10 Mbps.

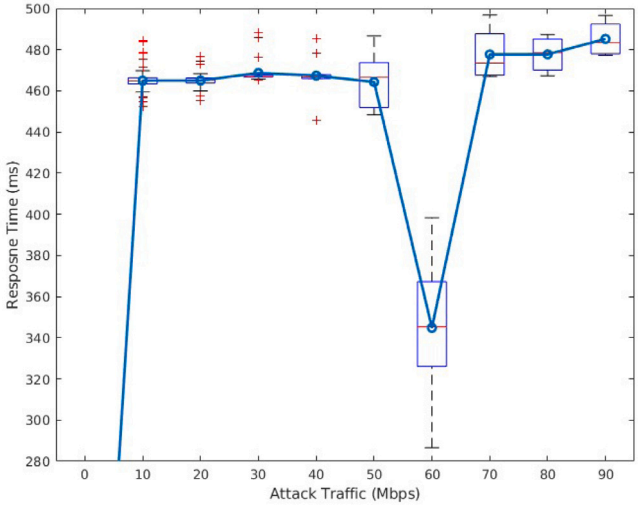

Fig. 13. Device 2 replay response.

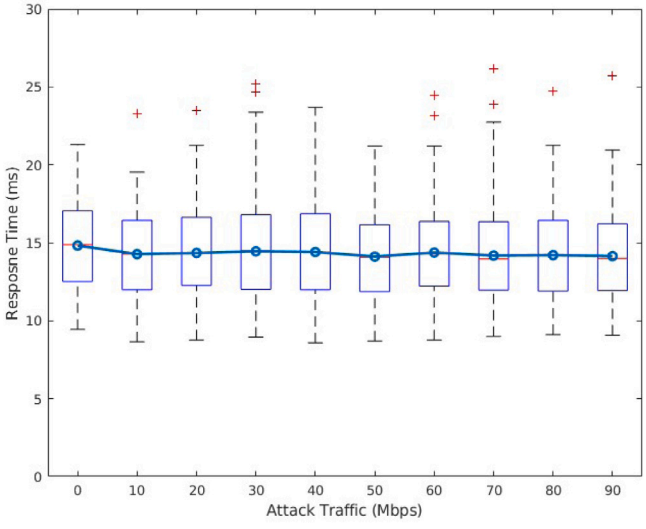

Fig. 12. Device 2 DoS response.

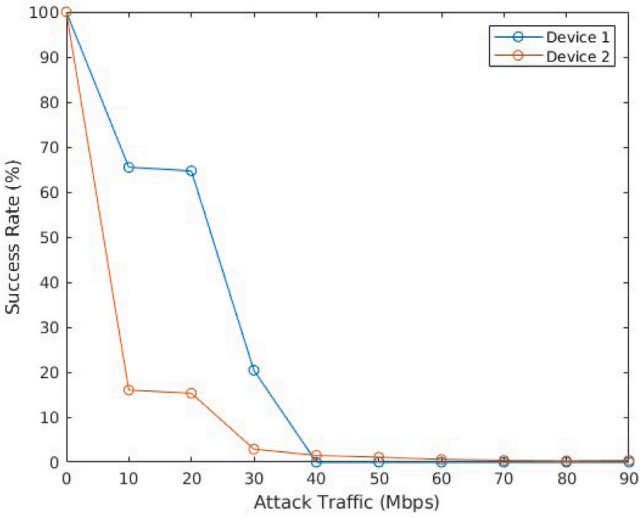

Fig. 14. Successful response rate under replay attack.

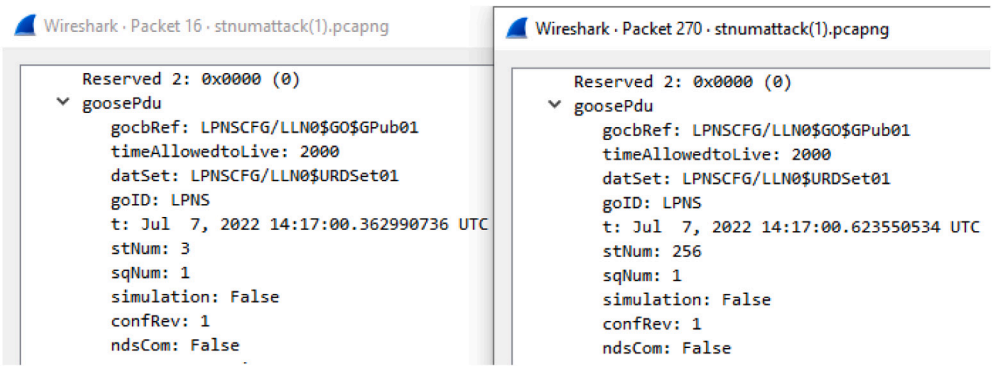

Fig. 15. State number attack.

5.3. False data injection attack results for both devices

The final attack executed against the devices we evaluated is the FDIA. For this attack, a state number attack was executed at the defined data rates of 10–90 Mbps. The testbed allows for all packet fields to be edited and can also inject false measurements into a targeted device. In our testing we chose ***state number*** updates with a fixed boolean value for keeping the contact open as one such targeted measurement to outline the efficacy of the FDIA. Under these attack loads, the devices were unable to register a single correct response. The cause for this is likely due to the speed at which the state number is increasing, as shown in Fig. 15. At an attack traffic rate of just 1 Mbps, with a packet size of 125 bytes, 1000 packets are sent each second to the device. For comparison, under normal operating conditions, such as those performed by the GOOSE Publisher, 1000 cycles take over 30 min to complete. This means that, with valid timestamps, one second of attack traffic will disable the relay for at least 30 min.

However, the attack was not always detrimental to the recovery of both devices. After an attack, the state of the subscriber in Device 1 would timeout depending on the ***timeAllowedtoLive*** field. Once the last packet of the FDIA attack was sent and the timeout occurred, new packets with lower states would be properly processed. In this way, the device was more robust to the FDIA attack. With higher traffic rates, the FDIA would cause packets to be discarded due to a lack of buffer space, as shown in Fig. 16. The code shown, "Out of Sequence", appears to indicate a self-recovery feature within the device. Since Device 1 was able to recover from all attacks that did not disable it, the device may utilize the codes shown to allow for new publications with lower state numbers to be processed. However, Device 2 did not operate under the same timeout conditions and would become inoperable until either the state number of valid packets surpassed the current subscriber count number, or the device was power cycled. This shows Device 2 is robust against attacks that target a specific Ethernet interface. However, attacks that target protocol-specific vulnerabilities do not implement self-recovery features.

6. Future work

This paper introduced a flexible OT testbed to test device behavior against cyberattacks targeting the IEC-61850 GOOSE protocol as a case study. While these results are promising, and show the impact that varying hardware implementations of the same protocol can have on IED operations, there are additional protocols that can be tested, such as MMS, SV, Modbus, and DNP3. This tesbed can also be added to previously mentioned HIL simulators to measure the impact of attacks on fully modeled substations.

Additional studies on devices from other manufacturers would also provide a means to contrast different device behaviors and identify common trends within IED operations. This testbed can also assist in identifying device-specific vulnerabilities and aid manufacturers in resolving potential issues in both pre-release and post-release product testing. The testbed can also be expanded to work against emulated substations, to identify widespread effects of attacks targeted at specific GOOSE publications and other protocols and attacks.

7. Conclusion

With the convergence of the OT and IT sectors, the introduction of protocols that increase data visibility across the power grid have also resulted in an expanded attack surface for malicious actors. Each protocol implements automation and security differently, and some protocols may exhibit more security vulnerabilities than others. What is needed is a more comprehensive approach to identifying and rectifying these vulnerabilities. In this paper, we demonstrate this capability provided by our developed OT testbed, by performing a case study on the IEC-61850 GOOSE protocol to evaluate its hardware implementation within two commercially available and widely deployed IEDs. The GOOSE protocol was chosen due to its high-priority messaging and un-encrypted packet structure.

The testbed presented in this paper utilized dynamically generated attack traffic, which can be employed to identify specific device vulnerabilities for the IEDs with specific protocol support. For this case study, the vulnerabilities of the GOOSE protocol were identified during DoS, Replay, and FDIA attacks. The testbed's utilization of a separate attacking agent and scripting node allows for the creation of customizable test configurations, which gives access to adjustable traffic rate, duration, and frequency. This is a clear advantage over existing testbeds that utilize predefined attack data sets or user input for each attack deployed and do not allow user-defined data rates for specific attacks.

These attacks were then implemented against the two devices at data rates ranging from 10–90 Mbps. The DoS attack had limited effect on the performance of both devices tested, but the Replay and FDIA attacks had lasting impacts on each device. The first device tested was disabled after a high enough data rate, determined to be 58 Mbps for Replay and 53 Mbps for FDIA attacks occurring for a minimum of 30 s. For FDIA attacks that did not disable the device, it was able to recover when the previous packet timed out. Contact with the manufacturer verified this vulnerability was resolved through a firmware update. Device 2 saw an immediate impact with response times increasing to over 450 ms for any attack traffic. This device was also unable to recover after a state number attack, which required the valid publication to subsequently exceed the last attack state number. These contrasting results show that while devices advertise the same protocol capabilities, their performance can differ greatly, and it is therefore of utmost importance to be able to evaluate and quantify these behavior variations.

These findings clearly highlight the use case for this OT testbed. While the devices are marketed with the same GOOSE protocol capability, their performance in the face of attacks varies greatly. While the

```
GOOSE Receive Status

 MultiCastAddr    Ptag:Vlan AppID  StNum       SqNum       TTL     Code
-------------------------------------------------------------------------------
LPNSCFG/LLN0$GO$GPub01
 01-0C-CD-01-00-03 0:1     3      2011        0           2000    OUT OF SEQUENC
 Data Set: LPNSCFG/LLN0$URDSet01


=>goo s 1

 SubsID 1
-------------------------------------------------------------------------------
 Ctrl Ref: LPNSCFG/LLN0$GO$GPub01
 AppID   : 3
 From    : 07/06/2022 21:23:33.389  To: 07/07/2022 14:21:32.095

 Accumulated downtime duration                      : 0016:57:54.601
 Maximum downtime duration                          : 0016:46:18.196
 Date & time maximum downtime began                 : 07/06/2022 21:23:33.427
 Number of messages received out-of-sequence(OOS)   : 208
 Number of time-to-live(TTL) violations detected    : 3
 Number of messages incorrectly encoded or corrupted: 0
 Number of messages lost due to receive overflow    : 1795
 Calculated max. sequential messages lost due to OOS: 9
 Calculated number of messages lost due to OOS      : 1794
```

Fig. 16. Device 1 GOOSE status [11].

so-called Defense in Depth strategy is recommended for OT systems, the identification and mitigation of each device's vulnerabilities can further improve the robustness of their operation, as consequently also that of the power grid.

CRediT authorship contribution statement

Matthew Boeding: Methodology, Software, Investigation, Visualization, Writing – original draft. **Michael Hempel:** Conceptualization, Validation, Supervision, Writing – review & editing. **Hamid Sharif:** Conceptualization, Project administration, Resources, Supervision, Writing – review & editing. **Juan Lopez Jr:** Project administration, Methodology, Validation, Supervision, Writing – review & editing. **Kalyan Perumalla:** Project administration, Supervision, Writing – review & editing.

Declaration of competing interest

The authors declare the following financial interests/personal relationships which may be considered as potential competing interests: Hamid Sharif reports financial support was provided by Oak Ridge National Laboratory. Hamid Sharif reports financial support was provided by University of Nebraska-Lincoln Nebraska Center for Energy Sciences Research.

Funding

This research has been supported in part by the Department of Energy Cybersecurity for Energy Delivery Systems program, and the Oak Ridge National Laboratory, under grants 4000175929 and 4000193048. It has also been supported in part by the University of Nebraska-Lincoln's Nebraska Center for Energy Sciences Research (NCESR) under Cycle 16 Grant# 20-706.

Data availability

The authors are unable or have chosen not to specify which data has been used

References

[1] M. Boeding, K. Boswell, M. Hempel, H. Sharif, J. Lopez, K. Perumalla, Survey of cybersecurity governance, threats, and countermeasures for the power grid, Energies 15 (22) (2022) http://dx.doi.org/10.3390/en15228692, URL https://www.mdpi.com/1996-1073/15/22/8692.

[2] IEC62351-1:2007, Power Systems Management and Associated Information Exchange—Data and Communications Security Part 1: Communication Network and System Security—Introduction to Security Issues, International Electrotechnical Comission, 2007.

[3] I. Kharchouf, A. Alrashide, M.S. Abdelrahman, O.A. Mohammed, On the implementation and security analysis of routable-GOOSE messages based on IEC 61850 standard, in: 2022 IEEE International Conference on Environment and Electrical Engineering and 2022 IEEE Industrial and Commercial Power Systems Europe, EEEIC / I&CPS Europe, 2022, pp. 1–6, http://dx.doi.org/10.1109/EEEIC/ICPSEurope54979.2022.9854415.

[4] S. Mocanu, J.-M. Thiriet, Experimental study of performance and vulnerabilities of IEC 61850 process bus communications on HSR networks, in: 2020 IEEE European Symposium on Security and Privacy Workshops, EuroS&PW, 2020, pp. 584–593, http://dx.doi.org/10.1109/EuroSPW51379.2020.00085.

[5] J. Zhang, J. Li, X. Chen, M. Ni, T. Wang, J. Luo, A security scheme for intelligent substation communications considering real-time performance, J. Mod. Power Syst. Clean Energy 7 (4) (2019) 948–961, http://dx.doi.org/10.1007/s40565-019-0498-5.

[6] IEC61850-7-1:2011, Communication Networks and Systems for Power Utility Automation - Part 7-1: Basic Communication Structure - Principles and Models, Vol. 2011, International Electrotechnical Commission, 2011.

[7] M. Hemmati, M.H. Palahalli, G. Storti Gajani, G. Gruosso, Impact and vulnerability analysis of IEC61850 in smartgrids using multiple HIL real-time testbeds, IEEE Access 10 (2022) 103275–103285, http://dx.doi.org/10.1109/ACCESS.2022.3209698.

[8] S.S.M. Reshikeshan, M.S. Illindala, Systematically encoded polynomial codes to detect and mitigate high-status-number attacks in inter-substation GOOSE communications, in: 2020 IEEE Industry Applications Society Annual Meeting, 2020, pp. 1–7, http://dx.doi.org/10.1109/IAS44978.2020.9334776.

[9] J. Hong, T.-J. Song, H. Lee, A. Zaboli, Automated cybersecurity tester for IEC61850-based digital substations, Energies 15 (21) (2022) http://dx.doi.org/10.3390/en15217833, URL https://www.mdpi.com/1996-1073/15/21/7833.

[10] P. Jafary, A. Supponen, S. Repo, Network architecture for IEC61850-90-5 communication: Case study of evaluating R-GOOSE over 5G for communication-based protection, Energies 15 (11) (2022) http://dx.doi.org/10.3390/en15113915, URL https://www.mdpi.com/1996-1073/15/11/3915.

[11] M. Boeding, M. Hempel, H. Sharif, J. Lopez Jr., K. Perumalla, A testbed for evaluating performance and cybersecurity implications of IEC-61850 GOOSE hardware implementations, in: 2023 IEEE Consumer Communications & Networking Conference, 2023.

[12] S. Ashraf, M.H. Shawon, H.M. Khalid, S.M. Muyeen, Denial-of-service attack on IEC 61850-based substation automation system: A crucial cyber threat towards smart substation pathways, Sensors 21 (19) (2021) http://dx.doi.org/10.3390/s21196415, URL https://www.mdpi.com/1424-8220/21/19/6415.

[13] T.S. Ustun, S.M. Farooq, S.M.S. Hussain, A novel approach for mitigation of replay and masquerade attacks in smartgrids using IEC 61850 standard, IEEE Access 7 (2019) 156044–156053, http://dx.doi.org/10.1109/ACCESS.2019.2948117.

[14] J. Hoyos, M. Dehus, T.X. Brown, Exploiting the GOOSE protocol: A practical attack on cyber-infrastructure, in: 2012 IEEE Globecom Workshops, 2012, pp. 1508–1513, http://dx.doi.org/10.1109/GLOCOMW.2012.6477809.

[15] J.G. Wright, S.D. Wolthusen, Stealthy injection attacks against IEC61850's GOOSE messaging service, in: 2018 IEEE PES Innovative Smart Grid Technologies Conference Europe, ISGT-Europe, 2018, pp. 1–6.

[16] J. Noce, Y. Lopes, N.C. Fernandes, C.V.N. Albuquerque, D.C. Muchaluat-Saade, Identifying vulnerabilities in smart grid communication networks of electrical substations using GEESE 2.0, in: 2017 IEEE 26th International Symposium on Industrial Electronics, ISIE, 2017, pp. 111–116, http://dx.doi.org/10.1109/ISIE.2017.8001232.

[17] A.A. Memon, K. Kauhaniemi, Real-time hardware-in-the-loop testing of IEC 61850 GOOSE-based logically selective adaptive protection of AC microgrid, IEEE Access 9 (2021) 154612–154639, http://dx.doi.org/10.1109/ACCESS.2021.3128370.

[18] G. Ravikumar, B. Hyder, J.R. Babu, K. Khanna, M. Govindarasu, M. Parashar, CPS testbed architectures for WAMPAC using industrial substation and control center platforms and attack-defense evaluation, in: 2021 IEEE Power & Energy Society General Meeting, PESGM, 2021, pp. 1–5, http://dx.doi.org/10.1109/PESGM46819.2021.9638183.

[19] H. Reda, B. Ray, P. Peidaee, A. Anwar, A. Mahmood, A. Kalam, N. Islam, Vulnerability and impact analysis of the IEC 61850 GOOSE protocol in the smart grid, Sensors 21 (2021) 1554, http://dx.doi.org/10.3390/s21041554.

[20] S.M.S. Hussain, T.S. Ustun, A. Kalam, A review of IEC 62351 security mechanisms for IEC 61850 message exchanges, IEEE Trans. Ind. Inform. 16 (9) (2020) 5643–5654, http://dx.doi.org/10.1109/TII.2019.2956734.

[21] J. Claveria, A. Kalam, GOOSE protocol: Ied's Smart Solution for Victoria University Zone Substation (VUZS) simulator based on IEC61850 standard, in: 2018 IEEE PES Asia-Pacific Power and Energy Engineering Conference, APPEEC, 2018, pp. 730–735, http://dx.doi.org/10.1109/APPEEC.2018.8566413.

[22] Z. Pourmirza, A. Srivastava, Cybersecurity analysis for the communication protocol in smart grids, in: 2020 IEEE 8th International Conference on Smart Energy Grid Engineering, SEGE, 2020, pp. 58–63, http://dx.doi.org/10.1109/SEGE49949.2020.9182015.

[23] A. Alrashide, M.S. Abdelrahman, I. Kharchouf, O.A. Mohammed, GNS3 communication network emulation for substation GOOSE based protection schemes, in: 2022 IEEE International Conference on Environment and Electrical Engineering and 2022 IEEE Industrial and Commercial Power Systems Europe, EEEIC / I&CPS Europe, 2022, pp. 1–6, http://dx.doi.org/10.1109/EEEIC/ICPSEurope54979.2022.9854689.

[24] G. Elbez, H.B. Keller, V. Hagenmeyer, A cost-efficient software testbed for cyber-physical security in IEC 61850-based substations, in: 2018 IEEE International Conference on Communications, Control, and Computing Technologies for Smart Grids, SmartGridComm, 2018, pp. 1–6, http://dx.doi.org/10.1109/SmartGridComm.2018.8587456.

[25] S. Zemanek, I. Hacker, K. Wolsing, E. Wagner, M. Henze, M. Serror, PowerDuck: A GOOSE data set of cyberattacks in substations, in: Proceedings of the 15th Workshop on Cyber Security Experimentation and Test, CSET '22, Association for Computing Machinery, New York, NY, USA, 2022, pp. 49–53, http://dx.doi.org/10.1145/3546096.3546102.

[26] P.P. Biswas, H.C. Tan, Q. Zhu, Y. Li, D. Mashima, B. Chen, A synthesized dataset for cybersecurity study of IEC 61850 based substation, in: 2019 IEEE International Conference on Communications, Control, and Computing Technologies for Smart Grids, SmartGridComm, 2019, pp. 1–7, http://dx.doi.org/10.1109/SmartGridComm.2019.8909783.

[27] Libiec61850: Open source library for IEC 61850, [online] Available: http://libiec61850.com/libiec61850/.

[28] Raspberry pi operating system, 2023, [online] Available: https://www.raspberrypi.com/software/ (Accessed 15 February 2023).

[29] Wireshark: Network protocol analyzer, 2023, [online] Available: https://www.wireshark.org/. (Accessed 15 February 2023).