Security Fabric – Tailored Trustworthy Space
Part 1: Flexibility Based on Policy Management

Charles Speicher

McAfee
3965 Freedom Circle
Santa Clara, CA 95054-1203

Abstract

The Security Fabric framework is a commercial implementation of the "tailored trustworthy space" strategy developed by the White House Networking and Information Technology Research and Development (NITRD) Program and promoted by the Department of Energy for maintaining security of end-to-end intelligent grid environments. For end-to-end security, no one size fits all implementation is possible because of the variety of specific installation needs. The approach must have the flexibility to dynamically adjust to the policies that are appropriate to each individual situation. It must be suitable for the very smallest of situations, but it must also scale uniformly to support the largest of situations which involve millions of managed objects. In that there will be no single victor in the commercial marketplace for a single proprietary design for matters such as key management or other major functional concepts, the Security Fabric provides an interoperable framework that comfortably supports many solutions for individual components using varying standards that can be tested and certified for interoperability.

This is the first of a three part series of papers that describe the complete set of dimensions for the Security Fabric:

- *Part 1: Flexibility Based on Policy Management* – Develops the use of policy execution environments within distributed devices in a system to provide the "tailorable" aspect of a Tailored Trustworthy Space.
- *Part 2: End-to-End Management and Security* – Provides insight on the system and network management elements needed to support the end-to-end system "space".
- *Part 3: Close-up on Security Management* – Provides focus on the "trustworthy" aspect of an end-to-end management system controlling the framework.

1. INTRODUCTION

In May of 2010, the Department of Energy in conjunction with the Networking and Information Technology Research and Development (NITRD) Program published a handful of very important strategies they would like to use to come to grips with the complexity of the security issues surrounding the intelligent grid in the United States [1]. Prior to this time, important research work had been done in individual technical areas, but nothing until then dealt with the end-to-end nature of how the electric grid operates as a system of systems, or how to deal with the growing enormity of the problem. The problem increases with every day as more and more communications-based electronics are added to tune and control the grid, usually through IP communications, and through accidental exposure to the public Internet.

The official term for this end-to-end security technique is a *tailored trustworthy space* or "TTS" for short. A *trustworthy space* is an isolated collection of devices, services, policies, and data that are meant to interact in a secure, private, and reliable fashion. *Tailored* indicates the need for handling the multiplicity of situations that comprise an end-to-end system and a need to use design patterns in different combinations to be able to mass customize appropriate solutions for the different circumstances common to the intelligent grid. The following diagram shows this concept of isolating a finite number of devices that must interact with one another to carry out a specific mission.

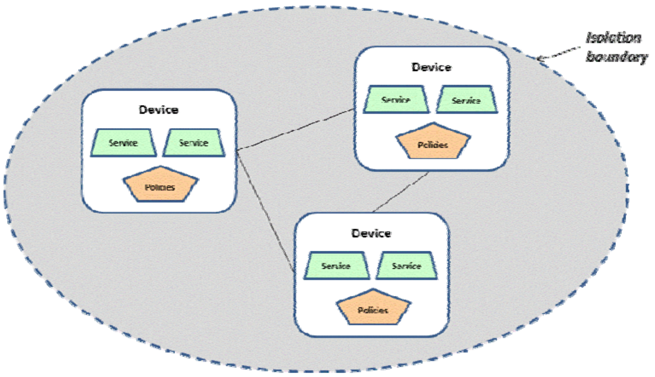


Figure 1. Control systems on the intelligent power grid are collections of devices.

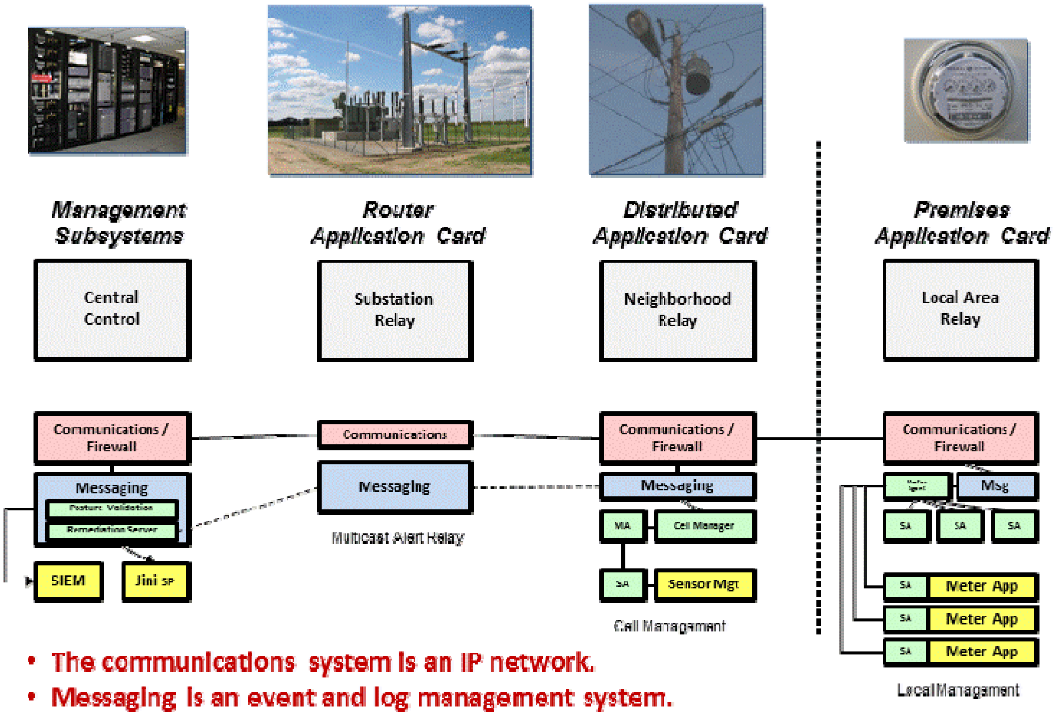
The actual deployment may of course have the devices interconnect over a cloud of other communications facilities which may or may not be secure unto themselves, but as will be shown, there are no specific assumptions that need to be made about the communications medium.

The approach suggests that the firmware operating inside the devices use a *service oriented architecture* (SOA) to modularize and organize its fundamental functions. In addition, the recommendation is for each device to also use a special service to house all local policy logic.

The main purpose of using policies is to guide the behavior of application services in devices by extracting and externalizing business logic from applications into sets of rules. In general, these rules are human readable and in some cases they are expressed as UML or other visual programming languages [2]. Modification of policies does not affect the underlying data and applications, resulting in a higher level of maintainability, variability, and manageability.

> **Note**: One of the challenges that this basic approach can help is that an "actor" in a TTS may not be a self-contained device, but rather a grouping. Clever use of SOA and bindings can help. This technique addresses this major problem of diversity – both for ongoing differing abilities as well as for retrofitting.

Control systems on the intelligent power grid are collections of devices that need to be TTS configurations to preserve the security of their operations despite accidental or intentional attempts to disrupt their operation. The physical environment being protected by the Security Fabric can vary with the situation. The diagram below illustrates a common end-to-end sequence of device subsystems that is typical in power distribution.



- The communications system is an IP network.
- Messaging is an event and log management system.
- Management interaction is separate from application interaction.

Figure 2. A typical situation involving power distribution.

Many devices are application-driven, but some are also intended to control and manage others in a hierarchical fashion. All devices in the end-to-end chain of control have policy decision and enforcement responsibilities that need to be tailored to the individual circumstances. Although every situation has the potential to have unique considerations, the application design pattern for the end-to-end control is pretty much always the same in that it is based on the physical pattern of distributing alternating current using substations, transformers, and power grooming devices worked out originally by George Westinghouse back in 1886. If there is potential for using a power control device in a power system, then there is equal potential for installing a control device there to measure and or control that power device at that location. (This hierarchical arrangement also offers an opportunity for help in certain scalability matters with regard to management delegation.)

In all communications-based control systems, the pattern is to have devices that need to be controlled, and one or more devices that are used to control them. Intelligent devices have an "agent" that handles all communications and interactions with the managing system, and the managing device has management routines that are meant to handle the coordination inside those devices. This concept is shown in the diagram below.

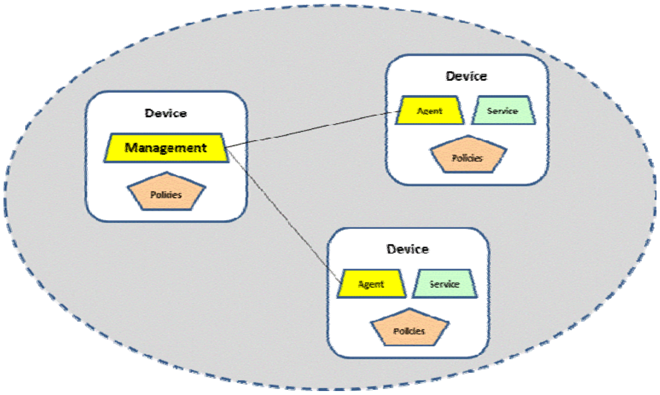


Figure 3. Some devices are managers of others.

To form a TTS of devices where a trustworthy environment is needed but not yet established, one device must take the responsibility of setting up the environment for a new TTS. Actually, for redundancy and reliability of operations, there may be an alternate manager for a TTS, but it must remain in standby mode until called into action so that there is no ambiguity as to who is in charge at any one moment. The multiple managers synchronize with one another and negotiate for seniority and how responsibility will be passed back and forth depending on the health of the devices in the TTS.

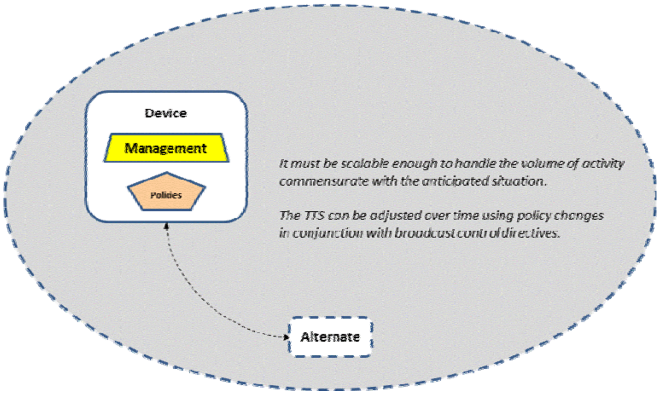

Device
Management
Policies
It must be scalable enough to handle the volume of activity commensurate with the anticipated situation.
The TTS can be adjusted over time using policy changes in conjunction with broadcast control directives.
Alternate

Figure 4. Multiple managers provide redundancy.

The management function has responsibility for creating the basis for the TTS collection, and then builds it up by responding to registration requests by individual devices, including the services running on them and the policy rulesets on them. Although somewhat rare in occurrence, the manager would also be responsible for systematically winding a space down at the conclusion of operations through notifications and unregistering of all the participating devices, if such a shutdown were deemed desirable. (Usually a total shutdown occurs when retiring a system as opposed to a normal operational failover.)

Some environments may actually have millions of participating devices. In an environment where there are a large number of participants, the management system must have a scalable architecture such that it can continue to function, even under duress, and maintain the required performance service level agreements. For large environments, parallel processing of some form is usually used to maintain the required levels of performance and availability.

Another phenomenon of the normal situation is that TTSs can be long lived. During the duration of the life of a TTS, the circumstances are likely to change, and thus the policies needed to control the TTS must change or otherwise evolve to keep up with the situation at hand.

To do this, management services, policies, and configuration data first have to be provisioned into the management device before commencing operations as is pictured in the following diagram.

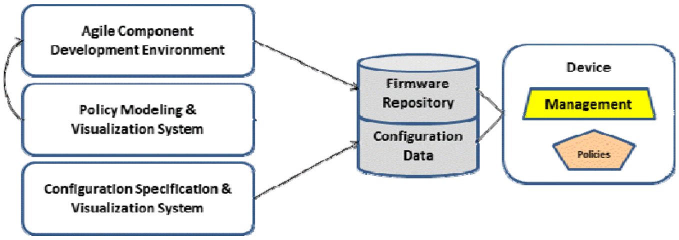

Agile Component Development Environment
Policy Modeling & Visualization System
Configuration Specification & Visualization System
Firmware Repository
Configuration Data
Device
Management
Policies

Figure 5. The management device controls the configuration of the TTS.

Developers create secure, reusable firmware components and policy rulesets using specific security development lifecycle steps. Services as shown in the earlier diagrams are firmware components with attributes and specified operations interfaces that are object-oriented in many ways. Firmware rulesets are also logic elements, but they are organized in slightly more constrained fashion to make it easier and safer to make changes every day. This component strategy for both services and rulesets is fundamental to the notion that eventually intelligent devices must be securely altered in the field in a graceful fashion through fairly simple and straightforward techniques.

In the Security Fabric implementation of a TTS, the firmware, policies, and configuration data can and will all change after the TTS is launched by the Security Fabric Management System to accommodate structural needs. But the strategy for evolution and reuse of common techniques is not a simple matter. Indeed, the NITRD has speculated that the nature of evolution for the elements probably needs to have characteristics somewhat similar to biological entities. In this sense the process of natural selection would allow systems administrators to continue to make improvements to the structure of the TTS in the face of a changing environment.

This sophisticated vision of flexibility and resiliency for the electric grid might seem somewhat esoteric in nature to hardboiled engineers and computer programmers steeped in knowledge of how many things have worked when such flexibility and resiliency did not seem to be important requirements. But examining how artificial intelligence systems have evolved, there are interesting concepts that we believe will be useful in being responsive to the vision of structuring some of the key control items. These concepts are fundamental to evolving the grid from how it works today to how people would like for it to control itself in the future. Students of Design Patterns [3] will recognize the technique for creating new patterns from old. One of the most interesting programming structures derived from this seminal work is called the “Genetic Pattern.” This software abstraction of the various important configuration patterns is actually based on the genetic pattern inspired by biological DNA replication. The detailed explanation of the use of the genetic pattern for both structure as well as policy logic is beyond the scope of this paper, but the technique is very useful in fulfilling the spirit of the flexibility required for deploying the intelligent grid. As will be developed further in this paper, the control center of the TTS configuration management is greatly simplified through using an abstraction layer that provides the handful of templates that allow the “genetic” pattern to be used to provide the flexibility and simplicity needed for deployment of what would otherwise be a complicated and chaotic system.

In addition to having a set of design patterns needed to keep the configuration under control, the related problem is that the configuration must be organized such that it can be visualized easily by human beings. Situational awareness controls are all built around the premise that both digital and human correlation engines are always needed. The visualization aspect of the configuration is meant to facilitate rapid human understanding of what is in place. This includes the logic as well as the structural aspects of configuration. It also is very helpful in creating rigorous rulesets for decision making that are intended to support critical aspects of the system. The goal is for the human operator in charge to be able to find the structure of the configuration rapidly, see or read the part that is important quickly, and comprehend what the situation is rapidly and unambiguously – so that timely decisions can be made. This may be a technical challenge, but it is nevertheless essential for preserving the reliable operation of the grid – even while under attack.

In terms of visualization, one strategy might be to give policies the look and feel of the Unified Modeling Language use cases and model diagrams – but with the underlying rule template generating machine-specific executable components. It is also essential to be able to see the “as planned” reality as well as the “as built” reality, as well as the “currently operational” reality – very quickly.

Once the management device is configured, initialized, and ready for operation, it is ready to interact with managed devices wishing to join the space and conduct operations and interactions. All the other devices also have their internal management agent for secure interaction with the central TTS environment manager to join the space.

In the Security Fabric framework, during the commissioning of a device before it is ready to be installed, the identity and address of the central manager is securely provisioned into the device. As a scalability feature, when a device is installed and activated during local stage of provisioning (or any other subsequent time), the manager can designate a distributed manager to the device for further delegated management authority. The managed device always retains the right to operate under its local policies if it cannot connect with its manager.

As managed devices power up and register with their manager the fact that they are ready to conduct business, along with what they are willing to do from a security standpoint, the TTS begins to take shape as pictured in the following diagram as the management device welcomes them on-line.

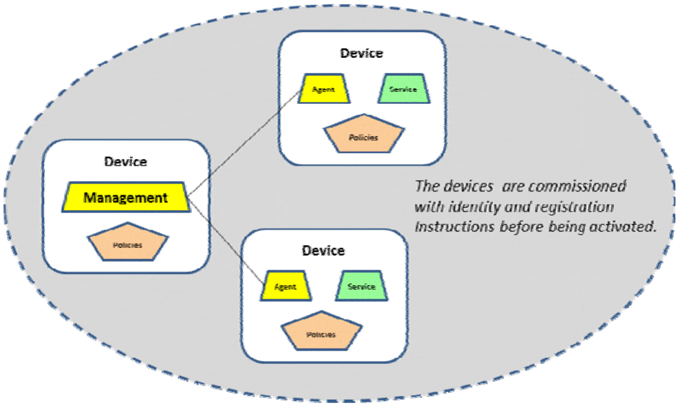

Device
Agent
Service
Policies
Device
Management
Policies
The devices are commissioned with identity and registration Instructions before being activated.
Device
Agent
Service
Policies

Figure 6. Devices must join the space to participate.

The management agent in each device is isolated for secure operation even if the device services are compromised by an attack. This gives the managed device the opportunity to discover that it has been compromised, and then take action to carefully reconfigure and transition the device and its services and policies back to a trusted state. In many cases this transition must take place without actually stopping the device from providing service! If for some reason the managed device loses secure connectivity with its managers, it must rely on its standby policies to continue to make policy decisions unilaterally until it can reconnect with its manager, confer relative to events that took place while off line, and potentially receive new instructions from its manager to remediate any continuing problems.

> **Note**: The logic that controls each local service and the procedural aspects of polices must be secured both in the firmware repository as well as in the field on the managed device. So must be the control parameters as well as the declarative configuration of the policies. The managed devices must be able pull new releases of logic as needed and to roll back to a previous point of integrity for logic as well as control data if an anomaly occurs. This includes rebuilding the entire set of dynamic elements in a device in the field if necessary.

Interactions between devices are actually messages that communicate between services that operate on those devices. Some of the messages are application-to-application interactions, and some of the interactions are agent-to-manager interactions. Ideally, the path for application-to-application messages is sequestered from manager-to-agent interactions so that compromises to applications do not interfere with the remediation that the management channel will use.

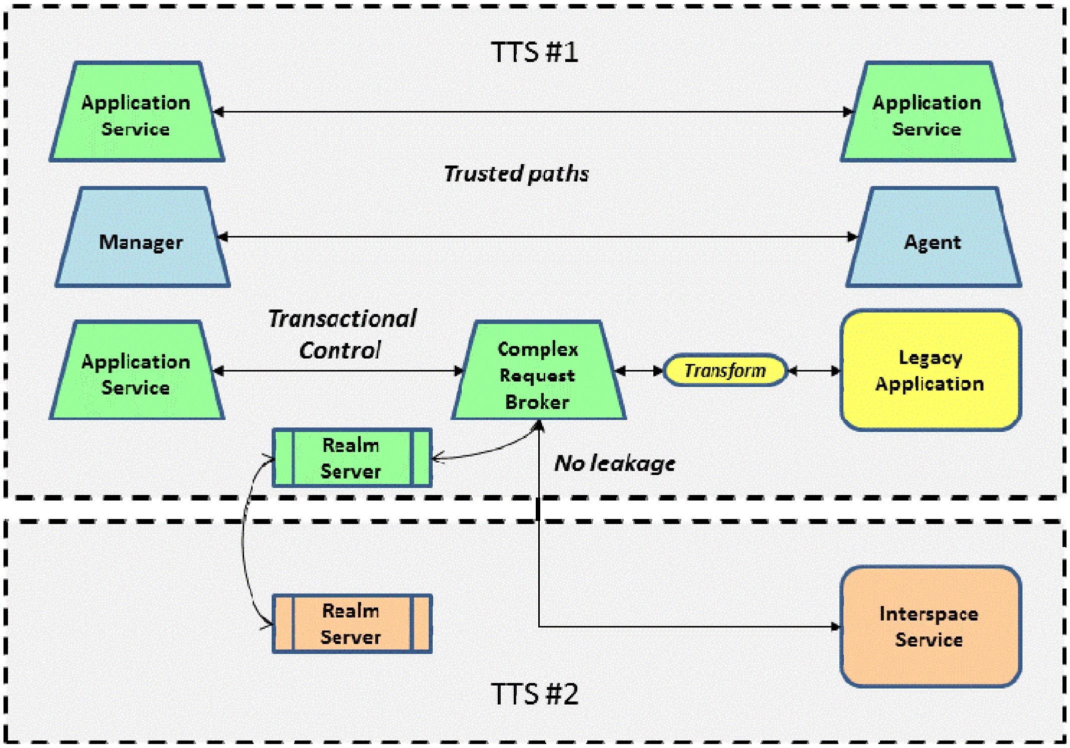


Figure 7. Devices interconnect using trusted communications paths.

Some of the interactions with legacy devices may require transformation of the messages to be compatible with the trustworthy space. Some may actually require transactional controls over longer periods of time to maintain integrity of their state in spite of unexpected outages at inopportune times. Such transactional mechanics will require a trusted third party, sometimes called a complex request broker, to maintain the transaction integrity and to back out partial updates during a recovery operation.

If the requested service actually originates from a managed device that also resides in a separate TTS, extreme care must be taken to not only allow the device to join the TTS space in trusted fashion, but also that the manager of that remote TTS vouches for the trustworthiness of the interspace service. Usually this is done by management interactions between the management in the two TTS spaces that contain managed devices that must interact. Kerberos provides such an interspace mutual authentication service using realm servers as shown in the previous diagram. But in addition, given the various approaches for reliability, availability, and failover, the interconnection between devices requires an authorization step following authentication. Each device in an interaction has the responsibility for protecting its own resources.

But devices must not only mutually authenticate who they are to each other; they must negotiate what basis they have for trusting each other in the current situation. This critical second step is important so that each device can defend itself, its services, and most importantly its data resources. Credentials can help identify who the parties are and what powers they have in the opinion of their controlling management, but ultimately a device that finds itself in isolation momentarily has the responsibility for protecting its own resources until command can be reestablished. This is true in the military analogy, and it remains good policy in the world of autonomic, policy-driven intelligent grid devices.

Some human interaction is also associated with a TTS – even though most interaction is machine-to-machine. During human interaction, a person can have different roles and needs depending on the situation. In some cases the person might have the role and responsibility to be the systems administrator for a TTS. In this case, the authorization policies in the system should allow the person to set policies, change policies, and oversee directives to all devices in an active TTS that they should transition themselves to enforce the new state of policies that are in production.

Sometimes the role might be that of a field person with the responsibility to install, test, and then activate devices at their service delivery point. Although the device will have been commissioned for duty by someone else, and the policies surrounding what can be done during activation and operation are predetermined, there is still the element of trust needed in the person installing the system for a variety of skills and decisions so that the service to the customer will be delivered as sold.

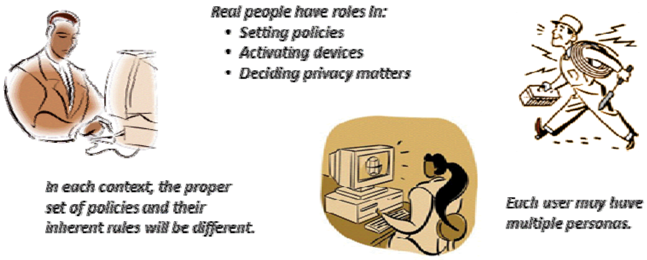

Real people have roles in:
• Setting policies
• Activating devices
• Deciding privacy matters
In each context, the proper set of policies and their inherent rules will be different.
Each user may have multiple personas.

Figure 8. Policy management must adapt to the situation.

But that same person may also be a consumer at his or her own home. In this situation, trust is more limited in some respects. But in matters of personal privacy, the consumer is king and the roles of administrator or field person are not allowed to even view the options chosen by the consumer if they affect privacy matters.

The policy management mechanism must also adapt to the skill level of the user and the subset of situations that are relevant to control the complexity. The policy control user interface needs to be intuitive enough for both the casual user and also for the expert user so that the articulation of the security requirements of the situation at hand is easy to see and understand. *The Laws of Simplicity* [4] provides useful guidance in the art of design in such matters.

Policies themselves operate in a controller/agent fashion – one set of policy rules at the controller form a *policy decision point* (PDP) while the agent policy rules form a *policy enforcement point* (PEP) [5]. These are policy rule execution environments that have different roles. The PEP encounters a situation and asks questions of the PDP in the management device as to what should be done, usually with a response of a yes or no answer, but sometimes supplemented with a few data attributes that are useful in clarifying the state of matters. The PEP then has the responsibility for faithfully carrying out the desires of the management given the PDP response. In order to do so, it has local policy logic that provides the arms and legs to carry out the decision.

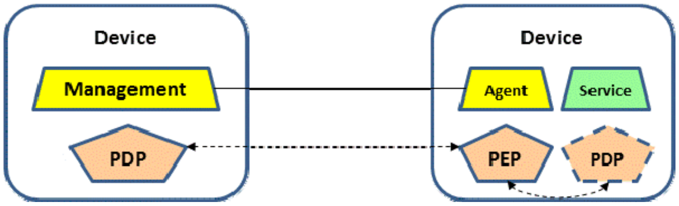


Figure 9. Policy decision points and policy enforcement points.

The agent usually asks for policy guidance from the remote manager PDP. But if offline or operating autonomically, policy guidance can come from the local device PDP. In either case, the request is a peer-to-peer interaction. When a device comes back online with the management, the usual sequence is for the device to attempt to resend significant event messages to the management to let the management know what took place while the device was offline from the central management element. In many ways the logic required to perform event management in a device is closely aligned with the logic that manages logging events for further use. Often these management functions are combined into a single management function to simplify operations.

2. SUMMARY OF PART 1

The emerging plan to use tailored trustworthy spaces as a model for providing end-to-end security for the intelligent grid has a lot of promise. The use of distributed policy management components greatly assists in the tailoring of these distributed systems so that they will operate with one another securely and in a coordinated fashion. The mechanisms can work well in a very small environment and well as massive environments.

The devices must be configured so that there is at least one manager, and then all the others join the group dynamically by registering with the chosen manager. The devices mutually authenticate using the manager as the trusted third party when they want to communicate with one another. Management communications is out-of-band from all application communications for both operational and security reasons. Legacy systems can participate through integration with a complex request broker. Even devices that are in two different TTSs can communicate without leakage using realm servers.

Human interaction when required also is controlled through policies. People may have multiple roles and personas that establish their permissions as well as what they want to keep private from other parts of the system.

The policy management system must provide for the creation and management of policies using visualization systems and language that makes policies easy to find, easy to read, and easy to understand. Policy execution is divided into Policy Decision Points where rulings can be requested and Policy Enforcement Points that ask questions and then must faithfully enforce current policy decisions.

References

[1] Landwehr, Carl E. “Toward a Federal Cybersecurity Research Agenda: Three Game-changing Themes,” www.nitrd.gov/.../NITRD_Cybersecurity_RD_Themes_20100519.ppt, May 19, 2010

[2] Reynolds, J. et al., *Security Fabric Policy Management System Concepts*, internal document, 2011.

[3] Gamma, Erich, et al., Design Patterns – Elements of Reusable Object Oriented Software, Addison Wesley, 1995.

[4] Maeda, John, *The Laws of Simplicity*, MIT Press, 2006.

[5] D. Durham, D., et al., *RFC 2748: The COPS (Common Open Policy Service) Protocol*, http://www.ietf.org/rfc/rfc2748.txt