On the Use of Peer-to-Peer Architectures for the Management of Highly Dynamic Environments

Carlos Kamienski, Djamel Sadok, Joseane Farias Fidalgo, Jennifer Lima
Universidade Federal de Pernambuco, Brazil
Börje Ohlman
Ericsson Research, Sweden
E-mail: {cak, jamel, joseane, jennifer}@gprt.ufpe.br, Borje.Ohlman@ericsson.com

Abstract

Scalable distributed management is a key challenge for current Internet services and necessary for future ubiquitous services of wireless mobile users. Policy-based Management (PBM) is seen as a practical solution for dealing with the needs of new advanced services for highly dynamic wireless environments. The IETF developed a two-tier client/server PBM framework, yet it requires some important extensions in such environments. In this paper we look into new management mechanisms for dealing with these limitations by proposing the Peer-to-Peer Policy Management Infrastructure (P4MI), a PBM framework based on peer-to-peer technology as the main enabling mechanism. We instantiate this abstract framework to Ambient Networks, a new concept aimed at creating network solutions for new mobile and wireless systems. We show that the P4MI is both a scalable and a complete PBM solution for coping with the challenges of Ambient Networks.

1 Introduction

The Internet is literally on the move. We need to review our management architectures to attend to such challenging new services. Policy-based Management (PBM) can be seen as a significant step towards automatic and dynamic configuration. PBM is an approach to simplify the administration of a complex network infrastructure by establishing policies to deal with situations that are likely to occur in a largely automated fashion. Policies represent business targets and objectives that describe how to allocate or configure resources in a general sense to meet them. In networking, they also refer to the ability to manage and control the access to the network using high-level abstracted rules and decisions. Formally, a policy may be defined as an aggregation of rules, where each one consists of one or more conditions and actions.

The PBM framework developed by the IETF [3] is a model for policy management comprised of Policy Decision Points (PDPs, also known as policies servers), Policy Enforcement Points (PEPs), policy repository and Policy Management Tool (PMT). The PDP is responsible for handling requests, querying the policy repository, making decisions and distributing them to the PEPs, which are the entities (e.g. routers) where the actions actually are implemented and/or enforced. The PMT support the specification, editing, validation and administration of policies, through a graphical interface. These policies are then stored in the policy repository. Some protocols are necessary within this framework, such as COPS for PDP and PEP interworking and LDAP for the PDP to be able to access policies in the policy repository.

The IETF PBM framework was not designed for having in mind heterogeneous scenarios where frequent changes are the rule, such as dealing with mobile and wireless users with highly dynamic usage patterns and unpredictable service needs. One of the most evident limitations is related to system scalability [1], since in the two-tier model adopted by IETF one PDP can only control a limited number of PEPs. Another source of scalability concerns is the very nature of the client/server solution, where features such as load balancing, fault tolerance and network self-configuration do not fit the model well. Some proposals have been emerging in the last years, acknowledging the important step represented by the IETF framework and extending it to deal with typical requirements of a more dynamic scenario. The Unified Policy-based Management (UPM) [1] proposes a three-tier model, by adding an intermediary entity between the PDPs and PEPs, known as PEA (Policy Enforcement Agent). This hierarchical model has some limitations, since it does not abandon the client/server

Proceedings of the Fourth Annual IEEE International Conference on Pervasive Computing and Communications Workshops (PERCOMW'06)
0-7695-2520-2/06 $20.00 © 2006 IEEE

model, yet it adds complexity to it. In [2], a PBM framework for Always Best Connected (ABC) users is proposed, introducing user, terminal and network profiles and also a policy based handover management. However, it also preserves the client/server model.

In this paper, we propose and investigate the benefits of using the P2P Policy Management Infrastructure (P4MI), a PBM framework based on the peer-to-peer (p2p) technology as the main enabling mechanism. A hierarchical p2p architecture, using a Distributed Hash Table (DHT) [4] is the core of P4MI. It introduces the Policy Decision Network (PDN), a network comprised of policy servers interconnected by a DHT.

As an application of P4MI, we develop PBMAN, a PBM solution for Ambient Networks (AN), which are a new concept aimed at creating network solutions for mobile and wireless systems beyond 3G [6]. The key AN concept is network composition, for allowing dynamic user access to services. Composition can be thought of as a mechanism for automatic negotiation of roaming and/or service level agreements (SLAs), which today are done manually, mostly off-line. PBMAN maps concepts of P4MI into the AN scenario, providing efficient and scalable mechanisms for network composition and policy distribution and retrieval. We also have implemented a prototype of PBMAN to demonstrate these ideas. To our knowledge, this is the first work that uses P2P architectures for PBM. We decided to consider P4MI and PBMAN as separate entities because we expected this will provide us higher flexibility and more formalism when modeling a different problem area.

There are three main motivations for proposing a new framework for policy management. First, the IETF framework is focused on specific policy areas, such as QoS and security, and on simpler problems from typical corporate networks. We are broadening the scope of applications for policy management considering service usage in the global Internet. Second, the 3G/4G scenarios targeted by our proposal consider a huge number of mobile wireless users with highly dynamic mobility and service usage patterns. Traditional PBM does not provide dynamic information sharing among policy domains. Requirements specified for Ambient Networks [4] give an idea of the complexity of such new environment. And third, for complying to important requirements associated with those environments, which are provided by the p2p technology, such as scalability, fault tolerance and load balancing.

The rest of the paper is structured as follows. Section 2 presents the abstract P4MI architectural framework, whereas in Section 3 the instantiation of P4MI to PBMAN is introduced. Finally, section 4 concludes the article with some final remarks.

2 P4MI Architectural Framework

The Internet growth dictates the adoption of a distributed architecture that guarantees rapid anywhere anytime access to resources and services. We found in the DHT structure an important technology to enable the building of such scalable architecture. Figure 1 shows the P4MI envisioned architecture, comprised of the Policy Decision Network (PDN), PEPs, policy users and their interaction. PEPs and users are called policy agents. The big picture shows a hybrid hierarchical p2p architecture based on super-nodes, yet having a structured organization. Policy servers interwork through a DHT network, policy agents communicate with each other also in a p2p fashion and agents communicate with policy servers in a client/server structure. This is aimed at providing both scalability and fault tolerance, since policy servers are robust and stable carefully chosen hosts, whereas policy agents may be low capacity, fragile equipments and present variable connectivity. The latter characteristics are one of the weaknesses of pure DHT network, since they may temporarily store important information.

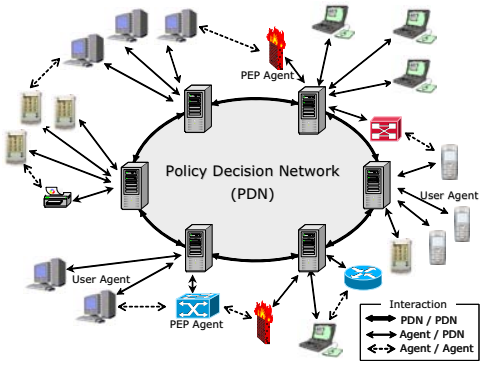

Figure 1 – P4MI architecture

2.1 Policy Decision Network

The PDN is the heart of P4MI, primarily responsible for functionalities related to PBM and p2p interworking, such as storing and retrieving policies and taking decisions upon receiving service requests. The PDN is comprised of two main entities, Decision Points and Repositories. A Decision Point, also called PDN Node or P-Node, is a policy server, which accepts some part of the whole PDN work. There is a significant difference between a PDP in the IETF PBM model and a P4MI P-Node. The latter is able to interwork with other P-Nodes by design via a distributed p2p DHT-based network, which is called PDN-ring. The PDN-ring provides P4MI with the inherent features of p2p systems, such as load

balancing, fault tolerance, scalability and ability to deal with system's heterogeneity.

The PDN has also two information repositories, the Policy Repository (PR) and the Management Information Repository (MIR). The PR must store policies according to some requirements, such as making easier the process of searching and retrieving policies. P4MI does not specify a particular storage technology, such as LDAP or a DBMS, as long as different implementations are able to interwork. In addition to the policies themselves, there is a need for keeping information about entities that are to be managed with policies, which is stored in the MIR. Typical information in the MIR is profile information for policy agents and targets (section 3.4.1), associations between policies and targets, policy to device mapping and configuration, control and management information of the PDN itself.

Bootstrapping the PDN is an important issue, yet not necessarily part of the P4MI framework, since this is also a design decision of current p2p networks. Each network is free to decide PDN bootstrapping rules, that may follows a wide variety of approaches, from manual configuration to automatic maintenance of the P-Node member list.

2.2 Policy Environment

Policy Agents are represented by hosts, equipments or devices used by users or networks for providing services and enforcing policies. The interaction between agents and the PDN is based on the hierarchical p2p DHT-based adopted approach. Agents may be comprised of two parts, which may be simultaneously present or not: PEPs and Users. PEPs are agents aimed at enforcing policies, such as routers and firewalls. PEP agents are also software and hardware for providing services, which must enforce policies of right of use, security, accounting, etc. Users represent devices or networks of connected devices that a given real user is using for accessing services.

Bootstrapping an agent is simple and only requires the address of one P-Node to be manually configured at the first time. At any time an agent connects to its PDN, it receives an up-to-date list of P-Nodes.

A typical policy transaction starts with an event, such as a service request arriving at a PEP agent, which in turn sends a service request to the PDN. In response, the PDN will select those policies that match particular conditions of that query (e.g., service, time, access). In that case, all policies related to this user must be recovered, i.e., policies associated directly to the user and indirectly, via target[1] relations (e.g. groups, services). The information model is used as the main guide in the selection process, defining policies, targets and their associations.

The basic algorithm can be summarized in five phases: policy retrieval → policy selection → conflict resolution → action analysis → action enforcement. A complete algorithm for a particular instantiation of P4MI would be more complex, considering different factors, such as the particular information model used. As a matter of fact, an algorithm for policy processing has been developed for PBMAN. Real policies are not presented in this paper, due to lack of space.

2.3 Policy Data Model

An information model is the abstract representation of managed entities and how they relate to each other, whereas a data model maps a given information model into a particular storage system [7]. P4MI provides a data model, structured upon the PDN, using a data management model that extends the DHT network. It is designed for enabling the mapping from different information models. One known limitation of DHT-based systems is that they only support exact-match lookups, i.e., lookup operations require the user to provide the exact key used to generate the hash table index for storing the information. This lack of flexibility requires additional data management features to be added, in order to make it able to deal with more complex data structures, such as lists and tables. OpenDHT shares a similar concern [11].

Most DHTs systems offer a basic mechanism for storing basic registers, which suffer from the exact key limitation. Therefore, P4MI adds a new layer for dealing with policy storage and retrieval. The data management architecture (Figure 2) is divided into three functional layers. Layer 1 includes a DHT network and a simple storage system. Layer 2 provides a higher-level view of the data management system, adding some new functionality, such as modeling entity relations and indexing, which are missing in the Layer 1. Layer 3 includes all policy readers and writers, such as Users, PEPs and the PMT.

The architecture includes three APIs. The Data Management API (DM API) is external (peer-to-peer) and the most important one from the standpoint of those entities who actually read and write policies. It provides functions aimed at hiding the complexity of the DHT/Storage system. The DHT API is an internal API used by the data management layer to access low-level functions. Finally, the Storage API (STO API) is used by the DHT network for storing records.

In Layer 1, P4MI models data as a DHT-based middleware extended with some storage functionality

[1] Targets are entities which policies may be associated to.

to deal simple records, which are comprised of a key field and a content field. The key field is the DHT hash key, further divided into a three sub-fields tuple {Key1, Key2, Key3}. Key1 must be filled with the network identifier, allowing multiple networks to coexist (i.e., by network composition, for PBMAN). Key2 represents the identity of the particular managed entity. Key3 is used to represent the record type, identified by reserved words (defined by particular instantiations). The content field is a single byte stream whose semantics are application specific. An example of mapping this data model to PBMAN is presented in section 3.5.

Layer 2 uses the DHT API to provide storage capabilities to a particular application. In other words, its design and implementation need to be customized each time P4MI is instantiated (such as PBMAN).

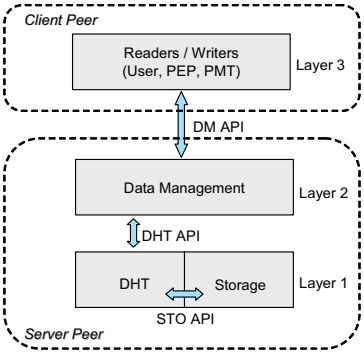

Figure 2 – P4MI Data Management Architecture

3 PBMAN - Ambient Networks Management

PBMAN (Policy-based Management for Ambient Networks) is an example instantiation of the general P4MI framework for dealing with Ambient Network requirements. We also developed a prototype called X-PBMAN that is a simplified proof-of-concept implementation, intended to be used for exploring and understanding the combined PBM and P2P approach adopted in PBMAN. X-PBMAN is implemented using the X-Peer [8], a hierarchical p2p middleware, which currently uses FreePastry [9], an open source implementation of Pastry [10] in Java. Although other hierarchical p2p approaches are also good candidates (e.g. [12]), X-Peer middleware fits the P4MI model well, where P-nodes are modeled as X-Peer super-nodes and policy agents are modeled as X-Peer clients.

3.1 Ambient Networks

Ambient Networks (AN) is a new networking concept, which aims to enable the cooperation of heterogeneous networks belonging to different operator or technology domains [6]. This cooperation should be transparent, under demand and "plug-and-play", i.e., no previous configuration or negotiation is required between network operators. The main innovative concept of AN is network composition, in order to allow rapid adaptation of the network domain topology as required for mobile users and moving networks. Formally, an Ambient Network is a collection of networks and/or devices sharing a common control plane, called Ambient Control Space (ACS). The ACS is comprised of a collection of functional entities (FE), each one reflecting different control and management tasks, such as composition, mobility, security and QoS.

Network composition is the key architectural concept and the main challenge of Ambient Networks, aimed at enabling control-plane interworking and sharing of control functions among networks. Composition goes beyond what the Internet and mobile networks can provide today in that interworking is not restricted to basic addressing and routing. Composition enables seamless mobility management, and improved network and service efficiency. It also hides interconnection details of cooperating networks to the outside. Additional information can be found in the Ambient Networks Project webpage[2].

Intuitively, composition can be thought of as a mechanism for automatic negotiation of roaming and/or service level agreements (SLA), which today are done manually. As a real example of network composition, we deployed a scenario of the video server, comprised of two services: basic and premium (with quality of service guarantees). When a remote user logs into the server, it informs its PDN, which in turn start a composition with the remote network (if they are not composed yet). After the composition is completed, the local PDN is able to perform authentication and authorization tasks, because it can access the policies of the remote network.

3.2 Access Control Space

The general PBMAN architecture is focused on the role and implementation of the access control space (ACS). The PDN concept of P4MI is extended in PBMAN for including the ACS functionality. Each P-Node, implements a part of the ACS. Three new FEs have been added for PBMAN, policy FE, p2p FE and Data Management (DM) FE. The policy FE embraces all PBM concepts, including the PDN and policy agent functionality. The p2p FE is comprised of all p2p functions, such as DHT-based policy location, routing, search and retrieval. Another function of the P2P-FE is

[2] http://www.ambient-networks.org

managing PDN rings, as well as enabling PDN/PDN, Agent/PDN and Agent/Agent interactions, as depicted in Figure 1. The DM FE is associated to the p2p FE and implements the Layer 2 of the P4MI's data model.

3.3 Network Composition in PBMAN

Figure 3 depicts the composition of two PDNs. The situation before the connection is shown in Figure 3a. Both PDN_A and PDN_B are single PDNs, each one having four P-Nodes. During the connection process, a new PDN ring (PDN_{AB}) is created and two P-Nodes of each PDN are chosen for being members of the PDN_{AB}. The actual number of P-nodes to take part in the new composed network is subject to a local policy. Since any P-node member of the new composition may act as a gateway between networks, fault tolerance and scalability are achieved in a transparent manner, even when composing multiple network hierarchies.

Composition may take some considerable time to be performed[3], but it will typically happen only at the first access to a remote network. It is necessary for the local P-Node to be able to perform authentication and authorization based on the remote policies. For all subsequent accesses, the P-Node will have instant access to the remote network's information and the response therefore should be immediate.

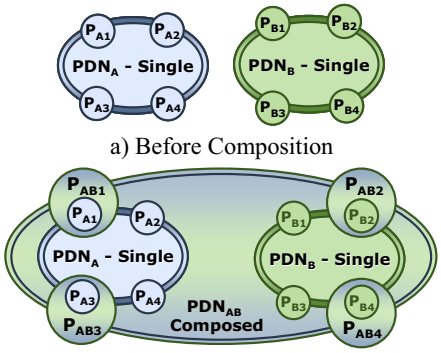

Figure 3 – Composition of PDN_A and PDN^B

The life cycle of a composition between PDNs is comprised of six phases: triggering, negotiation, setup, utilization, decomposition triggering and decomposition execution. Composition triggering is the event that motivates a new composition, usually a service request access from a user to a PEP and then to the PDN for policy processing. Before the policy retrieval is started, the P-Node must check whether a composition is needed. Decomposition triggering is a per-domain choice, such as a timeout when the network realizes that the composition is not being used any longer.

A simplified version of the negotiation is presented in the following algorithm. Notice that unusual situations, such as composition refusal and renegotiation are not included.

Step 1: a composition negotiation starts when a P-Node from the source PDN sends a composition request message to one selected P-Node of the destination PDN. These two P-Nodes will negotiate the composition, each one representing its own network.

Step 2: The composition request is accepted, and the source P-Node creates a new PDN ring (it becomes the bootstrap peer of the DHT) and publishes the list of P-Nodes that will take part of it. This list is obtained by combining its own list of P-Nodes from its home network together with the list of the destination (remote) network provided with the accept message.

Step 3: Both source and destination P-Nodes send join messages to all other P-Nodes of their PDNs that will take part on the new composed PDN.

Step 4: All P-Nodes join the new PDN and republish their policies and other policy-related information. Before that, P-Nodes need to be sure that a certain number of P-Nodes already joined the PDN, for avoiding excessive key redistribution in the DHT (that happens when new peers join a DHT network). To do that, P-Nodes obtain two lists that are published in the PDN: the PDN member list, which contains the actual P-Nodes that already joined the PDN; and the PDN "to be" member list, which contains the P-Nodes that were expected to be members the PDN (that was published in step 2). The decision of when to republish the information is based on the percentage of P-Nodes that already joined the network. If that percentage is reached, the information is republished. If not, the P-Node waits a random time and repeats the same process. For dealing with the situation where some P-Nodes take a long time to join the new PDN (or maybe never do it) there is a timeout mechanism. After a P-Node joins the PDN, it updates the PDN member list.

Step 5: Composition is finished when all P-Nodes joined the new PDN, or a timeout was exceeded. When the composition negotiation is triggered by a user service request, the source P-Node needs to know when the composition is ready to be used.

We specified and implemented a simple composition protocol for the X-PBMAN prototype, extending this basic algorithm. Our experience with its usability and performance will be reported in another paper.

3.4 Policy Information Model

The overall goal of an information model for PBMAN is allowing efficient and flexible policy

[3] We observed this long delay in our implementation.

processing and service usage decisions to be made. PBMAN information model comprises three main types of management entities: policies, targets and associations. One important step in a policy management system is the association between policies and targets. For PBMAN, targets are: user, group, network (AN), service, service bundle and access class. The latter is a technology-independent way of classifying access technologies by common features.

The PBMAN information model is presented to illustrate how the P4MI data model can be used. We are not advocating this model as a general solution to ambient networks, since various other models could be used. Its importance is that it has been successfully used in our prototype implementation.

3.5 Data Management Layer

As an instantiation of a general framework, the information model of PBMAN needs to be mapped to the data model of P4MI. This is done by using the Data Management Layer in such a way to allow policies, targets and associations to be efficiently stored and retrieved from the repositories. The need for seven reserved words for the record type Key3 field has been identified: info, ${entity}-list, index, translation, policy, target and map-${entity}. An example of using the data management layer is::

- {AN Id, Policy Id, info): Definition of a single policy; policy Id is a unique identifier and info contains the policy body, written in a policy language.
- {AN Id, Target Id, policy}: List of all policies associated to a given target; after retrieving the list, policies are retrieved one by one using the {AN Id, Policy Id, info) record.
- {AN Id, Policy Id, target}: a list of all targets associated to a given policy.

These records are primarily needed for retrieving policies during the policy processing process in response to a service request. As DHT only allows exact lookups, lists of entities and recursive lookups are needed. In our prototype, we implemented parallel lookups to improve data management efficiency. In this example, upon retrieving a policy list, the P-Node can retrieve all selected policies concurrently.

4 Conclusions

This paper described the P4MI framework, a flexible and new scalable Policy-Based Management framework based on p2p technology as the main enabling underlying mechanism. A hierarchical p2p architecture, using a DHT is the core model of P4MI. It is expected to be suitable for scalable distributed management needed by the future ubiquitous services provided for wireless mobile users. We also instantiated this abstract framework to Ambient Networks, that are expected to present high levels self-management features, required for providing dynamic and instant user access to services and resources.

We have implemented a prototype of PBMAN, using an extended DHT-storage middleware called X-Peer, based on Pastry substrate. We expect it will provide us enough feedback for being able to understand further the core features and limitations of P4MI.

As future work we will continue the development of the PBMAN prototype in order to be able to assess its real applicability for a scenario of instant service access and dynamic network composition. We also will work on other different instantiations of P4MI.

5 Acknowledgements

This work was supported by the Research and Development Centre, Ericsson Telecomunicações S.A., Brazil.

6 References

[1] Law, K. L. E. & Saxena, K., “Scalable Design of a policy-Based Management System and its Performance”, IEEE Communications Magazine, 2003.
[2] Chaouchi1, H. &, Pujolle, G. “Policy based management framework for Always Best Connected users”, 1st Intl. ANWIRE Workshop, April 2003.
[3] Yavatkar, R., Pendarakis, D. & Guerin, R., “A Framework for Policy Based Admission Control,” RFC 2753, January 2000.
[4] Balakrishnan, H., et al., ‘Looking Up Data in P2P Systems”, Communications of the ACM, February 2003.
[5] Kappler, C., “Connecting Ambient Networks - Requirements and Concepts”, Deliverable D3.1, Ambient Networks Project, August 2004,
[6] Niebert N. et al., “Ambient Networks: An Architecture for Communication Networks Beyond 3G”, IEEE Wireless Communications, April 2004
[7] Westerinen, A., “Terminology for Policy-Based Management”, RFC 3198, November 2001.
[8] Rocha Jr., J., Fidalgo, J., Dantas, R., Oliveira, L., Kamienski, C. & Sadok, D., “X-Peer: A Middleware for Peer-to-Peer Applications”, 1st Brazilian Workshop on Peer-to-Peer (WP2P), May 2005, (in Portuguese).
[9] FreePastry, Rice University, http://freepastry.rice.edu, last visited in 09/09/2005.
[10] Rowstron, A. & Druschel, P., "Pastry: Scalable, distributed object location and routing for large-scale peer-to-peer systems", 18th IFIP/ACM Intl. Conference on Distributed Systems Platforms, October 2001.
[11] Rhea, S. et al., “OpenDHT: A Public DHT Service and Its Uses”, ACM SIGCOMM 2005, September 2005.
[12] Garc´es-Erice, L., “Hierarchical Peer-to-peer Systems”, Parallel Processing Letters (PPL), December 2003.