Performance of a Multi-Tiered Policy-Based Management System

K.L. Eddie Law, Achint Saxena
University of Toronto

Abstract: Apart from providing device management on the Internet, it is essential to offer Quality of Service (QoS) to different users with different service requirements. Policy-based management provides policy control on network devices to achieve this objective. Internet Engineering Task Force (IETF) recommended a two-tiered policy-based management (PBM) architecture. This recommended design is based on Common Open Policy Service (COPS) protocol and Lightweight Directory Access Protocol (LDAP). There are COPS policy outsourcing and provisioning models. LDAP is for storing and fetching policy rules. However, several fundamental limitations exist in the recommended design. System scalability and cross-vendor hardware compatibility are the obvious drawbacks. In this paper, we study the system performance of PBM through experiments. Consequently, improved multi-tiered policy-based management architecture is proposed, and it is known as a unified policy-based management (UPM). For this new design, there are several extensions introduced that offer system flexibility and scalability. Particularly, an intermediate entity between policy server and network routers, the Policy Enforcement Agent (PEA), is introduced. In this proposed architecture, by properly extending network protocols, by installing multi-vendor hardware modules on-the-fly, and hence by interpreting and translating request and decision messages at the agents, the architecture enables a dynamic Unified Information Model throughout the control portion of the design. The multi-tier architecture provides flexible and scalable system design, and it allows executions of policy rules with dynamic addition of new equipment during run-time. To complete the design, communication protocols between policy servers and agents are established that facilitate load sharing and balancing mechanism and improve the system scalability issue. In the following, we discuss the architectural design and its system performance.

Key words: Unified Policy-based Management, Common Open Policy Service protocol, Lightweight Directory Access Protocol, network management

The original version of this chapter was revised: The copyright line was incorrect. This has been corrected. The Erratum to this chapter is available at DOI: 10.1007/978-0-387-35620-4_43

1. INTRODUCTION

Traditionally, Internet provides "best-effort" datagram service, which has limited efficiency for large volumes of varying kinds of application traffic. Performance related parameters, such as bandwidth and delay, in the Internet are currently unused for connection management. Managing the Internet and providing different Quality of Service (QoS) classes [1, 5] to different users becomes essential. This allows a variety of applications, for example, the Voice over Internet Protocol (VoIP) and the real-time data transmission, to operate harmoniously on the Internet.

To manage different QoS schemes and classes, policy-based management (PBM) has been proposed [1, 6] in which different levels of services are provided according to assigned policy rules. These policy rules define admission control, traffic shaping and scheduling mechanisms for different users and traffic connections. The rule parameters may include desired bandwidth, jitter, delay, duration and connection volume limitations.

Ignoring the directory server as shown in Figure 1(a), the recommended framework [1] from the Internet Engineering Task Force (IETF) is a two-tiered architecture that consists of policy manager (PM)/policy server (PS), and network routers at the edge or boundary of a policy domain (PD). With the Common Open Policy Service (COPS) [2] protocol, there are two entities defined in the framework and they are the Policy Decision Point and the Policy Enforcement Point. Whenever a Policy Enforcement Point (PEP) receives a service request, it forwards the request to a Policy Decision Point (PDP). Subsequently, PDP makes decisions by fetching policy rules that are stored in a centralized location using a Lightweight Directory Access Protocol (LDAP) [3] database. Upon returning the final decision, PEP is responsible for enforcing the policy rules in the domain. Usually PEP is located inside an Edge Router (ER), while the PDP operates within a Policy Server (PS). In general, there can have several PDPs in a policy domain, one for each kind of PEP device.

The COPS developed by IETF is a lightweight, and flexible client/server based protocol. It is designed for policy communications among the PEPs and PDPs. The aforementioned COPS mechanisms belong to the outsourcing model. Its extension, the COPS-PR [7], provides bulk DiffServ [1] type enforcements for policy provisioning. Typically, the current two-tier architecture assumes that the whole network uses a standard COPS protocol and message format. There are several benefits to this two-tier model. For example, it is simple and moderately extensible. It can be adapted to the Resource reSerVation Protocol (RSVP) paradigm easily. However it has several limitations. The most conspicuous limitation is the scalability issue in heterogeneous networks, in terms of router manufacturers and models.

Moreover, each router may have different implementations of COPS, ranging from something as minor as different versions to something crucial such as different message formats. Though the packet types of COPS are now standardized, the finalization of the message and content structures such as requests or decisions is still an ongoing process. Even if the message content is standardized, different vendors can still establish proprietary message structures and policy rules. Therefore, inter-vendor COPS compatibility will be a continuing problem.

Another consequence of this is that legacy routing equipment will not be supported. This is of major importance because it is a hurdle towards the deployment of QoS in current networks. To offset startup costs, any QoS system should integrate well with legacy equipment. Moreover, a single PEP could be connected to multiple PDPs, each may have a different client type [2]. On the other hand, if a single PDP fails then this may cause continuity problems at the PEP. A backup PDP may exist. However, the PEP will not automatically be aware of the location of the backup system. Though a list of backup PDPs may be manually configured into PEP at registration time, it may be problematic if all of them are down or unable to accept any more connections, or if the list is outdated.

As a result, the recommended two-tier architecture is not scalable, and it does not adapt itself easily to control different equipment from different vendors. There are no load sharing and balancing mechanisms at PDPs. Hence in large networks, we may have many congested and many idle PDPs. In this design, an PEP is associated with one PDP. The PEP keeps waiting for a timeout between each try if the PDP is busy. All these issues must be addressed before QoS and policy-based management become widely adapted on the Internet. With the limitations we discovered in the standard PBM, a testbed and an improved design have been constructed to improve the system performance. The improved architecture, Unified Policy-based Management (UPM), will be discussed in this paper.

2. SYSTEM ARCHITECTURE OF THE UNIFIED POLICY-BASED MANAGEMENT (UPM)

The system architecture of Unified Policy-based Management (UPM) is shown in Figure 1(b). It is a multi-tiered policy management system that is conceptually similar to those using middleware or agent technology. In each policy domain (PD), apart from the directory server, UPM is a three-tier architecture. The top tier is where the policy managers or the policy servers (PMs/PSs) locate. Their roles are responsible for making final policy decisions. The newly introduced middle-tier entity, the Policy Enforcement

Agent (PEA), coordinates the communications and creates transparency between network routers and the policy servers. There are several functional roles for the PEA. For example, if the edge routers do not understand the policy decisions, the PEA is responsible for translating and executing policy rule. The bottom tier consists of different network routers, such as the edge routers (ERs), boundary routers (BRs), and possibly core routers (CRs).

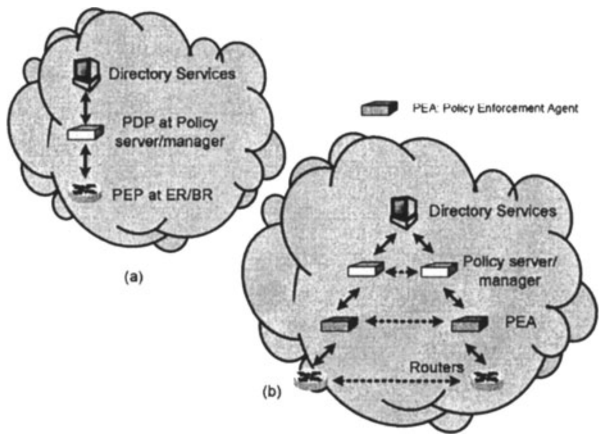

Figure 1. In a policy domain: (a) IETF's Recommended Model; (b) The UPM Model.

Each PEA at the middle tier enforces all received policy decisions from policy servers. Its objective is to provide guaranteed QoS flows at ERs for end-hosts within its PD and BRs from other PDs. The primary functions of the PEAs include: (1) relaying protocol messages between PS and ERs/BRs; (2) translating different policy rules; (3) distributing different policy sessions to PSs; (4) enforcing guaranteed QoS traffic at edge/boundary routers (5) informing ERs/BRs about other PEAs around, if required.

UPM works for both outsourcing and one-way decision provisioning. In Figure 1(b), a policy server receives a service request through either the COPS request from the edge routers or from the system administrators with domain level decision. Policy rules are obtained from the directory and the PS finalizes a to-be-applied policy rule and delivers this decision to an PEA at the middle tier for enforcement.

Another enhancement of the UPM is to provide load balancing and load sharing mechanisms at the middle and upper tiers. When there are numerous routers to control in one PD, there may have multiple PEAs. UPM provides inter-PS communications, inter-PEA communications and PS-PEA communications. Upon considering multiple vendor and multiple equipment scenarios, the UPM can be used to operate dynamically with a concept of unified information model (UIM) [8] between the policy servers and the middle-tier entities. For those who are interested in UIM can consult the reference [8]. This design unifies the system design at both the policy server and the middle-tiered entity. Therefore, we can add more PSs and PEAs at any time if required without disrupting any existing policy controlled services.

3. POLICY ENFORCEMENT AGENT

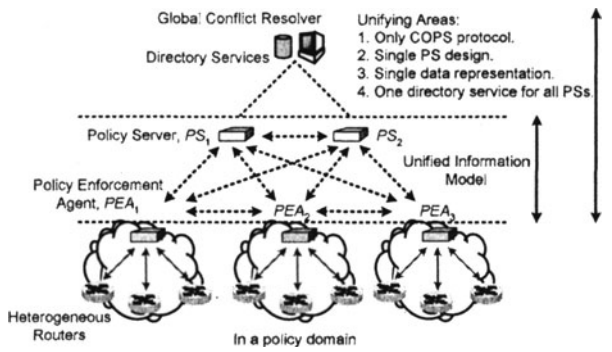

Figure 2. Implementation Model of the UPM System.

UPM reuses some components in IETF's two-tiered framework model. While some of these components have enhanced capabilities, some others have less complicated functionalities. In order to enable the unified information model, COPS is the only policy protocol used to communicate among PEAs and PSs. In UPM, all PSs and PEAs should use the same version of COPS, all network routers in the bottom tier are not required to understand COPS and they can run different versions of COPS or with different message contents. At the current design, PSs communicate with only one centralized directory server/database because of the static nature of the policy rule database. Most newly input policy rules will not be modified within short time duration. All network routers are required to communicate via the PEA under all conditions for traffic monitoring. Figure 2 shows a complete conceptual design of the UPM.

The PEA is the most important component in this multi-tiered architecture. It is a multi-functional unit that enables system flexibility and scalability. PEA is more than just a proxy for service requests; it resembles a gateway to the PDPs. More importantly, the presences of the PEAs are not supposed to be noticeable by other network components. It works as an PDP when it communicates with network routers. On the other hand, it operates as an PEP when it works with the policy servers. On summarizing, it provides features as shown follows.

- A central and transparent point of connection for all routers, and seamless integration with multiple policy servers. All routers connect to PEAs, which distributes the loads to PDPs.
- A transparent translation unit between routers and policy server, it translates different types of QoS and COPS requests into a version of COPS understood by the PDPs. Translation is done on a module basis, with the modules being dynamically loaded into the PEA.
- Dynamic load balancing/sharing for connections to and from routers.
- A dynamic TCP-based switching mechanism for COPS messages when translation is not required, to reduce latency.
- There may be multiple PEAs in a policy domain. They can discover and communicate with each other through the inter-PEA process.
- A user interface is designed to manage and monitor the network.

4. NOVEL DESIGNS IN UPM

4.1 System Improvements with PEA

4.1.1 COPS and Content Translation

One important function of the PEA is to translate unintelligible COPS to the native COPS version and format at PS. As shown in Figure 3, there exist two types of connections in UPM.

Non-COPS routers – These are the 'push' only routers and they are configured manually. A user interface can be used to monitor the router. A user-based input or a client program can replace the 'pull' mechanism for the router. The operations can then be pushed onto the router using those mechanisms that are acceptable by the routers, for example, the Command Line Interface (CLI) on telnet. For 'pull' capable routers with other, possibly proprietary, policy protocols, a basic store, convert and forward mechanism can be used at the PEA. The forwarding mechanism consists of opening a new connection or using an existing connection to PS, to send and receive request and decision messages.

COPS compliant routers/PEPs – Different vendors may have different implementations of COPS. Even if they are compliant to the standard, the request and decision message structures can be different from each other. In these cases, translations at the PEA will be required. There are also two different cases to consider: 1) for non-standard implementation of COPS, the same mechanism for the non-COPS routers can be re-utilized; 2) for COPS standard compliant routers, individual COPS message can be converted and then forwarded as necessary. This may require the PEA to know how to do proper translation. Rule translations are not covered in this paper, but the setup of the translation module repository will be discussed in the following subsection.

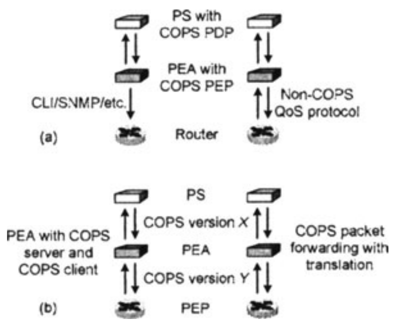

Figure 3. Translating Mechanisms for (a) Non-COPS Router. (b) Non-Native COPS PEP.

4.1.2 TCP-Bypass

TCP-bypass operates at the PEA. It represents one of the most important features in UPM. If router and PS communicate using the same COPS message format and information content, then it does not need any intermediary translation. TCP-bypass mechanism simply forwards incoming connections or requests at the transport layer to one of the PSs that is known to the PEA. It removes those unnecessary translation overhead and latency at the PEA. The design concept is illustrated in Figure 4.

The TCP-bypass operates like a transparent proxy but is subtly different. Incoming TCP connections are allowed to pass through the PEA, or are forwarded without processing, according to the firstly received COPS data packet. Using the information in the COPS header, the version and implementation signature may be deduced. If this matches the native COPS format and information content, then a new TCP connection is opened to a selected PS and the finite state information regarding to this TCP connection is cached. The TCP-switching can be done either at TCP-level, which is TCP connection-based, or at the COPS level, or both. This creates a "total transparent" connection between the PEP at router and the PDP at PS for they set up and maintain the persistent connections. To handle the bypassed COPS messages, careful designs are required to mirror and handle COPS connection establishment messages to the PSs, their corresponding replies from the PS, and those unsolicited PS messages.

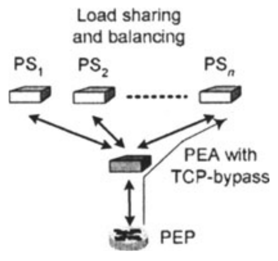

Figure 4. TCP-Bypass.

4.1.3 Load Balancing and Sharing Solutions

Load balancing will be mainly carried out by the PEAs. A table of PSs known to the PEA can be kept. Each PEA does not need to know all PSs at the beginning; it will learn dynamically the presences of different PSs because a set of inter-PS communication protocols is designed. Through the PEA-PS communications, each PEA is able to monitor those PS-related variables such as the CPU utilization, memory usage, link utilization, ping times, and link capacity/congestion. Different weighted values can be assigned to different parameters and to different PSs. As a consequence, new requests to each PEA can then be serviced in a Weighted Round Robin (WRR) fashion.

4.1.4 Multiple PEAs, Inter-Entities Communications

To allow system scalability and reliability, multiple PEAs are necessary. After PEA reaches a certain number of opened connections, CPU usage, network congestion, and other factors, the PEA may decide to issue COPS redirect message to those routers sending new requests. Additionally, it may be beneficial to connect to PEA that is closer to the routers, rather than the one that is far away in a large network. As a result, it may be desirable to have multiple PEAs in a PD. Moreover, when there are multiple PEAs, some of them can be module-specific, or vendor-specific, or area-specific if there is a high concentration of certain kind of routers in one small area.

To support system scalability, it is necessary to share information among PEAs, and among PSs. Therefore, inter-PEA, inter-PSs and PEA-PS communication protocols should be investigated. There can have sophisticated and simple alternative designs for these communication protocols, but they are too large to be included in this paper.

5. PERFORMANCE OF UPM

5.1 Setup of Current Prototype

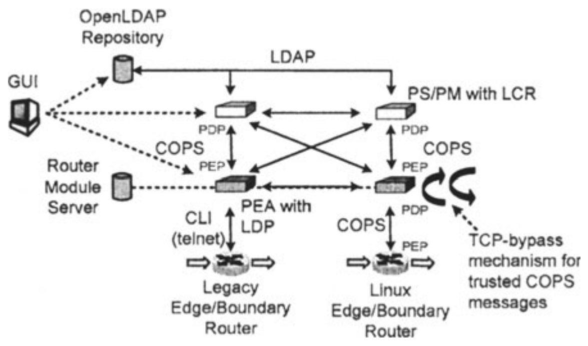
OpenLDAP Repository
LDAP
GUI
PS/PM with LCR
PDP
COPS
PEP
Router Module Server
PEA with LDP
CLI (telnet)
COPS
TCP-bypass mechanism for trusted COPS messages
Legacy Edge/Boundary Router
Linux Edge/Boundary Router

Figure 5. The UPM Testbed Prototype.

A system prototype for the UPM is shown in Figure 5. All servers and clients run on Pentium III computers with Linux kernel version 2.4.5. TCP-bypass is implemented at the Linux kernel level in PEA for concept verification. The implementations of most of these system components have not been optimized for heavy use; however, the results obtained are sufficient to demonstrate the superiority and feasibility of the UPM design.

5.2 System Performance of Two-Tiered PBM

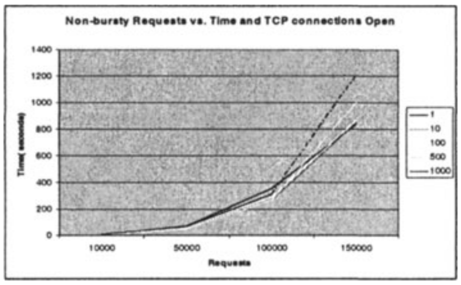

Figure 6. Non-Bursty Traffic with 1, 10, 100, 500 and 1000 TCP Connections Opened.

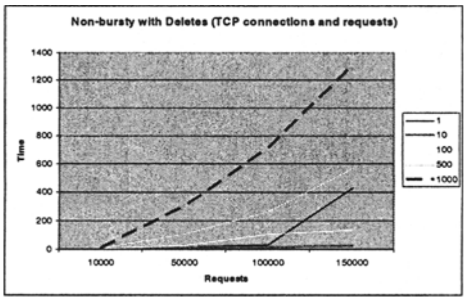

Figure 7. Non-Bursty Traffic with 1, 10, 100, 500, 1000 TCP Connections, DRQ Sent.

Two different kinds of experiments are made for this configuration. One series of tests is to send a burst of requests simultaneously to the policy server without waiting for any COPS decisions. We mark the test as "bursty" in the following graphs. For the second series of tests, the client only sends request when the decision for its previous request has arrived. We mark this test as "non-bursty" in the following graphs. Moreover, if a graph is marked with "deletes," it meant that one COPS Delete (DRQ) message is also sent whenever each decision has just received. Typically, it attempts to reduce the state information, and improves the performance of the policy server.

The recommended framework from IETF as shown in Figure 1(a) is tested for comparison. In Figure 6, we demonstrated the result of the "non-bursty" tests. Upon increasing the number of request messages, the time required to process these messages is growing exponentially. This graph demonstrates the limitation to have only one PS in PD would deteriorate the system performance when the number of routers increased in PD. Moreover, undeleted requests in this case meant more memory was used to store the states of the connections. Similarly, rehashing of the hash table for increased connections slowed down the response times. This explained the almost exponential increase in the total time taken to process the requests. The preliminary results for the "bursty" and "non-bursty" were very similar; therefore, the test graph for the "bursty" traffic is not shown.

In Figure 7, each request state is deleted whenever a decision has been made. We intend to test the computation performance of the policy instead of other limiting factors, e.g., memory size. In this case, the server keeps less information when compared to the test condition in Figure 6. However, the system performs better with smaller number of TCP connections. When the number of TCP connections gets larger, then there were 1000 TCP connections, it did not show any improvement at all. This indicates that the performance of the standard PBM will deteriorate rapidly with increasing number of service requests, i.e., the scalability and reliability problems.

5.3 System Performance of UPM

5.3.1 TCP-Bypass Overhead

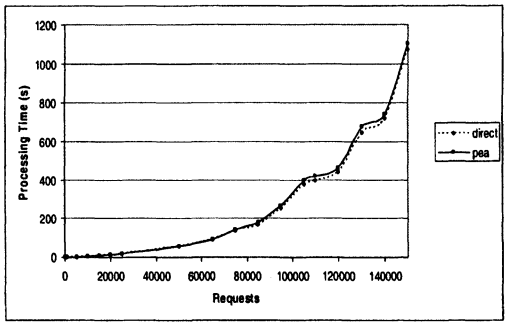

Figure 8. UPM with PEA.

In this set of tests, the TCP-bypass overhead at the PEA is measured. The total time spent from an PEP to an PS through the PEA in UPM configuration is compared to that in the standard PBM design. Figure 8 shows the processing time charts for both cases. A small overhead via TCP-bypass is observed from the measurements. For lower numbers of connections, the overhead is insignificant. Any variability is due to network issues. For a larger number of requests, > 12,000 TCP connections, the measured overhead is less than 5% with current implementation. In future, a faster indexing structure such as hash table in kernel space should be used.

5.3.2 Multiple-PS and Multiple-PEA Evaluation

In this section, we continue to demonstrate the scalability on the overall system performance. In these sets of experiments, there can have a number of PSs and a number of PEAs operating together. As shown in Figure 9, given a fixed number of policy servers, the performance of the system is about the same with one, two or three PEAs. This again demonstrates the transparency of the PEA in UPM. On the other hand, the throughput has been improved whenever one more policy server is added to the system as shown, whereas the standard PBM system can only achieve the throughput performance with only single PS in a system.

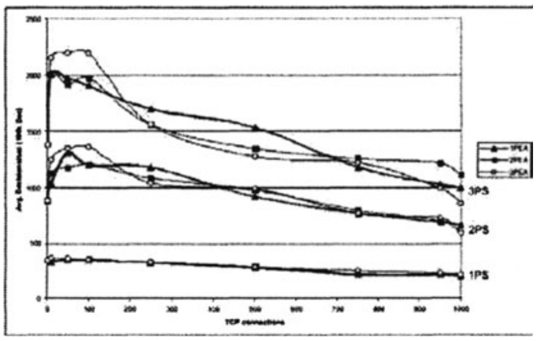

Figure 9. Multiple-PEA, Multiple-PS – Average Throughput for COPS Decisions.

6. CONCLUDING REMARKS

In this paper, we studied the drawbacks of the standard PBM system. This paper reports extensive experimental results. Through understanding the cons of the standard PBM, a multi-tiered unified policy-based network management (UPM) scheme has been proposed and discussed. By implementing a prototype, we have demonstrated the several advantages we have discussed in the paper. The TCP-bypass mechanism does not introduce excess latency on PEA that sits at the middle tier of the architecture. Besides, PEA provides load sharing, load balancing mechanisms. The design of UPM indeed offers some solutions to the problems facing the policy-based QoS industry today.

REFERENCE:

[1] R. Yavatkar, D. Pendarakis, R. Guerin, *A Framework for Policy Based Admission Control*, RFC 2753, IETF, January 2000.

[2] J. Boyle, R. Cohen, D. Durham, S. Herzog, R. Raja, and A. Sastry, *The COPS (Common Open Policy Service) Protocol*, RFC 2748, IETF, January 2000.

[3] M. Wahl, T. Howes, S. Kille, *Lightweight Directory Access Protocol (v3)*, RFC 2251, Dec. 1997.

[4] S. Blake, D. Black, M. Carlson, E. Davies, Z. Wang, and W. Weiss, *An Architecture for Differentiated Service*, RFC 2475, IETF, December 1998.

[5] R. Braden, D. Clark, S. Shenker, *Integrated Services in the Internet Architecture: an Overview*, RFC 1633, IETF, June 1994.

[6] H. Mahon, Y. Bernet, S. Herzog, and J. Schnizlein, *Requirements for a Policy Management System*, work in progress, November 2000.

[7] K. Chan, D. Durham, S. Gai, S. Herzog, K. McCloghrie, F. Reichmeyer, J. Seligson, A. Smith, and R. Yavatkar, *COPS Usage for Policy Provisioning*, RFC 3084, IETF, March 2001.

[8] K.L.E. Law, "*XML* on *LDAP* Network Database," in *Proc. IEEE Canadian Conf. Elec. & Comp. Engineering*, Halifax, Canada, 2000.