Self-Generation of Access Control Policies

Seraphin Calo
IBM Research
Yorktown Heights, New York, USA

Dinesh Verma
IBM Research
Yorktown Heights, New York, USA

Supriyo Chakraborty
IBM Research
Yorktown Heights, New York, USA

Elisa Bertino
Purdue University
West Lafayette, Indiana, USA

Emil Lupu
Imperial College
London, UK

Gregory Cirincione
Army Research Labs
Adelphi, Maryland, USA

ABSTRACT

Access control for information has primarily focused on access statically granted to subjects by administrators usually in the context of a specific system. Even if mechanisms are available for access revocation, revocations must still be executed manually by an administrator. However, as physical devices become increasingly embedded and interconnected, access control needs to become an integral part of the resources being protected and be generated dynamically by the resources depending on the context in which they are being used. In this paper, we discuss a set of scenarios for access control needed in current and future systems and use that to argue that an approach for resources to generate and manage their access control policies dynamically on their own is needed. We discuss some approaches for generating such access control policies that may address the requirements of the scenarios.

KEYWORDS

Dynamic Security, IoT, Machine Learning, AI, Autonomic Access Control

ACM Reference Format:
Seraphin Calo, Dinesh Verma, Supriyo Chakraborty, Elisa Bertino, Emil Lupu, and Gregory Cirincione. 2018. Self-Generation of Access Control Policies. In *SACMAT '18: The 23rd ACM Symposium on Access Control Models & Technologies (SACMAT), June 13–15, 2018, Indianapolis, IN, USA.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3205977.3205995

1 INTRODUCTION

Access control is a fundamental part of security in IT systems, and various approaches to properly authenticate and control access to sensitive resources have been developed [9], including role based [7] and attribute based [13] access control schemes, and their numerous variants [2, 8, 14]. While the need for dynamic access control is recognized, the solutions proposed for dynamicity have been grafted over a static access control paradigm [6, 11, 12]. Nevertheless, current approaches for access control tend to be relatively static, have to be managed by human administrators, and in many cases imposed as an external security mechanism to wrap around the operation of a resource.

As information systems become more dynamic, virtualized, grow in size and scale, and are increasingly used to interconnect physical systems, existing access control mechanisms may turn out to be inadequate to deal with the resulting complexities. In this paper, we argue that we need to envision a new type of access control paradigm, one in which resources determine their own access control policies from a dynamic assessment of their context of operations, the risks versus utility trade-off imposed in a specific situation and learn how to adapt their behavior. Static access control paradigms and supporting dynamicity by grafting dynamic access control post-facto, e.g., using a management system for dynamic access control, is no longer going to be sufficient. The intelligence to provide access control in a dynamic manner needs to be embedded within the resources themselves.

We begin this paper with a discussion of several scenarios in which existing access control paradigms may prove to be inadequate. These scenarios motivate the need for resources to create their own access control policies in a dynamic manner. Finally, we discuss some approaches by which the vision of resources that generate their own policies can be attained and conclude with a discussion about when each approach may be applicable.

2 SCENARIOS

The scenarios that we discuss below are ones that are going to be real in the imminent future. In some cases, one can even assert that the situation described is already present in many environments. In all the scenarios, we argue that the current access control mechanisms will be inadequate to address the security needs of the scenario.

Current access control mechanisms are derived primarily from a model in which "subjects"[1] try to access resources and a complete set of subjects and resources can be specified in advance in an access control matrix [19]. Access control is enforced by either considering the set of subjects that are allowed access to a resource (by maintaining an access control list) or by considering the set of resources a subject is allowed to access (capabilities). On this basic mechanism, one can add definitions of "roles" to group subjects and/or resources; or arrange these subjects and resources in different hierarchies to simplify the task of representing the access control matrix. However, the matrix remains relatively static, and rarely considers the fact that resources interact with each other and may require a continuously changing access control matrix. More recent attribute-based access control models [13], of which XACML [2] is a well-known example, try to provide richer access

[1] A subject can be a human user, a process, an application, a device and so forth.
[2] http://docs.oasis-open.org/xacml/3.0/xacml-3.0-core-spec-os-en.html

SACMAT '18, June 13–15, 2018, Indianapolis, IN, USA

ACM ISBN 978-1-4503-5666-4/18/06…$15.00
https://doi.org/10.1145/3205977.3205995

based on values of properties (e.g. attributes) characterizing subjects and resources and are also able to take context into account. However, permissions in such models are usually static and need to be issued by an administrator. Furthermore, these models require the acquisition of trusted attribute information. The static nature of access control may be inadequate in many situations. There are models which take into account context information. Some attention has been given to add dynamicity as an after-thought through management systems or as part of a federation architecture. For example, roles are delegated in a dynamic manner to deal with dynamic coalitions [11], an intelligent network management system reacts to current threat to adjust network security policies [6], higher level workflows are used to determine access control [12], or dynamic attributes like location are used to determine roles [3]. However, the context information is usually limited to location and time and even for this limited context there are no effective approaches for dealing with rapidly changing contexts (which is relevant especially for mobile systems). In order to cope up with the rapid dynamicity in the system, the use-cases below require resources to be able to determine their own access control policies.

2.1 Car Navigation Systems

A car navigation system has several types of information available to the driver, and information such as the home location and previous destinations is available to any person who is able to access the car. However, open access to this information creates known security exposures, e.g., a thief may break into a car parked at a movie theater, drive the car to the home location, open the garage door using the remote control in the car, and rob the house under the likely scenario that the home is unoccupied.

Adding a static access control paradigm to the information, e.g., defining the role of a driver and/or passenger, and assigning role or attribute-based access control permissions does not solve this problem. Similarly, identifying the driver using visual recognition with a camera and only allowing access to home location and/or recent destinations to the authorized driver is not a viable solution. The car may be driven on occasions by the spouse, children or neighbor of the most frequent driver, and authorizing each one to the car systems would be tedious. If a policeman or paramedic needs to take the driver, who may be sick, back to his or her home, they should be authorized to get the home information. If the driver is in the passenger seat of the car, the access should probably be authorized. On the other hand, if a previously unseen person has broken the window of the car to enter, that is likely to be a burglar, access should be denied. The starting location of the car, the presence/absence of the most frequent driver (see [3] for a preliminary notion of proximity-based access control), and the status of the car, all contribute to the decision as to whether the access to information in a navigation system (or other systems in the car) should be allowed. If we add the other issues which arise when a car is used by rental companies, where there is no normal driver, but an authorized driver or a set of authorized drivers has temporary authorization to drive the car, thus requiring configuration of the car access control systems on each rental, and revocation of the authorization when the car is returned, the situation becomes even more complex.

Dealing with all the dynamic contexts which could arise in the automobile and combinations of drivers cannot be achieved by means of current access control mechanisms. A dynamic access control scheme, where the navigation system, the car remote, and the car ignition system, make their own access control rules based on the current context is needed. The intelligence cannot be put into a management system since the connectivity of the car to the management system would frequently be interrupted depending on the usage pattern of the car.

2.2 Partner Information Access

Modern enterprise and university research has become more and more collaborative. Whether it is research, product development, or operational service creation, it is very common for several organizations to collaborate together to engage in a joint task. Each such cooperation requires access to resources and services that only the members of the alliance are allowed to access.

Most enterprise and government systems, however, remain bound to the concept of a static access control, where users outside the enterprise are considered potential threats and restricted in which services they can use. Although the use of client-side scripts and programs (e.g., Java or JavaScript code for video meetings) is prevalent, static protection systems would prevent such code from entering the enterprise network, and access to many useful services would thus be eliminated. The situation is made worse since many users frequently work remotely, so access control based on source and destination network addresses of users proves woefully inadequate.

Without denying the need for security for the enterprises, the identity, business arrangements and trust relationships must be considered to determine the dynamic context of a network access request, and enhanced access to services provided to trusted partners. Given the myriad of partnerships, alliances and collaborations any large organization engages in, the access granted needs to depend on the context and sensitivity of the information, and may need to consider history of previously granted accesses or previously accessed information.

The situation is shown in Figure 1. Organizations A and B have an existing agreement which allows them to share a set of common services. When employee X from Organization A tries to access a web conferencing service which is operated in an authorized manner by organization D under authorization by organization B, organization A's firewall access control policies should be smart enough to allow access to the client-side code that is needed for service participation on X's machine, as long as the web conferencing session includes an employee Y from the partner organization. However, if the same web conferencing service is being used by Organization C, which does not have an existing agreement with Organization A, any session which is an attempt by employee X to access the service with an employee of organization C should be denied. So even though the end-points of the communication for the two requests are the same, the context of operation, and the identities and affiliations of who else is in the conference as well as the relationships between organizations, need to be taken into account to determine whether access is to be allowed or not, and whether the code being sent by organization D is a threat to organization A.

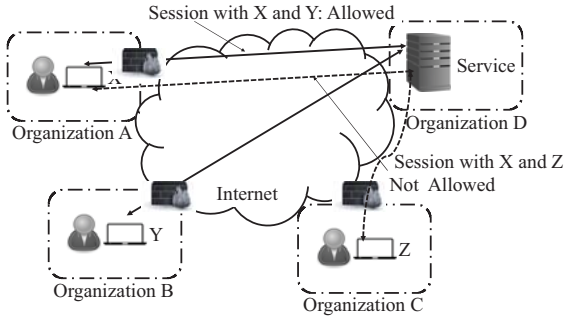

Figure 1: Partner Information Access Scenario

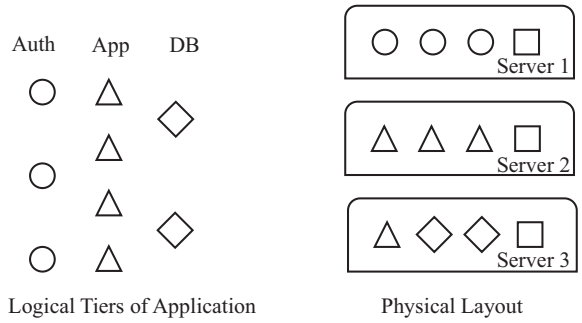

Figure 2: Elastic Data Center

While the software in an enterprise would typically be able to contact a management server and get the required dynamic access control policy, there are several organizations where partners cannot assume constant connectivity to a management service. Some examples include coalitions operating in a remote region, companies exploring for oil or minerals jointly in a remote area, or ships from different navies conducting a joint exercise.

2.3 Elastic Data Centers

Technologies such as virtual machines, Docker containers, Kubernetes and microservices-based architectures enable the concept of elastic services. In elastic services, a service may be running as multiple parallel tasks (a task being either a virtual machine, a container or a parallel service or micro-service instance).

Static access control mechanisms require setting up access control rules in different components of the system. With the high degree of dynamicity in the environment, manual configuration of such access control policies is challenging. An alternative would be to have a centralized system which generates access controls for the configuration as it changes. The master control would then be responsible for keeping track of the current configuration of a dynamic system and ensuring that the right access controls are in place.

However, centralizing the access control, while simple in concept, can be fraught with challenges when the set of virtual components are changing rapidly, and each change requires accessing a central location. There will be race conditions where the master configuration is out of sync with the actual configuration of the dynamic system, especially if every configuration change requires accessing the master server. It would be highly desirable that such central access be made for less dynamic aspects of the environment, while more dynamic issues such as access control for each device be able to be determined automatically by each element in the system.

An illustrative example is shown in Figure 2, in which an application running at the data center consists of three tiers of applications, let us say a tier of authentication services (circle shape), a tier of web application services (triangle shape), and a tier of database services (rhombus shape). The square shapes are for services that may belong to some other applications. Each tier consists of one or more services performing the task required of it. The authentication services, the application services and the database services are all virtual machines (or other virtual entities) which need to be configured physically so they allow access only to the other virtual machines belonging to the same application. The triangles permit access to the circles, and the rhombuses permit access to the triangles. The circles, triangles and the rhombuses should not permit access to the squares. The number of circles, triangles, rhombuses and even the squares in the system can vary over time.

Physically, the services can be running on physical servers in any configuration and share the physical capacity of the system. A possible configuration is shown to the right of Figure 2. This gives the ability of squares to send requests to the triangles, circles and rhombuses on the same or other machines. However, each virtual machine needs to protect itself to only allow requests from the tier ahead of it. This protection needs to be updated as new instances are added in each tier.

While the data centers do have a good connectivity to a local server that can provide them with dynamic access control guidelines, the rapid rate of changes in the system may make it more efficient if the virtual machines/containers could generate their own access control policies under a slower-rate supervision of the management server.

2.4 Coalition Resource Sharing

Military coalitions are becoming common with different countries joining forces in order to deal with the challenges of an increasingly complex world. When such military coalitions are formed, they often need to form dynamic communities of interest (CoI). A CoI consists of people, resources and infrastructure established for a relatively short period of time, e.g., a CoI may consist of a group of U.S. and UK soldiers that will be distributing food packets to a remote village which has been impacted by floods, and the operation will last for a couple of days. When a CoI contains local civilian organizations or NGOs, the degree of trust among organizations can vary widely.

An example of a CoI is provided in Figure 3, in which partner *A* provides people, drones and a mule, partner *B* provides people,

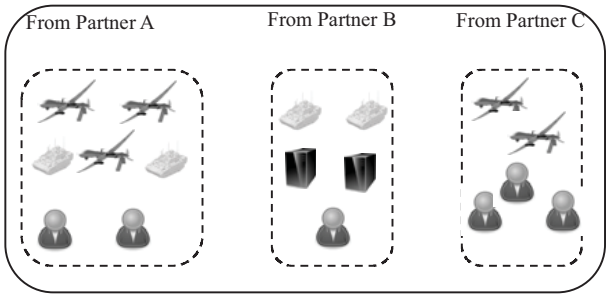

Figure 3: A Community of Interest

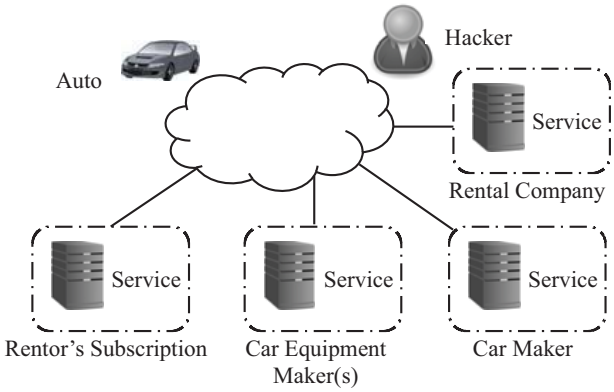

Figure 4: Client Side IoT Security Scenario

computing infrastructure and automated mules, and partner *C* provides drones and people. When dynamic CoIs are formed, resources among different coalition partners need to be shared and pooled together for the common task. The access control and authorizations for access to the shared resources need to be granted dynamically as these CoIs are formed and removed as the CoIs disband. Since the resources that are used for the CoI may also be used for other operations that a country may not want to share with other coalition partners, access to resources needs to be granted based on the situation. If the humanitarian mission mentioned above comes under an attack, and an attack drone with U.S. controls needs to expand its access control to its UK colleagues, then that access control ought to be provided based on the context. Given the range of dynamic situations that can arise in the context of a military operation, access control must be determined by resources themselves on a dynamic basis depending on the context, and re-evaluated as the context changes.

2.5 Client Side IoT Security

Devices that used to deal with physical processes, e.g. heaters, coolers, elevators, automobiles, centrifuges etc., are getting connected to Internet based services at an exponentially growing pace. This is happening in several domains, ranging from industrial manufacturing, automobiles, and semiconductor foundries to banking and homes. The benefits of connecting the devices to Internet based services is immense, allowing for dynamic updates to software and firmware, sharing of insights gained from different devices, the ability to mine the data for insights into operational efficiency, quick trouble-shooting and improved productivity. However, as more and more of these devices are getting connected, the risks associated with Internet based black hat attackers gaining access to these devices, and causing irreparable damage is also increasing.

As these devices are being accessed from the Internet, they need to be safeguarded. The accesses being provided to them must not cause a security risk, for example, being compromised and used in botnets [1], or even worse a safety risk. When safety is at risk, e.g. in an emergency situation, security norms may need to be violated. Static access control, which allows for restricted access to services from users on the Internet is inadequate for the task at hand. The person accessing the service from the Internet can steal credentials and masquerade as an authorized user. The IoT system which is being accessed must be able to protect itself when given dangerous commands over the Internet. This determination needs to be dynamic and based on the current context in which the device is being used. A simple and innocuous command, e.g., asking an automobile to accelerate from a service that appears to be from the manufacturer of the automobile, may be perfectly fine if the car is parked imperfectly in a parking lot, but would be dangerous if that causes the car to hit something in front of it, and is being sent by a hacker.

An illustrative example of the access control that an IoT device needs to enforce is shown in Figure 4. An automobile is being operated by a rental car company and is currently rented out to a driver. The automobile is connected via a cellular network to the Internet, and interacts with: (a) a tracking service run by rental company; (b) a maintenance and diagnostic service run by the maker of the car; (c) a maintenance and diagnostic service run by the supplier of the car; and, (d) an independent service that provides emergency assistance for which the renter has a subscription. Since the cellular network provider can change across different regions within which the car is driven, the car needs its own protection service.

Each of the services is allowed its own set of data restrictions which can change over time. The tracking service may be allowed access to the location of the car when the car is not rented out, but the tracking of an individual renter location when the car is rented out is prohibited. Each of the subcontractors who provided material to the car may be running a service, but they are only allowed access to some of the services within the car. Furthermore, the diagnostics that are permitted on the car have to be performed when the car is not being driven. Access control to car functions for services that need subscription from the renter needs to be added dynamically.

Furthermore, any requests to control the car from any of the service providers over the Internet needs to be validated to see that: (a) they are indeed from a valid service and not a hacker on the Internet; and, (b) they are permissible under the current context that the car is operating in. Similarly, any request going out from the car needs to be checked to ensure that: (a) it is going to one of the

authorized services relevant to the car; (b) it is not leaking sensitive information about the current user of the car; and, (c) is not due to the car being used for malicious purposes such as becoming part of a botnet. All these decisions vary over time depending on the rental status, the car location, and the condition of the automobile and its components.

Because of the dynamic nature of the access required, the car needs to be able to decide for itself which accesses are allowed, and which ones are not. The car is in the best position to assess its state, decide whether the requests coming from the Internet are worth addressing, and thus protect itself. Interactions with other cars in the neighborhood may also play a role in the decisions made by a single car.

In addition to the car, similar dynamic access control concerns may arise in many other IoT environments. Wearable health-care systems worn by a patient and the health-care providers may need to interact together, and each interaction may require adjustment in access control rules. In [5], a system which generates new access control rules when new devices are encountered is described. While they do not change access control rules, an environment in which people leave hospital premises would also require modification and changes in the access control rules.

The key requirement for access control as shown by these scenarios is that the right access control is very dependent on the context in which a request is being made. The same access request (i.e., from the same subject to the same resource) may be safe (or desirable) to grant in some conditions, while it may be undesirable to grant in other conditions. The resource being accessed is usually in the best position to determine whether the request should be allowed, and we need to develop access control mechanisms by which the resources themselves are able to decide whether a subject access request should be granted. We refer to these types of access control requests as dynamic access control approaches.

3 DYNAMIC ACCESS CONTROL APPROACHES

The previous section argued that we need to enhance static access control models, including role-based access control and more recent attribute-based access control, to deal with more dynamic scenarios. In this section, we present some potential approaches for a dynamic access control which depends on the local context. In order to do so, we introduce the abstraction of a device, which is a collection of resources and subjects. The devices can host subjects that can access resources on the same or on other devices. One way to understand the concept of the device is to consider each device as a collection of long running services. Each service can invoke other services, acting both as subjects as well as resources. Our end goal is to have each device be capable of determining the access control policies for its resources depending on the context in which the device finds itself.

3.1 Dynamic Refinement from Environment Specification

One approach to allow devices to generate their own access control policies is to provide them with a description of the environment in which they are operating, giving them a high-level guidance on how they ought to operate as regards to providing access control to other entities in the system. As an example, a manager can provide devices with information about other types of devices that are found in the system, and mechanisms to validate the new devices entering the system. The devices could then discover other devices in the system, and then decide to give them access on the basis of the restrictions provided by the manager, and also based on information about the devices (such as type of device [18], and organization or user owning the device).

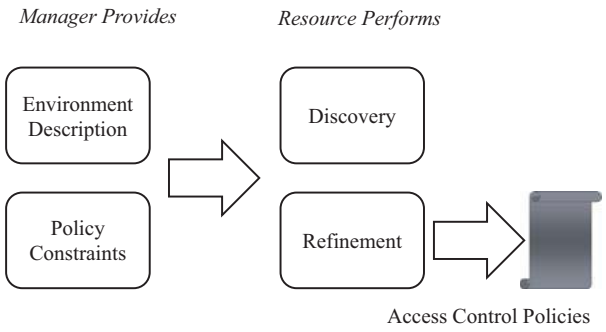

Figure 5: Dynamic Policy Refinement

The concept of dynamic refinement is schematically described in Figure 5. A set of guidelines from a management system provides each resource with a description of the environment in which it could expect to operate. This description would include the other types of subjects and resources it should expect to encounter during operation, and the properties expected of those other types of subjects/resources. Each device would also have a discovery process which would allow it to discover other entities in the system, both the other types of subjects, and other resources that are present in the environment. During the discovery process, resources may share accesses made by different subjects and thus maintain a distributed log of accesses that are made.

Each resource would also be provided with a set of constraints on the type of access control policies that they should explore. These constraints could be used to check the validity of the policies that the resource uses for its operation, some examples being a policy generation grammar, or a template for policies. The resource uses a refinement process to translate the constraints into a set of access control policies, which can then be used on a per request basis to determine access. The policies could be regenerated either periodically, or when a specific event happens, e.g., new subjects or resources are discovered in the environment. As the policies are regenerated, the high-level constraints are mapped (refined) into a part of the access control matrix with the currently discovered set of subjects, resources, their attributes and/or roles, and the history of the previous requests. The high level constraints may be defined in a teleo-reactive manner [5].

Such a system is described in [21], in which the manager defines the roles that devices may play within a large group, and provides each device with an interaction graph with the different roles in the system. The devices act as subjects for accessing services present on

other devices, with the services playing the role of resources. Each device is also provided with a template that allows it to generate access control policies based on the roles assigned to a new system. The combination of the three techniques, namely: (a) defining an interaction graph among different roles; (b) providing a way for devices to discover other devices in the environment and validate their roles; and, (c) providing each device with a mechanism to generate policies, e.g., a policy template or a policy generation grammar, allows devices the ability to generate access control policies in a dynamic manner.

Such dynamic refinement allows devices to generate their own policies in scenarios where the set of other subjects/resources in the environment fall into a few broad classes with known relationships among them, and the set of specific subjects/resources can be easily discovered. Examining the various scenarios presented in Section 2, the scenarios for dynamic data centers and coalition resource sharing can be supported well by dynamic refinement approaches.

3.2 Device Model based Access Control

Another approach for devices to determine their own access control is to provide them with a model of how their safety would be impacted if they were to allow any operation to be executed on them. For example, use of computation intensive services may deplete the energy resources of a device making it unable to answer some critical requests. The devices can use such a model to determine whether or not they should allow access to any resource or service that is being requested of them.

If we consider each resource as a collection of services, to which requests are being made by an external entity, the access control problem can be seen as a go/no-go decision on every attempt by the external entity to access a specific service. If the device has a model that determines whether the result of that particular service invocation would be safe or unsafe for that device, it can use that model to determine whether or not that access ought to be allowed.

The approach is described in Figure 6, where the device model is used to assess the validity of each request. On each request, the model of the device is used to understand the impact of approving for the requesting subject to access the requested resource. The device determines its state by reading its own sensors (including the status of the resources present on the device, and any prior history of requests from other subjects) and executes the model to assess the impact of the request. The assessment can be done by means of the digital twin paradigm [10], in which the device performs a simulation to understand the impact of the request.

As an example, if we consider that physical devices have a specific model that can map any state they have into a safe/unsafe classification, then every request can be run through a simulation of the model to determine whether the result of that service invocation is safe or not. Some of the simulations may be run when a device is put into operation, resulting in a static set of access control policies. Other types of requests would be done in a dynamic manner when a new request is received, and the resulting new state computed for the device as a function of the current state and the new request. This allows for a dynamic generation of the access control rules on a per-request basis.

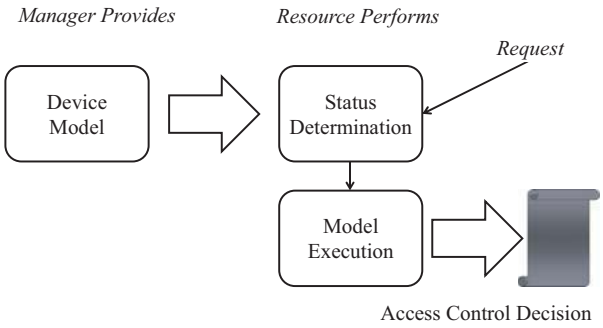

Figure 6: Model based Access Control

It is important to notice that such an approach is also critical for devices that are actuators and thus can execute actions that may impact the physical environment and thus undermine safety. Furthermore, since simulations of the systems can be time-consuming, the devices may choose to run simulations periodically to generate a set of access control policies which can be used as a faster way to do the determination of access on a per request basis. For access control decisions which can be modeled well, and the focus of securing access is on a specific device, model based access control can be very useful. It can be used effectively for physical safety of client-side IoT devices and be a building block for assuring safety, partner information access, and coalition resource sharing.

3.3 Networked Environment Model driven Access Control

While the previous subsection has discussed the model of individual devices, many security scenarios may involve considering the state of the other devices that are in the environment as well. The safety of a device may depend not just on its own state, but on the state of other devices in the environment. In addition, when dealing with a networked system of devices, many of which have actuation capabilities, assuring safety becomes even more challenging as the safety analysis becomes more complex as one must collectively consider the combined effects of several actions, each of which when executed alone is safe but when combined may introduce safety risks.

At a high level this system would work as shown in Figure 7. The manager provides an environment description to the device, and also provides models for understanding the behavior of each device. The device runs a discovery process which allows it to determine the set of other devices that are present in the system. The model execution phase performs the simulation after reading the status of all the devices in the system, and the simulation would consider the impact of all the devices in order to determine its final go/no-go decision.

As an example, if a building cooling system and a building heating system are controlled by independent controls, it is possible to launch an attack on the system by setting them both to run continuously, asking the cooling system to maintain a temperature below a threshold and asking the heating system to maintain a temperature

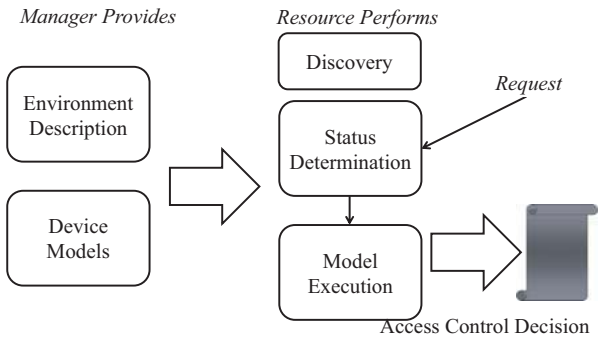

Figure 7: Networked Environment Model

above that threshold. On the other hand, if the devices have a way to discover other systems in the environment and their state, they could determine that a request made in this manner could lead to an unsafe situation and one or the other could adjust their settings to avoid running continuously. Therefore, in such context the devices must coordinate their access control decisions, rather than taking these decisions independently. Thus, some form of coordinated access control is required [20], combined with models of the devices and their deployment environments.

Networked environment models can be used to secure scenarios like car navigation systems and client side IoT security. They can also be used to provide dynamic access control for scenarios that use dynamic refinement such as elastic data centers and coalition resource sharing.

3.4 Machine Learning based Access Control

Whenever a model needs to be deployed, its specification and construction can be a difficult and tedious task. In some environments, models can be automatically built for the devices using machine learning techniques. In these cases, an initial specification could define the safe and unsafe state of devices to the best possible ability of a human. As the system operates in live environments, its state variables are recorded and the impact of any service request on the state variables also recorded. The system could then build the model for the device based on its actual operation.

Machine learning can also be used to characterize the behavior of external entities who are using the system. External entities which are using the system frequently with benign impact can be allowed access to the system. Such access has to be balanced against the threat of someone launching a reputation-based attack on the system. Similarly, entities which are present/endorsed by such an entity can be provided access. However, an unknown entity would be denied access to the system, unless it is in the presence of a known entity.

The machine learning aspect can be added to all three of the previous approaches described in this section. The machine learning algorithms can be used as the mechanism to refine the policy constraints into a set of access control policies or be used to make the determination on a request by request basis [17]. In the specific scenario of the car navigation system, a machine learning based system can recognize the faces of people who are present frequently with the driver in the car and provide them access to all required information. An unknown entity that tries to operate the car with the company of a known entity is also provided access. However, a completely unknown entity can be refused access to car navigation systems. Dynamic learning of entities using machine learning can provide a much better approach to handle access control policy generation than the other approaches discussed above.

3.5 Sharing of Models

While machine learning allows the ability for devices to determine their own models, it would provide a greater flexibility if devices could share the models that they have learnt with each other. This sharing of knowledge on how to determine policies in general has been introduced in [4] and can be used to further improve dynamic access control policy determination.

Devices in this model act like a community of managers of other devices. By contrast, in all previous approaches a (human) manager is typically responsible for providing some information to the devices. The devices can use that information to dynamically update any models that they have received. In a model sharing approach, instead, each device gets models from its peers and can update them locally – so the peer group is acting like several managers feeding models to each other.

One a device has updated its model, it can share the updated model and the conditions under which the updated model is best suited. Each device can use the set of models shared with it to determine if it should use the new model, the original model provided to it, or the model it has learned, the choice dependent on the conditions that accompany each model.

While the manager-provided original model would be the default mode of operation, such sharing of conditions and models would allow the devices to quicken the pace at which new models are created in the system. Models built on machine learning approaches could take a significant amount of time to be created – until sufficient data has been observed. The sharing of models allows the learning process to become much faster and efficient.

As with the machine learning approach, the model sharing approach can be used in conjunction with any of the three earlier approaches that were described.

4 RESEARCH CHALLENGES

We can generalize the various approaches for resources to generate their own access control policies described in Section 3 into an abstract one. In this abstract approach, each of the resources is provided some "high-level guidance" from a system manager; the resources then conduct a "discovery process" to determine their "situational context" of operation and generate their access control policies using some type of "learning" approach.

In order to enable the resources to learn their own access policies, several research challenges need to be addressed. Some of these challenges include:

(1) Defining the right "high-level guidance": Access control policies were introduced as the general approach for simplifying the application of access control mechanisms to a divergent

set of use-cases. We need to define a similar concept for "high level guidance" which will be applicable to various scenarios and contexts in a general way. The semantics to be used for high-level guidance needs to be defined. While that semantics can be defined for each of the different generation approaches discussed in Section 3, an open challenge remains that of identifying the abstraction that will create the concept of the "high level guidance" that will be applicable to a broad set of use-case scenarios regardless of the approach used to generate policies.

(2) Defining the right discovery process: Generating the right access control policy requires each of the resources in the environment to be aware of the other devices in the environment. Given the lack of widely adopted standards for discovery of resources, and the many security exposures that may result from a flawed discovery process, new discovery techniques that can provide sufficient security in many dynamic contexts are needed.
(3) Determining the situational context: The definition of the situational context, as well as its determination, remains a hard and open research challenge. Access control needs to account for not only the other resources that any given resource interacts with, but also for the resources that may indirectly use the information provided by that resource. Understanding the situation on the ground in sufficient detail to provide workable security is a challenge that can be solved on a case-by-case basis, but a general formulation remains an open challenge.
(4) Policy deconfliction without human guidance: When access control policies are generated by the resources, they may conflict among themselves, or with other policies that are imposed on the system to meet other requirements, such as performance, reliability or availability requirements. In the presence of a human, such conflicts can be resolved and deconflicted using the human provided input. Similar algorithms, which can operate without a human in the loop, need to be developed.
(5) Controlling emergent behavior: When resources create their own policies, there is a danger of uncontrolled emergent behavior [15]. Emergent behavior is behavior shown by a collection of systems which was not originally designed but happens due to the dynamic interactions among different intelligent systems. Such emergent behavior can be controlled, but the design needs to account for that. The control of emergent behavior in complete generality remains an open challenge, but there may be solutions that can be developed for specific domain of access control.
(6) Learning-related challenges: The approach used by resources to learn their access control policy has to use some type of learning approach, which creates a generalization based on its past experiences, but such learning mechanism has its challenges. The set of experiences for access control in each use-case has only a few small data points to draw from, which means that either existing learning techniques that use only a few data points need to be used, or new learning techniques requiring small data for training needs to be created. Learning also exposes new threats, e.g., a malicious actor may manipulate the data so as to force the device to learn an invalid model which could lead to generation of insecure access control policies. Solutions to such adversarial learning in uncontrolled environments need to be developed.
(7) Increase in attack surface. The increase in complexity due to autonomous generation of policies by devices can lead to new types of security vulnerabilities and threats, e.g. the adversarial learning example given above. This could lead to challenges with established assurance processes in enterprise and government networks, and the right way to provide assurance for systems using dynamically generated access control policies need to be developed.
(8) Power, computation and communication efficiency. While in the past several years there have been tremendous advantages in the computational capacity of small devices, devices that need to learn and manage their own policies may find themselves unable to use approaches that require specialized hardware, or significant computational power, e.g. learning mechanisms based on deep neural networks. For any solution developed with dynamic access control policies, system design should incorporate mechanisms that take into account the limitations on a resource, which usually manifests itself as limited amount of battery power, computational capacity or the ability to communicate at high bandwidth rates.

A more general discussion on research challenges associated with devices that try to generate their security policies in a broader context can be found in [16].

5 DISCUSSION

In this paper, we have looked at different scenarios that require a dynamic approach for access control, and require devices to generate their own policies, usually based on the context in which they are operating. The devices may operate under the constraints provided by a management system, or they may use models provided by a management system to determine their own access control policies. They can also use machine learning to learn their own models or improve upon the model that is provided to them by a machine learning system. Also access control mechanisms need to be tightly coupled with authentication systems so that access control can also be based on trusted information about the identity and the properties of the entities requiring access to devices.

Of the approaches discussed in this paper, the approach for dynamic refinement is closest to the current approaches for access control management. It can be viewed as an adaptation of role-based access control for dynamically formed groups that have a well-defined discovery process. Model based approaches provide more flexibility in the access control generation but face the challenge that a good model should accompany each device. The approach to use machine learning to learn the models for devices, and have the system determine its model and context is the most flexible option among all of them. Sharing the learned models among different devices can further improve the ability of devices to dynamically determine their context.

We believe that each of these approaches holds immense promise for improving access control for future systems and can be built upon to show their effectiveness in more details in several scenarios.

ACKNOWLEDGEMENTS

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.

REFERENCES

[1] E. Bertino and N. Islam. 2017. Botnets and Internet of Things Security. *IEEE Computer* 50, 2 (2017), 76–79.
[2] E. Bertino, P. Bonatti and E. Ferrari. 2001. TRBAC: A Temporal Role-based Access Control Model. *ACM Transactions on Information and System Security (TISSEC)* 4, 3 (2001), 191–233.
[3] M. S. Kirkpatrick, M. L. Damiani and E. Bertino. 2011. Prox-RBAC: a Proximity-Based Spatially Aware RBAC. In *Proceedings of the 19th ACM SIGSPATIAL International Symposium on Advances in Geographic Information Systems (ACM GIS)*. ACM, 339–348.
[4] E. Bertino, G. de Mel, A. Russo, S. Calo and D. Verma. 2017. Community-based Self Generation of Policies and Processes for Assets: Concepts and Research Directions. In *Proceedings of the 2017 IEEE International Conference on Big Data (BigData)*. IEEE Computer Society, 2961–2969.
[5] S. Marinovic, K. Tiwdle, N. Dulay and M. Sloman. 2010. Teleo-Reactive Policies for Managing Human-centric Pervasive Services. In *Proc. International Conference on Network and Service Management*. 80–87.
[6] A. K. Nayak, A.Reimers, N. Feamster and R. Clark. 2009. Resonance: Dynamic Access Control for Enterprise Networks. In *Proceedings of the 1st ACM Workshop on Research on Enterprise Networking*. ACM, 11–18.
[7] R. Sandhu , E. Coyne, H. Feinstein and C. Youman. 1995. Role-based Access Control Models. *Computer* 29, 2 (1995), 38–47.
[8] M. Bartoletti, P. Degano, G. L. Ferrari, and R. Zunino. 2015. Model Checking Usage Policies. *Mathematical Structures in Computer Science* 25, 3 (2015), 710–763.
[9] M. Gilbert. 1993. An Examination of Federal and Commercial Access Control Policy Needs. In *Proceedings of National Computer Security Conference on Information Systems Security*.
[10] M. Grieves and J. Vickersg. 2017. Digital Twin: Mitigating Unpredictable, Undesirable Emergent Behavior in Complex Systems. *Transdisciplinary Perspectives on Complex Systems* (2017), 85–113.
[11] E. Freudenthal, T. Pesin, L. Port, E. Keenan and V. Karamcheti. 2002. dRBAC: Distributed Role-based Access Control for Dynamic Coalition Environments. In *Proceedings 22nd International Conference on Distributed Computing Systems (ICDCS)*. IEEE Computer Society, 411–420.
[12] K. Knorr. 2000. Dynamic Access Control through Petri Net workflows. In *Proceedings of the Annual Computer Security Applications (ACSAC)*. ACM, 159–167.
[13] V. Hu, R. Kuhn, and D. Ferraiolo. 2015. Attribute-based Access Control. *Computer* 48, 2 (2015), 85–88.
[14] G. Wang, Q. Liu, and J. Wu. 2010. Hierarchical Attribute-based Encryption for Fine-Grained Access Control in Cloud Storage Services. In *Proceedings of the 17th ACM conference on Computer and Communications Security (CCS)*. 735–737.
[15] M. Mataric. 1993. Designing Emergent Behaviors: from Local Interactions to Collective Intelligence. In *Proceedings of the Second International Conference on Simulation of Adaptive Behavior*. 432–441.
[16] S. Calo, E. Lupu, E. Bertino, S. Arunkumar, G. Cirincione, B. Rivera and A. Cullen. 2017. Research Challenges in Dynamic Policy-based Autonomous Security. In *Proceedings of the 2017 IEEE International Conference on Big Data (BigData)*. IEEE Computer Society, 2970–2973.
[17] Q. Ni, J. Lobo, S. Calo, P. Rohatgi and E. Bertino. 2009. Automating Role-based Provisioning by Learning from Examples. In *Proceedings of the 14th ACM Symposium on Access Control Models and Technologies (SACMAT)*. 75–84.
[18] M. Miettinen, S. Marchal, I. Hafeez, N. Asokan, A.-R. Sadeghi and S. Tarkoma. 2017. IoT SENTINEL: Automated Device-Type Identification for Security Enforcement in IoT. In *Proceedings of the 37th IEEE International Conference on Distributed Computing Systems (ICDCS)*. IEEE Computer Society, 2177–2184.
[19] R. Sandhu and P. Samarati. 1994. Access Control: Principle and Practice. *IEEE Communications Magazine* 32, 09 (1994), 40–40.
[20] A. C. Squicciarini, M. Shehab and F. Paci. 2009. Collective Privacy Management in Social Networks. In *Proceedings of the 18th International Conference on World Wide Web, WWW*. ACM, 521–530.
[21] D. Verma, S. Calo, S. Chakraborty, E.Bertino, C. Williams, J. Tucker and B. Rivera. 2017. Generative Policies for Autonomic Management. In *Proceedings of 1st IEEE International Workshop on Distributed Analytics InfraStructure and Algorithms for Multi-Organization Federations (DAIS)*.