CNT 4603: System Administration
Fall 2013

Introduction To Policy Management – Part 1

Instructor : Dr. Mark Llewellyn
markl@cs.ucf.edu
HEC 236, 4078-823-2790
http://www.cs.ucf.edu/courses/cnt4603/fall2013

Department of Electrical Engineering and Computer Science
Computer Science Division
University of Central Florida

Policy Management

- Managing an IT infrastructure is hard. Management and operational expenses are taking a larger and larger share of the IT budget in many organizations, with a major part of it attributed to the complexity of the system that needs to be managed.
- This has driven research and industry to look for management frameworks that go beyond direct human manipulation of network devices and systems.
- One approach is to build policy-based management systems (PBMS). Policy-based management refers to a software paradigm developed around the concept of building autonomous systems or systems that manage themselves with minimal input from human administrators.

Policy Management

- This paradigm provides system administrators and decision makers with interfaces that let them set general guiding principles and policies to govern the behavior and interactions of the managed systems.
- Although the majority of IT management chores are still carried out manually and in an *ad hoc* manner, policy based management systems are maturing and can be found in areas such as data center management, privacy, security and access management, and the management of quality of service and service level agreements in networks.
- We're about to explore what policy-based management systems are, and how the policies that affect and IT organization are determined.

Formal Definition Of Policy

- The word "policy" has its origins in government and regulations and its source is Middle English and Middle French.
- Wikipedia defines the word "policy" as:

"A **policy** is a principle or protocol to guide decisions and achieve rational outcomes. A policy is a statement of intent, and is implemented as a procedure[1] or protocol. Policies are generally adopted by the Board of or senior governance body within an organization whereas procedures or protocols would be developed and adopted by senior executive officers. Policies can assist in both **subjective** and **objective** decision making. Policies to assist in subjective decision making would usually assist senior management with decisions that must consider the relative merits of a number of factors before making decisions and as a result are often hard to objectively test e.g. work-life balance policy. In contrast policies to assist in objective decision making are usually operational in nature and can be objectively tested e.g. password policy."

- The idea conveyed by the definition above is that a "policy" is a plan or course of action.
- In computer science, the word "policy" has been applied to access control policies, load balancing policies, security policies, back-up policies, firewall policies, and so on.

Formal Definition Of Policy

- Heuristically a policy is a set of considerations designed to guide decisions on a course of actions.
- Policies generally start as natural language statements. From these descriptions, many details need to be sorted out before policies can be implemented.
- Consider the following statement that is usually implemented as a default policy in Apache Web servers:

```
Do not allow the execution of CGI scripts.
```

- The policy is activated by setting the value of an appropriate variable in a configuration file. During the initialization the Web server reads the configuration file and adjusts its behavior in a way that when interpreting and serving documents to Web clients it will throw an exception if it encounters a CGI script as the source of the document to be rendered into the Web client.

Formal Definition Of Policy

- Compare the policy on the previous page to the following example from banking regulations:

```
A currency transaction report (CTR) must be filed with the federal
government for any deposit of $10,000 or more into a bank account.
```

- This statement, extracted from the Money Laundering Suppression Act enacted by Congress in 1994, is a typical policy regulation that banks must implement. In modern banking systems, the implementation would probably be handled using database triggers.

- The implementation of the two policies just discussed has little in common. However, there is significant commonality in the *specification*. First, both policies identify a target system: the computer where the Web server is running and the bank information system. Second, both policies express constraints over the behavior of the target system.

Formal Definition Of Policy

- From the point of vied of high-level policy specification, what the system is or how the system is implemented is not relevant.
- The policy merely indicates how to regulate the behavior of the system by indicating states that the system can or cannot take.
- In the Web server example, if we can take a snapshot of the state of the server at any moment in time, the policy indicates that we should never find a process associated with a CGI script that was started by the Web server.
- In the banking example, if we take a snapshot of the system and find a transaction containing a transfer of $10,000 or more, the snapshot must also contain the generation of a CTR.

Formal Definition Of Policy

- To specify a policy, you must identify three things:

1. The target of the policy, which is referred to as the target system. A target system may be a single device such as a notebook, or a single workstation, or a complex system such as a data center or a bank information system consisting of multiple servers and storage systems.

2. A set of attributes associated with the target system. The value of an attribute can be a simple number or text string, or it can be a complex structured object with many attributes.

3. The states that the target system can take at any given time, which are defined by an assignment of values to the system attributes.

Formal Definition Of Policy

- In practice, there are many alternatives for the definition and identification of target systems. For example, the computer system where the Web server is running could be identified by an IP address; or you could group subsystems together and identify them by a unique logical name, for example, all the computers on the second floor of the HEC building.

- There are also many different ways to define and get the values of system attributes. For example, an attribute of a computer system could be a set of objects representing the processes running in the system at a given time. These objects could be complex objects with testable properties that identify whether the object represents a process that has been started by the Web server, and whether it is a CGI script.

Formal Definition Of Policy

- The behavior of a system is not completely characterized by the set of states it is in.
- A definition of "behavior" needs to take into consideration how the system moves through these states.
- Given that policies constrain the behavior, it is not surprising to find policies that constrain these state transitions. Consider the following example:

```
If a credit card authorized for a single person has been
used within 1 hour in different cities that are at least
500 miles apart, reject the charge and suspend activity on
the credit card immediately.
```


Formal Definition Of Policy

- This policy is also a constraint, but the constraint is not imposed on a single state of the system, but on at least three states: the state of the system at the time the credit card is first used; the state at the time when a second use of the credit card is detected; and any state in the future where the credit card transaction must be rejected.
- Thus, the behavior of a system can be defined to be a continuous ordered set of states, where the order is imposed by time.
- Consider a system *S* that may behave in many ways. Let *B(S)* be the set of all possible behaviors the system *S* can exhibit (that is, any possible continuous ordered set of states).

Formal Definition Of Policy

- Based on the previous discussion, we can now provide a more formal definition of a policy:

> A policy is a set of constraints on the possible behaviors *B(S)* of a target system *S*; that is, it defines a subset of *B(S)* of acceptable behaviors for *S*.

- Note that this is a very generic definition, and it does not say how policies can be implements or enforced. Implementations will require systems to provide operations that can affect their behavior. If there is no way to affect the behavior of the system, we will not be able to implement policies.

- These operations are special attributes of the system that policies can use. These operations are generally referred to as actions. Note that even though the system states can change continuously, implementations will be able to observe only discrete changes.

Formal Definition Of Policy

- In many real-life systems, the state of the system may not be completely defined or known.
- Typically, the determination of the full state of a system is not necessary to use a policy based approach. Policies can be defined using only a small number of attributes of a system state and do not require the determination of the complete state *a priori*.
- Let's return for a moment to our Web server example. Activating the policy to restrict the execution of all CGI scripts is quite straightforward – set the appropriate variable in the Apache configuration file, restart the server, and it will take care of the rest by itself.

Formal Definition Of Policy

- Now let's consider a more complicated policy.
- We can create policies that will allow different sets of users to execute different sets of CGI scripts. The implementation of a policy like this in a standard Apache server is not that obvious.
- You could try to implement this policy by creating a directory structure that reflects the different sets of scripts with links to the scripts from the appropriate directories, and creating access control files for each directory with the different sets of users that have access to the scripts.
- In this fashion, the policy would be enforced by giving usernames and passwords to the users and forcing the users to authenticate themselves before executing any of the scripts.

Formal Definition Of Policy

- A major problem with this approach would be that changes in either the set of users of scripts would require reshuffling of the directories and changes in different access control files.
- The difficulty arises because there is no obvious connection between *what* the policy wants to enforce and *how* it is enforced.
- In the simple CGI policy, *how* the policy is implemented is hidden inside the implementation of the Web server and the implementer needs to simply set the policy on or off.
- For the second case, having only the possibility of setting the CGI script execution policy on or off is too restrictive because an essential component of *what* the policy wants to constrain is conveyed by the different sets of users and scripts.

Formal Definition Of Policy

- A policy-based management system aims to provide an environment to policy authors and implementers where they can concentrate their efforts on describing *what* the policy restricts, thereby alleviating the burden created by having to describe *how* the policy will be enforced.

- This separation of *what* from *how* varies widely among different systems and applications, and in practice most policy authors are still required to have at least a partial understanding of policy implementation.

Types, Nature, and Usage Of Policies

- As we've seen so far, policies are constraints on the behavior of a system, and system behavior is a sequence of system states. In turn, each state of a system can be characterized by the values of a collection of system attributes.
- The attributes of a system can be divided into three groups – a set of fixed attributes, a set of directly modifiable attributes, and other observable but not directly modifiable attributes.
- The fixed attributes cannot be modified directly or indirectly.
  - As an example, a server in a data center has a state characterized by attributes such as maximum number of processes, size of the virtual memory, amount of buffer space for network communication, processor utilization, disk space utilization, memory utilization, and so on. Among these, the size of physical memory is a fixed attribute for the purposes of system management – it cannot be changed until the hardware of the server is modified.

Types, Nature, and Usage Of Policies

- Directly modifiable attributes can be easily affected. Using the server in a data center as an example again, the attributes characterizing the server's state such as maximum number of processes, size of the virtual memory, amount of buffer space for network communication, can be modified directly by changing some values in a configuration file.
- Observable attributes, such as processor utilization, or memory utilization, cannot be modified directly. They can be manipulated only by modifying the direct parameters or taking some other actions – for example, killing a running process.

Types, Nature, and Usage Of Policies

- The set of directly modifiable attributes of a system will be called its configuration attributes.
- The set of attributes that are not directly modifiable, but can be observed or computed from observation of system attributes, will be referred to as system metrics.
- The simplest policy type specifies an explicit constraint on attributes of the state that the system can take, thereby limiting system behavior.
- We'll now look in more detail at various types of simple policies that specify explicit constraints on the attributes of the state of a system.

Configuration Constraint Policies

- A configuration constraint policy specifies constraints that must be satisfied by the configuration of the system in all possible states.
- These may include allowable values for an individual configuration attribute, minimum and maximum bounds on the value of an individual configuration attribute, relationships that must be satisfied among different configuration attributes, or allowable values for a function defined over the configuration attributes.
- Configuration constraint policies are often used to ensure correct configuration of a system, to self-protect from operator errors, and to prevent the system from entering the operational modes that are known to be harmful.

Configuration Constraint Policies

- Some examples of configuration constraint policies include:
  - Do not set the maximum number of threads attribute on an application server to more than 50.
  - The size of the virtual memory in the system should be less than two times the size of the physical memory.
  - Only users in the administration group have access to system configuration files.

Metric Constraint Policies

- Metric constraint policies specify constraints that must be satisfied by the system metrics at all times. Unlike configuration attributes, the metrics of a system cannot be manipulated directly.
- The system needs to determine in an automated manner how to manipulate the configuration of the system, or to take appropriate actions such that the constraints on the metrics are satisfied.
- Constraints on the metrics may include bounds on any observable metric, or relationships that may be satisfied among a set of system attributes including at least one metric attributes.
- Metric constraint policies that specify an upper or lower bound on a metric are generally known as goal policies because they provide a goal for that metric.

Metric Constraint Policies

- Examples of metric constraint policies include:
  - Keep the CPU utilization of the system below 50%.
  - All directory lookups on the name of a person should be completed in less than one second.
  - The end-to-end network latency should be kept below 100 msec.
- Metric constraint policies are often used to enable self-configuration of systems in order to meet specific performance requirements or objectives.

Action Policies

- Configuration and metric constraint policies specify constraints on a single system state.
- In many cases, a policy may be require explicit actions to be taken when the state of a target system satisfies some constraints. These types of policies are called action policies because they require the system take a specific set of actions.
- Action policies constrain a sequence of states. That is, when a particular state is observed then certain actions must be taken at a later point so that the target system will be in some other state.
- In most cases, an action policy will modify the configuration of the system is response to some condition being true.

Action Policies

- Action policies essentially provide a plan according to which the system should operate when it encounters a certain condition specified in the policy.
- Examples of action policies include:
  - If the CPU utilization of a server in the data center exceeds 70%, allocate a new server to balance the workload.
  - If the temperature of the system exceeds 95 degrees Celsius, then shut-down the system.
  - If the number of bytes used by a hosted site exceeds 100 GB in a month, then shut down access to the site.
  - If the inbound packet has a code-point for expedited forwarding (EF) per-hop behavior (PHB) in the packet header, then put it in the high priority queue.

Alert Policies

- The examples of the action policies on the previous page were used to manage the performance of a server or network, manage the effect of environmental conditions, limiting resource utilization, and for providing different QoS (quality of service) in communication networks.

- Not all action policies specify an action that can be directly executed on a system. One important type of action policy is the alert policy, which is commonly used to flag any conditions that may require operator intervention.

- An alert policy is an action policy where the action consists of a notification sent out to another entity. A notification is an action that does not modify the configuration of the system itself.

Alert Policies

- Notifications generally take one or more of the following forms: sending an email or an SMS message, making a phone call, logging a message in a file, or displaying an alert visually on a display.
- Some examples of alert policies are:
  - Notify all users who have not accessed their account for three months by email to warn them of possible account deletion.
  - If a system has not installed the latest version of anti-virus software, send an email to the employee and their manager.
  - If a system has failed, send a message to the administrator's pager.

Policy Information Models

- Although the policies we've seen as examples on the previous pages are specified in different styles, all of these policies can be restructured using a common pattern or model.
- Formally, this model is called the policy information model.
- One of the most widely used policy information models describes a policy using a condition-action rule, which means if the condition is true then perform the action.
- A more specific version of the condition-action rule is the event-condition-action (ECA) rule, which means upon occurrence of the event, if the condition is true then perform the action.

Policy Information Models

- Consider the metric constraint: "The end-to-end network latency should be kept below 100 msec." (from page 23).
- The can be rewritten as an ECA rule as follows:
  - Upon completion of measurement, if the end-to-end network latency is above 100 milliseconds, then record the violation in the system log file.
- We'll examine the policy information model in more detail in a later section of notes.

How Are Policies Used

- Since policies are defined as constraints on the operation of a system, how can the specification of such constraints help in the management of IT systems?
- In general, the specification of constraints on the state of the system can be used for several different purposes, such as:
  - When the demand or workload on a system changes requiring a reconfiguration of the system, the constraints can be used to determine a desirable new configuration.
  - When there is a contention for resources in the system, the constraints can be used to determine the manner in which to resolve that contention.
  - When any external entity attempts to access resources in the system, the constraints can be used to determine if the access should be allowed.
  - When a system violates certain constraints, it can determine and execute a set of actions that will allow it to remove the violation. (Autonomic computing)