Proceedings of the Singapore Cyber-Security Conference (SG-CRC) 2016
A. Mathur and A. Roychoudhury (Eds.)

doi:10.3233/978-1-61499-617-0-43

Data Driven Physical Modelling For Intrusion Detection In Cyber Physical Systems

Khurum Nazir JUNEJO [a,c,1], David YAU [a,b]
[a] *Singapore University of Technology and Design, Singapore*
[b] *Advanced Digital Sciences Center, Illinois at Singapore*
[c] *Karachi Institute of Economics and Technology, Pakistan*

Abstract. Cyber physical systems are critical to the infrastructure of a country. They are becoming more vulnerable to cyber attacks due to their use of off the shelf servers and industrial network protocols. Availability on World Wide Web for monitoring and reporting, has further aggravated their risk of being attacked. Once an attacker breaches the network security, he can affect the operations of the system which may even lead to a catastrophe. Mathematical and formal models try to detect the departure of the system from its expected behaviour but are difficult to build, and are sensitive to noise. Furthermore they take a lot of time to detect the attack. We here propose a behaviour based machine learning intrusion detection approach that quickly detects attacks at the physical process layer. We validate our result on a complete replicate of the physical and control components of a real modern water treatment facility. Our approach is fast, scalable, robust to noise, and exhibits a low false positive (FP) rate with high precision and recall. The model can be easily updated to match the changing behaviour of the system and environment.

Keywords. Cyber Physical Systems Security, Machine Learning, Intrusion Detection, Fault Detection

1. Introduction

Cyber Physical Systems (CPSs) are geographically dispersed, large-scale, life-critical, and expensive systems. Smart grids, water filtration and distribution, unmanned vehicles, and pervasive health care systems are some of the examples of such systems. They comprise of physical components such as actuators, sensors, and cyber components such as network components and commodity serves. These network can be wired or wireless, and run off the shelf industrial network protocols. Furthermore they maybe available on the World Wide Web for remote monitoring and reporting. Thus leaving them prone to attacks from not only criminals, hacktivists, or disgruntled employees, but also from enemy states, such as the attack on Iran's nuclear centrifuge [5]. A hacker in US infected a water filtering plant with a software that could alter the plants water treatment operations

[1]Corresponding Author: Singapore University of Technology and Design, 8 Somapah Rd, Singapore 487372; E-mail: khurum_junejo@sutd.ed.sg.

[4]. A disgruntled ex-employee took control of Australia's Maroochy Water Services and released one million litres of untreated sewage into local parks and rivers for many days [22]. Thus intrusion detection systems (IDS) are needed to secure these systems.

Traditional IDS systems try to detect intrusion in the network traffic layer only. They do so by either a dictionary of known attacks or by modelling the normal behaviour of the traffic through Machine Learning (ML) techniques. The former requires the cumbersome task of keeping the dictionary of attacks upto date, and are totally blind to zero day attacks. The later suffer from missed detections and high false alarms. Once the adversary has breached the traffic layer, the system is completely at his mercy. An IDS system is needed at the physical layer to serve as a last line of defence. Current systems tackle this problem by deploying a mathematical or some formal model of the physical process that simulates the system and classifies deviation from the expected output as an attack or an abnormal behaviour. These techniques are referred to as behaviour-specification approaches. They require intricate understanding of the physical process, physical laws that govern this process, and the equipment specifications. They are costly to come up with, are sensitive to noise, do not update to the change of the environment or the ageing of the plant, and lastly the behaviour of the physical process may not exactly follow the specification in operational manual.

Data driven (aka behaviour based) approaches do not suffer from these problems. They do not look for something specific and nor do they require an expert to build a formal model, instead they learn the behaviour of the system through historical data. They are fast, easier to develop, and are robust to noise and can adapt to the drift in the normal behaviour of the system with time and change in weather. However these approaches are very rare in the literature because they suffer from the lack of availability of training data under an attack scenario. So they depend on simulations to generate their results. These results may not be valid on real operational systems. We on the hand use the data from Secure Water Treatment (SWaT) testbed, a complete replicate of the physical and control components of a real modern water treatment facility to learn our model and validate results.

Since behaviour based approaches learn directly from the data, they are not blind to the operational behaviour of the CPS that is sometimes different than the vendor specification. This peculiar behaviour is missed by the simulations and specification based approaches. As an example, in SWaT, the closing and opening of a valve is not immediate, as specified in the operational manual. Instead, the valve is in a transient state for a dozen of seconds until it opens completely, followed by a similar delay in the water inflow to reach its desired rate as specified in the manual. This rate of inflow at times can be even greater than 11% as specified in the specification, operating under normal conditions. These phenomenon are brushed under sensor noise tag in a behaviour-specification approach. Sensor noise itself being a nuisance that needs separate modelling, as it even sometime shows a flow of water even when the valve is closed. Behaviour based approaches do not need to model these artefacts separately. Furthermore they can fit a more tighter bound around the operational behaviour of the system e.g. we observe that under the normal behaviour the level of the tank does not go above a little over 900 litres, whereas a behaviour-specification based approach would consider an upper bound of 1100 litres as stated in the specification and encoded into the PLC coding.

In this paper, in contrast to the dictionary, and behaviour-specification based approaches, we model the physical process of SWaT through a data driven approach to

detect attacks at the physical layer. Contrary to their usual critique, we show that these approaches do not suffer from a high FP rate, in fact, few of the best performing techniques generate no or very few FP with high precision and recall. In particular, we evaluate and compare the performance of nine supervised machine learning (ML) classifiers on data generated by SWaT injected with eighteen attacks of ten different types. We further compare the time to detect an attack, type of attacks detected by each approach, and time to build the model. We demonstrate that these ML approaches not only successfully detect almost all the attacks, but also detect them earlier than the behaviour-specification approaches.

The rest of the paper is organised as follows: Section 2 describes the SWaT testbed and the process that we are trying to model, followed by related work in Section 3. We describe the nine classifiers in Section 4. Dataset details and results are presented in Section 5, followed by conclusion in Section 6.

2. SWaT Testbed

Figure 1. A View of The SWaT Testbed.

Secure Water Treatment (SWaT) is a testbed that is jointly setup by the Ministry of Defence, National Research Foundation, Singapore and Singapore University of Technology and Design (SUTD). Its purpose is to enable experimentally validated research in the design of secure and safe CPS. It replicates the physical and control components of a real modern water treatment facility, see figure 1. SWaT consists of the following six-stage filtration process:

P1 Supply and storage
P2 Pre-treatment
P3 Ultrafiltration and backwash
P4 De-Chlorination System

P5 Reverse Osmosis (RO)
P6 RO Permeate Transfer, UF Backwash and Cleaning

The process begins by taking in raw water (P1), adding necessary chemicals to it (P2), followed by filtration via an Ultra-filtration (UF) system (P3), de-chlorinating it using UV lamps (P4), and then feeding it to a Reverse Osmosis (RO) system (P5). A backwash process (P6) cleans the membranes in UF using the water produced by RO, and transfers the clean water back to the raw water tank. All the process are interdependent, with each process dependent on the previous one. The cyber portion of SWaT consists of a layered communications network, Programmable Logic Controllers (PLCs), Human Machine Interfaces (HMIs), Supervisory Control and Data Acquisition (SCADA) workstation, and a Historian. Data from sensors is available to the SCADA system and recorded by the Historian for subsequent analysis. A separate PLC is dedicated for each of the six processes. Each of these PLCs is provided with a redundant hot-standby PLC.

The communication in the SWaT takes place over multi-layer communication links, consisting of different switches and routers. This communication can take place trough either a WiFi, or an Ethernet link running various industrial communication protocols. The sensor readings, and actuator commands are all communicated with the PLC over these (wired or wireless) communication links. These protocols suffer from network attacks. In our lab we have successfully injected Man In The Middle (MITM) attacks, which can inject false sensor readings, and commands.

2.1. Pre-Treatment Process

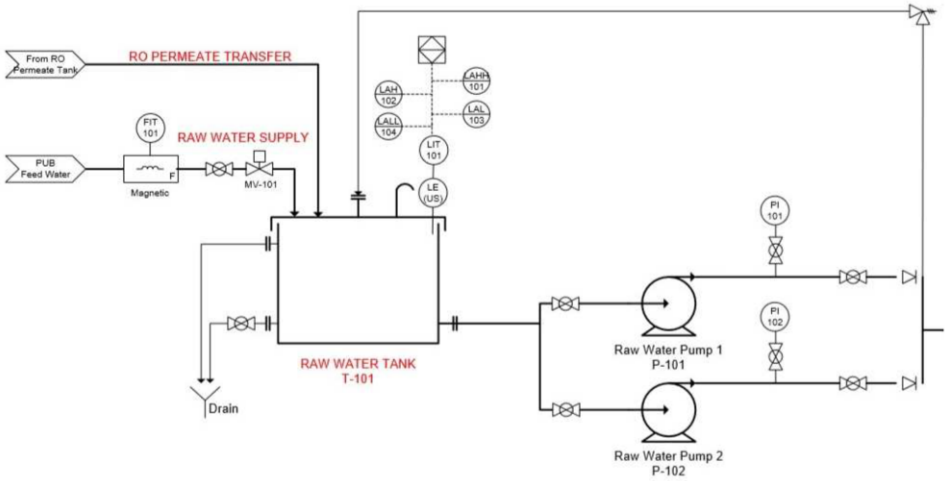

Figure 2. Details of the first stage of SWaT.

SWaT is a complex system, with six different processes interacting with each other. In this paper we try to model the intrusions in the process P1 only. Findings for this

process will be extended to rest of the processes by either extending the current model to incorporate one subsequent process at a time, or by learning a separate model for each of the process.

Process control of the P1 is illustrated in figure 2. The water flows into the raw water tank from two sources, city water supply system, and RO process (P6) after cleaning. The water from the water supply system is controlled by motor valve MV-101. It is opened when the water level goes below a predefined Low (L) threshold, and is closed when it reaches a predefined High (H) threshold. If for some reason, the water level still keeps on decreasing (or increasing) past the L (or H) threshold, then alarms are set off when it reaches the Low Low (or High High) threshold, signalling an emergency situation in which the system can be thought of as insecure. The rate of this inflow is measured by the flow indicator FIT-101. The water inflow from the RO process can not be controlled by P1, therefore at a given point in time water could be flowing into the tank from none, one, or both of the sources. Pump P-101 is turned on when the water level drops below L threshold in the UF tank (P3), and is turned off when water level rises above the H threshold in the UF tank. It can also be turned off when the level in the raw water tank drops below L, or the flow indicator FIT-201 (in process P2) drops below a certain predefined threshold. Pump P-102 is only for back up, it only operates when P-101 fails to do so.

Even though process P1 is not the most financially critical process of the testbed, but all the later process depend on it. A false sensor value for LIT-101 can overflow the tank, decrease the output of the plant, or burn out the pumps. In section 5 we define ten different attacks that are injected into the system.

3. Related Work

In literature, IDS have been categorized into three categories; knowledge based, behaviour-specification based, and behaviour based [14]. Knowledge based approaches look for specific runtime patterns that match a pre-maintained dictionary of pattern of known attack types [8]. These approaches are fast and have low FP rate, but require the dictionary of attacks to be kept upto date. They are totally blind to zero day attacks. Behaviour-specification approaches model the normal behaviour of the system by formally defining the legitimate behaviour, and signal an attack when the system departs from this model [15,10]. These approaches require a human expert to define this proper system behaviour, which is very time consuming. Secondly the systems behaviour may change because of environmental changes, or with ageing, thus departing from the behaviour that was previously thought as normal. Behaviour based approaches address these issues but suffer from high FP rate, but our study shows otherwise. Some behaviour based approaches use conventional statistics based techniques [25], they test if a sensor reading or an actuator setting is within some number of standard deviations of a mean. These approaches are parametric, not fully automatic, and difficult to come up for large number of interdependent sensor and actuators. We therefore limit our focus to non parametric approaches such as data mining and ML approaches.

Most of IDS model the network traffic [26,2]. Only few try to detect attacks at physical layer, and these approaches are behaviour-specification approaches [4]. Out of the forty IDS systems reviewed, only twenty four were behaviour based. More than half of

these IDS detect intrusions in non CPS systems, using internet and LAN traffic [24,11]. Other approaches try to detect attacks in the network traffic layers of CPS that use of the shelf industrial network protocols such as MODBUS, DNP3, Ethernet/IP, etc. [13]. We were not able to find an IDS that models only the physical process of a CPS using a ML approach. The closest work that we could find was done on the water storage testbed of Mississippi State University [16]. It is a simple testbed consisting a 2 litre water storage tank, a level sensor, and a pump. [7] use NN to detect attacks on this testbed with basically one network traffic attribute and one physical process attribute, namely, number of responses against a command, and water level. The only physical attacks that they simulate are the change in water level. [17] uses the same testbed but adopts a one class approach. They add network traffic features to increase the attribute set, thus rendering their approach to mostly a network traffic IDS. Whereas we use a two class approach due to the availability of attack data on SWaT, and we inject ten different attacks in the physical process. Lastly, our testbed is much more complex and realistic which validates our findings more.

4. Methodology

A data driven approach for CPS should be fast, scalable, and robust. It should be fast because the system operates in real time, it wouldn't be beneficial to signal an attack after it has happened. It should be scalable because CPS can be huge systems involving multiple readings per second from hundreds of sensors and actuators. It should be robust to sensor noise, and to the effects of weather and ageing. Therefore we choose to model this behaviour with the state of the art supervised Machine Learning (ML) techniques. Supervised techniques require training data under both the normal operating circumstances, and when under attack. It is rare to find the data when system is under attack, because these expensive and infrastructure critical plants are not available to researchers to carry out attacks. We on the other hand can generate the data under the attack scenario from the testbed. Even then, IDS problem is a skewed class problem. In all, we injected ten different types of attacks in the system. These attacks henceforth referred to as A1, A2,.., A10 are described below:

A1 FIT-101 is attacked and its value is changed to zero.
A2 MV-101 is attacked and is turned off.
A3 P-102 is attacked and is opened.
A4 P-101 is attacked and is closed down.
A5 MV-101 and P601 are closed down.
A6 FIT-101 is attacked by showing it operating value above its normal operating range.
A7 LIT-101 is attacked by increasing its value above the normal maximum level.
A8 LIT-101 is attacked by decreasing its value below the normal minimum level.
A9 LIT-101 is attacked by increasing its value more than normal rate of change.
A10 LIT-101 is attacked by decreasing its value more than normal rate of change.

The nine state of the art ML classifiers that we choose are fast and scalable, but some what sensitive to noise. We tackle this by reserving some part of the data for testing over which we tune the parameters of the algorithms to find the best performance parameters.

We choose three most widely used discriminative classifiers, namely, Support Vector Machines (SVM), Neural Network (NN), and Logistic Regression (LR). Three classifiers are decision tree based, namely, Random Forest (RF), J48, and Best First Tree. The remaining three are statistical classifiers, namely, Naive Bayes (NB), Bayes Network (BayesNet), and Bayes Logistic Regression (BayesLR). We briefly discuss each of these classifiers below.

4.1. Discriminative Approaches

Support Vector Machines (SVM) are non-probabilistic binary linear classifiers. They project the data in a higher dimensional feature space where a hyperplane is learned to discriminate between the points of the two classes. The hyperplane is such that it maximizes the margin between the closest points of the two classes, thus achieving a better generalization over the unseen data. This has led SVM to exhibit high performance on most of the real world classification problems including IDS [1].

Artificial Neural Networks (NNs) are a family of models inspired by biological neural networks and are used to approximate functions that can depend on a large number of inputs. In addition to the input and output layer, they consist of one or more hidden layer of neurons that try to learn non linear decision boundaries separating the classes. Along with SVM's, NN belongs to few of the most successful algorithms for intrusion detection [2].

Logistic Regression (LR) is an alternative to linear discriminant analysis, and can be seen as analogous to linear regression. However it is based on quite different assumptions. Firstly, the conditional distribution is a Bernoulli distribution rather than a Gaussian distribution. Secondly, the predicted values are probabilities through a logistic function. It nicely measures the relationship between the categorical dependent variable and one or more independent variables. It can outperform SVM and NN on problems with large number of features, it has been less successful on intrusion detection problems though [23].

4.2. Decision Tree Based Approaches

Decision tree have been a popular approach to ML since early nineties, mainly because they can be translated to simple IF-ELSE, OR and AND rules which are intuitive and easily understandable. They have been successfully applied to detect intrusion detection in network traffic layer [20,11] in recent years. A decision tree can be visualized as a tree structure that consist of nodes from root to leaf, in which each internal node denotes a test on an attribute, each branch represents the outcome of a test, and each leaf node represents a class label. A new record is traversed through the root node till the leaf, where it is assigned the label of the leaf node. The selection of attributes is based on some information theoretic measure. J48 uses Information Gain as the measure to select attributes at each node of the tree [19]. The tree is pruned to avoid overfitting. Best First Tree (BFTree) is similar to J48 [21], except that the best node is expanded first instead of the depth-first order. This approach allows new ways to prune the tree which make it less susceptible to overfitting.

Random forests (RF) are an ensemble approach proposed by [3]. They are based on notion of bootstrapping and selection of random subset of features i.e. they operate

by constructing a multitude of decision trees at training time on a random subset of the training examples. At each candidate split in the learning process, a random subset of the features is selected. The mode of the classes of the individual trees is the predicted class label for a record. RF have been recently perform very well on many problems. RF have been shown to be more robust to overfitting than other decision tree algorithms.

4.3. Bayesian Approaches

Bayesian classifiers are one of the most popular approaches in pattern recognition. They are probabilistic classifiers that predict class by means of probabilities, such as the probability that a given sample belongs to a particular class or not. Bayesian Networks (BayesNet) and Naive Bayes (NB) variants are the two most fundamental methods that belong to these class of classifiers. They have been successfully used for intrusion detection as well [12,24]. NB classifiers assume conditional independence given the value of the class. i.e. the attributes do not influence each other given the value of the class attribute. NB classifiers are highly scalable, requiring a number of parameters linear in the number of features. They have been shown to perform nicely on large dimensional problems, and in scenarios where part of the data is unobservable. They can be learned through Maximum-likelihood method that does not require multiple iterations over the training data. BayesNet are probabilistic graphical model that represents a set of random variables and their conditional dependencies via a directed acyclic graph [6]. Unlike NB, dependencies can be laid out between the variables by adding an edge between them. Since attributes are not completely independent in the real world, they tend to outperform NB classifier when number of features are less. BayesLR is a Bayesian approach to LR that uses a Laplace prior to avoid overfitting, and is shown to generate better results than LR especially for large feature spaces [9]. Bayesian classifiers are commonly used classification algorithms because of their simplicity, computational efficiency and good performance for real-world problems.

4.4. Optimal Parameters

In this subsection we report the parameters for which the classifiers give the best performance. For SVM, we tried out several parameters, including the kernel and the regularization parameter C. Performance for linear kernel did not exceed 95% despite choosing large values of C. SVM gave best performance using Radial Basis Function (RBF) kernel with $C = 8000$. NN gave best results with seven hidden units in the hidden layer, two units in the output layer, learning rate, momentum, and number of iterations equal to 0.3, 0.2, and 1000, respectively. LR worked best for the default parameters. RF and J48, gave the best performance for 200 trees, and confidence factor of 0.6, respectively. BFTree worked best without pruning, and minimum number of objects set at 4. NB does not have parameters, while BayesLR worked best for the default parameter values. For BayesNet, we used Simple Estimator, and TAN as the search algorithm with Entropy as the Score Type. Most of the algorithms have a few more parameters for which we tried out different values, but we do not mention them here because their best value turned out to be the default values.

	No. of Attacks	Attack	Normal	Total
Training Data	18	1260	18900	20160
Test Data	18	580	8060	8640

Table 1. Details of the dataset.

5. Dataset and Results

5.1. Dataset

Each second, the state of the sensors and actuators is recorded into a Historian database. For this study, we extracted eight hours of data from the database which totals around 28800 records. Since we are using supervised ML approaches, we divided this data into two sets, training and test set. We learn the model on the training set and then evaluate its performance on the test set. We tune the parameters of each algorithm on this test set to obtain their best performance. The details of these sets are given in table 1.

Since the probability of attack in real life is small, so in order to simulate a real scenario we have injected only few attacks into the system. We have injected 10 different attacks based on the sensor and actuator values. The first eight attacks have two instances each, while each of the last two attacks have one instance only. In the training set the duration of the first and second attack is kept at 30 and 120 seconds, respectively. In reality, the duration of attacks in the test set may be different from that of the training data, therefore we also keep it different in test data. In test data, the first attack lasts 10 seconds while the second attack lasts only 30 seconds. This totals to 18 attacks in the training and test set, corresponding to 1260 and 580 rows, respectively.

An important phenomenon observed in the data that is absent in the specification is that the turning on and off of valves and pumps is not immediate. It can take upto a dozen seconds for the valve to open, during which the state of the pump is neither open nor closed, instead its in a transient state. Furthermore, the water inflow takes similar amount of time to reach its normal or desired rate of flow. This makes the problem of detecting the attacks more challenging because now the valve is not closed and yet water is not pouring in. This problem is further aggravated by the sensor noise, e.g. the valve is closed but the value of the water inflow is greater than zero. Similarly the output pump is closed but the level of the tank still decreases. These issues can result in the degradation of the performance of the IDS, but we choose not to model them explicitly. We do this for two reasons, it would further complicate the model, and secondly since no modelling is perfect, therefore it may add some new artefact in the model. Instead we try to tackle these issues directly in the learning processing through regularization.

5.2. Metrics

Being a skewed class problem, accuracy is not a sufficient metric to measure the measure the performance of the IDS. Precision and recall are more suitable measures for this class of problems. Precision is basically a measure of how accurate the classifier is when it says that it has detected an attack. A higher value corresponds to a lesser number of false alarms (FP). Whereas, recall is basically how many of the attacks did the classifier actually detect. Higher value of Recall corresponds to a lesser number of missed attacks (FN). Ideally we want a classifier with high precision and recall because that corresponds

to lower FP and FN. F-Measure helps us to combine both into a single metric. It is the harmonic mean of precision and recall. It is a more conservative measure than the arithmetic mean of the two. The three measures are mathematically defined below:

$$Precision = \frac{TP}{TP+FP}$$

$$Recall = \frac{TP}{TP+FN}$$

$$F-Measure = \frac{2*Precision*Recall}{Precision+Recall}$$

where TP is the number of attacks correctly detected by the classifier. Fifth measure that we use is the Area Under the Receiver Operating Characteristic (ROC) curve (AUC). It is considered to be a more robust measure than F-Measure for highly skewed problems [18]. ROC curve is true positive rate (TPR) plotted against false positive rate (FPR) thresholded at various settings. This allows to select possibly optimal models by varying this threshold.

5.3. Pre-Processing

We treat every attribute as a numeric attribute, even the states of valve and pumps. Reason being that they are not a binary variables as they take on three values, where value 2 corresponds to the on state, 1 for the off state, and 0 for the transient state. We normalize all the attributes in the interval between zero and one. This is very important for certain algorithms e.g. normalization helped reduce SVM's training time from hours to just a few seconds, in some cases SVM also crashed on the un-normalize data. Lastly we introduced a new attribute in the data set as change in tank level, hoping that it would help us detect attacks that incur sudden changes in tank level. We will see in results that, it helps us to detect the sudden increase in the tank level, but not the sudden decrease in tank level. We calculate this attribute by subtracting the old reading from the current reading of the tank level.

5.4. Results

The results demonstrate the effectiveness of ML based algorithms to detect attacks in the physical process of a CPS. We performed various runs of each algorithm to find their parameters that would give their best performance on the test dataset. We summarize these results in table 2. All the five performance measures are reported as percentage values. The last column, Time To Build the Model (TTBM) is reported in seconds. The highest value for each metric is highlighted in bold. The results carry a bit of a surprise. In this particular problem when number of features is significantly smaller than the number of training examples, LR, and SVM with linear kernel should perform best. To the contrary LR and SVM with a linear kernel are one of the worst performers. Instead SVM with a RBF kernel takes the top spot in the accuracy section. NN on the other hand was supposed to perform equally better as SVM.

Coming to the tree based algorithms, rarely used BFTree performance is quite high, sharing the top spot with RF on three measures, and being second in the accuracy mea-

	Accurracy	Precision	Recall	F-Measure	AUC	TTBM
SVM	**99.83**	99.65	97.76	98.69	98.38	6.78
NN	99.08	99.10	99.10	99.10	96.30	54.23
LR	94.52	93.90	94.50	93.10	90.00	0.38
RF	99.78	**99.80**	**99.80**	**99.80**	99.60	10.95
J48	98.91	99.00	98.90	98.90	98.50	0.8
BFTree	99.78	**99.80**	**99.80**	**99.80**	98.30	0.15
NB	94.13	94.50	94.10	91.90	84.70	**0.03**
BayesNet	99.07	99.10	99.10	99.10	**99.70**	0.09
BayesLR	94.09	94.40	94.10	91.90	56.00	0.38

Table 2. Performance of Classifiers. All values are in percentages, except TTBM which is in seconds.

sure. Nonetheless, its AUC is significantly lower than that of RF, which demonstrates the befitting discriminative power of the RF.

Strikingly, BayesNet has the highest AUC. This means that its decision function can be tweaked to achieve better performance than rest of the algorithms. Its training time is also the lowest amongst the high performing classifiers.

It is interesting to note that in attack A7 to A10 the bounds are calculated using the normal behaviour of the system and not from the specification manual of the system. This is important, and allows us to fix more tighter bounds on the operation of the system, e.g. the specifications state that the level of the tank cannot be more than 1100 litres, so an alarm goes off if this so happens, but from the data we observe that the level of the tank never goes above 903.04 litres because MV-101 is turned off when it reaches 900 litres. Using this insight our model learns a value little more than 903.04 as an abnormal behaviour and signals it as an attack. Same is the case for the minimum level of the tank. Deriving these upper and lower bounds from data enables us to detect an attack way before it reaches the critical stage. On the contrary, for A6, the data helps us to loosen the bound on operational range of FIT-101, which otherwise would have resulted into an false alarm. Reason being that according to data, FIT-101 can operate at a 11% higher value than that stated in the specification.

5.5. Fault Identification

We further drill down into the results, and analyse which classifier failed to detect which type of attacks. Table 3 summarizes the type of attacks detected by each classifier, we discard the classifier with accuracy lower than 95% from this analysis. None of the classifier has been successful in detecting all of the attacks. The most difficult to detect was attack A10. RF and BFTree that seemed to perform exactly same also differ on detecting A10. In A10 attacker significantly reduces the level of the tank but still within its upper and lower bounds, hence being classified as normal behaviour by most of the classifiers. A9 is quite similar to A10, the level is increased significantly as opposed to decreasing it as in case of A10, rest of the conditions are same. A reason for its better detection could be that the change in water level is negative in this case, still it needs further analysis. Second most difficult to detect attack is A4, completely missed by NN, thus costing it position in the top rankings.

In the test data, the attack lasts from 10 to 30 seconds, so it could be that an attack is not detected for its whole duration but for a few seconds only. We therefore further

	A1	A2	A3	A4	A5	A6	A7	A8	A9	A10
SVM	2	2	2	2	2	2	2	2	1	0
NN	2	2	2	0	2	2	2	2	1	1
RF	2	2	2	2	2	2	2	2	1	0
J48	2	1	2	2	2	2	2	2	1	0
BFTRee	2	2	2	2	2	2	2	2	1	1
BayesNet	2	2	2	1	2	2	2	2	0	1

Table 3. Number of Attacks Detected. Two instance of each attack upto A8. Only once instance for A9 and A10.

	A1	A2	A3	A4	A5	A6	A7	A8	A9	A10
SVM	100	95.71	100	100	100	100	100	100	100	0
NN	100	100	100	0	100	100	100	100	100	10
RF	100	87.14	100	100	100	100	100	100	100	0
J48	100	85.71	100	100	100	100	100	100	100	0
BFTree	100	87.14	100	100	100	100	100	100	100	10
BayesNet	100	100	97.14	85.71	100	100	100	97.14	0	10

Table 4. Percentage of Records Detected For Each Attack.

analyse this by looking at the percentage of records detected by each classifier in table 4. This table shows a slightly different picture than 3. It shows that despite getting detected, attack A2 has given almost all the classifiers a tough time. SVM, RF, BFTree, and J48 miss some part of this attack. For attack A10, the three algorithms that eventually detected it, were only able to detect only 10% of it.

This raises a question; how long does it take to detect an attack? Delay in detecting an attack can lead to a catastrophe. We try to answer this question through table 5. A value of 0 means that the attack was detected the first second that it started, whereas ∞ means that the attack went undetected. Most of the attacks get detected at the very instant they start, or do not get detected at all. Only three attacks are detected at the fourth second. This is a good feature of ML based approaches in contrast to the specification based approaches where the model waits for the output of the system to deviate from the normal behaviour to be able to detect them.

These results demonstrate the effectiveness of using ML algorithms for modelling of the physical layer. An ensemble approach of complementing algorithms to detect all the attacks can be used to further improve performance, a direction for future exploration.

5.6. Training Time

Attack detection should be in real time, so it is necessary to train the model and classify the incoming examples as quick as possible. We report Time To Build the Model (TTBM) in table 2. Given the number of training examples, most of the algorithms were quick to learn the model. As expected, NN is the slowest, but it classifies new examples very fast. So does SVM, it took less then half a second to classify all the testing examples. TTBM should be taken with a grain of salt because it is affected by other processes running on the system. The experiments were run on a 3 GHZ Core i7 laptop with 8 GB RAM. The implementations were not parallelised, so the running times are for algorithms running on a single core. Memory footprint for each of the algorithm was also quite

	A1	A2	A3	A4	A5	A6	A7	A8	A9	A10
SVM	0	0	0	0	0	0	0	0	0	∞
	0	0	0	0	0	0	0	0		
NN	0	0	0	∞	0	0	0	0	0	0
	0	0	0	∞	0	0	0	0		
RF	0	3	0	0	0	0	0	0	0	∞
	0	0	0	0	0	0	0	0		
J48	0	∞	0	0	0	0	0	0	0	∞
	0	0	0	0	0	0	0	0		
BFTree	0	3	0	0	0	0	0	0	0	3
	0	0	0	0	0	0	0	0		
BayesNet	0	0	0	∞	0	0	0	0	∞	0
	0	0	0	0	0	0	0	0		

Table 5. Time (in seconds) to Detect an Attack.

small. Low memory footprint and low running time indicate that these approaches are highly scalable, and would be appropriate for detecting intrusion in all of the six stages of the SWaT testbed.

6. Conclusion

CPS are critical to the infrastructure and economy of a country. Securing them from cyber attacks that may be carried out by criminals, hacktivists, disgruntled employees, and enemy states is a high priority area for many governments. Even though many intrusion detection system (IDS) exist for the network layer, no further defences exist ones it has been breached. We therefore propose a behaviour based machine learning (ML) approach for intrusion detection that models the physical process of the CPS to detect any anomalous behaviour or attack that may try to change the behaviour of the CPS. We successfully demonstrate the effectiveness of such approaches by validating results of nine ML approaches on Secure Water and Treatment (SWaT) testbed, a complete replicate of the physical and control components of a real modern water treatment facility. We show that ML approaches are not only fast, robust, but detect attacks very early while exhibiting no or very little false positives. Furthermore our approach can be used in conjunction with the existing network traffic IDS systems.

Acknowledgment

This work was supported by the National Research Foundation (NRF), Prime Ministers Office, Singapore, under its National Cybersecurity R&D Programme (Award No. NRF2014NCR-NCR001-40) and administered by the National Cybersecurity R&D Directorate.

References

[1] Iftikhar Ahmad, Muhammad Hussain, Abdullah Alghamdi, and Abdulhameed Alelaiwi. Enhancing svm performance in intrusion detection using optimal feature subset selection based on genetic principal components. *Neural Computing and Applications*, 24(7-8):1671–1682, 2014.

[2] Omar Al-Jarrah and Ahmad Arafat. Network intrusion detection system using neural network classification of attack behavior. *Journal of Advances in Information Technology Vol*, 6(1), 2015.

[3] Leo Breiman. Random forests. *Machine learning*, 45(1):5–32, 2001.

[4] Alvaro A Cárdenas, Saurabh Amin, Zong-Syun Lin, Yu-Lun Huang, Chi-Yen Huang, and Shankar Sastry. Attacks against process control systems: risk assessment, detection, and response. In *Proceedings of the 6th ACM symposium on information, computer and communications security*, pages 355–366. ACM, 2011.

[5] Thomas M Chen and Saeed Abu-Nimeh. Lessons from stuxnet. *Computer*, 44(4):91–93, 2011.

[6] Nir Friedman, Dan Geiger, and Moises Goldszmidt. Bayesian network classifiers. *Machine learning*, 29(2-3):131–163, 1997.

[7] Wei Gao, Thomas Morris, Bradley Reaves, and Drew Richey. On scada control system command and response injection and intrusion detection. In *Proceedings of the 5th Annual Anti-Phishing Working Group eCrime Researchers Summit (eCrime)*, pages 1–9. IEEE, 2010.

[8] Wei Gao and Thomas H Morris. On cyber attacks and signature based intrusion detection for modbus based industrial control systems. *Journal of Digital Forensics, Security and Law*, 9(1):37–56, 2014.

[9] Alexander Genkin, David D Lewis, and David Madigan. Large-scale bayesian logistic regression for text categorization. *Technometrics*, 49(3):291–304, 2007.

[10] Dina Hadžiosmanović, Robin Sommer, Emmanuele Zambon, and Pieter H Hartel. Through the eye of the plc: semantic security monitoring for industrial processes. In *Proceedings of the 30th Annual Computer Security Applications Conference*, pages 126–135. ACM, 2014.

[11] Md Al Mehedi Hasan, Mohammed Nasser, Biprodip Pal, and Shamim Ahmad. Support vector machine and random forest modeling for intrusion detection systems. *Journal of Intelligent Learning Systems and Applications*, 2014, 2014.

[12] Levent Koc, Thomas A Mazzuchi, and Shahram Sarkani. A network intrusion detection system based on a hidden naïve bayes multiclass classifier. *Expert Systems with Applications*, 39(18):13492–13500, 2012.

[13] Hui Lin, Adam Slagell, Catello Di Martino, Zbigniew Kalbarczyk, and Ravishankar K Iyer. Adapting bro into scada: building a specification-based intrusion detection system for the dnp3 protocol. In *Proceedings of the Eighth Annual Cyber Security and Information Intelligence Research Workshop*, page 5. ACM, 2013.

[14] Robert Mitchell and Ing-Ray Chen. A survey of intrusion detection techniques for cyber-physical systems. *ACM Computing Surveys (CSUR)*, 46(4):55, 2014.

[15] Robert Mitchell and Ing-Ray Chen. Behavior rule specification-based intrusion detection for safety critical medical cyber physical systems. *Dependable and Secure Computing, IEEE Transactions on*, 12(1):16–30, 2015.

[16] Thomas Morris, Anurag Srivastava, Bradley Reaves, Wei Gao, Kalyan Pavurapu, and Ram Reddi. A control system testbed to validate critical infrastructure protection concepts. *International Journal of Critical Infrastructure Protection*, 4(2):88–103, 2011.

[17] Patric Nader, Paul Honeine, and Pierre Beauseroy. Mahalanobis-based one-class classification. In *Machine Learning for Signal Processing (MLSP), 2014 IEEE International Workshop on*, pages 1–6, 2014.

[18] David Martin Powers. Evaluation: from precision, recall and f-measure to roc, informedness, markedness and correlation. 2011.

[19] J. Ross Quinlan. Improved use of continuous attributes in c4. 5. *Journal of artificial intelligence research*, pages 77–90, 1996.

[20] Shailendra Sahu and BM Mehtre. Network intrusion detection system using j48 decision tree. In *Advances in Computing, Communications and Informatics (ICACCI), 2015 International Conference on*, pages 2023–2026. IEEE, 2015.

[21] Haijian Shi. *Best-first decision tree learning*. PhD thesis, Citeseer, 2007.

[22] Jill Slay and Michael Miller. *Lessons learned from the maroochy water breach*. Springer, 2008.

[23] Chih-Fong Tsai, Yu-Feng Hsu, Chia-Ying Lin, and Wei-Yang Lin. Intrusion detection by machine learning: A review. *Expert Systems with Applications*, 36(10):11994–12000, 2009.

[24] Liyuan Xiao, Yetian Chen, and Carl K Chang. Bayesian model averaging of bayesian network classifiers for intrusion detection. In *Computer Software and Applications Conference Workshops (COMPSACW), 2014 IEEE 38th International*, pages 128–133. IEEE, 2014.

[25] Nong Ye, Syed Masum Emran, Qiang Chen, and Sean Vilbert. Multivariate statistical analysis of audit trails for host-based intrusion detection. *Computers, IEEE Transactions on*, 51(7):810–820, 2002.

[26] Yichi Zhang, Lingfeng Wang, Weiqing Sun, Robert C Green, Mansoor Alam, et al. Distributed intrusion detection system in a multi-layer network architecture of smart grids. *Smart Grid, IEEE Transactions on*, 2(4):796–808, 2011.