Stealthy Injection Attacks Against IEC61850's GOOSE Messaging Service

James G. Wright
School of Mathematics and Information Security
Royal Holloway, University of London
Egham TW20 0EX
United Kingdom
Email: james.wright.2015@live.rhul.ac.uk

Stephen D. Wolthusen
School of Mathematics and Information Security
Royal Holloway, University of London
Egham TW20 0EX
United Kingdom
Norwegian Information Security Laboratory
Norwegian University of Science and Technology
Norway
Email: stephen.wolthusen@rhul.ac.uk
stephen.wolthusen@ntnu.no

***Abstract*—IEC61850 and IEC62351 combined provide a set of security promises for the communications channels that are used to run a substation automation system (SAS), that use IEC61850 based technologies. However, one area that is largely untouched by these security promises is the generic object oriented substation events (GOOSE) messaging service. GOOSE is designed to multicast commands and data across a substation within hard real time quality of service (QoS) requirements. This means that GOOSE is unable to implement the required security technologies as the added latency to any message would violate the QoS.**

The focus of the security research into the GOOSE messaging service has been on how it can be used to undermine IEC61850's security promise of availability, but these attacks will be detected in time. Given GOOSE's lack of security it is likely that the messaging service will be used to propagate a small number of messages to force the SAS to perform undesired actions, whilst avoiding undermining availability. It appears none of the analysis into GOOSE's security has considered this.
This analysis looks to find the minimum parameters that an adversary would have to fulfil, and probability of success, to inject a single malicious message into a intelligent electronic device (IED). This work develops a model for calculating the likelihood a malicious message will be successfully injected given a rate message injection, using a single $M/M/1/K$ queue. It then uses this model to test the effectiveness of various countermeasures, such as a message buffer, rate limiting, a threshold of the number of malicious messages in the required for detection, and stringent enforcement of the QoS requirements.

I. Introduction

The adoption of the smart grid (SG) design paradigm for the operation of electrical grids degrades the air gap security principle that has been used by the energy sector for the past few decades. The guiding principle of security through obscurity of supervisory control and data acquisition (SCADA) protocols is no longer tenable, as their communications networks begin to interact with internet technologies. Whilst both the academic and industrial research communities are now focusing on solving the unique security challenges created by using SG technologies, bespoke attacks against SG networks have already begun to emerge in the wild. In December 2016 the Ukrainian power grid was affected by the malware 'CRASHOVERRIDE', which proceed to deliver a payload that shut off a transmission substation[1]. The analysis of the malware showed that it was not only designed to undermine the security and QoS promises of the IEC60870 and IEC61850 SAS protocols, but use the authors detailed understanding of the protocols to use specific logical nodes (LN) to cause damage to the distribution network. A way of counteracting these threats is to develop secured communications standards that undermine a malicious actor's ability to gain a foothold in the network.

The GOOSE messaging service is one of the two models within IEC61850 that enables multicast association of devices in a substation. It is designed as a fast and reliable messaging system to distribute time critical data and commands to the relevant IEDs, such as commands to activate the substation's protection equipment. Due to the hard real time constraints $(3-4ms)$ and high level of reliability of all the messages the GOOSE messaging service is designed to be non-routeable, an entire message sent in a single packet, and received without acknowledgement. A GOOSE message is multicast across the substation's network using a publish-subscribe model, so different subscribers can be reached for different calibrated events[2].
The only security promises that the GOOSE messaging service must uphold are availability and integrity from transmission error. The standard does add that the GOOSE subscribers must be able to detect and dispose of duplicate messages. Whilst the IEC62351 does integrate traditional security promises into the IEC61850 standard, it explicitly does not add them to GOOSE. It declares "*for applications using GOOSE and IEC 61850-9-2 and requiring 4ms response times, multicast configurations, and low CPU overhead, encryption is not recommended*"[3]. This is due to the added latency of encrypting or digitally signing a GOOSE message using current IED hardware would violate

the time constraints of the QoS promises[4]. The only security feature that IEC62351 adds to GOOSE messaging serivce is a replay protection algorithm[3]. The IEC61850 and IEC62351 make no considerations for the other specific security problems faced by multicast protocols, which will be covered in section II. These are access control or group address obfuscation mechanisms. Having a policy on this would hinder an adversary's attempt to used the GOOSE service as an attack vector with which they could map and manipulate the communications network.

The combination of the lack of acknowledgement of message reciept and limited security protection makes the GOOSE service an ideal attack vector against an SG communications network. For this reason GOOSE has been one of the few models within IEC61850 that has been probed by the research community for vulnerabilities (an analysis will be provided in section II). The main focus of GOOSE attack research has been attacks against availability of devices, to prevent critical messages from being received, as well as easily propagated across the network due to multicast messaging services usually having a high workflow amplification factors[5]. However, there seems to have been no consideration about whether the service is susceptible to injection class attacks.

This analysis will describe the minimum necessary conditions that an adversary must meet for their injection attack to be successful, and remain undetected, given the stringent QoS time constraints of the GOOSE service. The objective of the adversary in this analysis is that they wish to achieve their disruption with only a few messages, instead of denying availability. It then goes on to demonstrate the effectiveness of this attack vector and consideration various counter measures. It is shown that rate limiting and having a small message buffer decreases the chance of a successful injection. Strict enforcement of the message arrival QoS is also discussed, but it is shown that it opens up another attack vector to be exploited.
The analysis, in section IV, is done using a model generated in the authors' queueing theory framework[6], which uses a $M/M/1/K$ queue. This framework is designed to model protocol semantic attacks against availability and de-synchronisation. The analysis uses some of the features of the framework, which are described in section III, to model messages entering an IED's communication channel.

II. Related Work

This section reviews the current state of the art of multicast and GOOSE security, as well as the use of queuing theory in security research.

There are taxonomies for the specific security considerations of a multicast system. Judge *et al.*[7] laid out the the three multicast features that require specific security considerations. They are that all group members receive the same message, the group membership is transparent to the source, and the source of a message doesn't have to be a member of the group. This taxonomy then goes on to proposing a series of solutions to negate each attack vector. Canetti *et al.*[8] discussed several of the security issues facing multicast technology, and then propose a solution for securing the source and message authentication. Their scheme was for each sender to hold a collection of keys, and each recipient held a subset of those. Their specific subset of the keys would allow them to authenticate the specific sender of a message.

Research into the security of the GOOSE messaging service is focused on what malicious actions an adversary can do within the rules of the service. Most of the attacks discussed below are done with the inclusion of, and sometimes caused by, the replay attack protection of IEC62351. Hoyos *et al.*[9] demonstrated a GOOSE spoofing attack where the adversary injects malicious copies of legitimate message, with an incremented $stNum$, the GOOSE variable the counts the number of events that have occured, to force the IED to ignore some of the future messages it will receive. They also flip any Boolean data in the transmitted $DatSet$. The aim of their attack is to get an IED to perform an undesirable action, such as ignoring a command to trip a circuit breaker, by providing it with incorrect information. Strobel *et al.*[10] qualitatively expand upon Hoyos' attack vector by discussing that if an adversary replayed a GOOSE message within two minutes of a $stNum$ rollover, after 2^{32} events the $stNum$ counter resets, they can undermine the availability promise of an IED for up to approximately 4.5 years, presuming no one notices. Whilst Kush *et al.*[11] also model an attack against availability by using a near rollover $stNum$, they model two other attacks. In the first attack the adversary floods a GOOSE subscriber with messages that have incrementally higher $stNum$ until they exceed the current value, and the second attack sees the adversary injecting malicious messages, with a marginally higher $stNum$, at a slightly higher rate than the regular messages. Whilst not developing an explicit adversary model El Hariri *et al.*[12] do explore an important consideration for the security of the GOOSE protocols whose QoS requirements demand that devices from different manufactures be interoperable. They demonstrate that different commercial devices and simulation libraries respond differently when presented with the same undesirable situations. The various implementations had differing responses with messages with old timestamps, and out of order $sTNum$, but with timestamps outside the 2 minute skew of the IEC62351 replay prevention algorithm.

The main use of queing theory formalism in the security domain has been to describe denial of service (DoS) attacks. It is a suitable formalism for this type of attack as DoS scenarios can be modeled without much abstraction. This is due to how queuing theory calculates the efficency of a series of objects being processed in a queue according to a specific set of rules. These rules can easily be mapped to represent

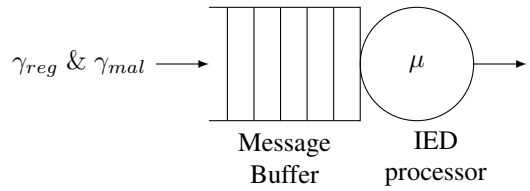

Fig. 1. A queue representing an IED with a message buffer (K) of 6, with 5 messages in the buffer. It has both regular and malicious traffic coming into it, but they are both processed at the same rate.

a processor with a specific memory allocation. Despite this natural affinity, seemingly little research has been persued in the modelling DoS attacks using queuing theory. Most of the research describes various packet level scenarios with either a single $M/M/1$ queue or an open Jackson network, which is a network of $M/M/1$ queues. Relying on $M/M/1$ limits what the user can discern from their models, as a queue of this type can hold an infinite number of objects. Given this underlying assumption, all that can be discerned from these models is the degradation of the queues performance. It doesn't tell the user at what point the system being modelled will fail to meet its promise of avaliability. However, Xiao-Yu *et al.*[13] does use a $M/M/c/K$ queue to investigate SIP INVITE request flooding scenario. Their solution is to create a queue that deals only with INVITE requests. Kammas *et al.*[14] created an open Jackson network of $M/M/1$ queues to model virus propagation across a network. Their state space included the internal transitions of state of each node, as well as the global state of the network. Wang *et al.*[15] developed a mathematical framework, using embedded two dimensional Markov chains, to allow the user to use different probability distribution functions for acceptance rates. Their model also allows for the separate analysis of the malicious message properties from the normal traffic's.

III. Queuing Theory Framework

For the models used to demonstrate the attacks a single $M/M/1/K$ queue is used[16], as shown in $Fig.1$, which can receive two different types of messages. The $M/M/1/K$ queue is used for this model because having a limit on the maximum number of messages in the queue puts an upper ceiling on the adversaries injection rate. If their injection rate is too high then they will end up denying any legitimate messages into the IED, which undermines their goals. In this model the two messages are the regular traffic and the malicious traffic injected into the channel. For the purposes of this analysis the single queue maps to a single communications channel of a subscriber that is within a GOOSE multicast group. The model assumes:-

1) Each queue obeys the first-in-first-out (FIFO) discipline for processing messages.
2) The processing time, μ, of the different types of messages are each assumed to follow an exponential distribution. However, the effective probability distribution describing the rate at which messages pass through a queue isn't an exponential distribution.
3) That the transition between states is memoryless.

The state space of the models represents the position of the different messages within the queue,

$$\mathcal{I} = \{(1, ..., K) \in \{n, m, \emptyset\}\} \tag{1}$$

where K is maximum number of messages in the queue. Each position in the queue can either regular, malicious, or empty, but there cannot be an empty space between two messages. The two types of transitions that can occur within this model are:

- A message entering the queue at a rate of γ_{reg} or γ_{mal}.
- A message being processed by the queue with a rate of μ_{reg} or μ_{mal}.

With the above rules the endogenous variables for the queue can be calculated with:

Variable	Equation
Probability of being full	$P(N = K) = \frac{(1-\rho)\rho^K}{1-\rho^{K+1}}$
Arrival rate	$\lambda = \gamma_{reg} + \gamma_{mal}$
Processing rate	$\mu = \mu_{reg} + \mu_{mal}$

where $\rho_i = \frac{\lambda}{\mu}$.
The arrival and processing rates are governed by assumption 2. Once a state transition matrix has been generated and solved, presuming the system is in a steady state, for the queue, a probabilistic view of the state space can be calculated. The view for this state space is the probability of that a particular type of message is in specific position in the queue

$$\pi(i = n) = \sum_{i=k \in \mathcal{I}} \pi(1, ..., K) \tag{2}$$

To further the understanding of the injection attack they're two additional views which are calculated in the same way. These views are the probability that there is only one a malicious message in the queue, and it is at the front of the queue, and there is a malicious message at the front of the queue with the number of malicious messages in the queue being below a certain threshold.
The standard queuing theory performance metrics are used in this analysis are:

Performance Metric	Equation
Mean Number of messages	$\bar{k} = \sum_{k=1}^{k} k\pi(k)$
Mean Response Time	$\bar{T} = \frac{\bar{k}}{\bar{\lambda}}$

IV. Injection Attacks

As stated in section II the research into attacks using the GOOSE message service focuses on seeing what capabilities the adversary can have due to the rules of the GOOSE model, but they don't discuss how the adversary achieved their foothold (since the security of the service is trivial to overcome). They also do not discuss how likely these attacks would be detected. If the adversary's goal is undermine availability completely, then it will be detected eventually. However, for an adversary that only wants to disrupt a few

messages it is likely that they wish for their actions to remain unnoticed by any intrusion detection system (IDS). None of the above works have commented on this particular use case, or how likely it is.
The following section will provide an analysis of the minimum adversarial capabilities required to inject a single malicious message. This injected message may be us to disrupt a few superseding messages or make the SAS perform an undesirable action, without undermining the promise of availability within the security network.

A. Message Injection Attack

For Strobel's DoS attack to succeed the adversary must time the injection of their malicious message correctly or they will have to wait around for another 4 years for the $stNum$ to rollover. Whilst Strobel gives the upper timing bound of the injection within two minutes of the rollover to pass the replay detection algorithm, they provide no consideration to the rate of injection needed for the attack to deny any messages after the rollover has happened. The analysis picks up from here and discusses likelihood of this kind of attack.

Using this starting point an adversary model can be developed for when they wish to complete their objective using only one message. For this attack to work the adversary will be required to have performed some passive surveillance of the target subscriber IED. This is so they can discover what the current $stNum$ is, and the rate at which they change. Whilst it is not necessary for the adversary to obey Kerckhoffs' principles[17], they must at least know that the GOOSE $stNum$ rollover occurs when it reaches 2^{32} and that they have to complete their injection within two minutes. Finally, they must be able to manipulate the rate of transmission of individual messages. First to delay their arrival, before accelerating it so it can be injected at the desired time.

If it is assumed that the subscriber's channel has no buffer of messages it receives, then the chances of adversary's malicious message being the first message they logarithmically with the increase in the rate of injection. This is shown in $Fig.2$, which shows that the adversary needs to be of order hundred times faster than the normal injection rate to guarantee their success.

B. Countermeasures: Message Buffer and Rate Limiting

By introducing even a small buffer of messages then the probability of the malicious message being first drops to below $\frac{1}{2}$. Adjusting the adversary model to allow for message duplicates to be injected, does not increase their probability of success greatly if they wish to maintain the objective of not DoSing the buffer with malicious messages. Another method of limiting the success of the adversary is by limiting the range of injection rates that will be accepted by the subscriber. These two methodologies are demonstrated in $Fig.3$, where

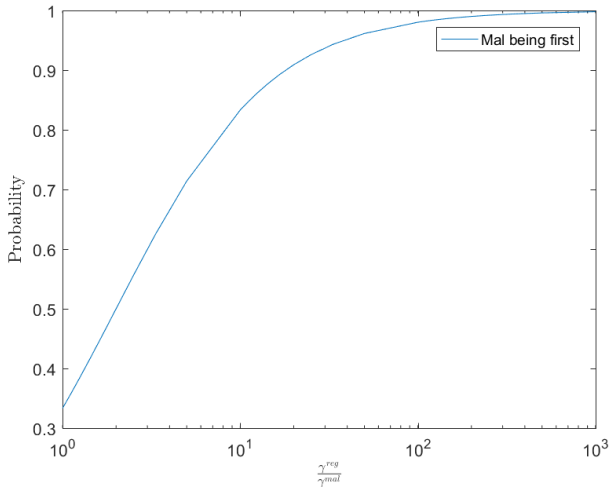

Fig. 2. The probability of success of an injection, given the ratio of regular traffic against the malicious traffic $\left(\frac{\gamma^{reg}}{\gamma^{mal}}\right)$.

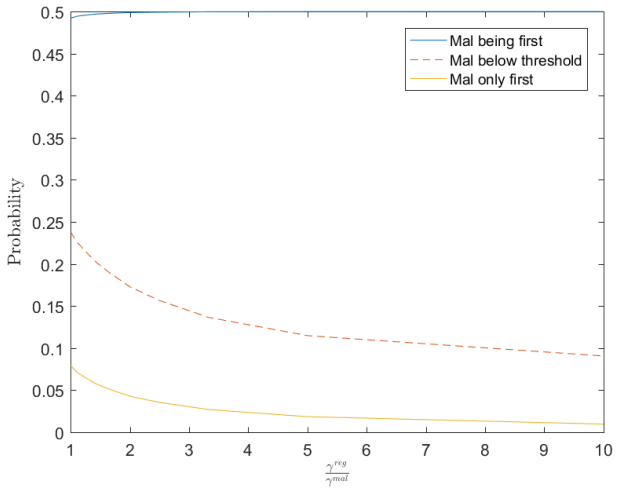

Fig. 3. The probability of success of an injection, given the ratio of regular traffic against the malicious traffic $\left(\frac{\gamma^{reg}}{\gamma^{mal}}\right)$. There is a message buffer and a injection rate limit of only being ten times faster than regular traffic.

the buffer is only five messages long, and the threshold of detection is two or more.

C. Countermeasure: Evasion of buffers & rate limitation and Implementing Inflexible QoS

In the above analysis it is presumed that the malicious message is the same size as the regular traffic. However, if an adversary gains the ability to edit and increase the number of data sets in the GOOSE message it can be shown that they may be able to circumvent the above the countermeasures. In this attack the adversary model is expanded so that they can intercept and edit a GOOSE message. The knowledge that they require is knowledge of the upper limit of size of the $DatSet$ that a GOOSE message can carry without tripping the

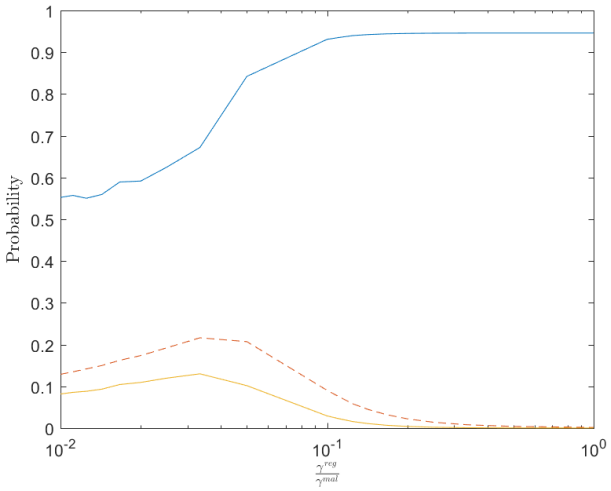

Fig. 4. The probability of success of an injection, given the ratio of regular traffic against the malicious traffic $\left(\frac{\gamma^{reg}}{\gamma^{mal}}\right)$, given a message buffer, when the malicious message is two orders of magnitude larger than the regular message.

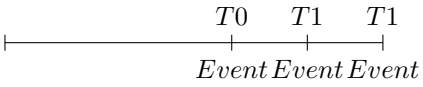

Fig. 5. The layout of the attack against a tap changer, when QoS is strictly enforced. Where $T0$ & $T1$ are different QoS time requirements defined in IEC61850-5[18]. An 'event' is a new event that causes the GOOSE publisher to start a new $stNum$.

integrity check. As shown in $Fig.4$, contrary to the previous section, the optimal rate of injection to evade the discussed countermeasures with the enlarged message is slower than the regular rate of injection. This means for the evasion to be successful the adversary would also be required to do some passive surveillance to observe the rate of arrival of regular messages so they could time their injection correctly. Whilst this attack vector doesn't increase the adversary's odds of remaining undetected, it does increase the chance of their injected message being first. A counter measure to the above attack vector would to have more stringent enforcement on the QoS transmission standard. Currently IEC61850 standard still allows for messages that arrive late to be processed. If the standard tightened this QoS promise, there would be no guarantee that publishers message could get through. An adversary would manipulate this counter measure, using the previously stated adversary model, to invalidate a single GOOSE event in a sequence of GOOSE events. That is to say that a $stNum$ is changed without any time for retransmission between the change. An example of this would be in the issuance of tap changer commands[19]. The commands to adjust come in a succession of GOOSE messages, as shown in $Fig.5$,and if the tap does not end in the correct position it is likely to damage the physical equipment. $Fig.6$ shows that a less than one order of magnitude increase in the size of the GOOSE message can create a greater delay in the expected

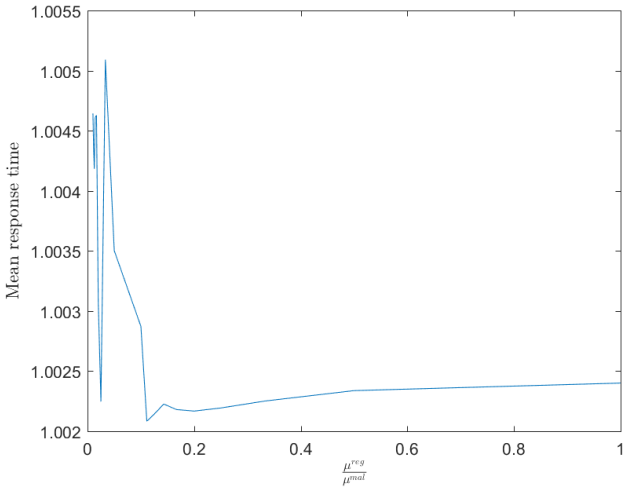

Fig. 6. A demonstration of the increased processing time when enlarging a malicious message in relation to regular traffic. The x-axis is the ratio of malicious and regular processing times $\left(\frac{\mu^{reg}}{\mu^{mal}}\right)$.

processing time of messages. All the variables in $Fig.6$ are set to the same, and only μ^{mal} is increased in relation to μ^{norm}. This delay would make subsequent messages miss their QoS permitted window.

This look at the stricter enforcement of QoS presumes that there is no change in the order of magnitude between the translation of a change in the size of the message and the time it takes to process a message. The change in size is demonstrated in the queueing theory model as an increase in time of processing of the malicious message, which has now been uncoupled from the processing rate of regular traffic.

A counter measure that would help prevent the adversary's subversion of the rate limiter, would be to bind the $ConfRev$ variable with the $DatSet$. If the number of data sets changes but doesn't have a corresponding event then the IED disposes of it in the integrity check. This would undermine the adversary being able to increase the size of the GOOSE message.

V. Conclusion

The above analysis describes the minimum capabilities an an adversary needs to inject a single message to cause the GOOSE messaging service to perform undesirable actions using a $M/M/1/K$ queue. The analysis goes on to look at the effectiveness of different countermeasures to attack vector. These were creating a message buffer for the GOOSE communication channel, rate limiting the arrival of messages and binding the GOOSE message $ConfRev$ variable with the message size, so it can be used for integrity checking.

For these countermeasures to be effective they should be implemented at the standard level, because if they are left to for implementers level there is no guarantee that the parameters of the countermeasures will be the same. A

difference in countermeasures parameters may provide an adversary with new attack vectors to exploit the GOOSE messaging service.

Despite the assumptions built into the model, such as the lack of transience in the message arrival rate distribution, it does provide a benchmark for those developing IDS systems for the GOOSE service. It can be used to show the probability of injection success given a threshold of the number of malicious messages in the buffer required for detection. Alongside this, the model can be used to calculate the injection probability against other protocols. It would require the user to input the QoS requirements from other standard.

The future direction of this work is to further develop the model presented, so new detection metrics can be created for IDSs. Part of it will incorporate joint probability distributions calculation, so the user can find a benchmark of the successful injection given other events occurring in the service. The authors would also like to be able to model multiple communications channels in an instance, by expanding the model to use $M/M/c/K$ queues. The authors are also expanding the queuing theory framework to model other GOOSE message attacks. The next investigation is to see if the GOOSE messaging service can be made to lose its state.

ACKNOWLEDGMENT

This work is supported by an EPSRC Academic Centres of Excellence in Cyber Security Research PhD grant.

REFERENCES

[1] R. M. Lee, "CRASHOVERRIDE: Analysis of the Threat to Electric Grid Operations," Dragos inc, Tech. Rep., 12-06-2017.

[2] C. Kriger, S. Behardien, and J. Retonda, "A detailed analysis of the goose message structure in an iec 61850 standard-based substation automation system," vol. 8, pp. 708–721, 10 2013.

[3] TC 57 - Power Systems Management and Associated Information Exchange, "Power Systems Management and Associated Information Exchange, Data and Communication Security - Part 6: Security for IEC 61850. IEC standard 62351-6," International Electrotechnical Commission, Tech. Rep., 2007.

[4] A. K. Pathan, *Securing Cyber-Physical Systems*. Boca Raton, FL, USA: CRC Press, Inc., 2015.

[5] J. G. Wright and S. D. Wolthusen, "Access control and availability vulnerabilities in the iso/iec 61850 substation automation protocol," in *Critical Information Infrastructures Security*, G. Havarneanu, R. Setola, H. Nassopoulos, and S. Wolthusen, Eds. Cham: Springer International Publishing, 2017, pp. 239–251.

[6] A. for review.

[7] P. Judge and M. Ammar, "Security issues and solutions in multicast content distribution: a survey," *IEEE Network*, vol. 17, no. 1, pp. 30–36, Jan 2003.

[8] R. Canetti, J. Garay, G. Itkis, D. Micciancio, M. Naor, and B. Pinkas, "Multicast security: a taxonomy and some efficient constructions," in *INFOCOM '99. Eighteenth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE*, vol. 2, Mar 1999, pp. 708–716 vol.2.

[9] J. Hoyos, M. Dehus, and T. X. Brown, "Exploiting the GOOSE Protocol: A Practical Attack on Cyber-Infrastructure," in *2012 IEEE Globecom Workshops*, Dec 2012, pp. 1508–1513.

[10] M. Strobel, N. Wiedermann, and C. Eckert, "Novel weaknesses in iec 62351 protected smart grid control systems," in *2016 IEEE International Conference on Smart Grid Communications (SmartGridComm)*, Nov 2016, pp. 266–270.

[11] N. Kush, E. Ahmed, M. Branagan, and E. Foo, "Poisoned GOOSE: Exploiting the GOOSE Protocol," in *Proceedings of the Twelfth Australasian Information Security Conference - Volume 149*, ser. AISC '14. Darlinghurst, Australia, Australia: Australian Computer Society, Inc., 2014, pp. 17–22. [Online]. Available: http://dl.acm.org/citation.cfm?id=2667510.2667513

[12] M. E. Hariri, T. A. Youssef, and O. A. Mohammed, "On the implementation of the iec 61850 standard: Will different manufacturer devices behave similarly under identical conditions?" *Electronics*, vol. 5, no. 4, pp. 201–213, 2016, http://www.mdpi.com/2079-9292/5/4/85.

[13] X. Y. Wan, Z. Li, and Z. F. Fan, "A SIP DoS flooding attack defense mechanism based on priority class queue," in *2010 IEEE International Conference on Wireless Communications, Networking and Information Security*, June 2010, pp. 428–431.

[14] P. Kammas, T. Komninos, and Y. C. Stamatiou, "A Queuing Theory Based Model for Studying Intrusion Evolution and Elimination in Computer Networks," in *The Fourth International Conference on Information Assurance and Security*, Sept 2008, pp. 167–171.

[15] Y. Wang, C. Lin, Q. Li, and Y. Fang, "A Queueing Analysis for the Denial of Service (DoS) Attacks in Computer Networks," *Comput. Netw.*, vol. 51, no. 12, pp. 3564–3573, Aug. 2007. [Online]. Available: http://dx.doi.org/10.1016/j.comnet.2007.02.011

[16] D. Gross, J. F. Shortle, J. M. Thompson, and C. M. Harris, *Fundamentals of Queueing Theory*, 4th ed. New York, NY, USA: Wiley-Interscience, 2008.

[17] A. Kerckhoffs, "La cryptographic militaire," *Journal des sciences militaires*, pp. 5–38, 1883.

[18] T. . P. S. Management and A. I. Exchange, "Communication Networks and Systems for Power Utility Automation - Part 5: Communication Requirements for Functions and Device Models. IEC standard 61850-5," International Electrotechnical Commission, Tech. Rep., 2013.

[19] N. Sichwart, A. Eltom, and G. Kobet, "Transformer load tap changer control using iec 61850 goose messaging," in *2013 IEEE Power Energy Society General Meeting*, July 2013, pp. 1–5.