Comparison of Design-Centric and Data-Centric Methods for Distributed Attack Detection in Cyber-Physical Systems

Jennifer Leopold, Bruce Mcmillin, Rachel Stiffler, Nathan Lutes

Chapter 1

A COMPARISON OF DESIGN- AND DATA-CENTRIC METHODS FOR DISTRIBUTED ATTACK DETECTION

Abstract Cyber-physical systems (CPS) are vulnerable to a variety of cyber, physical, and cyber-physical attacks. The security of a CPS can be enhanced beyond what can be achieved through firewalls and trusted components by building trust from observed and/or expected behavior. These behaviors can be encoded as invariants. Information flows that do not satisfy the invariants are used to identify and isolate a malfunctioning device or cyber intrusion. However, often it is the case that distributed architectures for cyber-physical systems contain multiple access points, which may be physically and/or digitally linked. Thus, invariants may be difficult to determine and/or computationally prohibitive to check in real-time. Researchers have employed a variety of methods to determine these invariants based on analyzing the design of and/or data generated by cyber-physical systems such as water treatment plants, thermal power plants, and electric power grids. This paper compares the effectiveness of detecting attacks on a secure water treatment plant using design-centric invariants versus data-centric rules, the latter generated using a variety of data mining methods. We compare the approaches in terms of maximization of true positives and minimization of false positives.

Keywords: cyber physical attacks, invariants, data mining, water treatment plant

1. Introduction

Modern cyber-physical systems (CPS) typically contain multiple entry points, particularly through edge devices which provide access into a service provider core network; such a device could simply be a smart meter, a house monitor, or a camera. Often it is the case that in distributed architectures for CPS, the access points are physically and/or digitally linked; hence a computationally prohibitive number of (combinatorial) conditions could need to be checked in real-time to enforce security of the overall system.

Efforts to secure a cyber-physical system must contend with a number of challenges, including the following:

- Combining cyber and physical information flows affords the use of a single model to represent both. However, the semantics of combined cyber-physical information flow among security domains often is not well understood; this is necessary if automation of securing information flows is to be achieved.
- Information flow is fundamentally bidirectional between two security domains [9]. Information flow either simultaneously preserves integrity and not confidentiality, or simultaneously preserves confidentiality but not integrity. This needs to be made clear to system designers.
- Mining a system's behavior by observing the system's operations may not uncover all of its operational modes, or may result in an overly complex model with a surplus of generated rules.
- Observations of the physical system may not be timely, accurate, or complete due to malicious information, bad data, communication delays, or noisy data.
- Proliferation of individual security domains causes high model complexity. Merging too many of them can result in a trivial, all-encompassing security domain.

One solution to the current vulnerability of cyber-physical systems is to enhance the security of a CPS beyond what can be achieved through firewalls and trusted components, instead building trust from observed and/or expected behavior. These behaviors are encoded as invariants. Redundant, yet inconsistent, information flows that do not satisfy the invariants can be used to identify and isolate a malfunctioning device or cyber intrusion. Researchers have employed a variety of methods based on analyzing the design of and/or data generated by cyber-physical systems such as water treatment plants and electric power grids. Herein we empirically compare the effectiveness of detecting attacks on a secure water treatment plant using design-centric invariants versus data-centric rules generated by a variety of data mining methods.

The remainder of this paper is organized as follows. Section 2 briefly discusses related work in data-centric and design-centric approaches for generating invariants for CPS. Section 3 provides background information about the secure water treatment plant used as a test dataset for our study and the metrics used to evaluate the different rule sets. In

Section 4 we analyze the accuracy of one design-centric and eight data-centric models. Directions for future work are discussed in Section 5, followed by summary and conclusions in Section 6.

2. Related Work

An invariant is a condition that should hold during the flow of information through a device or process from one state to the next. Invariants for a CPS can be derived using a variety of methods. In a design-centric approach, invariants are determined by examining the CPS design and control algorithms. This approach was studied in [1, 2], with the Distributed Attack Detection (DAD) system of [2] being notable for its use of only continuous (i.e., non-discretized) data. In contrast, data-centric approaches determine invariants by analyzing the data generated by the cyber-physical system under (labeled) normal and attack operating conditions. Various data mining and machine learning methods have been utilized to generate invariants including association rule mining, support vector machines, decision trees, and neural networks; see [6, 10] for examples of such studies.

In 2017, the authors of [10] conducted a study to compare design-centric and data-centric approaches for deriving invariants for a secure water treatment plant. For the design-centric approach, system component specifications and State Condition Graphs (SCGs) for the plant were examined and 39 invariants were manually defined; see [3] for a more in-depth discussion of SCGs. For the data-centric approach, association rule mining was used, resulting in hundreds of invariants. The analysis performed in that study yielded two important conclusions: (1) the data-centric approach did not find all of the invariants defined by the design-centric approach, likely because of loss of information from the data discretization that was necessary for association rule mining and the lack of data that tested certain possible/expected conditions, and (2) the design-centric approach did not include all the invariants defined by the data-centric method, most likely because of the comprehensive nature of the data mining method, which considered all possible combinations of conditions. There were 9 invariants that were common to both the data-centric and design-centric lists of rules. The authors of the study hypothesized that an integrated approach for invariant generation might be advantageous; however, no test results were reported for such a comparison.

3. Background

In this section we provide a description of the cyber-physical system we used as a test dataset for analyzing design-centric and data-centric rules for detecting attacks. We also provide a brief discussion of the metrics that were the basis for evaluating the rule sets.

3.1 SWaT: A Secure Water Treatment Plant

The test dataset used for this study is Secure Water Treatment (SWaT) [4], a testbed for research in cyber security. The plant produces 5 gallons/minute of treated water, and can operate non-stop, 24 hours per day, 7 days a week in a fully autonomous mode. There are six stages (i.e., processes) in the plant, each of which is controlled by a Programmable Logic Controller (PLC). The state of each stage is measured using a sensor, and the control is modified via an actuator. In all there are 68 sensors and actuators; some actuators serve as standbys and are used only when the primary actuator fails. See Figure 1 for a diagram of the system's primary components [10].

A multi-layer, ring network enables communications across all components of SWaT. Hence PLCs at one level are able to communicate with sensors and actuators that are at another level, and in a different stage. Operation of the plant can be monitored and controlled by an operator at a Supervisory Control and Data Acquisition (SCADA) workstation. Physical attacks can occur by acts such as replacing or removing sensors, disconnecting wires between components (i.e., sensors, actuators, and/or PLCs), or interrupting the power flow to any of the electronic components.

Each instance in the dataset has either a discrete or continuous value for each of the sensors, and a timestamp denoting when the reading was taken; readings are taken every second over a 24-hour period, operating for a certain number of days under normal conditions and a certain number of days under attacks. The originally published SWaT dataset [5] consisted of 890,298 instances labelled as sensors operating normally and 54,621 instances labelled as sensors under attack. However, when that dataset was created, the system was operating in a variety of non-standard (i.e., atypical) conditions, with no indication of state recorded in the published dataset. As will be discussed in the next section, some of the design-centric rules are sensitive to the system operating state. Also, values of some attributes in the dataset were not within a normal range and/or were abnormally fluctuating in value during certain states or state changes, although the system was not under attack. Without knowing the system state, this dataset was inappropriate for either

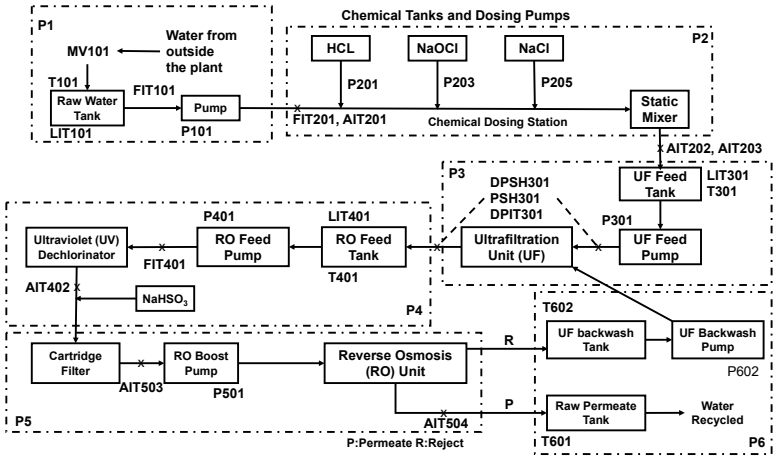

Figure 1: Secure Water Treatment (SWaT) Architecture [10]

design-centric or data-centric testing. The SWaT dataset of this paper (currently available at [5]) does not include collection of data in atypical states (e.g., initial startup, backwash, etc.). Additionally, this dataset did contain state information. This is the dataset that was used for all testing in this study; it consisted of 13,070 instances labelled as sensors operating normally and 1,926 instances labelled as sensors under attack (i.e., a binary attribute *is_attack* labels whether an instance is or is not in a state of attack). We were not made aware of the nature of the attacks.

3.2 Evaluation Metrics

For the purposes of evaluating the effectiveness of design-centric versus data-centric rules for detecting attacks on the SWaT system, we considered a *true positive* (TP) to be an instance labelled as an attack (i.e., $is\_attack = 1$) that was correctly detected as an attack, a *false positive* (FP) to be an instance labelled as normal (i.e., $is\_attack = 0$) that was detected as an attack, a *true negative* (TN) to be an instance labelled as normal that was detected as normal, and a *false negative* (FN) to be an instance labelled as an attack that was detected as normal. In the cybersecurity domain, the highest priority is to maximize the number of true positives and minimize the number of false negatives. A false negative (i.e., missing an attack) can result in severe consequences. While one does not want to "cry wolf" by incorrectly classifying a normal instance as an attack (i.e., a false positive), such situations carry less severe consequences.

Accuracy, the proportion of the total number of predictions that were correct, is not the only metric that is informative for evaluating a decision model (e.g., a rule set). *Precision* and *sensitivity* (also known as *recall*) are two different ways to measure the proportion of instances that were correctly identified as positive amongst instances that were correctly or incorrectly identified. *Specificity* is the proportion of negative instances correctly identified. In general, precision and sensitivity are also important in that they are viewed as metrics of exactness and completeness of testing, respectively. The equations for accuracy, precision, sensitivity, and specificity are given in eqs. 1 - 4, respectively.

$$accuracy = (TP + TN) / (TP + FP + FN + TN) \tag{1}$$

$$precision = TP / (TP + FP) \tag{2}$$

$$sensitivity = TP / (TP + FN) \tag{3}$$

$$specificity = TN / (TN + FP) \tag{4}$$

Yet another useful metric for evaluating a decision model is the *Kappa statistic*; it indicates how much better the predictive model is than simply random guessing. The equation for calculating the Kappa statistic is given in eq. 5; the higher the value, the better the predictive model is than a random predictor. In this equation, N is the number of instances in the dataset, *SuccessesNonRandom* is the number of instances that the (non-random) model correctly predicted, and *SuccessesRandom* is the number of instances that a (system generated) random model correctly predicted.

$$Kappa = (SuccessesNonRandom - SuccessesRandom) / (N - SuccessesRandom) \tag{5}$$

4. Rules for the SWaT System

Motivated by the limited work reported in [10], we wanted to compare the effectiveness of design-centric and data-centric rule sets for a cyber-physical system, namely the SWaT dataset described in Section 3.1. In

this section, we present the rules produced by a variety of methods and discuss how well each method was able to detect attacks in the SWaT dataset.

4.1 Design-Centric Method

In a study by Umer et al. [10], 39 invariants were identified, generated manually by examination of the SWaT State Condition Graphs and system specifications. When we attempted to test the accuracy of those rules on the SWaT dataset, we encountered a number of problems.

Each rule is specified using the syntax *antecedent* $\rightarrow$ *consequent*. The rule is intended to be checked in the following manner, where n is the number of seconds to wait for the consequent to become true once the antecedent has been satisfied:

If the *antecedent* is true
  Wait n seconds
  If the *antecedent* and the *consequent* are true
    Then the system is functioning normally
    Else the system is not functioning normally
  End-if
End-if

Interestingly, we do not deal with the situation where the antecedent becomes false during the n-second wait period, a condition that occurred seven times in the SWaT dataset. We are only concerned that the antecedent (and the consequent) are true at the end of the n-second period. This could be a problem if, as an extreme example, a water tank reaches a certain level (satisfying the antecedent), leaks some water on the floor because of a malfunction or some malfeasance during the wait period (thereby no longer satisfying the antecedent), but refills to a certain level (again satisfying the antecedent) before the end of the wait period.

No time constraints were published for the design-centric invariants. We had to obtain the time constraint for each rule by contacting one of the authors of [10] (Adepu). Even then, the time constraint for several rules had to be increased by a few seconds to best fit (i.e., increase the attack detection accuracy of) the SWaT dataset that we were testing.

Other problems that we encountered with the originally published set of design-centric rules included specification of attributes that were not included in the dataset, unspecified threshold values (labelled only with names such as *delta*), reversal of conjunctions and disjunctions, and critical missing conditions in both the antecedents and consequents. These problems necessitated the deletion of a few rules (i.e., those involving attributes that were not in the dataset), and resulted in a proliferation

of false positives and false negatives. Many of these problems were resolved with the help of one of the authors of [10] (Adepu), while other problems were resolved by examination of the data and/or specifications in the SWaT operating manual [7].

The resulting set of 31 design-centric rules, with each rule's respective time constraint in seconds, is shown in Table 1. The order in which the rules are listed has no significance. Values of 1 and 2 for pumps (e.g., P101) represent off and on, respectively. Values of 1 and 2 for valves (e.g., MV201) represent closed and open, respectively. Attributes with the prefix PLC represent the state of the system (e.g., initial startup, backwash, etc.). See the SWaT operating manual [7] for more information about each of the various attributes.

This rule set was treated as a set of invariants. Starting from any instance in the dataset, all rules had to be satisfied or else an attack was considered to have occurred. Because multiple rules, with different time constraints, had to be satisfied for an attack *not* to have occurred, it was not possible to calculate the true negative rate (i.e., the number of labelled normal instances detected to be normal). For similar reasons, false negatives (i.e., the number of labelled attacks detected to be normal) could not easily be determined. Consequently, we could not evaluate accuracy, specificity, or sensitivity.

When this set of design-centric rules was tested on the SWaT dataset, the false positive rate (i.e., instances labelled as normal but detected as attacks) was 55.47% and the true positive rate (i.e., actual attacks correctly detected as attacks) was 44.53%. In all, there were 446 occurrences of false positives. Of those 446 false positives, 14 were caused by Rule 1 and 432 were caused by Rule 5. These problems could have been due to a couple of faulty components. For example, in the case of Rule 5 a motorized valve (MV201) may have had trouble closing; however, further increasing the time constraint for Rule 5 only resolved 8 of the 432 false positives attributed to that rule. Of graver concern than the high number of false positives was the unacceptably low rate of true positives. One possible explanation for this was that the threshold values in the design-centric rules may not have been realistic for the actual performance of the individual components in the current system. For example, the LIT301 attribute was required to be less than or equal to 800 in Rule 4; however, as will be seen in one of the data-centric rule sets, LIT301 was required to be less than or equal to 793.7869. This was the case for several of the attributes that the design-centric and data-centric rule sets had in common. Another possible explanation for the low true positive rate could have been the time constraints. Upon manual examination of the dataset we found several cases where it took longer than

the specified amount of time for the consequent in a rule to be satisfied. When we attempted to increase the time constraint for some of those rules, the increase in the true positive rate was often coupled with an increase in the false positive rate.

4.2 Data-Centric Methods

We employed a variety of data mining methods to generate rules to detect attacks in the SWaT dataset. Unlike a set of invariants, each of these rules had as the consequent a single binary condition indicating whether or not an attack had occurred; and, unlike a set of invariants, it was not necessary for an instance to satisfy all rules in the rule set. It also should be noted that none of these methods took into account the temporal nature of the physical system; that is, unlike the design-centric rules, they did not consider that a certain number of seconds would need to transpire between satisfaction of the various conditions in a rule. For that reason, we used 10-fold cross validation to test these methods; breaks in time sequence between instances were not as critical as they were in the design-centric testing.

All methods were implemented in R. None of the methods required discretization of the SWaT data. However, attributes with values that did not change from row to row were removed prior to creating the data-centric models (i.e., AIT401 and all of the pumps except P101, P203, P205, P301, P401, and P601); these attributes were not informative and would not have been included in any decision model.

The methods (some of which were ensemble methods) that produced the best accuracy were: J48 and C5.0 (both decision tree methods with pruning), rfRules (random forest specifying a maximum of 300 trees), AdaBoost.M1 (boosting with decision trees), a Naïve Bayesian network (NB), a support vector machine (linear SVM with a polynomial kernel), JRip (association rules with reduced error pruning), and a multi-layer perceptron neural network (NNet). Other methods such as various linear regression models and other association rule methods were tried; however, those results are not included herein because their accuracy was not competitive with and/or their results were not as informative as the other methods that are presented. For the reader not familiar with the aforementioned data mining methods, see [11].

Table 2 provides a very brief comparison of the aforementioned methods based only on accuracy and the Kappa statistic. Figure 2 shows a more extensive comparison in terms of accuracy, the Kappa statistic, sensitivity, specificity, and precision for the eight methods. Based on these results, we determined that the two ensemble methods (rfRules

Table 1: Design-Centric Invariants for SWaT

Rule #	**Rule as *Antecedent* → *Consequent***	**Time Constraint (sec)**
1	LIT101<=500 → MV101=2	14
2	LIT101>=800 → MV101=1	12
3	LIT101<=250 → P101=1 and P102=1	2
4	LIT301<=800 and PLC1!=1 → P101=2 or P102=2 or PLC1!= 2	12
5	LIT301>=1000 → P101=1 and P102=1 and MV201=1	3
6	LIT301<=800 and PLC2!=1 → MV201=2	10
7	FIT201<0.5 → P201=1 and P202=1 and P203=1 and P204=1 and P205=1 and P206=1	2
8	AIT201>260 and FIT201 >0.5 → P201=1 and P202=1	2
9	MV201=2 and FIT201>0.5 and AIT503<260 and AIT201<250 and PLC2!=1 → P201=2 or P202=2	7
10	AIT503>=260 and MV201 > 0.5 → P201=1 and P202=1	2
11	MV201=2 and FIT201>0.5 and AIT503<260 and PLC2!=1 → P201=2 or P202=2	2
12	AIT202<6.95 → P203=1 and P204=1	2
13	MV201=2 and AIT202>=7.05 and FIT201 > 0.5 and PLC2!=1 → P203=2 or P204=2	8
14	AIT203>500 → P205=1 and P206=1	2
15	P101=2 or P102=2 and MV201=2 and AIT203<=420 and FIT201> 0.5 and AIT402<250 and PLC2!=1 → P205=2 or P206=2	6
16	AIT402>=250 → P205=1 and P206=1	2
17	MV201=2 and FIT201>0.5 and AIT402<240 and PLC2!=1 → P205=2 or P206=2	6
18	LIT301<=250 → P301=1 and P302=1	4
19	LIT401>=1000 → P301=1 and P302=1	2
20	LIT301>250 and LIT401<=800 → P301=2 or P302=2 or PLC3!=7	2
21	LIT401<=250 → P401=1 and P402=1	4
22	LIT401<=250 → UV401=1	4
23	P401=1 and P402=1 → UV401=1	4
24	FIT401<0.5 → UV401=1	4
25	P401=1 and P402=1 and PLC4=4 → UV401=1	4
26	AIT402<240 → P403=1 and P404=1 or PLC4!=4	2
27	AIT402>=250 → P403=2 or P404=2 or PLC4!=4	2
28	P401=1 and PLC5=12 → P501=1	5
29	UV401=1 → P401=1 or P402=1	4
30	FIT401<0.5 and PLC5=12 → P501=1 or P502=1	4
31	LIT101>1100 → P601=1	2

and AdaBoost.M1) did not improve upon the individual decision tree methods (J48 and C5.0), and did not justify the additional complexity and memory requirements of an ensemble. We also eliminated the Naïve Bayesian network and the neural network from our top candidates due to their lower accuracy, Kappa value, sensitivity (in the case of the Naïve Bayesian network), specificity, and precision. Although the support vector machine performed well, we decided to further investigate only the J48, C5.0, and JRip models as they not only performed well, but were the most comparable models in terms of their expression as rule sets.

Table 2: Accuracy and Kappa for Eight Data Mining Methods

Method	**Accuracy**	**Kappa**
J48	0.998044	0.991256
Naïve Bayes	0.871880	0.212844
SVM	0.998311	0.992461
rfRules	0.984352	0.926389
AdaBoost.M1	0.999289	0.996819
C5.0	0.997866	0.990452
JRip	0.998399	0.992863
Neural Network	0.871533	0.000000

Stacking was employed using the three selected methods (J48, C5.0, and JRip) as the base classifiers and the best performing of the three, JRip, as the ensemble method. This was done to see if an ensemble of the best three could improve upon the performance of the individual methods. The resulting accuracy was 0.998399 and the Kappa value was 0.992813. This was not an improvement over the individual JRip method, and thus was not further considered in the interest of memory efficiency.

The J48, C5.0, and JRip predictive models are expressed as rule sets. Even rules derived from pruned decision trees (e.g., J48 and C5.0) can result in unnecessary conditions. Thus, in order to construct the most efficient rule sets, we implemented a program in R to check for and drop unnecessary conditions; the strategy that we utilized (originally presented in [8]) is summarized below.

Let R be a rule of the form $A \rightarrow C$ where A is the antecedent, C is the consequent, and the antecedent A contains more than one condition. Let a more general rule R' be of the form $A' \rightarrow C$ where A' is obtained by deleting one condition X from A'. Consider the contingency table shown in Table 3, where the counts *Y1, Y2, E1,* and *E2* have been collected considering instances in the dataset that satisfy condition A'.

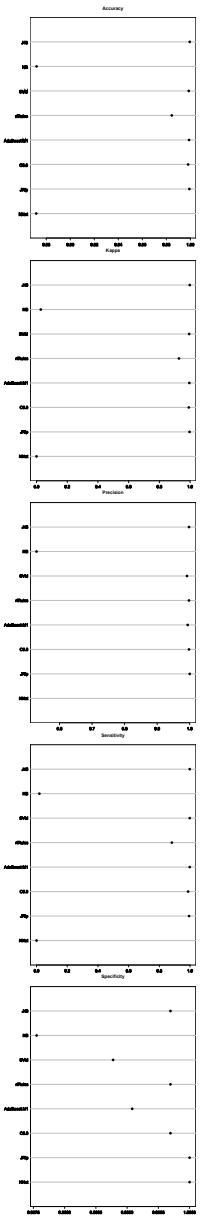

Figure 2: Evaluation Metrics for Eight Data Mining Methods Applied to SWaT Dataset

Let $U_{25}(E, S)$ be a binomial distribution using a 25% confidence level for a sample size S with E negative cases in the sample. We estimate the error rate of rule R as $U_{25}(E1, Y1+E1)$ and estimate the error rate

Table 3: Contingency Table for Instances Satisfying *A'*

	C	**!C**
X	Y1	E1
!X	Y2	E2

of rule *R'* as $U_{25}(E1+E2,\ Y1+Y2+E2+E2)$. If the error rate of *R'* is no greater than that of *R*, we delete condition *X*. Using a greedy approach, we remove the condition that produces the lowest error rate of the rule, continuing until no more conditions can be deleted. The choice of 25% for the confidence level is considered a conservative value for maintaining high accuracy in rule sets [8].

Table 4 shows the number of rules before dropping conditions, the number of rules that had conditions dropped, and the average number of conditions in the antecedent per rule (after dropping conditions) for each of the three methods that we continued to investigate. The resulting JRip and C5.0 rule sets are shown in Tables 5 and 6, respectively; due to space constraints, the J48 rule set is not included.

There are only a few rules that are identical in these three methods, those being in the J48 and C5.0 rule sets, which were both decision tree methods. Yet there are some commonalties between the rule sets. Fig. 4 shows the attributes that are common (and different) in the J48, JRip, and C5.0 rules. Of particular interest are the six attributes that were included in all three rule sets: AIT202, AIT203, LIT301, LIT401, AIT402, and AIT201. These attributes could be the primary "canaries in the coal mine" with respect to detecting attacks, and possibly were the direct targets of the attacks.

Table 4: Profile of Data-Centric Rules Before and After Conditions Dropped

	J48	**JRip**	**C5.0**
No. of Rules Before Dropping Conditions	25	11	19
No. of Rules That Had Conditions Dropped	19	2	1
Avg. No. of Conditions in Antecedent	5.68	2.64	2.42

Not surprisingly, those six attributes also featured prominently in the design-centric rule set. However, the design-centric rule set was distinctive in frequently specifying conditions on various pumps (attributes with the prefix P followed by three digits such as P201, P302, etc.) and the ultraviolet dechlorinator (UV401), whereas only one pump attribute (P301) was used in the J48 and C5.0 rules, and the dechlorinator at-

Table 5: JRip Rule Set for SWaT

Rule #	Rule as *Antecedent* → *Consequent*
1	AIT202 <= 8.931236 and LIT301 >= 923.2809 → is_attack = 1
2	AIT402 <= 6.613689 and PIT503 >= 113.800949 and PIT501 <= 159.590485 and LIT101 >= 596.9952 and DPIT301 <= 1.245278 and DPIT301 >= 1.229272 → is_attack = 1
3	AIT402 <= 6.613689 and AIT402 >= 5.075622 and LIT301 <= 829.274536 and LIT101 <= 769.3148 → is_attack = 1
4	AIT402 <= 6.664958 and LIT301 >= 938.781738 and AIT203 >= 262.4968 → is_attack = 1
5	LIT301 >= 1024 and AIT502 >= 69.21302 and FIT503 >= 0.610313 and LIT401 <= 864.932068 → is_attack = 1
6	LIT301 >= 1027.94153 and AIT203 <= 240.810043 → is_attack = 1
7	LIT301 >= 1112.57532 and AIT202 >= 9.202961 and AIT201 >= 126.24968 → is_attack = 1
8	LIT401 <= 787.2597 → is_attack = 1
9	FIT601 >= 0.000384 and LIT301 >= 1024 and LIT401 >= 848.936157 → is_attack = 1
10	DPIT301 <= 1.104424 → is_attack = 1
11	else → is_attack = 0

tribute was never used by J48, C5.0, or JRip. In fact, the P203 and P205 attributes had identical values in the SWaT dataset, eliminating the necessity to test more than one of them. Another point of interest is that, of the six stages of SWaT (labelled P1-P6 in Fig. 1), the least often referenced attributes in both the design-centric and the data-centric rules are from P1, P5, and P6. One wonders if the attacks were focused on the P2, P3, and P4 stages and/or if there is more of a "ripple effect" between sensors in those particular stages.

5. Future Work

The data-centric rules produced in this study are based on a single dataset and hence may be subject to overfitting, despite undergoing 10-fold cross validation which divided the data into multiple training and test datasets. In the future we would like to test these rules on the live SWaT system. This is comparable to what has been done for the work reported in [2].

There are other structured CPS test sets available from the iTrust Center for Research in Cyber Security [5], namely the Water Distribution (WADI) test set and the Electric Power and Intelligence Control (EPIC)

Table 6: C5.0 Rule Set for SWaT

Rule #	**Rule as *Antecedent* → *Consequent***
1	AIT202 <= 9.383043 and LIT301 <= 923.2008 → is_attack = 0
2	LIT301 > 923.2008 and LIT401 > 878.4286 and AIT201 > 126.0895 → is_attack = 0
3	AIT402 > 6.664958 → is_attack = 0
4	LIT301 <= 923.2008 and LIT401 > 954.0246 → is_attack = 0
5	AIT202 > 8.931236 and LIT401 <= 805.5627 → is_attack = 0
6	AIT202 > 8.931236 and PIT503 <= 113.6728 → is_attack = 0
7	P301 > 0 and AIT202 > 8.931236 and LIT401 <= 849.5514 → is_attack = 0
8	AIT202 > 8.931236 and AIT202 <= 8.996923 → is_attack = 0
9	AIT203 <= 262.3686 and FIT501 > 0.80223 and AIT201 <= 126.0895 → is_attack = 0
10	AIT203 <= 262.3686 and FIT504 > 0.209781 and AIT201 <= 126.0895 → is_attack = 0
11	AIT202 <= 9.020315 and LIT401 > 865.4319 → is_attack = 0
12	AIT202 > 8.931236 and LIT301 > 923.2008 and LIT301 <= 938.3812 → is_attack = 0
13	LIT301 <= 793.7869 → is_attack = 0
14	AIT202 > 9.070623 and FIT503 <= 0.606984 → is_attack = 0
15	AIT402 <= 3.409382 and FIT401 > 0.802484 → is_attack = 0
16	AIT202 > 8.996923 and AIT402 <= 6.664958 and LIT401 > 805.5627 and LIT401 <= 849.5514 → is_attack = 1
17	AIT202 > 9.383043 and AIT203 > 243.2453 and AIT402 <= 6.664958 and LIT401 <= 954.0246 → is_attack = 1
18	AIT202 > 9.39554 and LIT301 > 793.7869 and AIT402 <= 6.664958 → is_attack = 1
19	LIT301 > 923.2008 and AIT402 <= 6.664958 → is_attack = 1

test set. In the future, we plan to repeat our comparison of design-centric and data-centric models for attack detection on these other datasets. However, that first will require preparation of design-centric invariants, a task for which we are interested in developing semi-automated generation methods using the schematics and other design documents available for those systems.

We also plan to investigate other methods for generating data-centric models to detect attacks. One particular area that may be promising is to model a cyber-physical system as a graph and use dynamic graph anomaly detection algorithms to deduce invariants. Modeling the system as a graph is a natural extension of the interconnected physical design

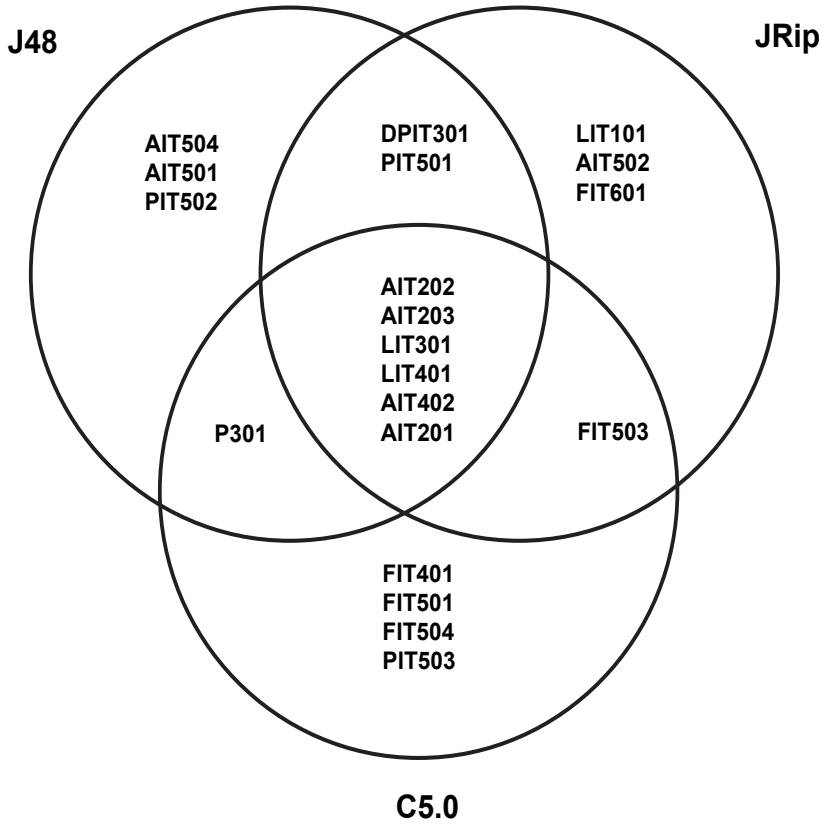

Figure 3: Common (and Different) Attributes in Data-Centric Rule Sets Based on Their Reference in the Antecedents and Consequents of the Rules

of such systems, and may provide a bridge to developing automated methods for generating design-centric invariants as well.

6. Summary and Conclusions

The detection of attacks in cyber-physical systems is a challenging task, in large part due to multiple access points that are physically and/or digitally linked. One approach for detecting attacks is the enforcement of invariants, conditions that must hold during the operation of a physical process and, when violated, are indicative of a cyber-attack or a component fault. Invariants can be generated by manual examination of the design of a system (i.e., a design-centric approach) and/or by deduction of patterns observed in data collected when the system is

operating under normal versus abnormal conditions (i.e., a data-centric approach).

Herein we compared the predictive quality of one design-centric rule set and multiple data-centric rule sets. In this limited study with one CPS test set (SWaT), we found that the data-centric methods were far better in terms of correctly predicting attacks and minimizing false negatives (i.e., incorrectly detecting attacks as normal). The results of this study cannot be generalized, and certainly should not serve as a chastisement of design-centric invariants. A design-centric approach has the advantage of domain expert consideration of how the system should behave under a variety of conditions. However, because it is a time-consuming (typically manual) task, it may not include all contingencies. Furthermore, design-centric rules may require adjustments in threshold values to account for the wear-and-tear on the system's physical components over time. A data-centric approach can more easily generate an optimal combination of the numerous possible conditions that identify anomalous behavior in the system. And, as shown by the results in this study, a data-centric approach learns from a real-time system (which can be re-run on a periodic basis) that may no longer correspond to the original design specifications. But a data-centric rule set is only as complete as the data upon which it is based; if the data model is not given a dataset that tests all possible situations, the ensuing rule set will not check for all contingencies. Overall, the best strategy is to consider the information provided by both approaches.

7. Acknowledgments

This work was performed with the assistance of three students at Missouri University of Science and Technology: Nathan Lincoln, Michael Macke, and Alex Warhover.

This work was supported by the National Science Foundation (NSF) under Grant No. CNS-183747.

References

[1] S. Adepu and A. Mathur, Generalized attacker and attack models for cyber physical systems, *Proceedings of the IEEE Fortieth Annual Computer Software and Applications Conference*, pp. 283–292, 2016.

[2] S. Adepu and A. Mathur, Distributed attack detection in a water treatment plant: Method and case study, *IEEE Transactions on Dependable and Secure Computing*, 2018.

[3] S. Adepu, A. Mathur, J. Gunda and S. Djokic, An agent-based framework for simulating and analysing attacks on cyber physical systems, in *Algorithms and Architectures for Parallel Processing*, G. Wang, A. Zomaya, G. Martinez and K. Li (Eds.), pp. 785–798, 2015.

[4] J. Goh, S. Adepu, K. Junejo and A. Mathur, A dataset to support research in the design of secure water treatment systems, in *Critical Information Infrastructures Security*, G. Havarneanu, R. Setola, H. Nassopoulos and S. Wolthusen (Eds.), pp. 88–99, 2016.

[5] iTrust, Singapore University of Technology and Design (`itrust.sutd.edu.sg/`).

[6] K. Junejo and D. Yau, Data driven physical modelling For intrusion detection in cyber physical systems, in *Proceedings of the Singapore Cyber-Security Conference*, A. Mathur and A. Roychoudhury (Eds.), pp. 43–57, 2016.

[7] Secure Water Treatment System (SWaT) Operating Manual, Singapore University of Technology and Design (`itrust.sutd.edu.sg/testbeds/secure-water-treatment-swat/`), 2014.

[8] J. Quinlan, *C4.5: Programs for machine learning*, Morgan Kaufmann Publishers, San Mateo, California, USA, 1993.

[9] D. Sutherland, A model of information, *Proceedings of the Ninth National Computer Security Conference*, pp. 175–183, 1986.

[10] M. Umer, A. Mathur, K. Junejo and S. Adepu, Integrating design and data centric approaches to generate invariants for distributed attack detection, *Proceedings of the Workshop on Cyber-Physical Systems Security and Privacy*, pp. 131–136, 2017.

[11] I. Witten, E. Frank and M. Hell, Data mining: Practical machine learning tools and technique, *ACM Software Engineering Notes*, vol. 36(5), pp. 51–52, 2011.