# Masquerading IEC 61850 GOOSE Protocol: Cyber-Physical Experiments and Detection

Hermenegildo da Conceição Alberto*
Karlsruhe Institute of Technology (KIT)
Karlsruhe, Germany
University Gaston Berger of Saint Louis (UGB)
Saint Louis, Senegal
hermenegildo.alberto@kit.edu

Gustavo Sánchez*
Karlsruhe Institute of Technology (KIT)
Karlsruhe, Germany
sanchez@kit.edu

Jean Marie Dembele
University Gaston Berger of Saint Louis (UGB)
Saint Louis, Senegal
jeanmarie.dembele@ugb.edu.sn

Idy Diop
Ecole Supérieure Polytechnique (ESP/UCAD)
Dakar, Senegal
idy.diop@esp.sn

Ghada Elbez
Karlsruhe Institute of Technology (KIT)
Karlsruhe, Germany
ghada.elbez@kit.edu

Veit Hagenmeyer
Karlsruhe Institute of Technology (KIT)
Karlsruhe, Germany
veit.hagenmeyer@kit.edu

## Abstract

In a Hardware-in-the-Loop digital substation, we show that a network attacker can spoof a GOOSE trip to open a breaker. We propose two detectors—one machine-learning, one rule-based—and illustrate both attack and defense in an interactive demo for awareness.

## CCS Concepts

• **Security and privacy**; • **Hardware → Power and energy**;

## Keywords

Security, Smart Grid, testbed, IEC 61850, digital substation.

**ACM Reference Format:**
Hermenegildo da Conceição Alberto, Gustavo Sánchez, Jean Marie Dembele, Idy Diop, Ghada Elbez, and Veit Hagenmeyer. 2025. Masquerading IEC 61850 GOOSE Protocol: Cyber-Physical Experiments and Detection. In *The 16th ACM International Conference on Future and Sustainable Energy Systems (E-ENERGY '25), June 17–20, 2025, Rotterdam, Netherlands.* ACM, New York, NY, USA, 2 pages. https://doi.org/10.1145/3679240.3734685

## 1 Introduction

IEC 61850 GOOSE delivers fast control/status messages but lacks cryptographic safeguards, enabling spoofing attacks [1]. We experimentally forge a masquerade trip (ATT&CK T1036) that opens a breaker. Unlike prior studies [2, 3] we demonstrate the attack on real hardware and share all artifacts. Finally, we evaluate both ML and rule-based IDS.

*Both authors contributed equally to this research.
[1]https://gitlab.kit.edu/ah5021/eenergy-masquerading-goose


*E-ENERGY '25, Rotterdam, Netherlands*

ACM ISBN 979-8-4007-1125-1/25/06
https://doi.org/10.1145/3679240.3734685

## 2 Background

A GOOSE message carries two counters—state (*st*) that defines a new event and sequence (*q*) for retransmission; besides *st* and *q* fields, it includes gocbRef (dataset reference), timeAllowedToLive, datSet, goID, confRev and all-data containing the command in boolean; see our parser in the repo for full details. A new event(legit and non-legit trip/non-trip) is added by incrementing the *st*, and resetting the *q* to zero.

## 3 Methodology

The experiment used a controlled laboratory setup replicating a substation environment. A GOOSE dissector was developed to decode network traffic and study communication.

**Testbed.** Our testbed replicates a digital substation with merging units and bay controllers on the process bus, and relay RTUs with SCADA on the station bus; detailed network diagrams are available in our GitLab repo.

**Attack.** We craft packets with the purpose of injecting a malicious GOOSE message into the process bus. Because GOOSE frames lack any authentication, an attacker with access to the process bus can forge Ethernet frames carrying GOOSE payloads that exactly match the legitimate packet structure, causing unsuspecting control bay to accept injected trip commands. The forged message mimics a legitimate trip command by replicating key parameters from Section 2. The Boolean command flag in the GOOSE payload is set to True, instructing the circuit breaker to open. The crafted packets are injected into the process bus, encapsulated in an Ethernet frame using a typical GOOSE multicast address. The packet is transmitted to emulate normal traffic, causing the breaker controller to accept the open command. Figure 1 shows circuit breaker state changes over time. Of 3 total trips, 2 were legitimate while 1 attack-induced event occurred during state 149, demonstrating successful injection.

**Detection.** We developed a framework emulating process-bus behavior using Random Forest (RF) and rule-based techniques. We

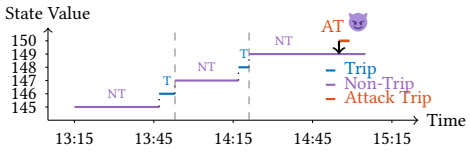


**Figure 1: GOOSE trip events sample in our digital substation.**

generated packet streams mimicking normal substation communication with fixed ratios of legitimate GOOSE behavior, validated on a real testbed. In normal conditions, breaker state changes are indicated by GOOSE messages where state numbers increment and sequence numbers reset to zero [5]; during retransmissions, sequence numbers increment until the next state change. **GOOSE Protocol Model.** Let $s_t$ (state) and $q_t$ (sequence) denote the $t$-th packet's counters. Under normal (non-trip) operation, $s_t = s_{t-1},\ q_t = q_{t-1}+1$, and at a legitimate trip, $s_t = s_{t-1}+1,\ q_t = 0$. **Raw-Feature RF Classifier.** We train an RF directly on absolute `stnum` ($s_t$) and `sqnum` ($q_t$), labeling packets 0 (legitimate) or 1 (injected). Training uses synthetic GOOSE streams of 10 000 packets (to vary attack- 0.1/normal ratio -0.001) plus hardware-in-the-loop data (70 % train / 30 % test). The raw features model processes packets independently, training on absolute values of `stnum` and `sqnum` with labels (0 for legitimate, 1 for injected). The RF learns normal packet feature distributions, but reliance on absolute values potentially leaves detection vulnerable to subtle adversarial manipulations. **Differential-Feature RF Classifier.** To enhance robustness let's $(s_b, q_b)$ be the last legitimate counters; define $\Delta s_t = s_t - s_b,\ \Delta q_t = q_t - q_b$. One expects $\Delta s_t = 0,\ \Delta q_t = 1$ (normal) and $\Delta s_t = 1,\ \Delta q_t = -q_b$ (trip). Training on $(\Delta s, \Delta q)$ also yields > 99% accuracy with improved robustness. **Evaluation.** We used `train_test_split` (with `random_state=42`, `test_size=0.3`, and `stratify=y_raw`) to split the data into 70% training and 30% testing sets while preserving the original class distribution. We present confusion matrices that give indications on how many false positives exist in the Git repository; the results vary from experiment to experiment, but we are consistently observing results in the order of 3 false positives within 875 injections ( about 0.3% false positives) and achieving an accuracy > 99% accuracy. **Rule-Based Detector.** Complementing the ML approach, we developed a deterministic rule-based detection mechanism that capitalizes on the expected sequential behavior. Let $A(t)$ be an indicator function that signals an anomaly at packet $t$. The rules are expressed as:

$$A(t) = \begin{cases} 1, & \text{if } s_t \neq s_{t-1} \text{ and } s_t \neq s_{t-1}+1, \\ 1, & \text{if } s_t = s_{t-1} \text{ and } q_t \neq q_{t-1}+1, \\ 1, & \text{if } s_t = s_{t-1}+1 \text{ and } q_t \neq 0, \\ 0, & \text{otherwise.} \end{cases}$$

**Adversarial Evaluation.** An adversarial perturbation search varied $(s_t, q_t)$ around expected values [4]. The raw-feature RF occasionally misclassifies crafted candidates, whereas the differential RF and rule-based detectors consistently identify all anomalies.

**Results.** The masquerade attack successfully opened the circuit breaker, potentially causing blackout. Figure 2 shows normal operation (breaker Q0 and disconnector Q1 energized) versus post-attack state (Q0 open, line isolated). Continuous packet injection prevented breaker operation and lockout reset, causing DoS. Other attacks (volumetric DoS, replay, random data injection) had no impact, highlighting masquerade attack severity. ML classifiers with raw/differential features combined with rule-based detection provide robust defense, aiding incident analysis and response.

**Live Demonstration.** We developed a web interface for executing live masquerade attacks and monitoring SCADA impact. Users function as substation operators to restore operations, and train/evaluate ML models with rule-based IDS, serving as a training and awareness resource.

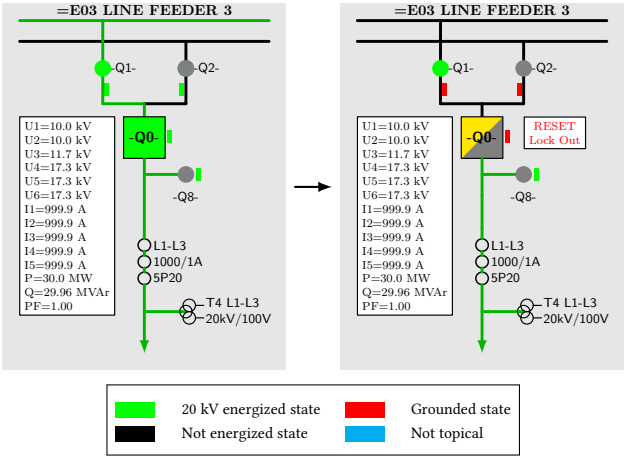


**Figure 2: *Normal* (left) and *attacked* (right) system diagrams.**

## 4 Conclusion

This study demonstrates a security gap in the GOOSE protocol, simulated in lab conditions that recreate real situations. This injection vulnerability poses a risk to substation operations, as the unauthorized opening of a circuit breaker can lead to widespread outages or equipment damage. To counter this, our detection strategy is capable of reliably identifying malicious injections in GOOSE communications, thereby contributing to the improved security of digital substations in smart grid systems.

## Acknowledgments

Funded partially by Engineering Secure Systems of the Helmholtz Association (HGF) and KASTEL Security Research Labs (structure 46.23.02), and by the Regional Scholarship and Innovation Fund (RSIF).

## References

[1] Hermenegildo da Conceição Aberto, Jean Marie Dembele, Idy Diop, and Alassane Bah. 2024. Review of Intrusion Detection Systems for Supervisor Control and Data Acquisition: A Machine Learning Approach. Springer Nature Switzerland.
[2] Juan Hoyos, Mark Dehus, and Timthy X Brown. 2012. Exploiting the GOOSE protocol: A practical attack on cyber-infrastructure. In *2012 IEEE Globecom W.*
[3] Nishchal Singh Kush, Ejaz Ahmed, Mark Branagan, and Ernest Foo. 2014. Poisoned GOOSE: Exploiting the GOOSE protocol. In *AISC'14.*
[4] Gustavo Sánchez, Ghada Elbez, and Veit Hagenmeyer. 2024. Attacking Learning-based Models in Smart Grids: Current Challenges and New Frontiers. In *e-Energy.*
[5] Shiming Wang, Fuyou Yang, Xu Yan, and Tianze Liu. 2020. Analysis of GOOSE message and the engineering application for GOOSE message in the intelligent substation. *The Journal of Engineering* 2020 (2020), 207–212.