Attack Rules: An Adversarial Approach to Generate Attacks for Industrial Control Systems using Machine Learning

Muhammad Azmi Umer
DHA Suffa University
Karachi Institute of Economics and Technology
azmi.umer@dsu.edu.pk

Chuadhry Mujeeb Ahmed
University of Strathclyde
mujeeb.ahmed@strath.ac.uk

Muhammad Taha Jilani
Karachi Institute of Economics and Technology
m.taha@pafkiet.edu.pk

Aditya P. Mathur
Singapore University of Technology and Design
aditya_mathur@sutd.edu.sg

ABSTRACT

Adversarial learning is used to test the robustness of machine learning algorithms under attack and create attacks that deceive the anomaly detection methods in Industrial Control System (ICS). Given that security assessment of an ICS demands that an exhaustive set of possible attack patterns is studied, in this work, we propose an association rule mining-based attack generation technique. The technique has been implemented using data from a Secure Water Treatment plant. The proposed technique was able to generate more than 300,000 attack patterns constituting a vast majority of new attack vectors which were not seen before. Automatically generated attacks improve our understanding of the potential attacks and enable the design of robust attack detection techniques.

KEYWORDS

Attack Detection, Attack Generation, ICS Security, Machine Learning, Adversarial Learning, Association Rule Mining.

1 INTRODUCTION

Most of the modern critical infrastructures (CI) are controlled by Industrial Control Systems (ICS). Examples of such systems are the electric power grid and water treatment plants, where an ICS controls the physical process in a CI. Computing and communication elements such as Programmable Logic Controllers (PLCs) and Supervisory Control and Data Acquisition systems (SCADA), and communication networks are the important technologies used to realize the control. PLCs control the physical process via actuators based on the sensor measurements. The SCADA workstations are used to exert high-level control over the PLCs, and the process, and provide a view into the current process state [3]. The automation makes it easy to monitor and control the critical infrastructures but it opens up these critical systems to malicious entities as evident from several widely reported successful attempts such as those reported in,[8, 16, 22]. It is observed that while air-gaping a system might be a means to consider securing a CI, it does not guarantee to keep attackers from gaining access to the ICS and launch successful attacks.

Due to the increase in attacks on ICS, a lot of attention is given to the development of security mechanisms to prevent, contain, and react to cyber-attacks [6]. This led to research in two major directions, 1) rigorous security testing of ICS and 2) design of robust intrusion detection techniques. The outcome of those efforts either relied on the design of these complicated systems or the data from the CI. On the security analysis front, one approach was to come up with realistic threat models, accurately modeling the capabilities and intentions of an attacker specifically applied to an ICS [1, 19]. These earlier works provide interesting frameworks to model attacks in an ICS based on an attacker's capabilities and intentions but require design knowledge and domain expertise in the specific process. In this work, we propose a data-based automatic attack generation technique to drive an ICS to an unsafe state, this type of resource would prove to be a key technique to cover as many bases as possible.

Other efforts towards generating the anomalous data rely on the normal process data to come up with examples not present in the normal data [18]. However, data generated this way might be anomalous but the particular anomaly doesn't need to have any strategy to cause any damage and labeled as the attack data. Studies looked into generating more samples of attack data than really present, e.g., SMOTE [7], but those do not add any new attack patterns rather interpolate the existing ones. Therefore, we take up this study to design an attack generator that can learn from the attack goals of the attacks designed by domain experts. Association Rule Mining (ARM) was used to find attack scenarios autonomously using the attack samples. Earlier manually generated attacks [1] were crafted by a human expert to take the system to an unsafe state. Given those sets of actions and knowing that these will take a system into an unsafe state, we can learn more about such patterns that were not explored before. This leads to, 1) discovering attacks that were not seen before, 2) analyze the impact of those attacks. We plan to exhaust the possibilities to bring a physical process in an unsafe state and figure out sensor/actuator combinations to be manipulated.

In this work, a water treatment testbed is used as a case study and our proposed technique is applied to generate attack patterns. As expected we obtained 329069 attack patterns that were impossible to find manually. SWaT testbed engineers spent a lot of effort and ran the plant continuously for 4 days to come up with 36 attack patterns across the plant [11]. This is considered a remarkable effort, given the limited research facilities and lack of data availability especially

, ,

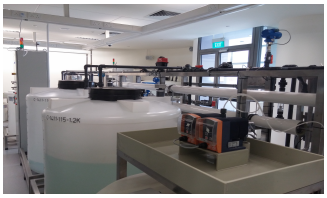

Figure 1: Case Study: SWaT Testbed

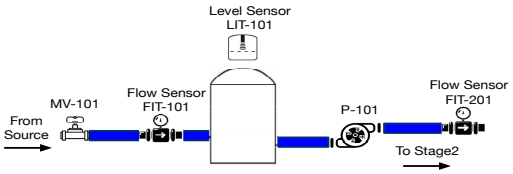

Figure 2: Stage 1 of the SWaT Testbed.

the attack data. However, for data-hungry machine learning algorithms, this is still a limited amount of data and lacks completeness if someone wants to design a supervised learning algorithm for attack detection. The generated attack patterns are verified to be attacks using the well-known *process invariants* technique [21] that mined the process invariant from the normal operational data of the plant.

Contributions: Contributions of this work are twofold. 1) Security analysis of the physical process by exploring the attack patterns. This is to explore the combination of possible sensor's and actuator's states to lead to unsafe operation. 2) Providing the data for improving the design of intrusion detection systems. Data is the first requirement for machine learning-based intrusion detection techniques. Normal data could be easily obtained from the physical operation of the process but attack data is rarely available. In cases where the attack data is collected, the attacks generated by human experts are not exhaustive. However, we have tried to provide all the combinations that can lead to successful attacks.

2 BACKGROUND

2.1 SWAT: A live water treatment plant

The Secure Water Treatment (SWaT) is an industrial replica of a water treatment plant which is available at Singapore University of Technology and Design (SUTD) [17]. It has been used by various researchers to validate their defense mechanisms for ICS [12]. A pictorial view of the SWaT plant is shown in Figure 1. It has the capacity to produce 5 gallons/minute of treated water. Water is treated first using ultrafiltration and then followed by reverse osmosis. SWaT is a distributed control system consisting of six stages. Each stage is installed with a set of sensors and actuators. Sensors are used to measure the quantity of water like the level in a tank, flow, and pressure. They are also used to measure the quality of water like pH, oxidation-reduction potential, and conductivity. Motorized valves and electric pumps serve as actuators.

Stage 1 treats the raw water. Chemical dosing is performed in stage 2 that depends on the measurements from the water quality sensors. Ultrafiltration occurs in stage 3. Dechlorination is performed in stage 4 before it is passed to the reverse osmosis units in stage 5. The treated water is distributed in stage 6 and it also performs the backwash by cleaning the ultrafiltration unit. Level 0 network is used for data communication between sensors/ actuators and PLCs. Inter PLCs communication is done over a level 1 network.

2.2 Motivating Example

Figure 2 shows stage 1 of the SWaT testbed. It consists of a raw water storage tank, a level sensor (LIT-101) on top of the tank, an inlet motorized valve (MV-101), a flow meter (FIT-101) on the inlet pipe, a pump (P-101) on the outlet, a standby pump (P-102 not shown in the figure), a flow sensor (FIT-201) and a motorized valve (MV-201) on the outlet of the tank and on the stage 2 of the SWaT. Now, having seen the critical components let's see how does this process is controlled by a PLC. The purpose of the tank is to store the raw water to be supplied to subsequent stages for the purification process. Listing 1 shows a snippet of a part of the control logic in the PLC.

```
# LIT-101 is level sensor in tank1
# MV-101 is the inlet valve controlled by a PLC depending
      on LIT101 measurements.
if LIT-101_value == Low_Level:
    MV-101 = Open
elif LIT-101_value == High_Level:
    MV-101 = Close

```

Listing 1: Control Logic Example in Stage1 of SWaT

There are designed operational limits for the tank to hold the water, i.e., the Low_Level is the lowest limit beyond which stage 1 would not be able to meet the water requirement of the subsequent stages and High_Level is the upper limit beyond which the tank would overflow. According to the control logic shown in Listing 1 when the tank is at a high level, i.e., full then the inlet supply via MV-101 should be cut off to avoid overflow by closing the MV-101. If MV-101 does not close ultimately tank would overflow leading to a flooding situation. Now imagine that an attacker's goal is to overflow the tank, this goal could be achieved in many ways. One way is to force the MV-101 to stay open although the water level is high. This attack scenario compromises only one component MV-101 and succeeds but this attack could be easily detected by an operator monitoring the SCADA system by observing that the level sensor measurements have gone way higher than those should be. Another possible attack could compromise both the level sensor LIT-101 and the MV-101, by keeping the MV-101 open and spoofing the sensor reading within low and high levels. This way simply monitoring level sensor would not raise any alarms and the SCADA system would also fail to report this attack. We could see that by changing more than one component our attack becomes more sophisticated. Stage 1 is quite simple but still has got 7 variables that could be modified in different combinations to create a lot of attack scenarios. Given that the other stages in SWaT and real-world systems are much more complex with sometimes 100s of devices, then it is really hard to generate all possible attacks manually.

Therefore, in this work, we have proposed a data-based autonomous technique to generate possible attack patterns.

3 ATTACKS GENERATION USING ASSOCIATION RULE MINING

Lack of comprehensive attack scenarios and data is a major concern while working on anomaly detection problems. The key idea of the proposed technique is to use the available attack data [11] created by human experts and generate an exhaustive list of possible attack patterns. Attack data on the SWaT testbed is collected for 4 days while running the plant continuously. Plant engineers and researchers from SUTD came up with a rigorous design of attacks modeling the attacker's intentions, capabilities, and attack goals. An example of one such attack is the tank overflow attack as explained in the previous section. Therefore, an attack model is a tuple composed of sensor/actuator pair it shall attack with a consequence on the real physical process. This attack generation exercise led to the generation of 36 attack patterns in total. This attack dataset is not exhaustive in any sense but it is really useful for machine learning-based attack/defense techniques. Our proposed idea leverages that dataset and comes up with new attack patterns. The key observation was that each of those 36 attacks had a goal in mind, e.g., tank overflow and attack is executed by operators with design knowledge to incur the required damage. We model those intentions from the sample attack dataset and tried to come up with all the possible ways in which that goal of an attacker is fulfilled. To capture those rules set we have used the Association Rule Mining (ARM) [2] for attack generation. ARM is a rule-based unsupervised ML technique. It is very common in traditional market basket analysis. However, it also has important applications in intrusion detection [5]. In the current study, ARM was used for attack generation on a secure water treatment plant (SWaT) as described in algorithm 1. Attacks were generated using the attacked data of SWaT [12]. The following type of attacks was generated using the proposed approach:

$$X \implies Y \tag{1}$$

There are several algorithms including Apriori, Eclat, and FP-Groth which can be used for ARM. FP-growth was used in the given case study which is available in the Orange-Associate library in Python. ARM works in two phases. In the first phase, it generates frequent itemsets, while in the second phase it generates association rules using the frequent itemsets generated in the first phase. The complete flow of work is described in Figure 3.

3.1 Frequent Itemsets

An itemset is either a value of a single attribute or values of multiple attributes in a dataset. Itemset that qualifies the minimum support threshold is considered as a frequent itemset.

3.1.1 Support: Let there is an itemset 'I' in dataset 'D'. The support for an itemset 'I' in 'D' can be calculated by counting the number of transactions or rows 'r' of 'D' which contains the 'I'.

$$S(I) = \frac{|r \in D; I \in r|}{|D|} \tag{2}$$

3.2 Association Rules

Frequent itemsets are used to generate association rules which are described in equation 1. Only rules which qualify the minimum confidence threshold are considered as association rules.

3.2.1 Confidence: The rule described in equation 1 can be divided into two parts. One is 'X' which is placed at the left side of $\implies$ and called as antecedent, while one is 'Y' which is placed at the right side of $\implies$ and called as consequent. Confidence of any rule is calculated using the combined support of antecedent and consequent, and the antecedent alone.

$$C(X \implies Y) = \frac{S(X \cup Y)}{S(X)} \tag{3}$$

Algorithm 1: Attack Generation using Association Rule Mining

1: Data collection by capturing the network packets
2: Decoding of network packets to store state information of sensors and actuators.
3: Send the state information to the historian.
4: Feature/ Attribute Transformation followed by Feature Selection on collected data.
5: Frequent itemsets generation using the transformed data.
6: Generation of association rules from the frequent itemsets.
7: Validation of attacks (association rules) against the normal patterns.

3.3 Feature Transformation and Selection to Generate Attack Patterns

SWaTs' dataset consists of 51 attributes consist of different sensors and actuators. They are binary (e.g. pumps), ternary (e.g. motorized valves), and real-valued (e.g. level sensors) attributes. However, ARM works only on binary-valued attributes. Therefore, all other types of attributes need to be transformed into binary-valued attributes. While converting them into binary-valued attributes, some of the attributes become useless, for example after conversion into binary the whole vector of attribute gets converted into a single number i.e. either 0 or 1. This type of data is useless for ARM. Therefore, it needs to be removed from the dataset. There is a total of 15 attributes were selected for attack generation. Another reason for this reduced set of attributes is the baseline validation of attack patterns with normal patterns of SWaT generated in [21]. We selected the same attributes as done in [21].

3.3.1 Transformation of Attributes. There are some motorized valves in the dataset which have three possible states, i.e., Close (1), Open (2), and Transition (0). This makes them ternary valued attributes. The transition state lasts for less than 10 seconds. To convert these attributes into binary attributes, the transition state (0) was transformed into either Open (2) or Close (1) state. This was done by looking at the corresponding FIT value. If the corresponding FIT Value against the transition state (0) is less than 0.5 then the transition state was converted into Close (1) state else into Open (2) state as shown in Listing 2. The FITs which are the real-valued attribute

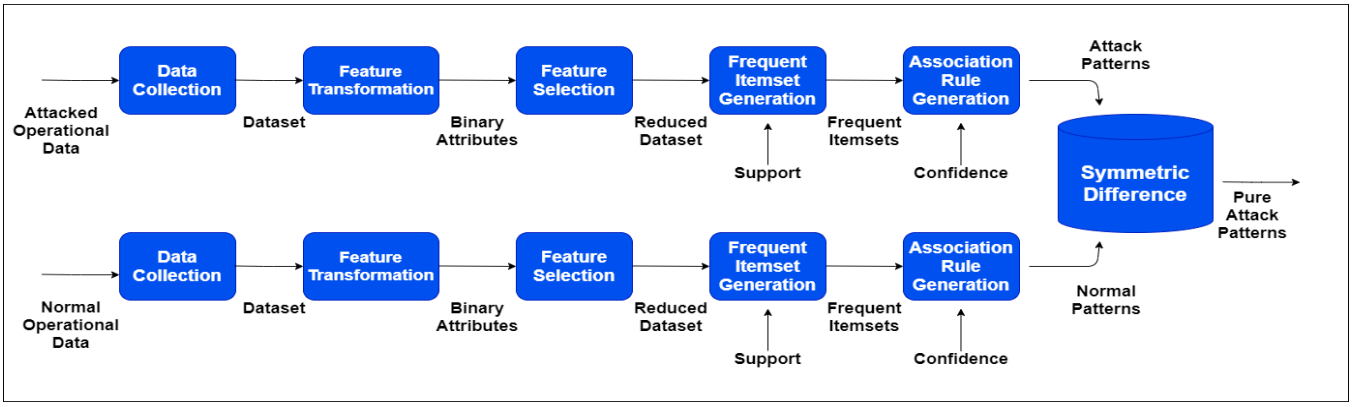

Figure 3: Attacks Generation using ARM

Attributes	Description
Flow meters	
FIT-101	Flow measurement at inlet of T-101.
FIT-201	Flow measurement between Process 1 and 2.
FIT-301, FIT-601	Flow measurement in Ultra filtration Process, and backwash respectively.
Motorized valves	
MV-101, MV-201	Motorized Valve at inlet of T-101, and T-301.
MV-301, MV-303	To control the Ultra filtration-backwash.
MV-302, MV-304	To control the flow towards the de-chlorination unit, and Ultra filtration backwash drain respectively
Pumps	
P-101	Pump to supply raw water in Process 2.
P-203, P-205	Dosing pumps for HCl , and NaOCl respectively.
P-302, P-602	To supply water from T-301 to T-401, and from T-602 to Ultra filtration unit.

Table 1: Attributes Selected for Attacks Generation.

was transformed into the binary-valued attribute. If its' value is equal to or greater than 0.5 then flow was assumed otherwise not.

```
# MV-101 is the inlet valve controlled by a PLC depending
    on LIT101 measurements.
# FIT-101 is use to measure the flow towards Tank T101.
if MV-101 == Open:
    MV-101 = Open
elif MV-101 == Close:
    MV-101 = Close
elif MV-101 == Transition:
    if FIT-101<0.5:
        MV-101 = Close
    else:
        MV-101 = Open

```

Listing 2: Transformation of Motorized Valve into Binary-Valued Attribute

3.3.2 Large Set of Rules. Defining an optimal value of support is very crucial to the number of rules. If it is too high then the number of rules would be too low, therefore, it is inappropriate to get the actual behavior or patterns in the data. While if support is too low then a very high number of rules gets generated. There was an attribute in SWaT i.e. P602 which remained open only in 3164 transactions of normal SWaT data, the rest of the time it remained close. To get the patterns relevant to P602 in Open state, we need to drop the support level up to 3164/410400, i.e. 0.77%, where 410400 is the total number of transactions in the dataset.

4 EVALUATION AND DISCUSSION

To validate the generated attack patterns, we replicated the experiments performed in [21] using the data-centric approach. The challenge was that the attack dataset used also had the normal data samples thus making it difficult to identify attacks among the generated patterns based on the defined rules. For this purpose we used the seven days normal operational data from SWaT [12] to generate the physical invariants under the normal operation of the plant. All attributes were transformed into binary attributes and feature selection was performed to derive the same attributes as done in [21]. Frequent itemsets were generated using the transformed dataset with 0.7% support. The frequent itemsets were used to mine the association rules with 100% confidence level. We compared these normal patterns of SWaT against the attacks generated in the current case study using the same support and confidence thresholds. For this purpose, all the attack patterns of SWaT were converted into a set, say A, where each entry is an attack pattern. Likewise, all normal patterns of SWaT were converted to a set B where each entry is a normal pattern. We thus obtained the symmetric difference of sets A and B and stored it in set C ($C = A \setminus B$). Now C contains all the attack patterns that can be differentiated from the normal patterns.

A list of attacks generated using ARM along with sample attack patterns is shown in Table 2. We were able to generate both local and global attacks. Here "local attack" refers to intra-process attack patterns and "global attack" refers to inter-process, i.e., across 6-stages, attack patterns. We generated attacks from the antecedent

size of 2 to the antecedent size of 14 (as a total of 15 sensor/actuator pairs considered in this work). Attack patterns are normally distributed with respect to antecedent size as shown in Figure 4. Figure 4 highlights a key result that is "using a 1 or 2 sensor/actuator pairs, an attacker can launch a small number of attacks but when we increase this number the attack patterns increase exponentially." Results show that by leveraging more sensors and actuators, an attacker can inflict more harm but at some point increasing the devices involved does not result in a successful attack as all the devices involved might not have any physical relationship. With 8 devices involved, we get the maximum possible attack scenarios. If antecedent $\longrightarrow$ consequent is true then the system should be considered as under attack. These attack patterns are useful in signature-based intrusion detection. Usually these approaches suffer from a lack of attack patterns in complex physical processes. Through our approach, we were able to generate more than 300,000 attack patterns for SWaT. We plan to share our results and dataset publicly.

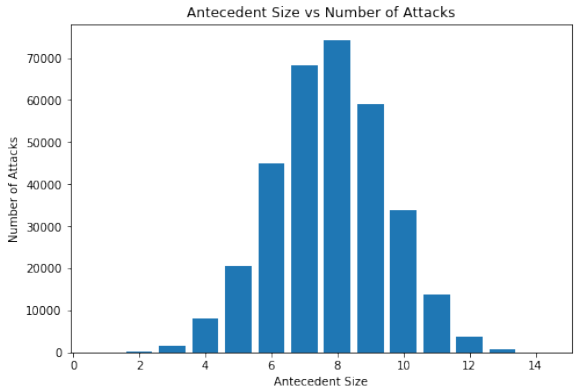

Figure 4: Number of Attacks with respect to Antecedent size

5 RELATED WORK

Earlier attempts to manually generate attacks could not scale well and are limited by the attacker's expertise. However, the use of machine learning to automatically generate attacks has received attention [15]. Most of the work is focused on adversarial learning that attempts to generate evasive attacks. In some cases [23] the researchers are able to generate stealthy attacks for a specific type of detector using the attack dataset from SWaT [11]. Another work [14] focused on the SWaT dataset to gradually poison the input samples during the training phase to avoid detection during the test phase. They chose a set of seven attacks from the attack dataset and derived stealthy attacks for a threshold-based detector operating on the residual signal. These schemes make use of publicly available attack dataset [11] and modify those attacks to generate a limited number of stealthy attacks for the specific detection method. In [10, 15], the authors used Generative Adversarial Networks (GANs) for learning anomaly detector classifiers and for the generation of malicious sensor measurements that will go undetected. A gradient-based adversarial attack scheme was proposed in [13] which was able to deceive Recurrent Neural Network (RNN) based anomaly detector in two-real world CPS namely WADI[4] and SWaT [17]. Erba et al. [9] has designed attacks to evade detection by an autoencoder in a water distribution systems [4]. They considered a white-box attacker that generates two different sets of spoofed sensor values for the PLC and the detector. These studies focus on adversarial learning to deceive a specific classifier. Some techniques modify sensor data while others modify actuator data. In our method, we do impose any such limitations. Our proposed technique is an attempt to generate attack patterns autonomously using the limited attack dataset.

Besides adversarial learning, there are techniques to generate synthetic data from a limited set of attack examples. A recent study uses the normal data from multiple sources and then generates anomalous data by sampling out of distribution data [18]. This method can generate anomalous data but that need not necessarily be the attack data, hence, not quite useful for security testing. There are a few well-known techniques to generate data samples, using the labeled examples from the minority class (anomalous data) [20]. One such technique named Synthetic Minority Oversampling Technique (SMOTE) [7], interpolates the current data samples to generate the artificial instances. However, though this oversampling method solves the imbalanced data problem for machine learning algorithms but does not provide any new scenarios but extends the provided minority examples. In contrast, the proposed technique provides thousands of new attack patterns, those never seen before and almost impossible to derive manually.

6 CONCLUSIONS

A rule-based machine learning technique is proposed to autonomously generate attack patterns for the physical process of an ICS. More than 300,000 attack patterns were generated in a real-world six-stage water treatment plant. This is an incredible amount of attack scenarios as compared to 36 public attack scenarios designed by domain experts on the same testbed. These insights are useful in testing the security of an ICS and providing additional data for signature-based intrusion detection systems.

ACKNOWLEDGEMENTS

This work is supported in part by the National Research Foundation and Cyber Security Agency of Singapore, under its National Cybersecurity RD Programme (Award No. NRF2018NCR-NSOE005-0001). Any opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not reflect the views of National Research Foundation and Cyber Security Agency of Singapore.

REFERENCES

[1] S. Adepu and A. Mathur. 2016. Generalized Attacker and Attack Models for Cyber Physical Systems. In *2016 IEEE 40th Annual Computer Software and Applications Conference (COMPSAC)*, Vol. 1. 283–292.

[2] Rakesh Agrawal, Tomasz Imieliński, and Arun Swami. 1993. Mining Association Rules Between Sets of Items in Large Databases. In *ICMD*, Vol. 22. ACM, New York, NY, USA, 207–216.

[3] Chuadhry Mujeeb Ahmed, Martin Ochoa, Jianying Zhou, and Aditya Mathur. 2021. Scanning the Cycle: Timing-Based Authentication on PLCs *(ASIA CCS '21)*. ACM, New York, NY, USA, 886–900. https://doi.org/10.1145/3433210.3453102

[4] Chuadhry Mujeeb Ahmed, Venkata Reddy Palleti, and Aditya P. Mathur. 2017. WADI: A Water Distribution Testbed for Research in the Design of Secure Cyber Physical Systems. In *CysWater*. ACM, NY, USA, 25–28. https://doi.org/10.1145/3055366.3055375

Table 2: A sample of attacks generated using ARM. *The comma (,) is equivalent to Boolean AND. If antecedent ⟶ consequent is true then the system should be considered as under attack.

Antecedent Size	No. of Attack Patterns	Antecedent*	Consequent
2	96	MV304=Close, FIT301<0.5	P302=Off
3	1554	MV301=Open, MV302=Close, MV303=Close	MV304=Close
4	8133	P302=Off, P203=Off, MV303=Close, MV304=Open	P602=Off
5	20485	P101=Off, FIT201<0.5, MV201=Open, MV302=Open, P302=On	MV101=Open
6	45006	FIT101<0.5, MV101=Close, P203=Off, MV304=Open, P302=Off, FIT601<0.5	MV303=Close
7	68294	FIT101<0.5, MV101=Close, P101=Off, P203=Off, FIT301<0.5, MV301=Close, P602=Off	MV201=Close
8	74348	MV101=Open, MV201=Open, P203=Off, FIT301>=0.5, MV301=Close, MV302=Open, MV303=Close, MV304=Close	P602=Off
9	58936	MV101=Open, P101=Off, FIT201<0.5, P203=Off, FIT301<0.5, MV301=Close, MV302=Open, MV304=Close, P602=Off	MV201=Close
10	33903	FIT101>=0.5, P101=Off, FIT201<0.5, MV201=Close, P205=Off, MV302=Open, MV303=Close, MV304=Close, P302=Off, FIT601<0.5	P602=Off
11	13832	FIT101<0.5, MV101=Close, MV201=Close, P203=Off, P205=Off, MV301=Close, MV302=Close, MV304=Open, P302=Off, FIT601<0.5, P602=Off	MV303=Close
12	3802	MOFIT101=Close, MV101=Close, FIT201<0.5, P203=Off, P205=Off, FIT301<0.5, MV301=Close, MV303=Close, MV304=Close, P302=Off, FIT601<0.5, P602=Off	P101=Off
13	632	FIT101>=0.5, MV101=Open, P101=Off, MV201=Close, P203=Off, P205=Off, FIT301<0.5, MV301=Close, MV302=Open, MV303=Close, P302=Off, FIT601<0.5, P602=OFF	MV304=Close
14	48	FIT101<0.5, MV101=Close, P101=Off, FIT201<0.5, MV201=Close, P203=Off, P205=Off, FIT301<0.5, MV301=Close, MV303=Close, MV304=Open, P302=Off, FIT601<0.5, P602=Off	MV302=Close

[5] Chuadhry Mujeeb Ahmed, Muhammad Azmi Umer, Beebi Siti Salimah Binte Liyakkathali, Muhammad Taha Jilani, and Jianying Zhou. 2021. Machine Learning for CPS Security: Applications, Challenges and Recommendations. In *Machine Intelligence for Cybersecurity Applications*. Springer, 397–421.

[6] Chuadhry Mujeeb Ahmed and Jianying Zhou. 2020. Challenges and Opportunities in Cyberphysical Systems Security: A Physics-Based Perspective. *IEEE Security Privacy* 18, 6 (2020), 14–22. https://doi.org/10.1109/MSEC.2020.3002851

[7] Nitesh V Chawla, Kevin W Bowyer, Lawrence O Hall, and W Philip Kegelmeyer. 2002. SMOTE: synthetic minority over-sampling technique. *Journal of artificial intelligence research* 16 (2002), 321–357.

[8] Pamel Cobb. 2015. German Steel Mill Meltdown: Rising Stakes in the Internet of Things. https://securityintelligence.com/german-steel-mill-meltdown-rising-stakes-in-the-internet-of-things/.

[9] Alessandro Erba, Riccardo Taormina, Stefano Galelli, Marcello Pogliani, Michele Carminati, Stefano Zanero, and Nils Ole Tippenhauer. 2019. Real-time Evasion Attacks with Physical Constraints on Deep Learning-based Anomaly Detectors in Industrial Control Systems. (07 2019).

[10] Cheng Feng, Tingting Li, Zhanxing Zhu, and Deeph Chana. 2017. A deep learning-based framework for conducting stealthy attacks in industrial control systems. *arXiv preprint arXiv:1709.06397* (2017).

[11] Jonathan Goh, Sridhar Adepu, Khurum Nazir Junejo, and Aditya Mathur. 2017. A Dataset to Support Research in the Design of Secure Water Treatment Systems. In *CRITIS*. Springer, Cham, 88–99.

[12] iTrust. 2021. Dataset and Models. https://itrust.sutd.edu.sg/itrust-labs_datasets/dataset_info/.

[13] Yifan Jia, Jingyi Wang, Christopher M. Poskitt, Sudipta Chattopadhyay, Jun Sun, and Yuqi Chen. 2021. Adversarial attacks and mitigation for anomaly detectors of cyber-physical systems. *International Journal of Critical Infrastructure Protection* 34 (2021), 100452.

[14] Moshe Kravchik, Battista Biggio, and Asaf Shabtai. 2020. Poisoning Attacks on Cyber Attack Detectors for Industrial Control Systems. *arXiv preprint arXiv:2012.15740* (2020).

[15] Zilong Lin, Yong Shi, and Zhi Xue. 2019. IDSGAN: Generative Adversarial Networks for Attack Generation against Intrusion Detection. arXiv:1809.02077 [cs.CR]

[16] Robert Lipovsky. 2016. New wave of cyber attacks against Ukrainian power industry. http://www.welivesecurity.com/2016/01/11.

[17] A. P. Mathur and N. O. Tippenhauer. 2016. SWaT: A water treatment testbed for research and training on ICS security. In *International Workshop on Cyber-physical Systems for Smart Water Networks (CySWater)*. IEEE, USA, 31–36.

[18] Truong Son Pham, Quang Uy Nguyen, and Xuan Hoai Nguyen. 2014. Generating artificial attack data for intrusion detection using machine learning. In *Proceedings of the Fifth Symposium on Information and Communication Technology*. 286–291.

[19] Marco Rocchetto and Nils Ole Tippenhauer. 2016. *On Attacker Models and Profiles for Cyber-Physical Systems*. Springer International Publishing, Cham, 427–449. https://doi.org/10.1007/978-3-319-45741-3_22

[20] Georg Steinbuß. 2020. *Calibration and Evaluation of Outlier Detection with Generated Data*. Ph.D. Dissertation. Karlsruher Institut für Technologie (KIT). https://doi.org/10.5445/IR/1000120534

[21] Muhammad Azmi Umer, Aditya Mathur, Khurum Nazir Junejo, and Sridhar Adepu. 2020. Generating invariants using design and data-centric approaches for distributed attack detection. *IJCIP* 28 (2020), 100341.

[22] Sharon Weinberger. 2011. Computer security: Is this the start of cyberwarfare? *Nature* 174 (June 2011), 142–145.

[23] Giulio Zizzo, Chris Hankin, Sergio Maffeis, and Kevin Jones. 2020. Adversarial Attacks on Time-Series Intrusion Detection for Industrial Control Systems. In *(TrustCom)*. IEEE.