

Fingerprinting for Cyber-Physical System Security:

Device Physics Matters Too

Qinchen Gu, David Formby, and Shouling Ji | Georgia Institute of Technology

Hasan Cam | US Army Research Laboratory

Raheem Beyah | Georgia Institute of Technology

Due to the increasing attacks against cyber-physical systems, it is important to develop novel solutions to secure these critical systems. System security can be improved by using the physics of process actuators (that is, devices). Device physics can be used to generate device fingerprints to increase the integrity of responses from process actuators.

The field of cyber-physical systems, or CPSs, is growing rapidly. In recent years, a variety of CPS applications in different domains have flourished. For example, 80 million smart home devices were delivered worldwide in 2016, a 64 percent increase from 2015.¹ The global industrial control system (ICS) market, one of the most important areas in CPSs, was valued at 58 billion US dollars in 2014 and expected to be worth 81 billion US dollars by 2021, growing at an annual rate of 4.9 percent.² Meanwhile, attacks targeting CPSs have become more frequent as well. In March 2000, a former contractor of Maroochy Water Services took control of 150 sewage pumping stations using a laptop computer and a radio transmitter. This was not discovered until an engineer examined every signal passing through the system, by which time one million liters of untreated sewage had been released into a stormwater drain.³ More recently, malware specially crafted to attack the Ukrainian electric utility caused a blackout in a portion of its capital equivalent to a fifth of its total power capacity.⁴ The most significant concern regarding attacks targeting CPSs on which we depend is that they pose a threat not only to the equipment in the CPSs themselves, but also to the physical world in which we live. This very

threat calls for innovative and effective techniques to be developed. With the growing number of threats in the space, it is clear that novel solutions are required. One clear approach to improving the security of CPSs is to rely on process physics as a defensive side channel. In fact, C. McParland and colleagues make the case for the need to leverage process physics to secure CPSs.⁵ While we agree that process physics should be used to secure CPSs, we argue that physics of the devices also matters. Note that for the rest of this article, “device” refers to the “actuator” in the CPSs.

Identifying Threats in Cyber-Physical Systems

Common components in CPSs are the process, actuator(s), sensor(s), and controller(s). Figure 1 shows the interconnections among these elements using an air conditioning system as an example. Despite various attack vectors to penetrate the system and take control of one or more components, it would be a fair assumption that a potential goal for an attacker is to drive the physical plant to an unsafe state. Attacking the actuator(s), sensor(s), or controller(s) serves to achieve this goal. One way to do this is spoofing control

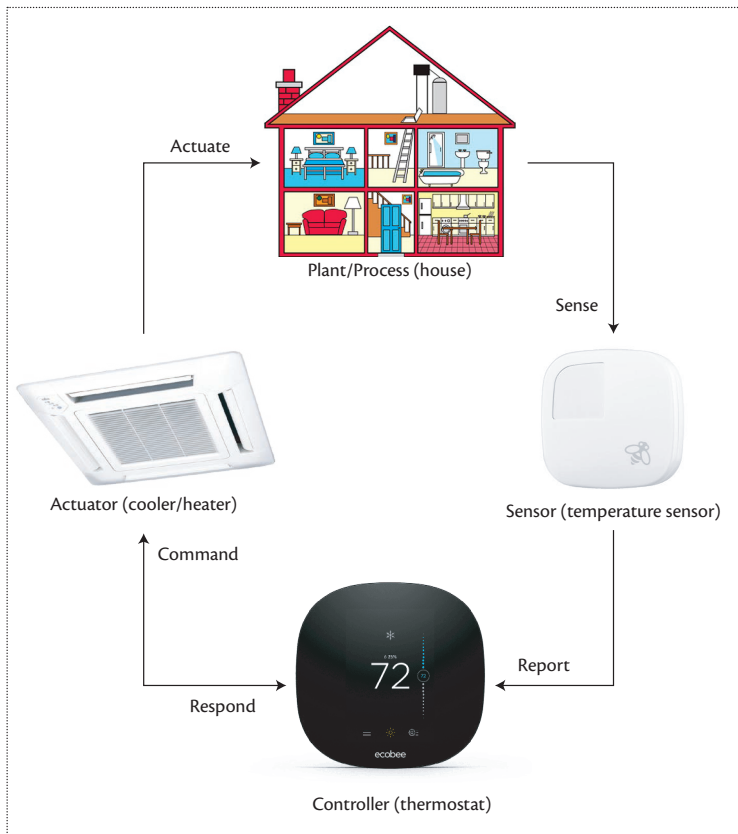


Figure 1. Interconnection between different elements in CPS.

commands to the actuators or sending incorrect sensor values to the controller. Because defending against direct physical access to the plant falls outside the realm of the cyber world, we limit the scope of our discussion to securing the other three components.

The most well-known attack that targeted the controller is Stuxnet.⁶ By reprogramming the programmable logic controllers (PLCs), Stuxnet was able to operate the PLCs according to the attacker's intention. More specifically, Stuxnet spreads itself using traditional network infrastructure as well as removable storage media to reach the targeted computer. It then modifies the code on the PLC, causing it to send out commands that drive the centrifuges fast enough to tear themselves apart. It is worth noting that the cost of such an attack is high: four 0-day flaws were exploited and two digital certificates were compromised. In addition, part of the path Stuxnet took was through air-gapped networks, because the key computer was unlikely to have outbound Internet access. Clearly, many defensive techniques used in the traditional computer network domain should also be used in such environments where security is highly demanded. However, as we explained later, not all network security tools (for example, virtual private network [VPN], encryption) can be easily applied in the CPS

environment due to the insufficient computational and memory capabilities of a large portion of legacy devices present in CPSs. Also, note that as illustrated in the Stuxnet example, even though air gapping physically isolates a protected network from insecure networks, it does not ensure 100 percent security, as attackers may use flash storage and other media to circumvent this. In fact, attackers may come from the inside, which renders any defensive means against outside attackers useless.

Compared to the controllers, actuators and sensors are more tightly coupled with physical plants, as they serve as the direct inputs and outputs, respectively, of the physical process. Attacking sensor devices can be carried out in two ways: integrity attacks and denial-of-service (DoS) attacks.⁷ In the former case, the attacker can inject any sensor value so the controller receives false non-zero values, performing a false data injection attack. In the latter case, the attacker simply cuts off the communication link between the sensor and the controller. Attacking the actuators may be carried out by compromising the actuator and impersonating it so the attacker can send out false reports back to the controller (for example, the breaker or valve has opened). Because the actions of an actuator apply directly to the underlying plant and thus may potentially cause the fastest and most effective damage, it is clearly a critical point to defend in a control system. Thus, our work aims to secure the CPS at the individual actuator level, by leveraging the fingerprints generated by the physical devices. The fingerprints can be used to authenticate the actuators, ensuring that these responses are not spoofed by attackers.

To summarize the problem, assume the global set of all devices in CPSs consisting of devices of various models and configurations. Given a finite-time observation of any device in a network, the goal of our device physics-based fingerprint method is to identify which model and configuration the observation corresponds to.

Formulating Our Device Physics-Based Approach

Based on the existing studies in CPS security, which extend from the traditional IT networks and physics modeling techniques, we develop our novel device physics-based approach. In this section, we briefly explain the existing solutions for protecting CPSs before introducing our approach.

Existing CPS Security Research

Many existing solutions for protecting CPSs can be divided into two categories:

- examining the network traffic to look for abnormal packets similar to what a traditional intrusion detection system (IDS) does,⁸ and

- modeling the system behavior and comparing the values output from the model's sensors with those from the real-world sensors, which leverages the knowledge about the system specifications, thus seeking to detect potential hazardous states.^{5,7,9}

For the first category, it is possible to provide a level of security of the control system's network by treating it as an instance of IT networks and applying mature secure access technologies (for example, VPN, firewall, and so on). C. Neilson proposed to secure the control system from cyberattacks with traditional IT solutions such as VPN.⁸ Although the author listed pros and cons of each solution, none took the physical system being controlled into account. This means that standard access control solutions are inadequate because they are not able to stop insider attackers from sending commands that can drive the CPSs into dangerous states, since they have already been granted access to the system. This network-based type of solution considers only the cyber domain and ignores the unique physical attributes, which may overlook important information about the physical state of the system. Such an approach may be the easiest to deploy and is generally agnostic to the specific communication protocols or details of the physical system.⁸ However, the amount of protection added by such technologies would be very limited in the ICS environment, which again is one of the biggest subgroups of CPSs. A high cost of deployment, poor support for legacy equipment, and rare software patching further aggravate the problem.¹⁰

Solutions in the second category attempt to incorporate knowledge specific to the CPSs. Some researchers propose to leverage physics of the system to solve this issue. A. Cárdenas and colleagues identified several challenges for the CPS security research community including new vulnerabilities, threats, and consequences of potential attacks on networked control systems, and proposed linear system models to detect such attacks.^{11,12} The authors showed that they were able to detect stealthy attacks that change the physical behavior of the targeted control system by incorporating knowledge of the physical system. D. Urbina and colleagues studied whether physics-based attack detection can limit the impact of stealthy attacks in ICS and showed that the impact of such attacks can be mitigated by the proper combination and configuration of detection schemes, including a stateful model of the physical system.⁹

A more recent work proposed a framework to monitor for physical constraint violations by leveraging specification-based intrusion detection. McParland and colleagues were able to leverage the model of the physical plant and check the model against its

corresponding physical limitations.⁵ In their paper, the authors demonstrated their approach with several scenarios, including a boiler with a heater to heat or cool the water depending on the on/off status of the heater. They created scripts to passively monitor the communication and track boiler behavior to alert upon out-of-range conditions. They did so by leveraging control theory to infer the transition states of the system model given the actions defined in the captured packets. This modeling process may require an understanding of the entire physical interactions between all actuators and the plant, as well as between the plant and all the sensors, in advance. Compared to the solutions that try to secure the CPSs by securing the network access to the system, process modeling-based solutions target the underlying physical process, thus increasing the likelihood of detecting an insider attack.

Our Device Physics-Based Fingerprinting Approach

In our previous work,¹³ we proposed fingerprinting CPS devices by measuring the network level actuator response time, which in turn is determined by the physical properties inherent to each individual device. More specifically, for an actuator as shown in Figure 2, we monitor the time it takes between the event that a command from the controller is being sent to the actuator (for example, a TCP packet containing the command to open a valve) and the event that the corresponding response is received by the controller (for example, a TCP packet containing the response from the actuator that confirms the valve is open). We will show that this time difference, called the *operation time*, is tightly coupled with the physical characteristics of the device, and we perform experiments to validate our hypothesis. The operation time for a certain actuator is one method to fingerprint the device. In contrast to the concept in the general network domain, where the fingerprint of a device is mostly related to hardware and software components and configurations, that of a CPS device can be additionally affected by the unique physical composition of the device (that is, device physics). Thus, timing features of the messages returned by an actuator related to the operation it performs can be used as the main attributes for fingerprinting the device.

Device-modeling techniques can be used in conjunction with the technique proposed in McParland and colleagues' process-modeling techniques (in "Monitoring Security of Networked Control Systems: It's the Physics"⁵) to better secure CPSs. While that work examines and monitors the physical system as a whole (that is, the physical process), we combine the information from both "cyber" and "physical" domains and monitor the signatures at the individual device level.

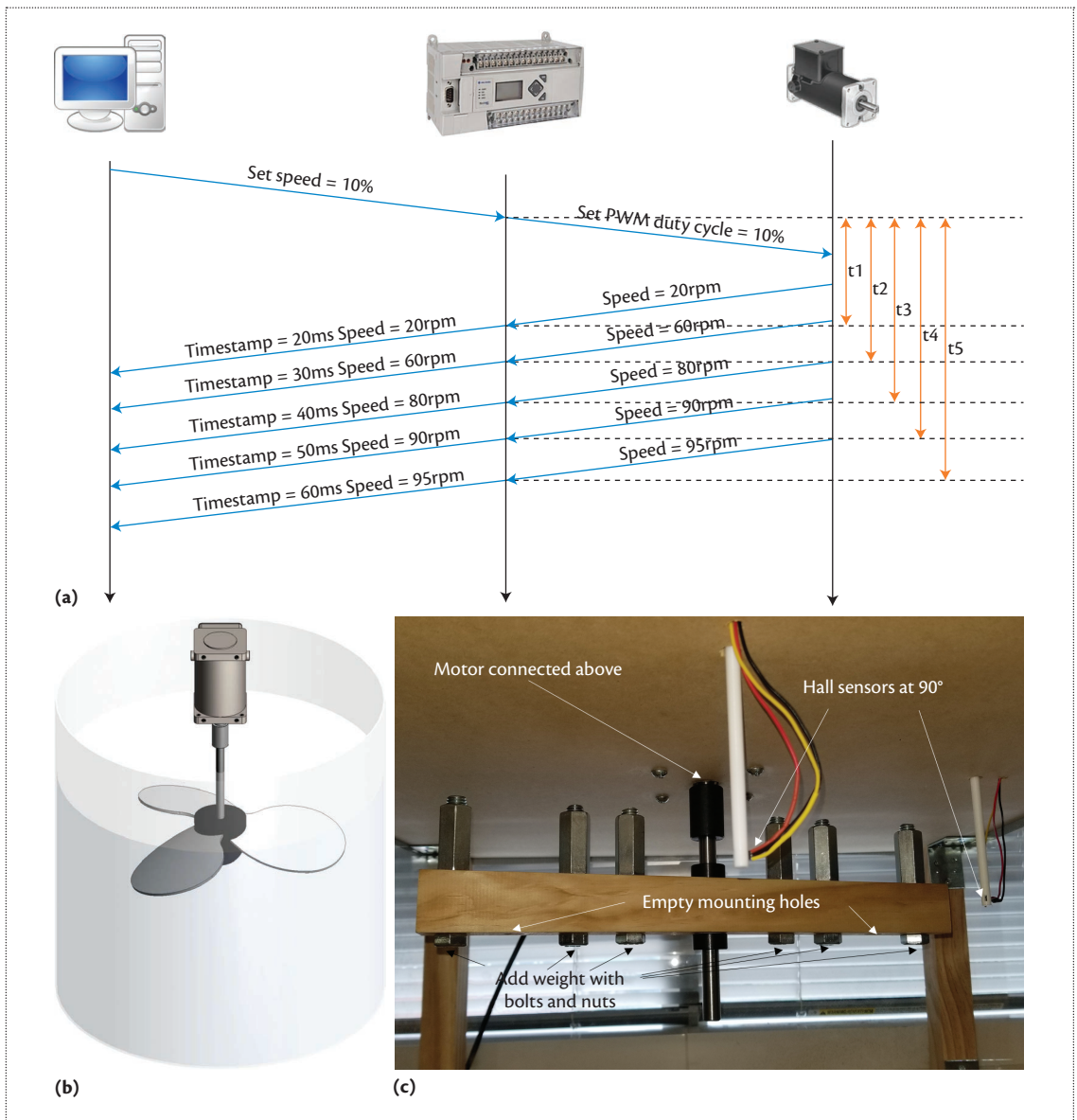


Figure 2. Device model used for the experiment. (a) Network packet timing diagram of the experiment setup. (b) Industrial mixer device that our model emulates. (c) Motor device showing sensor positions and load adjustment.

Because the behavior of an individual device is largely dependent on its own state and attributes, this approach requires less knowledge from the overall system dynamics, and thus is still useful when the system gets complicated and difficult to model. Furthermore, by combining information at the individual device level with that at the system level, it becomes much more difficult for an attacker to exploit vulnerabilities when device physics is also used to detect attacks. In addition to having to monitor and model the dynamics of the process, the attacker would need to forge actuator response times with strict timing and value constraints for each device in order to circumvent the detection mechanism.

Applying Our Approach

In earlier work,¹³ we proposed two device type fingerprinting methods designed to augment existing intrusion detection methods in the ICS environment. We used relay switches as an example and analyzed the fingerprints generated by the distribution of response times when issuing open/close commands from the PLC. We were able to predict the time it takes to open or close the latching relay from measured operation times as well as models of the device dynamics under normal conditions. We showed that there was a significant difference in operation time for the same action between the same types of devices from different vendors. Such

differences enabled us to accurately fingerprint the device's type. We also showed that forgery of the fingerprint can be detected.

Demonstration Scenario

In this article, we generalize the technique in our earlier work¹³ and argue that device physics can also be widely applied to CPS security. Specifically, we design an experiment with several controllable variables to test the robustness of our device physics-based fingerprinting technique. A high-level network diagram can be seen in Figure 2a, as a PC acts as a host device sending out commands to a PLC as well as receiving responses from the PLC regarding the status of the physical devices. The PLC then connects to actuators and sensors and executes the command on the corresponding devices. For the actuator, we pick a motor as it is an extensively used device in the ICS environment. It is also a high-value target for attackers because a motor usually outputs a much higher power than many other actuators such as valves and relays, thus may deal more damage to the surrounding equipment and personnel when sabotaged.

An example of such sabotage is faking the response of a request to reduce the revolutions per minute (RPM) of a motor, leading to a potentially unsafe scenario for individuals in the plant or the plant in general.

We choose to emulate the use case of a motor as in an industrial mixer shown in Figure 2b, as such equipment is widely used and usually comes in many variants. For example, we found a series of six heavy duty electric mixers from a company named INDCO,¹⁴ with horsepower ranging from 1/3 HP to 3 HP. The various horsepower ratings represent different possible options when choosing a motor at a specific point within a CPS, and may generate different fingerprints. At the end of the mixing shaft of the mixer is a blade that can propel the fluid being stirred. The fluid exerts a drag force on the blade, which slows it down. When the mixer is in use, the fluid being stirred usually consists of the same material, while the viscosity of the fluid can vary depending on its concentration and density (for example, different concentrations of maple syrup solution). Such differences also make a difference in the fingerprints generated by the mixer in response to control commands sent from the host. To emulate the mixer and quantify the behavior of the device from both the cyber and physical domains, we leverage a device model we built as shown in Figure 2c.

We set five different levels of power input to a single electric motor used in our experiment (to emulate variants with different horsepower), and 16 different load levels on the motor output (to emulate various fluid viscosity), giving a total of 80 different operating configurations for the motor. Each configuration generates a fingerprint that is later used to classify the corresponding configuration. In an industrial environment, the total number of configurations may vary. Thus, we also later discuss the effect of a different number of configurations on the performance of our fingerprinting method. The goal of the experiments is to show that a remotely observable fingerprint can be generated according to the physical attributes of an actuator to effectively identify the different configurations of the device (which naturally extends to identifying different devices).

Experiment Setup

For the communication protocol between the host PC and PLC, we choose Modbus as it is an open standard protocol widely used in ICSs, and is easier to implement compared to DNP3. The

Modbus protocol does not inherently contain timestamp information in its packet; therefore, we address this issue with a modification in the PLC ladder logic program and utilize the PLC to timestamp the command and response packets

to achieve real-time accuracy. The PLC acts as a Modbus slave and waits for read and write requests from the host PC running a Python script that acts as a Modbus master. The host sends commands to the PLC to set the operating speed of the motor and receives a series of response packets from the PLC containing the measurement of the angular velocity of the rotating load along with timestamps at which the measurement is taken, as depicted in Figure 2a. Note that the timestamps are measured in reference to the time when the PLC received the corresponding command. The operating speed can be adjusted in a range of values by changing the pulse-width modulation output of the motor driver connected in between the PLC and the motor. The motor spins a load that can be adjusted by adding or removing weights on it as shown in Figure 2c. The base of the load is a uniform lightweight wood bar with eight mounting holes positioned vertically at proportional distance to the center of spinning axis. The mounting holes are symmetrical to the spinning axis to keep the center of mass aligned with the center of rotation, thus minimizing

“A remotely observable fingerprint can be generated according to the physical attributes of an actuator to effectively identify the different configurations of the device.”

the rotating imbalance and the wobbling movement of rotating structures. A bolt and a coupling nut of known masses are mounted at each hole to adjust the moment of inertia of the overall rotating load. Two Hall effect sensors are placed near the circular track of the tips of the rotating bar; they produce signals when either of the two magnets attached to the tips of the load passes by. These signals are then picked up by the PLC connecting to the sensors, and used to send responses back to the host.

Extracting Features by Modeling the Physics of Device

To classify the fingerprints generated by the mixer under different configurations, features must be extracted from the raw sensor readings. Because each configuration differs only in its physical attributes, a straightforward source of features to be used for classification is the mathematical model that corresponds to the dynamics of the device. Thus, we derive a simple mathematical model of the device, starting with Newton's second law for rotation,

$$\tau - \hat{\tau} = I\alpha,$$

where τ is the torque exerted by the motor to the load, $\hat{\tau}$ is the frictional torque assumed to be constant relative to the angular velocity of the load, I is the overall moment of inertia of the load, and α is the angular acceleration of the load. For a DC motor like the one used in our experiment, the relationship between the torque it generates and the input voltage can be derived from equations,

$$i = \frac{E_s - E_o}{R},$$

$$P = E_o i, \text{ and} \\ E_o = ZnF/60,$$

where i is the current through the armature in the motor, E_s is the source DC voltage and E_o is the induced voltage in the armature conductors as they cut the magnetic field produced by the permanent magnet inside the motor, P is the mechanical power of the motor, n is the rotation speed in RPMs, and R , Z , and F are the armature resistance, winding coefficient, and flux per pole, respectively, and are constant regarding the specific motor build. Turning attention to the torque, we know that the mechanical power P is given by the expression

$$P = \tau n \times \frac{2\pi}{60}.$$

Thus, we have

$$\tau = \frac{ZF(E_s - ZnF/60)}{2\pi R}.$$

Now we consider the case where the load is initially at rest and accelerated by applying a time-invariant voltage E_s onto the motor. The angular velocity ω satisfies the equation

$$\omega(t) = \int \alpha dt = \int \frac{ZF(E_s - ZnF/60)}{2\pi RI} dt - \int \frac{\hat{\tau}}{I} dt$$

under the boundary condition that $\omega(0) = 0$. By solving the differential equation and substituting $n = \frac{30}{\pi}\omega$, we get an exponential decay equation

$$\omega = -Ae^{-t/B} + A,$$

where

$$A = \left(\frac{ZF}{2\pi R} E_s - \hat{\tau} \right) \frac{4\pi^2 R}{Z^2 F^2} \text{ and}$$

$$B = \frac{4\pi^2 R}{Z^2 F^2} I.$$

In our experiment, we measure all timestamped velocities at the time when a magnet passed by a sensor and calculate the instantaneous velocity based on the inverse of the time it takes since the last time a magnet passed by. Therefore, during an acceleration process, such calculated velocity is slightly lower than the actual value at the timestamp. Thus, we introduce a delay variable t_d into the equation

$$\omega = -Ae^{-\frac{t-t_d}{B}} + A. \quad (1)$$

Recall that the only two variables in the equation are the moment of inertia I and the supply voltage E_s , which determine how the angular velocity ω changes over time. We will show that the configuration of our mixer device consisting of these two variables can be inferred from the features extracted from the raw sensor readings in the response packets (that is, corresponding timestamps and angular velocity measurements). Because of the asymptotic nature of the curve described by Equation 1 and the noise introduced in actual measurements, as can be seen in Figure 3, it becomes infeasible to define an exact operation time as we did in our previous work.¹³ Instead of using a single event to mark the completion of an operation, we choose to characterize the timestamps in reference to the angular velocity measurements and obtain a trend of "operation times"

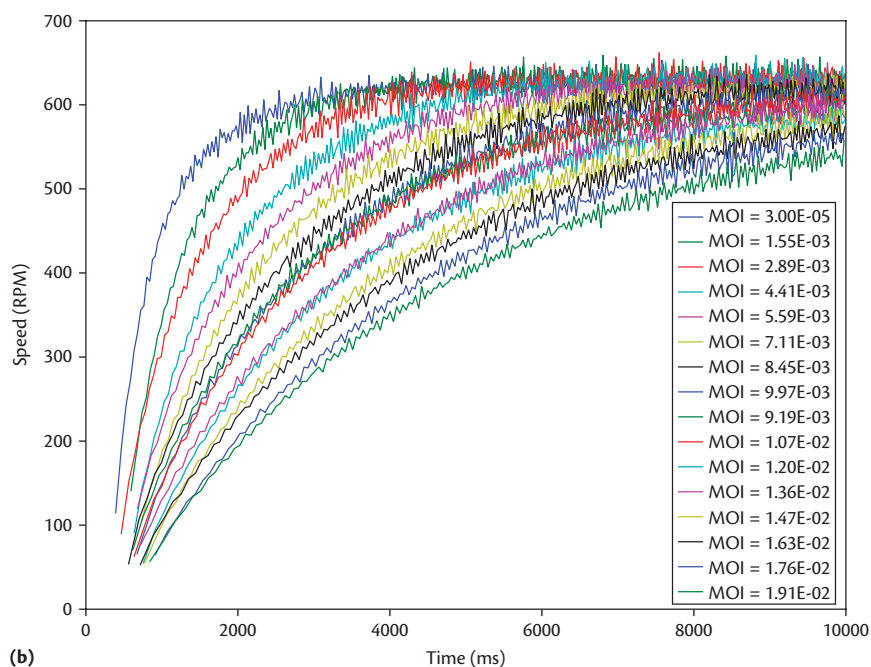
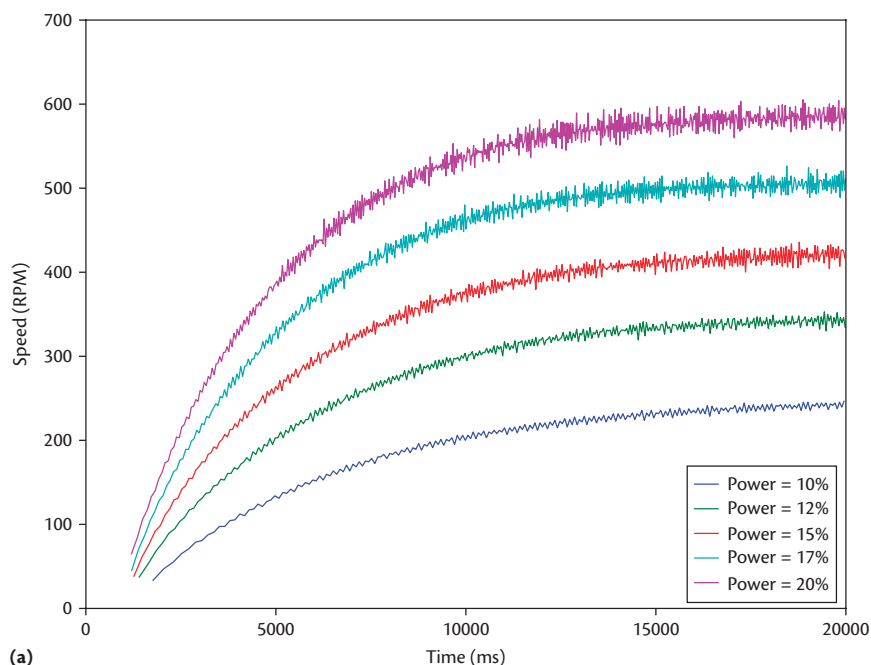


Figure 3. Plot of angular velocity over time under two different settings. MOI stands for moment of inertia and the numbers are in units of $kg \cdot m^2$. (a) Five power input settings with fixed load, and (b) 16 load settings with fixed power.

given by the series of timestamps generated in response to a command. Thus, by fitting Equation 1 to the measured timestamps and angular velocity values from the experiments, we can generate various features on which to classify, consisting of the coefficients in the equation.

Classifying Different Devices Based on Their Fingerprints

There are two stages in our approach, namely a training stage and a classification stage for fingerprinting each actuator. During the first stage, the operation time is

generated and stored in a database. In the classification stage, a series of measurements of the operation times is taken from the devices in the control system and compared against the database generated during the training stage. From the test results, conclusions can be drawn about whether the actuator as seen from network traffic corresponds to the actual configuration of the device or one of its variants.

To collect data needed for training and testing, we alter both the input power to the motor as well as the load driven by the motor and measure timestamped angular velocity measurement response under a total of 80 different configurations formed by the combination of the two variables. Recall that the various input power emulates variants of the same model of mixer with different horsepower, and various loads corresponding to different viscosities (that is, concentrations) of fluid being stirred by the motor-drive mixer. For each configuration, we perform 50 runs of measurement by accelerating the load from rest to a stable angular velocity. In our case, we heuristically set the capture time for each run to 30 seconds. Each measurement run generates a fingerprint associated with the physical characteristics dependent on the two variables. To get some intuition on the separability of these fingerprints generated by different configurations, we randomly pick one out of the 50 runs for each configuration, and plot the angular velocity measurements against corresponding timestamps (see Figure 3). For clarity, we show the measurements under two sets of configurations, namely varying power with fixed load—denoted as S_{power} —and varying load with fixed power—denoted as S_{load} . This shows that the same configuration can generate stable fingerprints distinguishable from different configurations.

We then step forward and build a classifier to quantitatively measure how well these fingerprints can be used to identify the configuration of a device (or, as a natural extension, a number of different devices). The features we use are extracted by fitting Equation 1 to the selected angular velocity/time measurements in each run (that is, the first 20 seconds of data after the command was sent) and taking the coefficients generated after the fitting (that is, A , B , and t_d). The entire measurement dataset is split into training and testing sets using stratified K-fold method, with K set to 10. For our experiments, we choose three basic supervised machine learning classification methods, namely decision tree, naive Bayes, and k-nearest neighbors (KNN) implemented in the Python scikit-learn machine learning library. These classification algorithms generated highly accurate classification results. For example, the naive Bayes classifier achieves both 1.0 precision and recall scores when varying only the power input to the motor,

and 0.98 precision and 0.97 recall scores when varying only the load connected to the motor. All classifiers have a slight performance drop when classifying fingerprints generated by all 80 configurations when combining the two variables, namely the input power and load. For example, the decision tree classifier achieves a 0.89 precision score and 0.89 recall score. We take this difference as a deficiency in our approach and argue that this becomes a challenge when the device has so many possible configurations that they have similar signatures. However, this does not become an unsolvable problem in an industrial environment, as the number of different design configurations of a device (thus the fingerprints) is small. Furthermore, a device is usually expected to operate within a reasonable range of states, which allows slight deviations from its theoretical operating state. Such deviations can sometimes cause the fingerprints to be difficult to differentiate from those generated by a slightly different device configuration.

Effect of Network Delay

As timestamps play a significant role in our fingerprinting technique, their integrity is critical to the success of the proposed technique. Generally, there are three options for obtaining the actuator timing values in CPSs:

1. If the protocol natively supports timestamps (for instance, DNP3), the operation time can simply be sent as part of the packet.
2. Protocols that do not have native support for timestamps (for example, Modbus as we used in the experiment) can have PLCs store such information as values in their registers, and later send these values back to the host.
3. If neither of the above two methods is available, for the first two options, a timestamp is taken by the PLC in real time and is relatively accurate (the results above illustrate this scenario). In the last option, the operation time is taken using a tapping point that monitors the packets flowing in the network as we did in our previous work.¹³ In such case, network delay can add to the measurement of operation time and thus impact the accuracy of the timestamps.

Recall that our results thus far represent scenarios 1 and 2 above where the scheme is impervious to network perturbations. This is because the actuation time is placed as a value inside the packet. Here, we seek to infer that time from the timing of the request/response packets received/sent from/to the actuator, as done in our previous work.¹³ Thus, network delay *would* affect the technique. This method would have to be used if the hardware and protocol used in the CPS do not support

the ability to transmit timestamps in the packet, the timestamp in the packet is not trustworthy, or the timestamp is not available (for instance, encrypted). In this scenario, we would use a network traffic monitoring device to timestamp the packets locally. To better understand the effect of network delay on the performance of our fingerprinting technique, we use a statistical model fitted to real data¹⁵ to generate network delay time values without having to perform an enormous amount of experiments. Thus, physical devices may receive the command from the controller earlier or later than in the previous scenario due to the network perturbations.

We take the network delay values generated through the statistical model and add them to the collected timestamps to simulate the data collected by the traffic monitoring device (tapping point) under the influence of network congestion and delay. We tweak the shape parameter k in the gamma distribution function used in the model while keeping the scale parameter θ at default value 1.0. The mean is thus $k\theta = k$ and variance is $k\theta^2 = k$, with the unit of time in ms. Note that jitter can be modeled by the variance as its definition is the variation in the delay of received packets. We repeat the classification tasks under different network delay distributions determined by k as well as under three different sets of configurations, S_{power} , S_{load} , and S_{both} , which denote varying only power input with fixed load, varying only load with fixed power, and varying both, respectively. We plot the classification result in Figure 4. Note that a larger k value corresponds to more severe network delay and 0 simply means no delay. The performance of almost all classifiers degrades as k increases; however, both decision tree classifier and KNN classifier remain at more than 0.8 precision and recall, and more than 80 percent accuracy even under a large network delay and jitter. The result suggests that the fingerprinting technique can be robust under the influence of network delay and jitter up to a reasonable level (approximately 500ms).

Resistance to False Modeling Attacks

As illustrated above, physics-based device fingerprinting can be used to authenticate actuators. Accordingly, it is important to understand the efficacy of such an approach when under attack. Thus, we introduce the concept of a *false modeling attack*. In a false modeling attack, we assume an attacker launches the attack on a CPS either inside or outside of the CPS network, and his objective is to sabotage the physical system (that is, the process) by spoofing commands to the actuators and sending emulated responses to the controllers in the CPS. We further assume that the attacker has some, but not complete knowledge of the physical components in the system. Such an assumption is usually

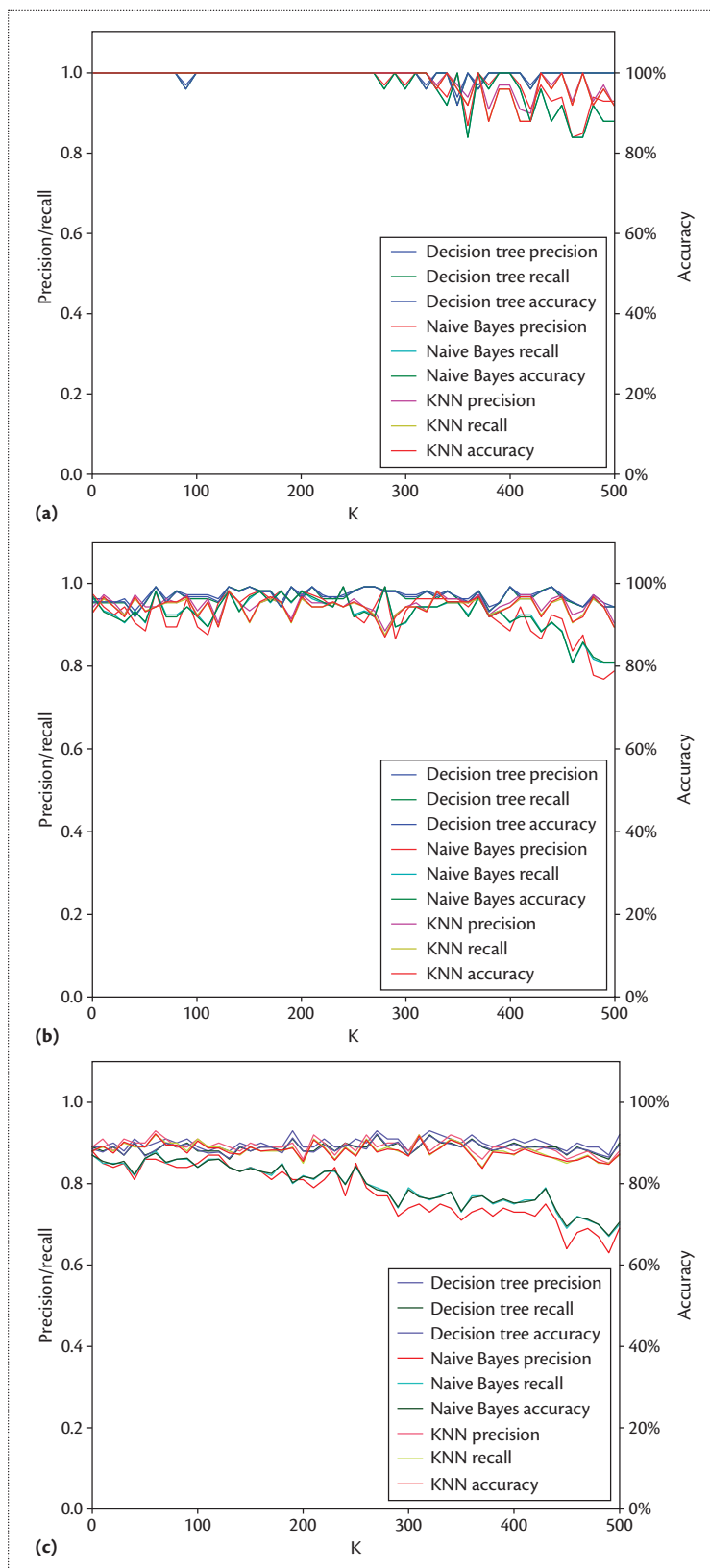


Figure 4. Precision, recall, and accuracy scores when performing classification on the fingerprints generated under various network delays. (a) Varying only power input to the motor. (b) Varying only the load connected to the motor. (c) Varying both power and load.

reasonable in most cases, as even a system administrator may not have all details about the underlying physical system, such as the exact model of every device. To achieve his objective, the attacker would need to deceive the detection mechanism with network response packets as long as possible, so that he could perform a long enough attack to cause significant damage to the system. As the attacker does not hold every detail of the system, he needs to make assumptions about the missing details in order to generate the response packets. Without complete knowledge of the exact model of a device, the attacker has to randomly guess the model and generate the response packets based on the potentially incorrect model, for instance, assuming a motor of unknown model to be a 24V, 2 HP model driving a load equivalent of $0.2kg \cdot m^2$. Hence, this attack is deemed a false modeling attack.

Our proposed device physics-based fingerprinting technique can be used to defend against false modeling attacks. Specifically, a two-phased approach is used. During the first phase, a classifier is trained with the fingerprint of each device of interest. Such fingerprints could be generated experimentally (black box model), or could be generated by an accurate modeling of the device (white box model).¹³ A more general method could be a mix of both (gray box model). Because the attacker may have limited details of the detection method and actual models of the devices used in a CPS, it is very likely that he will assume a device model different from the one actually being used. Under ideal conditions, the possibility P_{diff} of the attacker guessing the wrong device model is $P_{diff} = 1 - 1/N_{model}$, where N_{model} is the number of models a device can have. Note that our technique may fail to detect the spoofed response packets if the attacker picks the correct device model or a similar one with the fingerprints in the packets close enough to the expected response generated by the actual device.

We perform a simulated test to experimentally measure the performance of our technique against false modeling attacks. Using the classifiers we trained, we test the classifiers with the fingerprints generated by randomly chosen configurations of the motor. This test data represents the fingerprints generated by an attacker who does not know which model of the motor is used in the actual CPS. We again perform the experiment under three different device configuration combinations, S_{power} , S_{load} , and S_{both} . Under each combination, the attacker correspondingly chooses a device model among the configurations. Our results show that the detection rate is very close to the ideal value. For example, when varying only power input (five different values), the naive Bayes classifier correctly identifies 79.1 percent of the attacks on average.

When varying only the load (16 different values), 93.7 percent of the attacks can be correctly identified. The number gets even higher, to 98.9 percent, when varying both parameters and all 80 configurations are available. Thus, with an increasing number of models and configurations for a device, our device physics-based fingerprinting technique has an increasing success rate at detecting false modeling attacks. Clearly, as the recall in some cases is not equal to 1.0, a false alarm will occur. However, this problem can be mitigated by:

- adjusting the tolerance of the difference between the inferred configuration and the true configuration, thus taking a tradeoff between precision and recall depending on the specific application, and
- incorporating our device physics method with other ones, for example, process physics methods as we argued at the beginning of this article.

In reality, if there is a high cost associated with the actions taken in the case of false alarms, a less aggressive defense can be taken at first to minimize the loss due to the shutdown, such as checking network logs to determine if there has been an intrusion.

In the future, we will automate the physics-based model extraction process. Another important consideration for this work is the effect of wear and tear on the devices' operating times, which can potentially compromise the effectiveness of the technique over time. Because the fingerprints are tightly associated with the physical operation of the device, they are subject to change gradually as the device inevitably degrades. Taking this factor into account might not only help to reduce false alarms as device fingerprints deviate from the time they were first collected and stored for reference, but also increases the difficulty for the attacker to spoof the device. ■

References

1. D. Olick, "Why 2017 Will Finally Be the Year of the Smart Home: Consumers Figure It Out," CNBC, 4 Jan. 2017; www.cnn.com/2017/01/04/why-2017-will-finally-be-the-year-of-the-smart-home-consumers-figure-it-out.html.
2. "Global Industrial Controls System Market to Grow at CAGR of 4.9% from 2015 to 2021," Transparency Market Research, Sept. 2015; www.transparencymarketresearch.com/pressrelease/industrial-controls-market.htm.
3. J. Slay and M. Miller, *Lessons Learned from the Maroochy Water Breach*, Springer, 2008.
4. A. Greenberg, "Crash Override: The Malware That Took Down a Power Grid," WIREd, 12 June 2017; <https://www.wired.com/story/crash-override-malware>.

5. C. McParland, S. Peisert, and A. Scaglione, "Monitoring Security of Networked Control Systems: It's the Physics," *IEEE Security & Privacy*, vol. 12, no. 6, pp. 32–39, 2014.
6. N. Falliere, L.O. Murchu, and E. Chien, "W32.Stuxnet Dossier," *Symantec-Security Response*, February 2011, pp. 1–69.
7. A.A. Cárdenas et al., "Attacks against Process Control Systems: Risk Assessment, Detection, and Response," *ASIACCS*, 2011, p. 355.
8. C. Neilson, "Securing a Control Systems Network," *ASHRAE*, 2013.
9. D.I. Urbina et al., "Limiting the Impact of Stealthy Attacks on Industrial Control Systems," *CCS*, 2016, pp. 1092–1105.
10. A.A. Cárdenas et al., "Challenges for Securing Cyber Physical Systems," *Workshop on Future Directions in Cyber-physical Systems Security*, DHS, 2009.
11. A.A. Cárdenas, S. Amin, and S. Sastry, "Research Challenges for the Security of Control Systems," *HOTSEC*, 2008, p. 6.
12. A.A. Cárdenas, S. Amin, and S. Sastry, "Secure Control: Towards Survivable Cyber-Physical Systems," *IEEE 28th International Conference on Distributed Computing Systems Workshops*, 2008, pp. 495–500.
13. D. Formby et al., "Who's in Control of Your Control System? Device Fingerprinting for Cyber-Physical Systems," *NDSS*, 2016.
14. "Mixing Equipment—Industrial Equipment—Shopping Cart by INDCO, New Albany, Indiana," <https://www.indco.com/shop/industrial-mixers/ansi-flange-mixers>.
15. T. Holleczeck, V. Venus, and S. Naegele-Jackson, "Statistical Analysis of IP Delay Measurements as a Basis for Network Alert Systems," *IEEE International Conference on Communications*, 2009, pp. 1–6.

Qinchen Gu (Member, IEEE) received an MS degree in electrical and computer engineering from Georgia Institute of Technology (Georgia Tech). He is currently a PhD student in the School of Electrical and Computer Engineering at Georgia Tech, and a graduate research assistant of the Communications Assurance and Performance (CAP) group. His research primarily focuses on the security for cyber-physical systems. Contact him at qgu7@gatech.edu.


David Formby (Member, IEEE) received MS and PhD degrees in electrical and computer engineering from Georgia Tech. He is currently a postdoctoral researcher in the School of Electrical and Computer Engineering at Georgia Tech, and a member of the CAP group. His research primarily focuses on network security for industrial control system networks. Contact him at djformby@gatech.edu.

Shouling Ji is a ZJU 100-Young Professor in the College of Computer Science and Technology at Zhejiang

University and a member of the research faculty in the School of Electrical and Computer Engineering at Georgia Institute of Technology. He received a PhD in electrical and computer engineering from Georgia Institute of Technology and a PhD in computer science from Georgia State University. His current research interests include big data security and privacy, big data driven security and privacy, AI security, and adversarial learning. He is a member of IEEE and ACM and was the Membership Chair of the IEEE Student Branch at Georgia State (2012–2013). Contact him at sji@gatech.edu.

Hasan Cam received a PhD in electrical and computer engineering from Purdue University and an MS degree in computer science from Polytechnic University, New York. He is a computer scientist at US Army Research Laboratory. He currently works on the projects involved with assessment and management of cyber vulnerability, risk, resilience, agility, mission assurance, active malware defense over wired, mobile, and tactical networks. His research interests include cybersecurity, machine learning, data analytics, networks, algorithms, and parallel processing. He serves as the government lead for the Risk area in Cyber Collaborative Research Alliance. He has served as an editorial member of two journals, a guest editor of two special issues of journals, an organizer of symposiums and workshops, and a Technical Program Committee Member in numerous conferences. He is a Senior Member of IEEE. Contact him at hasan.cam.civ@mail.mil.

Raheem Beyah is the Interim Steve W. Chaddick School Chair and Motorola Foundation Professor in the School of Electrical and Computer Engineering at Georgia Tech. He is also a co-founder of Fortiphed Logic, which provides cybersecurity solutions for industrial control systems. He received the National Science Foundation CAREER award in 2009 and was selected for DARPA's Computer Science Study Panel in 2010. He is a member of AAAS, ASEE, a lifetime member of NSBE, a senior member of IEEE, and an ACM Distinguished Scientist. Contact him at rbeyah@ece.gatech.edu.


Read your subscriptions through the myCS publications portal at <http://mycs.computer.org>