Generating Client Side Policies for Cyber-Physical Safety*

Dinesh Verma[1] and Seraphin Calo[2] and Elisa Bertino [3] and Geeth de Mel[4]

Abstract— **Cyber phyiscal systems are increasingly connected to the Internet for reasons of convenience and efficiency. However, such cyber-physical systems are exposed to new vulnerabilities since they may be compromised by an attack from the Internet. In addition to traditional network security mechanisms, such systems need additional mechanism which can prevent the exploitation of their physical vulnerabilities. We propose an architecture in which the behavior of the system can be controlled by having it generate the policies for its own protection automatically.**

I. Introduction

Cyber-physical systems [1] are systems that interact with the physical environment while being connected to a computing and communications infrastructure. Such cyber-physical systems are used in many environments including but not limited to industrial manufacturing, power generation, and building management. With the wide availability of network connectivity through the ubiquity of communications networks, such systems are also connected to the Internet, resulting in sub-domains such as the Internet of Things [2], the Industrial Internet of Things [3], Internet of Battlespace Things [4], etc.

The interconnection of cyber-physical systems to the Internet adds significantly to the convenience of accessing and managing such systems remotely. However, this comes with an increased risk of vulnerability from the attackers on the Internet. In order to address the security vulnerabilities, several approaches have been proposed which can be found summarized in various surveys [2], [5]–[7]. However, the primary focus of existing work has been an extension of the traditional computer and network security mechanisms to the domain of connected things.

While the need for extending traditional security mechanisms such as encryption, authentication, authorization, access control, etc. is definitely present, such extension alone is inadequate to address the unique security and safety risks that arise when cyber-physical systems are connected to the Internet. A unique security vulnerability that arises in such systems is the ability of an attacker to cause physical damage using Internet based attacks, e.g., damage done in a manner similar to Stuxnet [8] but leveraging the Internet based connection for the same (Stuxnet itself was not launched via the Internet but is an example of physical damage caused using malware).

In the next section, we introduce the environmental assumptions we are making and provide a limited enumeration of the type of attacks that can be launched on the connected system. This is followed by the description of a system for which we propose an architecture for preventing physical damage to devices. A key new approach for protection of physical systems is the automatic generation of policies which allow/deny some types of requests on those cyber-physical systems. We discuss the approach for generating these security policies. Finally, we conclude with some example scenarios of cyber-physical systems and some simulation results on their protection.

II. The Cyber Physical Environment

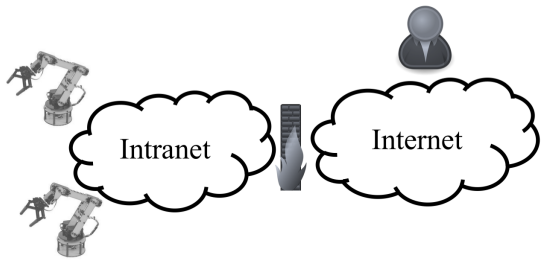

Fig. 1. The environment assumed for Cyber Physical Systems. Cyber phyiscal systems are connected and made accessible over the Internet. The most common usage will be a manager or legitimate user issuing control commands to the IoT device.

The cyber-physical environment we assume for this paper is shown in Fig. 1. One or more cyber-physical devices are operational within an enterprise, and are present in the Intranet of that enterprise. These cyber-physical devices could be controllers for building management systems, equipment used for industrial manufacturing, devices used for power generation, or any of the various other functions which require robotic controls. The intranet could also represent

*This research was sponsored by the U.S. Army Research Laboratory and the U.K. Ministry of Defence under Agreement Number W911NF-16-3-0001. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the U.S. Army Research Laboratory, the U.S. Government, the U.K. Ministry of Defence or the U.K. Government. The U.S. and U.K. Governments are authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation hereon.

[1]D. Verma is with IBM T. J. Watson Research Center, Yorktown Heights, NY 10598, USA dverma at us.ibm.com

[2]S. Calo is with IBM T. J. Watson Research Center, Yorktown Heights, NY 10598, USA scalo at us.ibm.com

[3]E. Bertino is with Computer Science Dept, Purdue University, West Lafayette, IN-47907, USA bertino at purdue.edu

[4]G. de Mel is with the IBM Research UK, Hartree Center, Warrington, UK geeth.demel at uk.ibm.com

a home network when the cyber-physical system is used for devices within a home.

The devices can be accessed remotely over the Internet. The Internet based remote access provides a convenience to the user or the administrator of the machine. The building manager can check the controller and its state remotely. The foreman of a manufacturing plant can check or operate a factory floor machine remotely if needed. A technician can trouble-shoot a malfunctioning machine remotely. Devices located in an isolated area can be maintained by a remote administrator. There are many advantages in enabling the systems to be accessible and managed remotely.

We assume that all the precautions taken to ensure secure access and authorization are followed. However, the following vulnerabilities would still exist within the environment which is made accessible via the Internet.

- **Stolen Credentials**: The credentials of a legitimate user can be stolen by a malicious person. The attacker can then masquerade as the legitimate user to access the remote devices. The attacker can then launch different types of destructive commands on the devices. These commands could cause physical damage in the environment.
- **Disgruntled Employee**: A legitimate user may be disgruntled for a variety of reasons. The disgruntled employee may decide to issue commands which can cause damage to the physical devices.
- **Human Error**: A remote user who may be performing legitimate operations on the device may accidentally issue commands which can be damaging in the environment.

These attacks and mistakes cannot be corrected by means of basic authentication, authorization and access control mechanisms. The access by the user is allowed and appears to be legitimate. The actions invoked by the user are also legitimate. However, their impact could be physical damage to the system.

The physical damage can be caused in many subtle ways. As an example, if an environment may consist of both heating equipment and cooling equipment, the system can be configured so that the heater tries to maintain a temperature above a high threshold, while the cooling equipment is instructed to maintain a temperate below a low threshold. Both pieces of equipment will be operationally negating the impact of each other, causing an increase in the electricity costs incurred by the building. In other examples, devices may be instructed to take actions that can be possibly damaging, e.g., instructing a remote 3D printer to print worthless artifacts, asking a machining tool to take a sequence of steps which causes a high level of vibrations and associated damage, or shutting down critical ventilation functions in an enclosed space. Some physical damage such as leakage of fluids, or excess vibrations, may not be reversible.

Since network access controls can not prevent these types of attacks, we need to devise schemes which span the boundary of physical systems and cyber-security to ensure safe and secure operation. The danger that can come from the operations performed in this environment arises not just from a breach in the network security, but from the operations that are performed over the network. Thus, we need to ensure that the operations do not change the state of the device in a manner that is dangerous.

In order to design the security architecture for this environment, we assume an abstract representation of the environment as shown in Figure 2. The protected intranet consists of several devices and a controller, along with a firewall. The firewall is responsible for protecting all of the devices from unauthorized access from the Internet, and provides the ability to authenticate and authorize the user accessing the secure environment from the outside. The devices perform the required task within the protected environment. The controller receives all inbound requests for reading information from, or performing any control operations on any device, and is the logical entity through which devices are monitored and controlled.

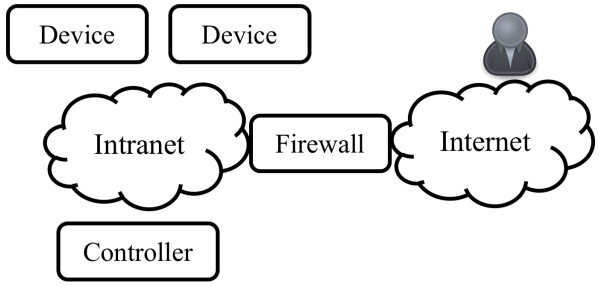

Fig. 2. The components in the abstract representation of the environment. Different devices that can impact the physical environment are accessed via a controller, and the network containing the devices and controller is protect ed from the Internet via a firewall

III. Client Security Architecture

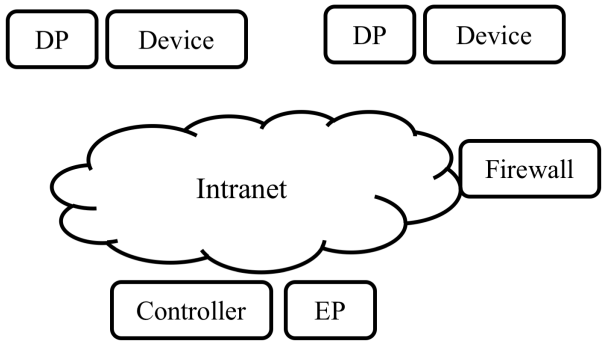

Fig. 3. The protection model for devices and controllers.

In order to protect against physical damage from network based attacks, we assume an architecture in which each device is augmented with a device protection component (DP

as shown in Figure 3). The device protection component is responsible for protecting the devices from commands that can potentially cause damage to it. Similarly the controller is augmented with an environment protection function (EP), which is responsible for ensuring that different devices working together do not get into a state where the environment is not safe. The firewall directs any requests from the remote user to the EP, which decides whether or not the request ought to be allowed or not. When the controller makes a request to the device, the device similarly determines whether or not to allow the request.

The EP function and firewall function are very distinct, whereby the firewall is responsible for ensuring that only authorized users are allowed to access the devices on the Intranet, and the EP is responsible for ensuring that the inbound requests from users who are authorized (or able to masquerade as authorized users) do not cause physical damage to the devices, or to the overall ecosystem in which the devices are operating. In some cases, the EP and the firewall may be physically packaged in the same hardware box, but they are two distinct logical functions.

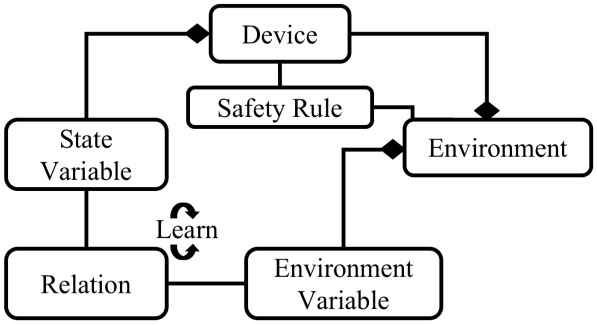

Fig. 4. UML diagram modeling the system.

In order to perform their functions, the EP and the DP need to work with a model for accesses to the devices. We assume that each device is characterized by a set of one or more state variables which provide a model for the operation of the device. Similarly, each environment is modeled by one or more environment variables. Each external user is allowed to invoke one or more actions on the device, and each action has the impact of changing the state variables. Each of the actions is abstracted as consisting of changing one or more state variables into some other value. The state variables of different devices may have an impact on the environment variables, which is captured by a set of relationships. In order to characterize the bounds which define safe or unsafe operation of the system, a set of safety rules can be defined. The net resulting model of the system is described in a UML diagram shown in Figure 4.

As an example, a building space may be monitored as consisting of two devices, a heater and a cooler, each with one state variables – an on/off state. The environment variable is the temperature of the building space, which may also be a read-only state variable. When the state variable of the cooler is on, the temperature of the building decreases, while when the state variable of the heater is on, the temperature of the building increases. If both are off, the environment temperature is an independent variable that can take any value. If both are on, the temperature may increase or decrease depending on the strength of the two devices. The safety rule would put upper and lower bounds on the temperature that the environment should have.

While the state variables, environment variables, safety rules, and the actions permitted on each device can be defined manually, defining the relationship between the different state variables and environment variables is hard to do manually. This relationship needs to be learned for every different environment, since it depends on several factors that are environment-dependent. In the example given above, the impact of turning on a cooler or heater on the change in the temperature depends on the position and distance between the cooling/heating vents and the temperature sensor. As a result, Figure 4 shows that the relationship between the different variables ought to be learned by means of a loop.

With the abstract representation of the environment, we assume that the EP and the different DP modules in the environment are policy-driven, i.e., they are designed to use a set of access control policies which will determine which operations are permitted to be taken on the device, and which operations will be denied, even if they are issued by an authorized user of the system. This results in a client security architecture as shown in Figure 5

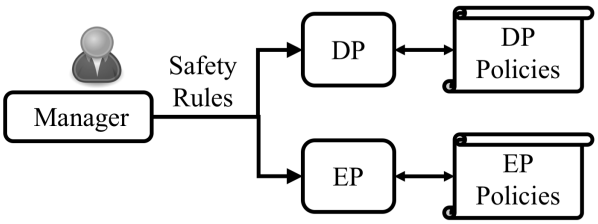

Fig. 5. Client-Side Security Architecture

In this client security architecture, a policy manager provides an interface for a human to define safety rules for the environment. These safety rules are sent to the device protectors (DPs) and environment protector (EP) present in the environment. The DPs and EP generate access control policies for different operations that may be invoked upon them, and enforce those policies. Each access control policy allows or disallows the operation of a command on the device when invoked by the remote user. Each of these access control policies depends on the current set of safety rules, the current state of the devices and the environment, and the current set of relationships that have been learned about the

different device variables and environment variables.

Note that in this model, the policies do not apply if the authorized user is in the secure intranet itself, which allows for an over-ride mechanism to deal with those situations when the automatically generated policies block some operations which are considered critical by a human administrator to be performed.

IV. Policy Generation

In order for the devices and the controller to generate their access control policies, we use the generative policy management architecture [9] as the basic approach and customize it for the specific details of the cyber-physical safety scenario (See Figure 6). In order for each of the protectors (the multiple DPs and the EP) to generate their own policies, they need to discover each other. The discovery process allows them to find out the set of state variables in the other devices, and use that to learn the relations between different state variables. The discovery process is controlled by means of a context provided by the policy manager to the different protectors.

There are three key components of the context, an interaction graph which tells each protector what other types of protectors are in the system, a set of safety rules for the device which provides hard bounds on device and environment variables that can not be exceeded, and a set of configuration parameters, such as the time period for which the protectors ought to look ahead to ensure that the safety rules are being satisfied.

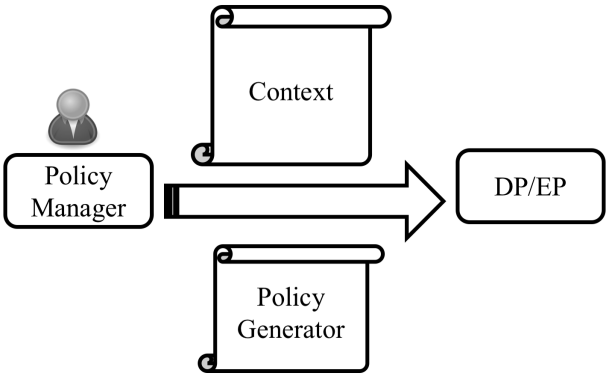

Fig. 6. Generative Policy Architecture

The other information provided by a human to the protectors via the policy manager is a set of policy generators that can be used to create new policies. The policy generator could be a template which specifies the format of the policy that is to be generated, where specific values in the template are created automatically by the DP and EP components. Other alternative approaches include providing a grammar from which to generate the policies, or using an Answer Set Programming approach [10].

Each of the protectors performs the steps shown in Fig. 7, checking continuously for new types of protectors in the system. Once the protectors find new devices in the system, they check for the relationships between other devices and their own state variables, and combine those relationships with the safety rules to generate the policies.

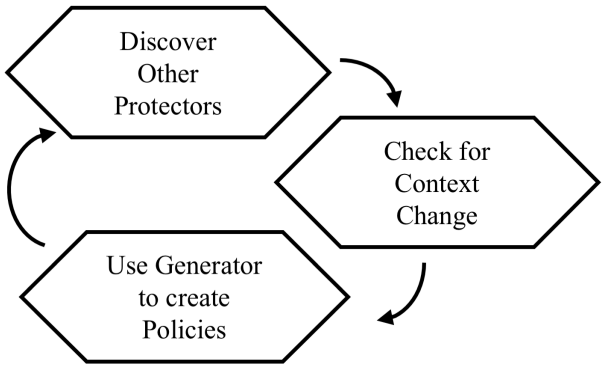

Fig. 7. Activities of each protector

The second step of the process, generation of the policies happens according to the logic dictated in Fig. 8. An initial set of relationships among different devices may be specified and available to each protector. The machine learning process associated with each protector finds the relationship between the different elements, and updates the set of relationships that exist between the variables. The comparison between the new relations and the safety rules provided to the system are used to generate the new access policies, which determine which actions would be allowed and which actions would be disallowed in the environment. In this model, new policies will be generated if new devices enter the system, safety bounds are changed, or if the machine learning process determining the relationship between different variables results in a new model.

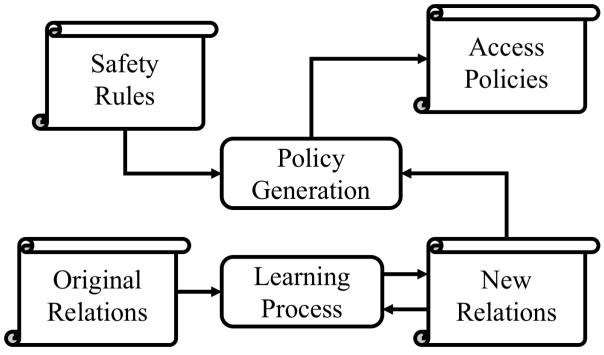

Fig. 8. Process for Generation of Access Control Policies

The policy generation information that comes from the policy manager to each protector includes a set of safety rules, along with the time-period which needs to be considered to stay within the safety bounds. The learning rules among the variables finds the rate of change between the different environment variables caused by changes in the device state variables. For any action that can be invoked on a protector, the system calculates if the change in state

variables would cause the environment variables to exceed the safety threshold. If so, a policy indicating that that action is not allowed is generated. Otherwise, a policy indicating that the action is allowed is generated.

V. Scenarios

In this section, we discuss some of the scenarios where improved security can be provided by means of the system described here.

A. Numerically controlled Machines

Many current machines such as lathes, water-jet cutters and spark-machines are equipped with numerical control, which essentially provides a local computer to program the machine to perform the task of converting a *blank* (unshaped piece of metal or other material) into a shaped mechanical part. By means of the computer program, any type of mechanical part can be shaped, using computer programs to alter the configuration of the machine as needed to shape the required part. This allows a lathe to automatically adjust the distance of the cutting blade from the center of the piece, and to adjust the speed of the rotation of the cutting tool, or a drill to adjust the number, depth and position of holes made into the blank. Similarly, a spark machine's intensity and location of the sparks, or the speed and location of water-jets that cut a part, can be adjusted automatically.

Currently, such machines will have a local control by which an operator would load the blank into the machine, and program its operation. A human operator watches over the machine as it operates, allowing the operator to stop the processing if the machine takes an unexpected turn, e.g. vibrates too much. In many cases, the loading of the blanks and removal of the shaped part from the machine is also performed by means of a robotic arm, and a sequence of numerically controlled machines can be used to perform complex operations in conversion of a blank into a very complicated final machine part.

The convenience that can be offered in controlling a machine of elements with remote access is tremendous, since the operator can be remote while the required parts are getting machined. However, it also removes the watchful eye of the operator in case an unexpected error arises in the operation of the machine.

The state variables of a machine like a lathe can be viewed as the state of the machine, the position of its cutting edge (as measured from the center), the speed at which the blank part is being rotated, and the type of blank that is put into the system. Additional variables would be the vibration of the machine, and its temperature which could be monitored by a sensor. An environmental variable would be the noise produced in the environment due to the machining process.

The safety constraint in this scenario would be upper bounds on the vibration of the machine, its temperature and the threshold on the noise produced. During the normal operation of the machine, the relationship between the type of metal, the position of the machine, the speed of the blank, and the temperature, vibration and noise level produced by the tool is learned. This allows the environment protector to generate rules which will prevent the remote loading of a component when the machine is running hot, or likely to vibrate too much, as well as the device protector to generate a rule to shut down the machine in case an existing program is likely to make the machine overheat in some safety time-period.

B. Machine Room Temperature Control

A machine room may contain a large number of computing and related equipment, e.g., processing, memory, storage, communications and power regulating devices. The overall space may thus require several cooling elements for dispersing the heat generated, and some heating elements for maintaining a reasonable temperature for people who may need to work there.

Improper setting of the cooling and heating units can cause erratic behavior or permanent damage to the electronic components within the computing equipment. The safety of each of the devices thus depends not just on its own state, but on the state of all the other devices in the environment, since they all contribute to the heat generated in the machine room. In addition, assuring safety becomes quite challenging since the safety analysis becomes more complex as one must collectively consider the combined effects of changes in state of each of the devices in the room (e.g., powered on, powered off, working at low utilization, working at high utilization).

It is possible to launch an attack on the system by manipulating the states of the computing equipment to generate more heat than expected, or manipulating the cooling equipment to provide less cooling than needed. In our protection model, each device would have a DP with rules that would not allow operations that would increase the temperature of the device beyond sustainable levels. The DP would also be aware of the devices closest to it that would have the most effect on its own temperature by means of the interaction graph.

The DPs on the cooling units would not allow cooling to be decreased unless the temperature within the regions that they most affect was below a certain threshold. The EP would need to be aware of the effects of operations on the devices that would increase their local temperature, the cumulative effects of heat generation at each of the devices on the room temperature, and the states of the cooling units. For example, it would not allow devices to be powered on in regions where the additional heat generated would adversely affect surrounding devices.

It would be difficult to manually generate the policies needed to protect the system, since the room temperature varies over time, and is a function of the states of all the devices in the room, their locations with respect to the cooling units, and the settings of the cooling units. The relationships between the local state variables and the environment variables need to be learned.

VI. Conclusions

We have presented an architecture for controlling the behavior of cyber-physical systems that are connected to the

Internet for reasons of convenience and efficiency. In our approach the elements of the system learn the relationships between their states and the state of the system environment, and generate the policies needed for their own protection automatically. Such policies are meant to prevent the exploitation of their physical vulnerabilities.

REFERENCES

[1] R. Rajkumar, I. Lee, L. Sha, and J. Stankovic, "Cyber-physical systems: the next computing revolution," in *Design Automation Conference*. IEEE, 2010, pp. 731–736.

[2] L. Atzori, A. Iera, and G. Morabito, "The internet of things: A survey," *Computer networks*, vol. 54, no. 15, pp. 2787–2805, 2010.

[3] A. Gilchrist, *Industry 4.0: the industrial internet of things*. Apress, 2016.

[4] A. Kott, A. Swami, and B. J. West, "The internet of battle things," *Computer*, vol. 49, no. 12, pp. 70–75, 2016.

[5] A.-R. Sadeghi, C. Wachsmann, and M. Waidner, "Security and privacy challenges in industrial internet of things," in *2015 52nd ACM/EDAC/IEEE Design Automation Conference (DAC)*. IEEE, 2015, pp. 1–6.

[6] C. Maple, "Security and privacy in the internet of things," *Journal of Cyber Policy*, vol. 2, no. 2, pp. 155–184, 2017.

[7] S. Li, T. Tryfonas, and H. Li, "The internet of things: a security point of view," *Internet Research*, vol. 26, no. 2, pp. 337–359, 2016.

[8] R. Langner, "Stuxnet: Dissecting a cyberwarfare weapon," *IEEE Security & Privacy*, vol. 9, no. 3, pp. 49–51, 2011.

[9] D. Verma, S. Calo, S. Chakraborty, E. Bertino, C. Williams, J. Tucker, and B. Rivera, "Generative policy model for autonomic management," in *2017 IEEE SmartWorld*. IEEE, 2017, pp. 1–6.

[10] M. Law, A. Russo, E. Bertino, K. Broda, and J. Lobo, "Representing and learning grammars in answer set programming," 2019.