The Meta-Policy Information Base

Andreas Polyrakis, University of Toronto
Raouf Boutaba, University of Waterloo

Abstract

The recent considerable growth of computer networks has revealed significant scalability and efficiency limitations to the traditional management techniques. Policy-based networking (PBN) has emerged as a promising paradigm for configuration management and service provisioning. The Common Open Policy Service (COPS) and its extension for policy provisioning (COPS-PR) are currently being developed as the protocols to implement PBN. COPS-PR has received significant attention and seems efficient for several management areas. However, the rigidity of its policy-enforcing mechanisms constrains the intelligence that can be pushed toward the managed devices. This work aims at relaxing this limitation by using meta-policies, rules that enforce the appropriate policies on the devices. Meta-policies are stored and processed by the devices, independent of their semantics, thus making the model more efficient, scalable, distributed, and robust. The additional functionality is implemented through a novel policy information base we have defined, the Meta-Policy PIB.

Policy-based networking (PBN) has emerged as a promising paradigm for network operation and management [1, 2]. It is based on high-level control/management policies [3-5], that is, rules that describe the desired behavior of the network in a way as independent as possible of network devices and topology. Two basic entities are distinguished: the policy enforcement points (PEPs) and policy decision points (PDPs) [6]. The PEPs typically reside on the managed devices. Their role is to enforce the commands they receive in the form of configuration data from the PDPs. The PDPs process the high-level policies along with other data such as network state information and generate configuration data for the PEPs. The configuration data for each PEP are generated according to its specific role, capabilities, and limitations. If the network state or policies change, the PDP may readjust the behavior of the devices by sending updated configuration data.

The key concept in PBN is that by describing goals ("what") rather than procedures ("how"), which happens with the traditional management techniques, the policies are separated from the network details. The high degree of abstraction and the automation implied from this model make the network easier to manage and ensure consistency in the behavior of the devices across it. Besides, the dynamic binding of the policies with the network details that takes place in real time at the PDPs allows new types of policies to be constructed and gives extra flexibility to the network managers. A PBN is illustrated in Fig. 1.

The IETF Resource Allocation Protocol (RAP) working group [7] attempts to standardize communication between PDPs and PEPs through the Common Open Policy Service (COPS) [8] protocol and its extensions. COPS has already received significant attention, and applications based on it have emerged [9–12].

The COPS protocol is designed to communicate self-identifying policy-related information, exchanged between the PDP and the PEP. In COPS, each PEP may support one or more clients of different client types; different client types exist for the different policing areas (security, QoS, admission control, accounting, etc). By supporting the appropriate client types, the PEP provides a way to control the various management aspects of the device. Such client types are currently being developed by the IETF. These client types are considered extensions of the base COPS protocol, since they define details for the format and semantics of the configuration data exchanged between the PDPs and PEPs. COPS for Policy Provisioning (COPS-PR) [13] is one of those client types.

COPS-PR uses special structures called *policy information bases* (PIBs) that store policy information at the PEPs and control the behavior of the devices. However, the rigidity of its mechanisms constrains the intelligence that can be pushed toward the managed devices. This work aims at relaxing this limitation by using meta-policies, rules that enforce the appropriate policies on the devices. Meta-policies are stored and processed by the devices, independent of their semantics, thus making the model more efficient, scalable, distributed, and robust. The additional functionality is implemented through a PIB we define that stores and handles meta-policies. This way, although the philosophy of the conveyed policing information is now different, no modifications are required to the COPS-PR protocol.

The structure of this article is as follows. We present PBN, COPS, and COPS-PR. A small example demonstrates how COPS-PR works. We present the motivation of our work and introduce the concept of meta-policies through the same example. We present and analyze the proposed PIB. We report the implementation status of our work, and conclude by summarizing the presented work and discussing future goals.

COPS-PR

The Protocol

RAP developed COPS-PR [13] as an extension (or client type) of COPS. COPS-PR was initially biased toward DiffServ policy provisioning [14]. However, it appears to be suitable for

This research wprk is supported by research grants from Nortel Networks and the Natural Sciences and Engineering Research Council of Canada.

several other management areas (accounting [15], IP filtering [13, 16], security [17], etc.).

As its name implies, COPS-PR operates in a provisioning mode. The clients connect to the appropriate PDP, report their capabilities and limitations, and request the initial policies to be downloaded into them. The PDP processes the request of each client and, according to the global policies and network state, generates and downloads the appropriate configuration data into the PEPs, which serve all incoming events according to these data. If the network state or policies change, the PDP may update these configuration data in order to keep the behavior of the managed devices consistent.

The Policy Information Base

In COPS-PR, each client has to maintain a special database, the PIB [18], where all the received configuration data are stored (Fig. 2). The PIB is a structure similar to a management information base (MIB), and can be described as a conceptual tree namespace, where the branches represent structures of data, or provisioning classes (PRCs), and the leaves represent instances of these classes, called provisioning instances (PRIs). PIBs are defined by COPS-PR only as abstract structures; the details of each PIB (PRCs and their semantics) are specified in separate standard documents (e.g., Internet drafts or vendor private documents). Different PIBs are defined in order to cover the various management areas (DiffServ, accounting, security, etc). PIBs are defined at a high abstraction level in order to hide the details of the underlying hardware and provide to the PDP a unified way to control the behavior of the devices regarding a specific management area across the entire network.

PRIs are identified within the PIB through a PRI identifier (PRID). Policies are formed as a set of PRIs in the PIB; by adding or removing PRIs, the PDP can implement the desired policies to be enforced at the device.

It is important to highlight that the policies each PIB can implement are predefined (in the document that defines this PIB). In order to control a device, the PDP has to map the high-level network policies and network state into policies that can be implemented in the PIB of the PEP.

PIBs are defined using the Structure of Policy Provisioning Information (SPPI) specification [19].

A COPS-PR Example

We shall use a small filtering PIB in order to demonstrate how COPS-PR works. The network of our example is the network of a small company (Fig. 3), with the following topology:

- LAN address range: X.Y.0.0/16
- Subnets X.Y.1.0/24 (public), X.Y.2.0/24 (administrators), X.Y.3.0/24 (employees)
- A central router A that routes the LAN and Internet traffic and serves as the Internet gateway

Suppose that the following high-level access rules have been set:

- Internal LAN traffic is always allowed.
- The administrator can always access the Internet, whenever and from wherever he/she is logged in.
- During overall congestion, traffic between the employee domain and the Internet is denied.
- The Internet can be accessed only during working hours (Monday to Friday, 9:00-17:00).

In this example, the first rule has the highest priority, the fourth the lowest, and the term "overall congestion" is evaluated according to whether router A is congested (i.e., based on the load of its interfaces).

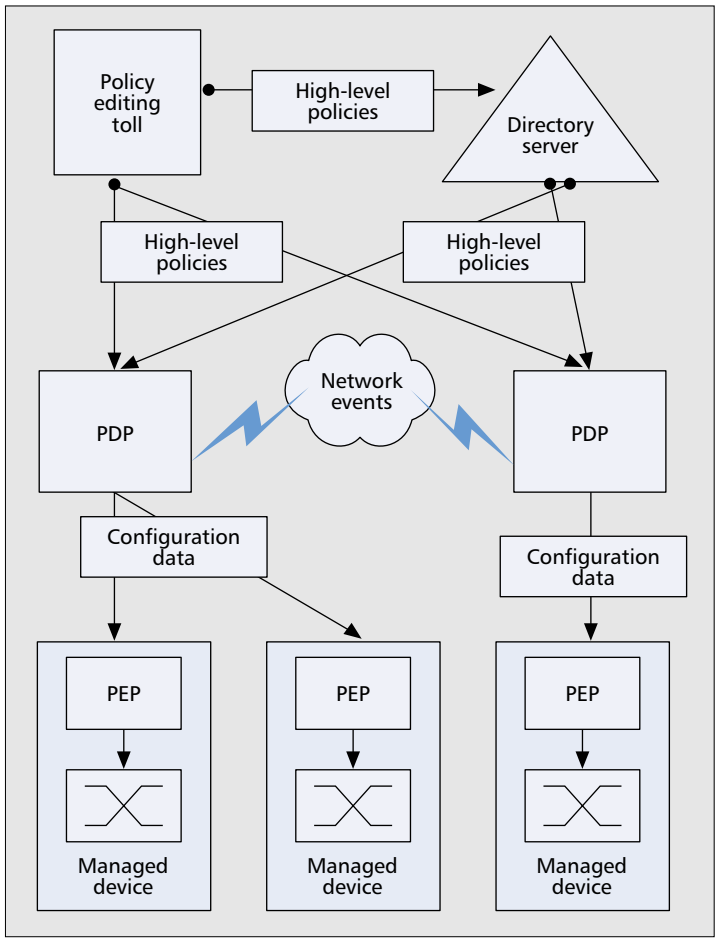

Figure 1. *Policy-based networking.*

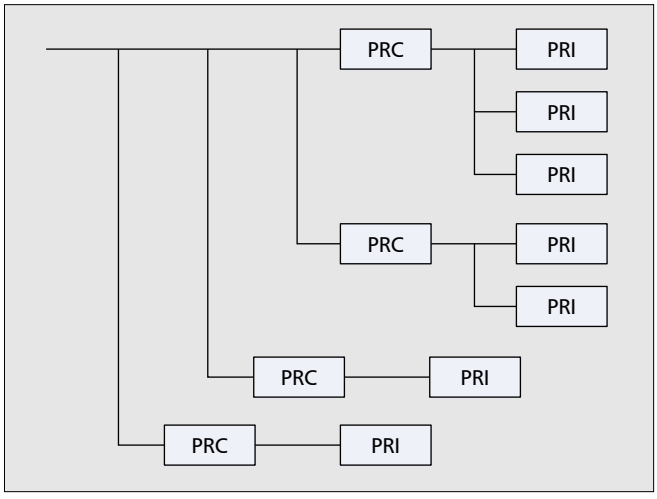

Figure 2. *PIB structure.*

Suppose that the (PEP of the) routers of the network support a PIB with a single PRC. PRIs of this PIB describe source/destination criteria that allow access to IP traffic within the network. Each PRI in this PIB is a standalone policy of the form

If ((Source matches Srcaddr/Srcmask) and (Destination matches Destaddr/Destmask)) then allow

Traffic that matches at least one PRI in the PIB is allowed. Traffic that does not match any criteria (policies in the PIB) is, by default, denied.

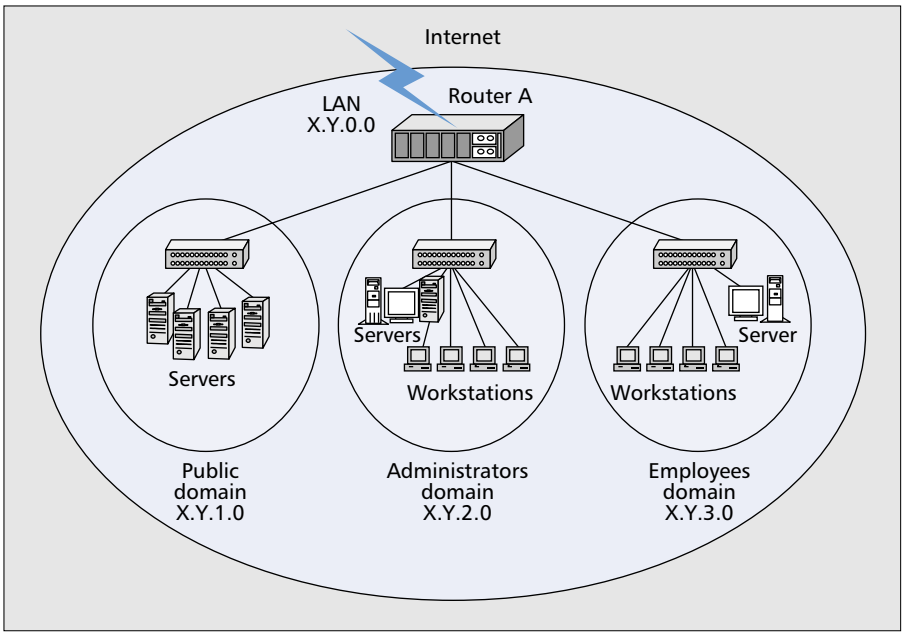

Figure 3. *The topology of the company example network.*

Suppose now that the following events take place:
08:59: No administrator logged on
09:00: start of working day
11:00: congestion detected
11:05: no congestion
15:08: congestion detected
15:11: administrator logs on at X.Y.3.7
15:20: no congestion
17:00: end of working day
17:15: administrator logs out

Figure 4 demonstrates snapshots of the PIB of router A during the day. Initially, the PDP activates policy #1 by installing a PRI that allows all LAN traffic (PRID #1). At 09:00, policy #4 becomes applicable, and a PRI that allows traffic to/from the Internet is added into the PIB (PRI #1 is now redundant, and the PDP may keep it or not). When congestion is detected (11:00), the PDP attempts to install policy #3. This policy conflicts with the already installed policy #4; however, policy #3 has higher priority; hence, the employee subnet is banned from Internet traffic. At 11:05, the network becomes decongested, and the PIB is restored to its previous state. When the network becomes congested again (15:08), the PIB has to be updated again, as before. When the administrator logs on at the employee subnet (15:11), traffic to/from the Internet to his/her IP is allowed. Note that policy #2 conflicts with policy #3, which bans traffic to the employee subnet; however, the former wins since it has higher priority. When the network becomes decongested (15:20), policy #3 is uninstalled, and policy #4 is installed again. At the end of the working day (17:00), policy #4 is also uninstalled, and finally, when the administrator logs out, policy #2 is uninstalled as well, denying all Internet access.

Meta-Policies in COPS-PR

Motivation

The previous example reveals the two shortcomings that motivated our work.

First of all, in this model the PEPs have no memory of the previous events or network state. Assuming that the management policies remain the same, the PIB of a PEP should be the same whenever the same network state (i.e., combination of the same network events) occurs. However, in this model the PEP cannot be directed to restore a previous state of its PIB if a network state reoccurs. Instead, the PDP needs to update the PIB by installing and removing PRIs to achieve the desired content. From another point of view, the current model does not take advantage of the correlation between the network state (or events) and the PIB content.

A second limitation lies in the rigidity of the PIBs. PIBs are predefined structures, and the high-level policies cannot directly map into them. For instance, in the previous example the policy "During overall congestion traffic between the employee domain and the Internet is denied" cannot fit into the PIB, and has to be processed by the PDP. The latter, depending on the overall network state, produces the PRIs that are in conformance with the initial policy for the given congestion status. Then the PEP enforces the policies these PRIs describe. Notice that the high-level policy has to be processed partially by the PDP and partially by the PEP. The PDP needs to query the MIB of router A in order to determine if there is congestion, then send the appropriate policies back to the router's PIB. Obviously, the processing of this policy could be carried out entirely at the PEP level. The rigidity of the PIBs, though, does not allow any other kind of policies to be evaluated by the PEP apart from those supported by the PIB, thus making the presence of the PDP necessary, even in cases where this could be avoided. This limitation increases the volume of communication and makes the model vulnerable to PDP errors or malfunctions and network error situations, such as congestion or network failures.

The previously discussed limitations motivated our work. The intelligence in the COPS-PR model seems to be concentrated at the PDP level. PDP decisions always download policies into the PEP, even when the same events recur. The PIB is a rigid structure that allows only limited types of policies to be pushed into the PEP. The PEP depends on the PDP presence, even in cases where this is not absolutely necessary.

This work extends the policy functionality of the PIB so that the PEP will be able to take more decisions simply by examining events. Initially, the PDP downloads all applicable policies and directs the PEP on how to react to certain events. After that, the PDP controls the PEP mainly by communicating events that modify its PIB indirectly, rather than by updating it directly. Also, the PEP can be programmed to monitor some of these events by itself and initiate the appropriate actions. Assuming this extended functionality, the PDP is able to control the PEP mainly by communicating events rather than policies. Also, the PEP is able to make certain policing decisions by itself. Thus, intelligence is pushed toward the PEP.

The described functionality is achieved through meta-policies, which are defined and discussed next.

The Concept of Meta-Policies

We define a meta-policy as a rule that describes how policies are enforced. Meta-policies have the form:

if (condition) then {actions}

where condition is a logical expression, for example, "(C>80%) and (D=true)," and "actions" are a set of PIB commands that install PRIs into the PIB.

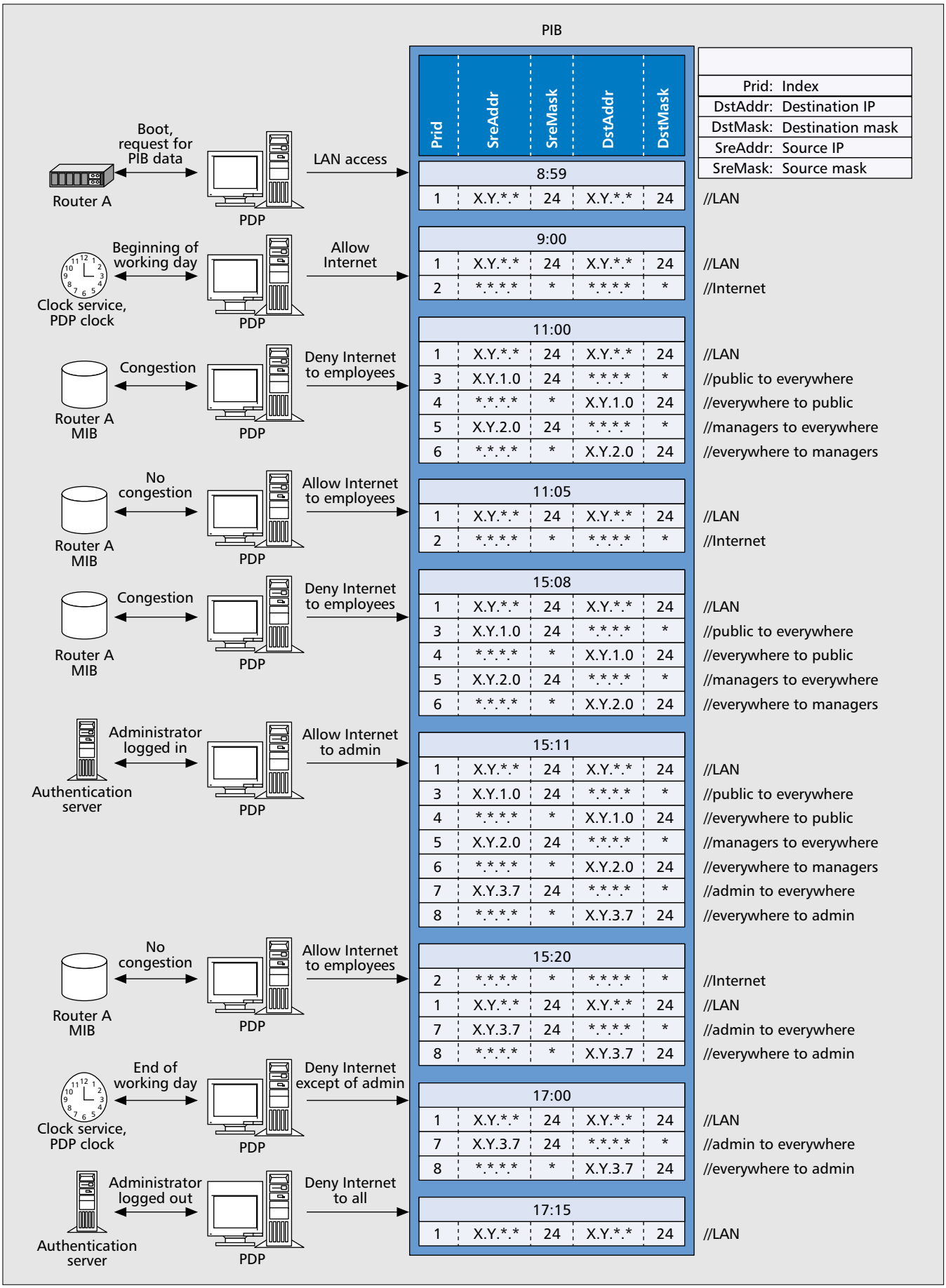

Figure 4. *Instances of the PIB of router A.*

Meta-policies are generated by the PDP and consumed by the PEP. The PEP evaluates the condition of each meta-policy, and when if evaluates true, it enforces the actions. Both the condition and the actions may contain parameters. For instance, the meta-policy "**if (AdminLogged) then {install (7, AdminIP, 24, *.*.*.*, 24,), install (8, *.*.*.*,24, AdminIP, 24)}**" contains two parameters: AdminLogged and AdminIP. Such parameters are evaluated by the PEP according to directions sent by the PDP. For example, the PDP may provide the value of these parameters, or direct the PEP to evaluate them from its MIB or by contacting another server or service.

The key idea in meta-policies is that the PEP can store and process them without understanding their exact semantics: the condition is treated as a logical expression; the actions, pregenerated by the PDP, just denote PRIs that must be installed. In this way, the PEP can process any meta-policy, independent of its complexity and meaning.

Example

Consider the company example we studied before. We shall examine how it is affected by meta-policies (Fig. 5).

First of all, policy #1, "Internal LAN traffic is always allowed," must always be enforced. Hence, the PDP directly enforces this policy by installing PRI #1 into the PIB when the router boots.

In addition, the PDP downloads to the PEP the following meta-policies:

- if (WorkTime) then {install (2,*.*.*.*,24,*.*.*,*,24)}
- if ((if1Util>80%) or (if2Util>80%) or (if3Util>80%)) then {
  install (3,X.Y.1.0,24,*.*.*.*,24), install (4, *.*.*.*,24, X.Y.1.0,24)
  install (5,X.Y.2.0,24,*.*.*.*,24), install (6, *.*.*.* ,24, X.Y.2.0,24)
  }
- if (AdminLogged) then
  {install(1,AdminIP,24,*.*.*.*,24), install(1, *.*.*.*,24, AdminIP,24,)}

and informs the PEP that the two first meta-policies are conflicting, and the second one has higher priority.

Since the meta-policies contain parameters, the PDP also has to inform the PEP of the evaluation methods for these parameters. In our example, the PDP sends the values of the parameters "WorkTime," "AdminLogged," and "AdminIP," and it directs the PEP to evaluate by itself the parameters "if1Util," "if2Util," and "if3Util" through the appropriate MIB variables that denote the usage of the router's interfaces.

The PEP monitors the parameters, and when their values change it reevaluates the affected conditions. While the condition of a meta-policy is met, the corresponding PRIs are installed in the PIB. In this way, the PIB always contains the PRIs that achieve the desired behavior.

Meta-policies allow the PDP to initially download the applicable policies and meta-policies and then control the PEP mainly by reporting network events. Moreover, some of these events can be monitored by the PEP itself without involvement of the PDP. Note that such events do not have to be local; the PEP can be programmed (e.g., by downloading and executing some scripts or through mobile agents) to monitor such events through another server or service: for instance, the parameter "WorkTime" could have been monitored by the PEP through a script running on it that gets the current time by contacting a network time service, without involvement of the PDP.

The Meta-Policy PIB

The proposed enhancements require meta-policing information to be exchanged between the PDP and the PEP, and stored and processed by the latter. We decided to use COPS-PR to communicate such data and define a PIB to store them at the PEP (as opposed to defining another protocol and/or storage structure, or extending existing ones). This decision was based on a number of factors. The provisioning style adopted by COPS-PR perfectly suits our needs. Our work is in line with the IETF and requires no new protocols to be developed. A PIB can easily be adapted by the Internet community (researchers and vendors). Finally, the design and implementation are simplified since the definition of a PIB is much simpler than defining a new protocol, and the implementation is based on existing tested tools.

It is important to highlight that the following PIB is meaningless by itself. The same PEP is supposed to implement one or more other PIBs, the contents of which are controlled by the meta-policy PIB.

PIB Definition

The PIB we have defined provides the necessary classes (PRCs) for handling meta-policies. The PIB is defined according to the IETF specifications (i.e., using SPPI). Although a detailed description of the PIB is out of the scope of this document, a brief overview is given in the following paragraph.

The PIB is divided into five groups (Fig. 6):

- The Capabilities Group contains the PRCs that store the capabilities and limitations of the PEP (as far as the meta-policy PIB is concerned). The PRIs of these classes are reported to the PDP when the PEP connects.
- The Base Meta-Policy Group contains the classes that form the meta-policies, define their relative priority in case of conflicts, and report their status.
- The Condition Group provides classes for forming the conditions of the meta-policies.
- The Action Group includes the PRCs that define the actions of the meta-policies.
- The Parameter Group contains the PRCs where the parameters, and their evaluation methods are stored.

Prior to installing a meta-policy, the PDP has to install its parameters into the meta-policy PIB (unless these parameters are already installed by another meta-policy). The parameters are stored in the **parameter** table (PRC). Depending on the evaluation method of the parameter, the **mibPibParameter** or **pdpParameter** class is used. The former points to the MIB/PIB variable from where the parameter will be evaluated; the latter stores the latest value sent by the PDP. The evaluation methods can be extended with new ones, as we shall see later on.

Having installed the parameters, the PDP installs the meta-policy by inserting a row in the **metaPolicy** table. If the meta-policy may be in conflict with other meta-policies under certain circumstances (i.e., it attempts to install the same PRIs with other meta-policies), the PDP also needs to declare how this conflict will be resolved through the **metaPolicyPriority** class.

In order to form a meta-policy, its condition and actions must be installed in the appropriate tables of the Condition and Actions Group: The **condition** is placed on the condition table. Each condition is either decomposed into simpler conditions of the same table, through the **complexCondition** table, or is a primitive, which is evaluated according to instructions found in the **booleanCondition** and **generalCondition** classes. Actions are declared it the action PRC. Depending on whether the action is parametric or not, the action itself is actually encoded in the **actionParametricValue** or the **actionValue** classes.

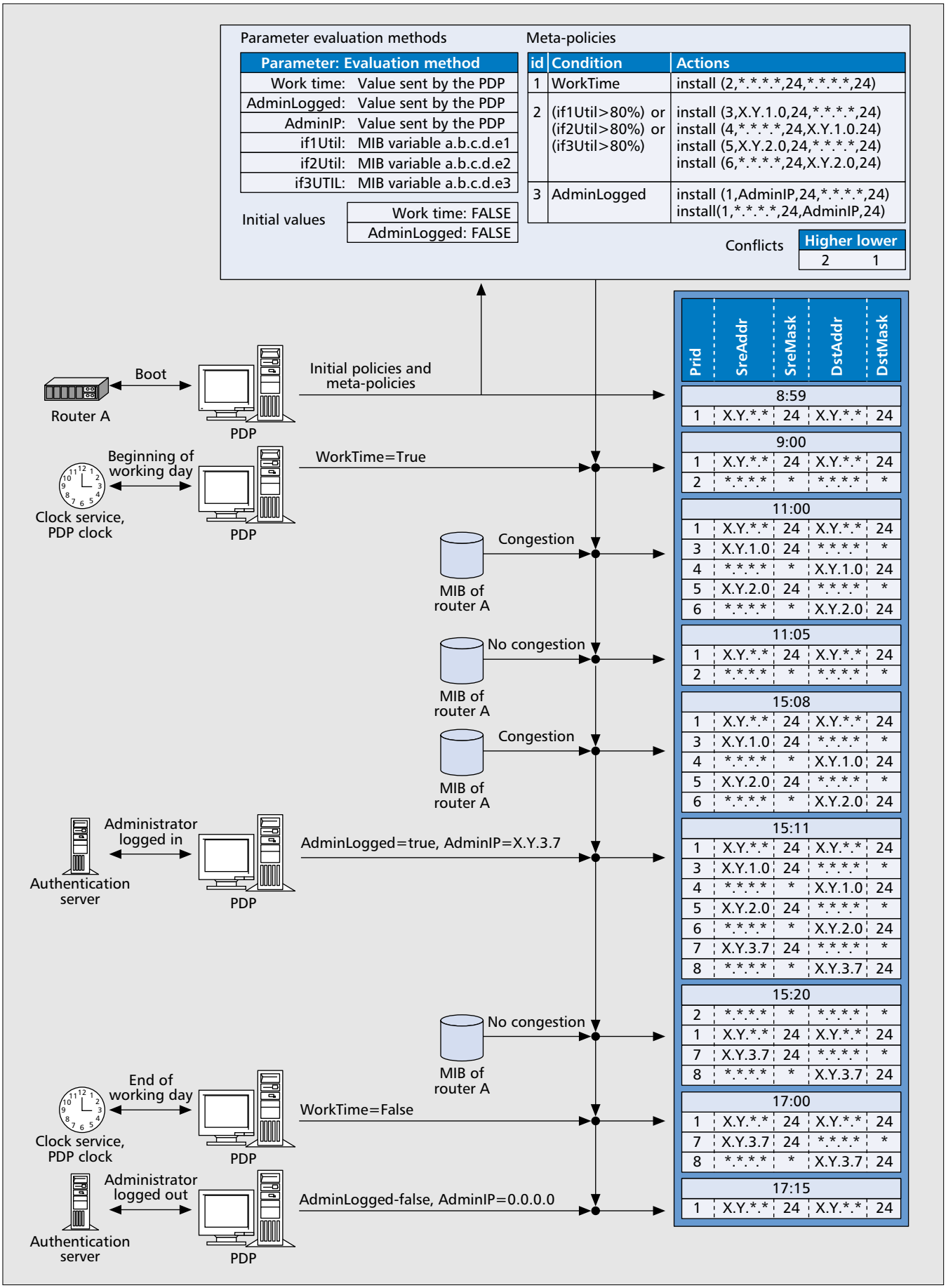

Figure 5. *Instances of the PIB of router A.*

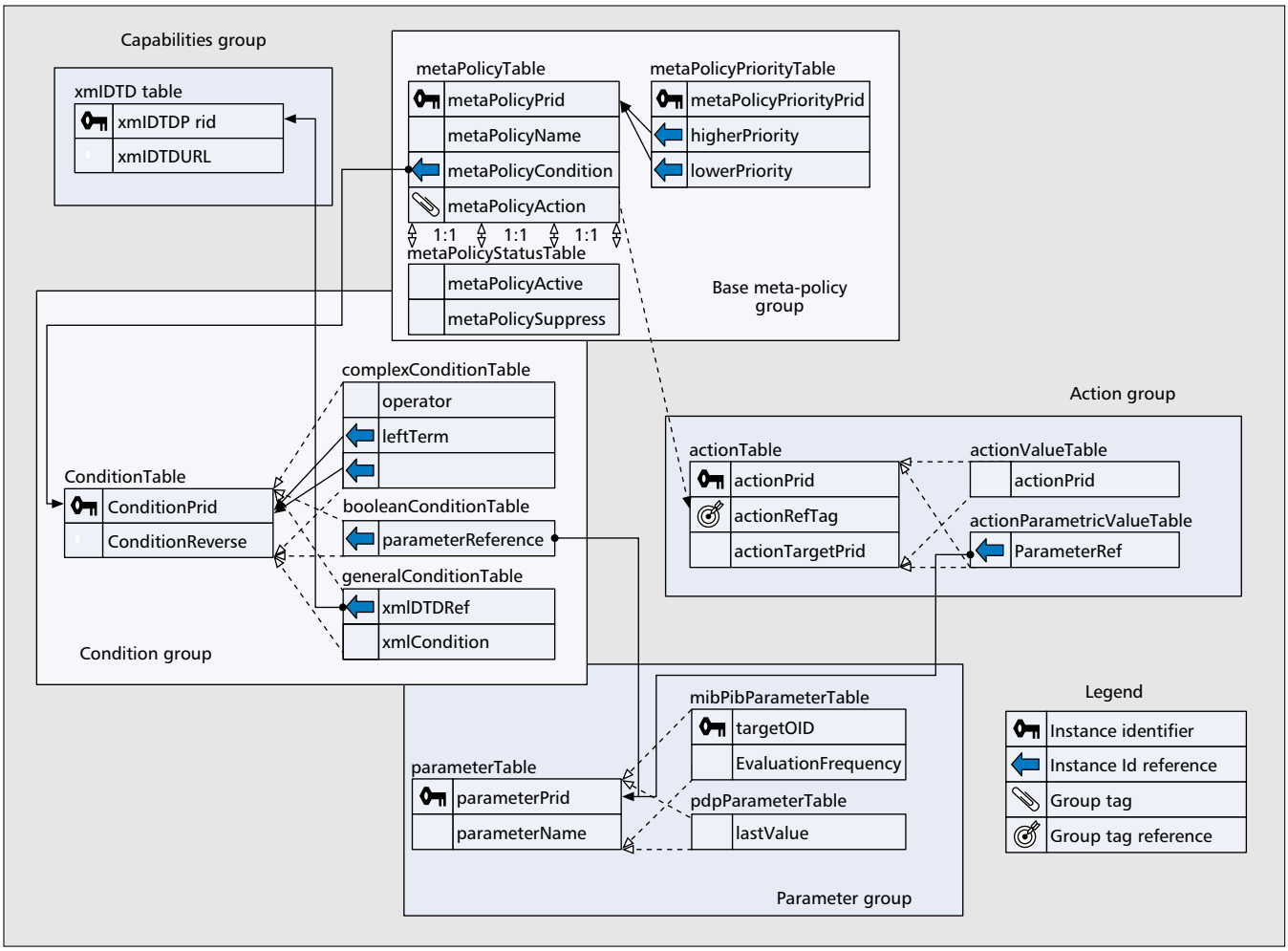

Figure 6. *The meta-policy PIB.*

Further description of the PIB is beyond the scope of this article. However, we would like to emphasize that the defined PIB is extensible. For instance, a network manager may want to extend the parameter evaluation methods by adding a new type of parameters that are evaluated by executing a certain script. In this case, the PIB is extended by adding the additional parameter evaluation methods as additional PRCs of the Parameter Group and associating these classes with the **parameter** PRC (and, of course, installing the scripts that will compute the value of the parameter into the PEP). In addition, the PDP should be aware of what these scripts do and how to use them. A realistic scenario is one where a vendor that develops a PDP also provides some code that extends the functionality of the meta-policy PIB in a specific way. Network managers download this code and install it on their devices (the code may be vendor-specific). In this way the PEPs that reside on such devices have extended functionality, and the PDP knows exactly how to use the additional mechanisms.

The previously described mechanisms that acquire information about the network state can be either standard-based or custom. For instance, a PEP could be enhanced by installing a module on it that acquires data through SNMP from any SNMP-enabled device on the network. COPS or LDAP could also be used to retrieve network state information. However, modules that use nonstandard protocols to retrieve information from custom servers or services can also be used.

PEP Behavior

We have also defined how the PEP must behave and how PIB data are interpreted.

Communication — In general, the communication between the PEP and the PDP as well as their behavior adheres to the COPS-PR protocol [13].

When the PEP connects to the PDP, it reports its meta-policing capabilities and limitations through a configuration request (REQ) message. This message reports the classes of the Capabilities Group. The PDP downloads all the appropriate meta-policies according to the reported capabilities and limitations.

Decision (DEC) messages are sent as solicited replies to REQ messages, or unsolicited, whenever the policing data into the PIB needs to be updated. Meta-policing data is handled as any other kind of PIB data; hence, the format of DEC messages and the way these are installed into the PIB are exactly as defined by COPS-PR. Notice, however, that meta-policy data may now report network events to the PEP, since the PDP may send values for parameters that represent such events (e.g., the PDP may report congestion by setting the value of a parameter in the PIB of the PEP). This means that the semantics of the exchanged data can now be different, conveying events to the PEP instead of configuration data that are mapped directly to its policy mechanisms.

According to COPS-PR, the PEP reports the success or failure of the DEC message with a solicited report message (RPT). In the case of meta-policies, such messages report whether the meta-policy itself was successfully installed/uninstalled (i.e., if the operation is valid according to the meta-policy PIB specification), not whether the meta-policy manages to install/uninstall its actions successfully. Unsolicited RPT messages can be accounting-related information. Such information is stored in the **activeMetaPolicy** class. Unsolicit-

ed RPT messages can finally report PEP errors that are not related to a specific DEC message. Such RPT messages can be triggered by badly behaving meta-policies, (e.g., that attempt to install invalid or conflicting PRIs).

Handling Meta-Policy Data — When meta-policy data are to be installed into the PIB, the PEP needs to perform integrity and consistency checks to ensure that these data conform to the PIB definition. If not, the entire DEC message is rejected, and an RPT message indicating the cause of the error is sent to the PDP (as defined in COPS-PR).

Parameters: The PIB defines two types of evaluation methods for the parameters, and more can be added by users such as network operators. Independent of the way a parameter is evaluated, a change in its value triggers the reevaluation of the conditions and actions in which it is contained.

Conditions: The condition of each meta-policy is decomposed into primitive logical expressions. Each logical expression contains a number of parameters, which must exist in the PIB before the logical expression is installed. When a logical expression is installed, it is evaluated according to the current values of its parameters. The overall condition is evaluated according to the evaluation of these logical expressions. If the values of the parameters change, the condition may need to be reevaluated.

Actions: When the condition of a meta-policy evaluates true, and if no conflicting meta-policy with higher priority is already active, the meta-policy is activated by installing the appropriate PRIs into the PIB. The meta-policy stays active while its condition is met, and no other meta-policy with higher priority is activated. In this case, the PRIs installed by this meta-policy are removed from the PIB.

Backward Compatibility

The proposed PIB does not create any backward compatibility issues when PDPs that support the proposed PIB are required to cooperate with PEPs that do not, and vice versa. If only the PDP does not support the additional functionality, the PEP is never directed to use the meta-policy PIB. If only the PEP does not implement the PIB, the REQ message will not report meta-policy classes, and the PDP will not attempt to install data in them, as defined by COPS-PR [13].

Trade-offs

The meta-policy PIB allows policing intelligence to be pushed toward the PEP. However, the additional functionality implies more complicated algorithms at the PEP that increase the CPU and memory requirements on the managed devices. Such resources, which are critical for the smooth operation of several network devices (e.g., routers), are usually limited and mainly dedicated to the critical operations of the device. On the other hand, the processing and memory capabilities of the modern devices tend to increase rapidly, and it can be predicted that devices in the near future will have the power to accommodate meta-policy functionality. Besides, compatibility with standard COPS-PR allows devices with limited resources not to implement the additional functionality and still communicate with PDPs that support meta-policies.

Apart from the increased resource requirements at the PEP, meta-policies require sophisticated algorithms at the PDP as well. However, PDPs are complex anyway, and they usually reside on servers that have adequate computing power for such complex computations.

An important issue with meta-policies is security. Meta-policies inherit COPS-PR security issues, but they do not introduce any important new ones since they introduce no new protocols or policing mechanisms on the devices. On the other hand, the extensibility of the meta-policy PIB with modules or scripts that execute code on behalf of the PEP makes the PEP and the device vulnerable to the security issues of the used modules or scripts.

Implementation

In order to test and evaluate the concept of meta-policies we have already built the proposed meta-policy PIB. This PIB is hosted into a PEP we have developed for testing purposes. The PEP also hosts a simple filtering PIB similar to the one described in the example studied earlier in this article. The PEP communicates through COPS-PR with a simple PDP that controls the contents of the filtering PIB either directly or through the meta-policy PIB. All the components (the PEP, its PIBs, and the PDP) are written in Java and run on Linux boxes.

Currently, the PEPs are being transported on two open-architecture Nortel Networks programmable Passport routers available in our lab. These routers can be programmed to run small java applications, called *oplets*, which will implement the additional functionality of the PEP and PIBs. The oplets run on an environment supported by the routers, the Oplet Runtime Environment (ORE), which provides access to the mechanisms of the routers. The PDP will control the behavior of the two routers by installing the appropriate policies and meta-policies on the PEPs residing on them. In addition, the programmability of the routers will allow us to write oplets that will communicate with third-party servers or services (e.g., network time service) and evaluate meta-policy parameters on behalf of the PEPs. The goal of these experiments is to test the proposed PIB with regard to its performance and efficiency, and compare it with the existing techniques (COPS-PR without meta-policies).

Conclusion

This article introduces the concept of meta-policies and demonstrates through a simple example how these can be used to overcome the limitation of the current COPS-PR model. The idea underlying the meta-policy concept is to push some of the COPS-PR PDP functionality and intelligence toward the PEPs through the meta-policy PIB. Finally, this article describes the defined meta-policy PIB and its implementation.

As a future work, we intend to study further enhancements to the meta-policy PIB. As a first step, we would like to investigate whether hierarchies of meta-policies can increase the intelligence of the PEP, and how the proposed PIB should be modified (if necessary) to support such hierarchies. Another research goal is to examine how other promising technologies, such as mobile agents or directories, can be exploited to enhance the functionality of the PEP, in combination with the proposed PIB. We believe that this will allow us to move most of the functionality of the PDPs into the managed devices. This will make our long-term goal, the existence of "smarter" devices and the concept of "plug-and-play" networks, seem more feasible and realistic.

References

[1] S. J. Shepard, "Policy-based Networks: Hype and Hope," *IT Prof.*, vol. 2, no. 1, Jan.-Feb. 2000, pp.12–16.

[2] "Introduction to Policy-Based Networking and Quality of Service," IPHighway, White Paper, Jan. 2000.

[3] R. Boutaba, K. El-Guemhioui, and P. Dini, "An Outlook on Intranet Management," *IEEE Commun. Mag.*, Special issue on Intranet Services and Communication Management, Oct. 1997, pp. 92–97.

[4] R. Boutaba and S. Znaty, "An Architectural Approach for Integrated Networks and Systems Management," *ACM-SIGCOM Comp. Commun. Rev.*, vol. 25, no 5, Oct. 1995, pp. 13–39.

[5] M. Sloman, "Policy Driven Management For Distributed Systems," *Int'l. J. Net. Sys. Mgmt.*, vol. 2, no. 4, Dec. 1994, pp. 333-60.
[6] A. Westerinen *et al.*, "Terminology," IETF, Internet draft, draft-ietf-policy-terminology-02.txt, Nov. 2000; http://www.ietf.org/internet-drafts/draft-ietf-policy-terminology-02.txt
[7] "Resource Allocation Protocol (rap)," http://www.ietf.org/html.charters/rap-charter.html
[8] D. Durham *et al.*, "The COPS (Common Open Policy Service) Protocol," IETF, RFC 2748, Jan. 2000; http://www.ietf.org/rfc/rfc2748.txt
[9] "Policy-Powered Networking and the Role of Directories," 3COM, White Paper, July 1998.
[10] "Policy Based Networking Products, Design and Architecture," IPHighway, White Paper, Jan. 2000.
[11] "Intel COPS client Software Development Kit," http://www.intel.com/ial/cops
[12] "COPS Download Page," http://www.vovida.org/protocols/downloads/cops
[13] K. Chan *et al.*, "COPS Usage for Policy Provisioning," IETF, RFC 3084, Mar. 2001; http://www.ietf.org/rfc/rfc3084.txt
[14] M. Fine *et al.*, "Differentiated Services Quality of Service Policy Information Base,"IETF, Internet draft, draft-ietf-diffserv-pib-03.txt, March 2001; http://www.ietf.org/internet-drafts/draft-ietf-diffserv-pib-03.txt
[15] D. Rawlins *et al.*, "Framework of COPS-PR Policy Information Base for Accounting Usage," IETF, Internet-Draft, draft-ietf-rap-acct-fr-pib-01.txt, July 2000; http://www.ietf.org/internet-drafts/draft-ietf-rap-acct-fr-pib-01.txt
[16] J. Ottensmeyer, M. Bokaemper, and K. Roeber,"A Filtering Policy Information Base (PIB) for Edge Router Filtering Services and Provisioning via COPS-PR," IETF, Internet draft, draft-otty-cops-pr-filter-pib-00.txt, Nov. 2000; http://www.ietf.org/internet-drafts/draft-otty-cops-pr-filter-pib-00.txt
[17] M. Li *et al.*, "IPSec Policy Information Base," IETF, Internet-Draft, draft-ietf-ipsp-ipsecpib-02.txt, Mar. 2001; http://www.ietf.org/internet-drafts/draft-ietf-ipsp-ipsecpib-02.txt
[18] M. Fine *et al.*, "Framework Policy Information Base," IETF, Internet-Draft, draft-ietf-rap-frameworkpib-04.txt, Nov. 2000; http://www.ietf.org/internet-drafts/draft-ietf-rap-frameworkpib-04.txt
[19] K. McCloghrie *et al.*, "Structure of Policy Provisioning Information (SPPI)," IETF, Internet draft, draft-ietf-rap-frameworkpib-06.txt, Feb. 2001; http://www.ietf.org/internet-drafts/draft-ietf-rap-frameworkpib-06.txt

Additional Reading

[1] "Internet Engineering Task Force," http://www.ietf.org

Biographies

ANDREAS POLYRAKIS (apolyr@cs.toronto.edu) works as a network engineer at the National University of Athens, Greece. Recently he received his M.Sc. degree from the University of Toronto, Canada. His main area of interest is policy-based networking, but he is also interested in other areas of networking, such as directory-enabled networking, active networks, and QoS provisioning in IP networks.

RAOUF BOUTABA (rboutaba@bbcr.uwaterloo.ca) has been teaching networks and distributed systems in the Department of Computer Science of the University of Waterloo since 1999. He conducts research in integrated network and systems management, wired and wireless multimedia networks, and quality of service control in the Internet. He has been program chair of the Technical Committee on Information Infrastructure of the IEEE Communications Society since 2000 and chair of the IFIP working group on network and distributed systems management since 2001. He was the recipient of Premier's Research Excellence Award in 2000.