

Policy-Based Quality of Service Management in Wireless Ad Hoc Networks

Kaustubh S. Phanse

Dissertation submitted to the Faculty of the
Virginia Polytechnic Institute and State University
in partial fulfillment of the requirements for the degree of

Doctor of Philosophy
in
Electrical Engineering

Committee:

Dr. Luiz A. DaSilva, Committee Chair

Dr. Annamalai Annamalai

Dr. Ing-Ray Chen

Dr. Yao Liang

Dr. Scott F. Midkiff

August 22, 2003

Alexandria, Virginia

Keywords: Policy-Based Network Management, Framework, Quality of Service, Ad
Hoc Network, Solution Suite, Testbed, Simulation

© Copyright 2003, Kaustubh S. Phanse

Abstract

Policy-Based Quality of Service Management in Wireless Ad Hoc Networks

Kaustubh S. Phanse

Managing mobile ad hoc networks (MANETs) presents new challenges due to the need for a distributed management mechanism that can efficiently adapt to the dynamic nature of these networks. In particular, provisioning and management of Quality of Service (QoS) in such networks remains a challenging task. Previous works in this field have focused largely on the monitoring and data collection aspects of network management; literature on the provisioning of devices and protocol support for MANET configuration is scarce.

One approach for QoS provisioning and management in the Internet that has met with considerable interest in the networking community is that of Policy-Based Network Management (PBNM). However, its application has been so far limited mainly to fixed high-bandwidth networks. In this research, we apply the PBNM concept, for the first time, for managing QoS in ad hoc networks. We formulate a framework to understand the various crucial components that should comprise an ad hoc network management system. We propose a taxonomy of policy architectures to classify the various feasible architectures into distinct categories. Based on our assessment using the taxonomy, we identify architectures that seem promising for managing ad hoc networks. We propose a solution suite to address the different challenges in deploying policy-based management in MANETs. These solutions include *k-hop clustering*, *Dynamic Service Redundancy (DynaSeR)*, *inter-domain policy negotiation*, and *automated service discovery*. We propose extensions to the standard Common Open Policy Service (COPS) protocol and suggest methods for cross-layer interaction to implement our solutions.

Our methodology focuses on both a prototype implementation and experimental analysis using wired and wireless testbed networks, and modeling and performance evaluation using simulation. The whole exercise of conducting experiments provided valuable insight into the challenges of operating in an actual ad hoc network environment; implementation and testing facilitated assessment of the feasibility of our proposed schemes. Simulation allowed us to evaluate our

solutions for different cluster sizes, network densities, and node mobility. The scalability of our solutions was tested with networks of up to 100 nodes.

In general, average service availability for the PBNM system improved as the cluster size increased, with decreased COPS connection overhead (the tradeoff is increased unpredictability, longer response time, and resource requirements at intermediate nodes to support larger clusters). We were also able to determine that, for a given cluster size, our proposed delegation scheme resulted in a 10 to 25% improvement in service availability. Using our proposed time-based heuristic, savings on the order of 50 to 400% were obtained in the service discovery overhead for larger cluster sizes. We also validated some of the simulation results against proof-of-concept experiments conducted using the testbed. We presented a working illustration of our PBNM system prototype by demonstrating its application for managing QoS for multimedia and real-time mission critical applications in a multi-domain ad hoc network.

The policy-based approach is a promising one for the management of MANETs, but it requires the flexibility to adapt to a constantly changing environment. Through experimental studies and simulation, we were able to determine that using our proposed solution suite and through the addition of a set of extensions to the COPS protocol, we can achieve our objective of a self-organizing, robust, and efficient PBNM system for managing MANETs.

To my parents, Aai and Dada
my brother and guardian, Kedar
and my friend, guide and philosopher, Joshi Kaka

Acknowledgements

This dissertation would not have been possible without the help of many people. I would like to take this opportunity to express my deep appreciation to all those who helped me in this arduous but extremely rewarding process.

First and foremost, I would like to thank two people – Professors Luiz DaSilva and Scott Midkiff who have influenced me the most as a researcher, and I hope I am able to live up to their high standards. I am grateful to my advisor, Dr. Luiz DaSilva for accepting me as his student four years ago. He was a very patient and supportive advisor, an excellent teacher and most importantly a great friend. While his expert guidance provided the spring to clear most technical hurdles, his enthusiasm and sense of humor played an equally important role in making this journey less daunting. I have come a long way as far as technical writing is concerned, and I owe it all to Dr. DaSilva and his relentless drive to improve his students' writing skills. I consider it to be my good fortune to have got a chance to work closely with Dr. Midkiff. He went beyond the limits of his duties to provide guidance, support, and encouragement at every stage of this research endeavor. His guidance, both in terms of technical advice on my research and in terms of professional advice, was invaluable. He always raised the bar inspiring me to strive for the best, and was always among the first ones to applaud any accomplishment.

I am grateful to Dr. Annamalai Annamalai, Dr. Ing-Ray Chen and Dr. Yao Liang for serving on my Ph.D. committee, and for their time and co-operation in reviewing this work. Their suggestions and comments helped immensely in improving the quality of this research and this thesis. Also, I would like to thank Dr. Chen for providing a jump-start to the analytical modeling aspects related to this research.

Research funding plays an extremely important role in a graduate (especially Ph.D.) student's life, and I was no different. I am indebted to the Office of Naval Research (ONR) for funding this research as a part of the Navy Collaborative Integrated Information Technology Initiative (NAVCIITI) project. There is no doubt that this research could not have been completed in such a timely fashion without their support.

I had a great time working with some very bright and nice people in the NAVCIITI Task 3.1 research group: Dr. Nathaniel Davis, Dr. Jahng Park, Palani Annamalai, Michael Christman,

George Hadjichristofi, Tao Lin and John Wells, and learned a lot from my extended association with them. In particular, I must express my special gratitude to Tao whose contribution to this research was invaluable.

I would also like to thank Karthik Channakeshava and Dr. Binoy Ravindran with whom I collaborated as a part of the NAVCIITI project. My many discussions with them helped me gain a broader perspective of the problem in hand, and in effect helped improve the content of this dissertation.

Thanks to Aniket Bhat for offering useful suggestions and criticism on my work. His efforts in improving the quality of the simulation code and with preliminary analytical modeling were useful.

Working at Virginia Tech's Alexandria Research Institute (ARI) for the majority of my graduate life was a good learning experience – different in many ways as compared to the main campus life in Blacksburg. I would like to thank everyone at the ARI for making my stay a pleasant one; in particular, I am thankful to Dr. Saifur Rahman, George Hagermann, and Latricia Nell. Thanks to Naresh Verma for the interesting brainstorming sessions and for patiently listening to many of my ideas and questions, and otherwise making us laugh with his humor. Thanks to Leila Ribeiro for some useful guidance and for her inspiring presence. Many thanks to Vivek Srivastava, Vara Prashanth Pushpagiri, and Yonael Teklu for their friendship; they played a major role in keeping me sane in this difficult journey that otherwise could have got lonesome in an extended campus environment.

There are many “behind-the-scene” people at Virginia Tech without whose support life can become very difficult for a graduate student. I would like to offer my appreciation to two such people -- Ms. Monica Gibson from the Graduate School and Ms. Cindy Hopkins from the Bradley Department of Electrical and Computer Engineering for their welcoming attitude and for always being there to take care of the administrative problems.

I would like to thank the technical support crew at Scalable Technologies Inc. for promptly answering my queries about QualNet; this immensely helped in learning the minutiae of the simulation tool in a short period of time. Also, thanks to the many participants on the mailing lists hosted by the Linux Advanced Routing and Traffic Control (LARTC) team, the IETF Resource

Allocation Protocol (RAP) working group, and the IETF Mobile Ad Hoc Network (MANET) working group, who helped me gain a better understanding of the various aspects of this research.

I owe a special debt of gratitude to Mr. Shrikrishna Joglekar (Professor, Pune University, India) for inspiring me to pursue graduate studies in the U.S., and helping me in the initial stages of making this decision.

I am grateful to the Almighty for blessing me with such wonderful family and friends; without their love, support, and encouragement, more than anything else, I would have never reached this stage in my life. The expression of my gratitude for them is beyond words!

Table of Contents

1.	Introduction	1
1.1	Motivation	1
1.2	Policy-Based Network Management (PBNM).	2
1.3	Preview: Research Challenges and Methodology	3
	1.3.1 Research Challenges.....	3
	1.3.2 Implementation.....	6
	1.3.2.1 Prototype Implementation.....	6
	1.3.2.2 Simulation.....	8
1.4	Contributions.....	8
1.5	Structure of Document.....	9
2.	Background and Related Work.....	11
2.1	Wireless Ad Hoc Networks.....	11
2.2	Related Work.	15
	2.2.1 Ad Hoc Network Management.....	16
	2.2.2 Quality of Service (QoS) in Ad Hoc Networks.....	19
2.3	Policy-Based Networking.....	20
	2.3.1 Overview.....	20
	2.3.2 Policy-Based QoS.....	23
2.4	Summary.....	25
3.	Policy-Based Management Framework for Wireless Ad Hoc Networks.....	26
3.1	Requirements for Management of Wireless Ad Hoc Networks.....	27
	3.1.1 Desirable Features.....	27
	3.1.2 Adapting to the Decentralized Paradigm.....	28
3.2	Systems Approach.....	29
	3.2.1 Policy Specification.....	30
	3.2.2 Policy Architecture and Distribution.....	30

3.2.3	Resource Discovery.....	35
3.2.4	Policy Provisioning.....	36
3.2.5	Policy-Based Routing.....	36
3.2.6	Policy Monitoring.....	37
3.2.7	Adaptation Logic.....	37
3.3	Summary.....	38
4.	Research Methodology.....	39
4.1	Testbed Network.....	39
4.1.1	Wired Testbed: Mobility Emulation using the Dynamic Switch.....	40
4.1.2	Wireless Testbed.....	40
4.2	Software Components.....	42
4.2.1	Intel® COPS Client SDK.....	42
4.2.2	Telia Research COPS API.....	42
4.2.3	OLSR Routing Daemon.....	42
4.2.4	OSPF-MCDS Routing Daemon.....	43
4.2.5	DiffServ on Linux Tool.....	43
4.2.6	Real-Time Application and Middleware.....	43
4.2.7	Video Conferencing Tool (VIC).....	45
4.2.8	BonnMotion and Supplementary Programs.....	45
4.2.9	Analysis and Other Utility Tools.....	46
4.3	Simulation Environment.....	47
4.4	Measurements and Evaluation.....	50
4.4.1	Performance Metrics.....	50
4.4.2	Factors.....	51
4.5	Summary.....	52
5.	Taxonomy and Experimental Evaluation of Policy Architectures	53
5.1	Introduction.....	53
5.2	Taxonomy of Policy Architectures.....	54
5.2.1	Characteristics of a Policy Architecture.....	54
5.2.2	Types of Policy Architectures.....	56
5.2.3	Summary.....	61

5.3	Preliminary Experimental Evaluation.....	61
	5.3.1 Comparison of Policy Architectures.....	62
	5.3.2 Multi-hop Ad Hoc Network (Wired Testbed).....	66
	5.3.3 Multi-hop Ad Hoc Network (Wireless Testbed): Proof of Concept.....	68
5.4	Summary.....	71
6.	Solution Suite: Protocol Support and Implementation.....	73
6.1	Cluster Management.....	73
	6.1.1 Taking Advantage of Proactive MANET Routing.....	75
	6.1.2 General Approach.....	76
6.2	Dynamic Service Redundancy.....	76
	6.2.1 Redirection.....	77
	6.2.2 Delegation.....	77
6.3	Policy Negotiation.....	79
6.4	Service Discovery.....	80
	6.4.1 Proposed Mechanism.....	80
	6.4.2 Time-Based Heuristic to Minimize Broadcast Overhead.....	82
6.5	Implementation.....	83
	6.5.1 Prototype Implementation.....	83
	6.5.1.1 Policy-Based Management Application.....	83
	6.5.1.2 Integration with Ad Hoc Routing Protocols.....	85
	6.5.1.3 Quality of Service Mechanisms.....	87
	6.5.2 Simulation Models.....	88
	6.5.2.1 COPS and COPS-PR.....	88
	6.5.2.2 Solution Suite.....	89
	6.5.2.3 Model Details.....	89
6.6	Summary.....	92
7.	Performance Evaluation.....	94
7.1	Simulation Environment.....	95
7.2	Simulation Results.....	96
	7.2.1 Effect of Cluster Size and Mobility.....	96
	7.2.2 Effect of Network Density.....	99

7.2.3	Service Discovery Overhead Minimization.....	100
7.2.4	Delegation.....	101
7.3	Experiments.....	102
7.3.1	Testbed Setup.....	102
7.3.2	Results.....	102
7.4	QoS Management in Multi-domain Ad Hoc Networks.....	104
7.4.1	Illustration 1: Video-streaming.....	105
7.4.2	Illustration 2: Real-Time Application.....	107
7.5	Summary.....	110
8.	Conclusion and Future Work	111
8.1	Summary.....	111
8.2	Directions for Future Work.....	114
	Glossary.....	118
	Bibliography.....	121
	Vita.....	134

List of Figures

Figure 1.1	Four phases of our research methodology.....	4
Figure 2.1	Wireless ad hoc network	12
Figure 2.2	Sample scenario showing deployment of wireless ad hoc networks in a military environment	13
Figure 2.3	Heterogeneous wireless devices (source: COMPAQ).....	14
Figure 2.4	Key architectural elements of a policy-based management system.....	22
Figure 2.5	Policy information base tree structure.....	23
Figure 2.6	Policy distribution models	24
Figure 3.1	Depiction of our proposed policy-based framework for wireless ad hoc networks, and its main components.....	29
Figure 4.1	Laptop with wireless interface PCMCIA type card.....	41
Figure 4.2	Antenna portion of the wireless PCMCIA card wrapped with aluminum foil “attenuator”.....	41
Figure 4.3	Sample soft timing constraints described using utility functions.....	44
Figure 4.4	GUI of the performance monitoring tool.....	45
Figure 4.5	BonnMotion and supplementary programs to generate mobility files for the Dynamic Switch.....	46
Figure 4.6	QualNet Animator GUI.....	48
Figure 4.7	Network protocols modeled as a Finite State Machine in the QualNet Designer.....	49
Figure 4.8	Illustration of policy response time and inter-decision time	51
Figure 5.1	Testbed setup for single hop experiments.....	62
Figure 5.2	Policy signaling overhead for outsourced (CCO) architecture.....	63
Figure 5.3	Policy response time and inter-decision time (in seconds) plotted as a function of bandwidth (Load = 25 policy requests/PEP; for the hybrid architecture about 60% of these requests are processed locally).....	64
Figure 5.4	Policy response time and inter-decision time plotted as a function of the number of hops between the policy client and server (wired testbed).....	67
Figure 5.5	Layout of our work area; placement of nodes for a 4-hop wireless ad hoc network topology is shown. Nodes A and E are the policy server and client respectively.....	69
Figure 5.6	Policy response time plotted as a function of the number of hops between the policy client and server (wireless testbed); the dotted lines indicate the confidence interval.....	70
Figure 5.7	Inter-decision time plotted as a function of the number of hops between the policy client and server (wireless testbed).....	71
Figure 6.1	k -hop cluster management; 1-hop ($k = 1$) clusters indicated by dotted lines.....	74
Figure 6.2	Cross-layer interaction for k -hop cluster management.....	75
Figure 6.3	Delegation of policy-based service.....	78
Figure 6.4	Signaling for inter-domain policy negotiation.....	80
Figure 6.5	Service discovery message format.....	81
Figure 6.6	Policy server user-interface.....	84
Figure 6.7	Policy client user interface.....	85
Figure 6.8	User-interface of a policy server showing topology information gathered from underlying OLSR routing daemon, and implementation of 1-hop cluster	86

	management.....	
Figure 6.9	User-interface of the policy client; as the client moves out of the 1-hop cluster of its original server, the client is redirected to another policy server.....	86
Figure 6.10	Redirection of a policy client (208.17.194.132) by one policy server (208.17.194.134) to another policy server (208.17.194.131), as a part of k -hop cluster management.....	87
Figure 6.11	Sample shell script executing commands for the <i>DiffServ on Linux</i> tool.....	88
Figure 6.12	Snapshot of the dataPtr structure (of type <i>struct pbnm_adhoc_str</i>) that maintains all the information for each node in a simulation.....	90
Figure 6.13	Snapshot of the format used to define the (a) PBNM application as compared to (b) existing QualNet applications such as FTP.....	92
Figure 7.1	Average number of COPS connections established per server as a function of mobility and cluster size.....	97
Figure 7.2	Percentage average service availability as a function of cluster size and mobility.....	97
Figure 7.3	Percentage service availability at different speeds for cluster size $k = 1$	98
Figure 7.4	COPS connection timeouts as a function of cluster size and mobility.....	99
Figure 7.5	Percentage average service availability as a function of cluster size and network density.....	99
Figure 7.6	Average service discovery broadcast overhead as a function of cluster size, in presence and absence of our proposed time-based heuristic.....	100
Figure 7.7	Improvement in service availability using our proposed delegation scheme.....	101
Figure 7.8	Comparison of average service availability obtained from the experiments and simulations.....	103
Figure 7.9	Comparison of average number of COPS connections obtained from the experiments and simulations.....	103
Figure 7.10	Demonstration scenario depicting a multi-domain wireless ad hoc network; hosts B and C are policy servers in distinct administrative domains.....	105
Figure 7.11	Layout of the area where we conducted the demonstration; nodes A, B and C are static, while the movement of node D is shown with dotted line. Node D is the policy client that “hands-off” from policy server B to policy server C.....	106
Figure 7.12	(a) Degraded video quality without policy negotiation (allocated bandwidth is 12 kb/s); (b) Acceptable video quality (bandwidth in the range 64 kb/s to 128 kb/s allocated) after policy negotiation.....	107
Figure 7.13	Without policy negotiation, the real-time mission critical applications are treated as best-effort along with background traffic as the source node moves into foreign domains.....	109
Figure 7.14	Almost seamless QoS is achieved for real-time mission critical applications in presence of policy negotiation even as the source node moves across different network domains.....	109
Figure 8.1	Representation of the client behavior as a closed network of queues to model service availability.....	116

List of Tables

Table 4.1	Machines Used in the Testbed Network.....	40
Table 5.1	Policy Architecture Taxonomy Matrix.....	57
Table 5.2	Policy Response Time and Inter-decision Time (for different bandwidths).....	65
Table 5.3	Policy Response Time and Inter-decision Time vs. Number of Hops (wired testbed).....	68

*Begin – to begin is half the work, let half still remain;
again begin this and thou wilt have finished.
- Marcus Aurelius*

*The beginning is the most important part of the work.
– Plato*

Chapter 1

Introduction

1.1 Motivation

Wireless ad hoc networks [Ily02, Per00, Toh01] are autonomous networks operating either in isolation or as “stub networks” connecting to a fixed infrastructure. Such networks have the potential to provide wireless and mobile computing capability in situations where efficient, economical and rapid deployment of communication is required, and where the use of a wired or an infrastructure-based wireless network is either too expensive or impractical. Ad hoc networks have found a growing number of applications: wearable computing, disaster management/relief and other emergency operations, rapidly deployable military battle-site networks, and sensor fields, to name a few. Some of the features that characterize such networks are dynamic topologies (host and/or network mobility), bandwidth-constrained variable capacity links, limited physical security and survivability, and nodes with limited battery life, processing power and storage capacity. These characteristics pose significant challenges to the management of mobile wireless ad hoc networks.

In particular, providing and managing Quality of Service (QoS) in an ad hoc network environment remains a challenge [CM01, CS99, PD03]. In many applications, ad hoc networks must support transmission of real-time traffic such as voice and video, still-imagery, and mission critical data traffic. Hence, although providing quantitative or absolute guarantees seems intractable, at least some means to differentiate among traffic flows is essential. Further, given

the hostile environments in which ad hoc networks are deployed and the limited resources they can count upon, service differentiation and dynamic bandwidth management can help provide graceful degradation of performance in case of an attack or network failure, and thus enhance the overall survivability of the network [DP02]. Various aspects of providing QoS in ad hoc networks, including QoS routing [SSB99], QoS medium access control [MG98], resource reservation [MST00], and QoS architectures [LAZ+00, XSL+00] have been widely studied. However, the management and control architecture required to support QoS provisioning and management in ad hoc networks is not yet well understood.

In general, traffic differentiation or the ability to reserve network capacity for certain flows opens up the possibility of unauthorized usage of available resources, e.g., providing malicious user(s) with another tool to initiate a denial-of-service attack. This is even more relevant for ad hoc networks typically deployed in hostile environments with limited security and survivability. Such a free-for-all QoS implementation could lead to a *tragedy of the commons* and possibly result in even worse than best-effort performance. In addition, it may be necessary to support admission control or bandwidth allocation based on one or more factors other than just the availability of bandwidth. These factors may include the owner of the traffic (identity of the user, application or organization the traffic is originating from), temporal elements (time of day), etc. Thus, a control structure that allows automated QoS management – authorized usage of network resources, dynamic bandwidth allocation, and admission control based on different policies – is required.

Network monitoring and data collection [CJS99, SSJ03] have been the focus of most published work on ad hoc network management. There is still a clear need for a management system that allows automated, efficient, and robust network configuration and provisioning.

1.2 Policy-Based Network Management (PBNM)

One approach for QoS provisioning and management in the Internet that has met with considerable interest in the networking community is Policy-Based Network Management (PBNM) [Kos01, Pol, Ver00]. Unlike legacy network management, which generally involves configuring and managing each network entity individually, PBNM configures and controls the network as a whole, providing the network operator with a simplified, logically centralized and automated control over the entire network. PBNM has made administration of complex

operational characteristics (also known as policy disciplines) of a network, such as Quality of Service (QoS), access control, network security, dynamic IP address allocation, etc., much easier.

As a result, it is not surprising that the concept of policy-based networking has become so popular in the last few years. The growing interest is evidenced by several research and development efforts in both academia and industry [LT00, RVS+99, VBJ01, VCA02], publication of books [Kos01, Ver00], working groups leading standardization efforts [Dis, Pol, Rap], new technical conferences, and new commercial products [Cis, Int] supporting PBNM.

However, the work on policy-based management so far has mainly focused on large, high bandwidth, fixed networks [Kos01, Ver00, VBJ01, VCA02], e.g., enterprise networks, content provider networks, Internet service provider (ISP) networks, etc. To our knowledge, policy-based networking has not been extensively studied in the context of wireless mobile ad hoc networks. Our intention is to apply the policy-based approach for provisioning ad hoc networks, and conduct a comprehensive study of the performance of a PBNM system in the context of a mobile ad hoc network environment.

1.3 Preview: Research Challenges and Methodology

Here we provide a preview of the research challenges we address in this dissertation and the methodology adopted.

1.3.1 Research Challenges

The challenges in provisioning ad hoc networks arise primarily from the salient features that characterize such networks and the environments in which they are typically deployed. This calls for first laying down the key requirements sought in an ad hoc network provisioning system. Secondly, it is necessary to understand the basic properties of policy-based network management and how they relate to ad hoc networks. This environment characterization formed the first step of our methodology as shown in Figure 1.1.

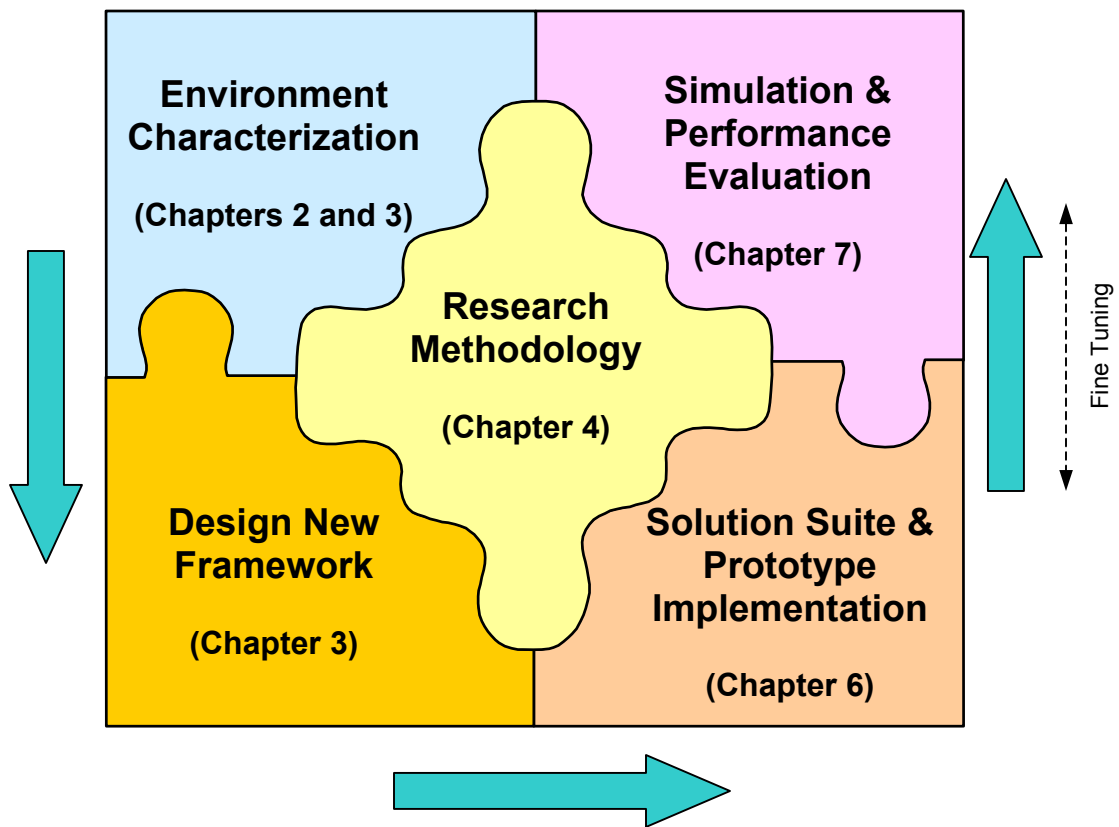


Figure 1.1 Four phases of our research methodology.

Deeper investigation of these factors led to the formulation of our problem statement. In this research, we address the following research challenges.

(1) Fundamental Challenge: Centralized versus Decentralized Paradigms

Policy-based management offers several features that make it a good candidate for QoS provisioning and management. However, in order to extend the policy-based approach to ad hoc networks, we have to address the fundamental challenge of adapting the conceptually centralized idea of policy-based management to a decentralized paradigm applicable to ad hoc networks.

We needed to build a management framework that is as automated as possible, requiring minimal human intervention, and is intelligent, meaning it is able to learn about changes in networking conditions. This would lead us to a self-organizing or adaptive control structure that automatically reacts to ad hoc network dynamics.

(2) Efficiency

Given the resource-constrained nature of ad hoc networks, it is very important that the proposed management system be efficient and lightweight. Minimal resources in terms of bandwidth, battery life, and computational means should be used up in network management.

(3) Robustness

The management system should allow secure exchange of management data among authorized hosts, and enhance the overall survivability of the network. This may require the means to authenticate and authorize users/hosts, and support encryption capabilities, and to allow reliable transmission of management data with means to detect failure and provide fault-tolerance.

(4) Interoperability in a Multi-domain Ad Hoc Network

In many applications, an ad hoc network is formed by groups of networks and nodes belonging to different organizations. For example, Kidston and Robinson [KR00] describe a Coalition Wide Area Network (C-WAN) – a mobile ad hoc internetwork formed by mobile networks (e.g., ships) and nodes belonging to the U.S. Navy and its allies. Network interoperability in such internetworks still remains a challenging task. Hence, we decided to tackle the problem of QoS provisioning in such multi-domain ad hoc networks – to support seamless QoS for mobile nodes as they move across the different network domains administered by individual organizations.

To address these research problems, we propose and implement an *automated, intelligent, efficient, and robust* policy-based management framework for wireless ad hoc networks, and to demonstrate its application for QoS management. The Merriam-Webster dictionary [Web98] defines a *framework* as “a basic conceptional structure (as of ideas)” or “a skeletal, openwork, or structural frame.” Our proposed *framework* serves the exact same purpose: a conceptual representation of an actual PBNM system implementation and the varied, complex functions that it needs to carry out.

In Chapter 3, we present our PBNM framework. We use a systems approach and identify seven critical components that together constitute the framework. Using a systems approach facilitates better understanding of the functional tasks of each individual component as well as the interaction and dependency among the different components.

We propose a policy architecture taxonomy (Chapter 5) and study the various feasible architectures that can be used to deploy PBNM. Based on our qualitative assessment and experimental evaluation, we suggest the use of hybrid type of architectures as suitable candidates for ad hoc network management.

To address the fundamental challenge (centralized vs. decentralized paradigms) mentioned earlier, and to implement a self-organizing and adaptive PBNM system, there is no single panacea or silver bullet. We recognize this and devise a suite of solutions and techniques that work together to address this challenge. Three modules in our solution suite are: *Cluster Management*, *Service Discovery Mechanism*, and *Dynamic Service Redundancy*. The primary goal of our service discovery mechanism is to allow client nodes to automatically discover policy server(s) in the network. *Cluster management* is used primarily from a management perspective. We propose a *k-hop clustering algorithm* to control the number of hops between a policy server and its clients; this is crucial for efficient and predictable performance, as will be seen from our results in Chapter 5 and 7. Finally, the *Dynamic Service Redundancy (DynaSeR)* solution is proposed to further enhance cluster management and improve the service coverage of our management system. To tackle the problem of providing seamless QoS in multi-domain ad hoc networks, we propose a *Policy Negotiation* scheme that facilitates inter-domain policy exchange.

In devising these solutions, we keep in mind the need for efficiency and robustness. This also dictates our design choices – the most important being the protocol used for policy distribution. We use the IETF standardized Common Open Policy Service (COPS) protocol [DBC+00] and its extension, COPS for PRovisioning (COPS-PR) [CSD+01]. The various features that make COPS-PR a good choice are described in Chapter 3.

1.3.2 Implementation

1.3.2.1 Prototype Implementation

Most research that evaluates and compares different solutions and protocols to be used in mobile ad hoc networks relies on simulation. Using simulation as the primary analysis tool is common since a network simulator allows rapid prototyping of new ideas and solutions. Most importantly, it also provides a controlled environment to analyze and compare different alternatives quickly.

However, we adopt a slightly unconventional approach. We focus on the implementation of our management framework in a real network environment. The implementation and experimental evaluation using a testbed network forms a major portion of this research. Three main factors that guide our approach are as follows.

- **Prior Experiences**

Very little has been done in terms of actual implementation and testing of algorithms and solutions in real-life ad hoc networks. We believe that the challenges of operating in an actual ad hoc network environment are not always grasped in a simulation environment. Also, any assumptions made while running simulations, may not always guarantee robustness of the simulation results. Similarly, theoretical estimation may not always be accurate.

Our concerns are confirmed by prior experiences of researchers [CHC+01, GGK01, MBJ99, RH00] who have ventured in actual implementation of and experimentation with wireless ad hoc networks. In [GGK01], Gupta *et al.*, study the scaling laws in ad hoc networks employing IEEE 802.11 technology. Their experimental results show that the throughput per node in a real ad hoc environment decreases much more rapidly than that shown to be attainable in theory. In [CHC+01], Clausen *et al.*, attribute the discovery of some of the shortcomings in their implementation to the practical experience attained through experimental analysis; this in turn helped them optimize their simulation-based evaluation of larger networks. [MBJ99, RH00] provide insight into the various practical aspects and importance of building and studying actual ad hoc networks.

- **Feasibility**

Given the key requirements of an ad hoc network management system, and the complexities involved in making different modules in the management framework work together, practicability of deploying our framework in a real network was essential. Also, the lack of real network data available from existing policy-based management systems led us to implement and test our proposed solutions on a testbed network. Finally, we also hope that our experiences and lessons learnt from this hands-on work will contribute to the research community at large.

- **NAVCITI Research Project**

This research is partially funded by the Office of Naval Research (ONR) as a part of the Navy Collaborative Integrated Information Technology Initiative (NAVCITI). One of the goals of the NAVCITI project was to setup an interoperability testbed network to allow implementation and experimentation of diverse technologies such as Quality of Service (QoS), network management, network security, mobile routing, etc., in a mobile wireless network environment. Involvement in

this project also greatly influenced our focus on the implementation and evaluation of our proposed framework in a real network.

1.3.2.2 Simulation

One limitation of our experimental study was the maximum number of nodes that were available. To overcome this limitation – to address scalability and get a better perspective of the management system performance in larger networks – we also conducted simulation-based study of the PBNM system. This exercise proved to be invaluable since the effect of varying some of the system parameters (e.g., cluster size) on network behavior under wider variety of conditions was visible only in simulations. We implemented and simulated our proposed schemes using the QualNet network simulator [Qua].

1.4 Contributions

In this dissertation, we analyze the problems posed by the challenges outlined in the previous section and design, implement and evaluate solutions to them that play a major role in making deployment of policy-based management in ad hoc networks possible, as well as in significantly improving the PBNM system performance.

The major contributions and benefits from this research are as follows.

- Formulation of a policy-based management framework for wireless ad hoc networks that comprehensively considers the main components that should comprise a management system.
- Taxonomy of policy architectures that provides a common platform to qualitatively compare the various feasible policy architectures and also the selection of one or more architectures that seem most promising for the network scenario of interest.
- A solution suite comprised of k-hop cluster management, Dynamic Service Redundancy (DynaSeR), service discovery, and policy negotiation techniques that facilitates deployment of a PBNM system for ad hoc network provisioning and management.

- A PBNM software package, which includes implementation of COPS and COPS-PR based policy server and client applications, proposed extensions to the COPS-PR protocol, and implementation of our solution suite.
- Integration of the PBNM software with two ad hoc routing protocols – the Optimized Link State Routing (OLSR) protocol [CJ03], and OSPF-MCDS [Lin03, LMP03] developed by researchers at Virginia Tech.
- Integration of the PBNM software with *Diffserv on Linux* QoS toolset [Dif] to facilitate dynamic bandwidth management.
- Dissemination of lessons learnt and insights gained in prototype implementation and working with an ad hoc network testbed.
- Simulation models we developed in QualNet to implement the COPS and COPS-PR protocols, and our solution suite.
- An extensive simulation-based and experimental evaluation of the PBNM system under different ad hoc networking conditions to gain insight into the system performance and provide pointers for future research and deployment of PBNM in ad hoc networks.
- It is noteworthy that the usefulness of this study and some of the proposed solutions is not limited to policy-based management; it can be applied in a broader context of general client-server systems in ad hoc networks.

1.5 Structure of Document

The remainder of this document is organized as follows. In Chapter 2, we discuss the background material and related work in the area of QoS provisioning and network management in ad hoc networks, and provide an overview on policy-based network management and QoS policy. In Chapter 3, we formulate and present our proposed policy-based management framework for wireless ad hoc networks. We describe in detail our testbed network and introduce the various software tools that constitute our research platform in Chapter 4. We also provide a short primer on QualNet, and describe our simulation methodology. In Chapter 5, we propose a taxonomy of policy architectures and discuss our findings based on experimental evaluation comparing the various architectures, and using COPS in a multi-hop ad hoc network. We describe our solution suite and the underlying protocol support in Chapter 6; we describe in detail the implementation of the various modules in our framework as a prototype as well as simulation models. We present our performance evaluation in Chapter 7, which includes simulation and experimental results, and

demonstrations illustrating the effectiveness of the PBNM system in managing multi-domain ad hoc networks. Finally, we conclude this document in Chapter 8, wherein we summarize the contributions of this thesis, and discuss the related areas of research and venues for future work.

*One's mind has a way of making itself up in the background,
and it suddenly becomes clear what one means to do.*

- A. C. Benson

History will be kind to me for I intend to write it.

- Sir Winston Churchill

Chapter 2

Background and Related Work

In this chapter, we survey the relevant work that serves as a background to our research. We begin by discussing wireless ad hoc networks and their salient features that make management of such networks particularly challenging. In Section 2.2, we survey the related work on network management and QoS provisioning in ad hoc networks. Section 2.3 provides a brief overview of the state of the art in the field of policy-based networking. Finally, we summarize the main points of the chapter in Section 2.4.

2.1 Wireless Ad Hoc Networks

A wireless ad hoc network is formed by a collection of two or more network nodes with wireless communication capability, as shown in Figure 2.1. It does not rely on a central entity or infrastructure (e.g., a base station or access point) for communication. Two nodes can communicate directly if they are within radio range of each other. If out of radio range, the nodes can communicate through one or more intermediate nodes. In such case, the intermediate nodes act as routers and relay packets from the source to the destination.

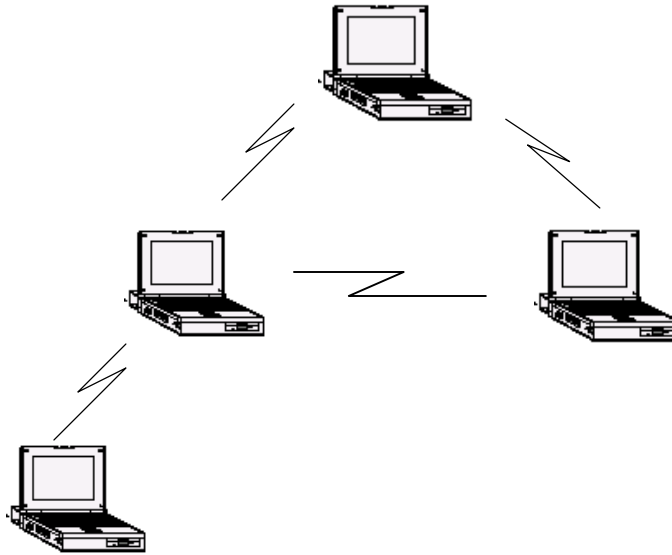


Figure 2.1 Wireless ad hoc network.

A wireless ad hoc network can operate either in isolation or as a “stub network” connecting to a fixed network, and it may be mobile – popularly known as a MANET (Mobile Ad hoc NETWORK) – or fixed.

The distinct features that characterize most wireless ad hoc networks and those that need to be addressed by any management framework to be deployed in such networks are as follows [CM99, CJS99, PD03].

(1) Low Bandwidth, Variable Capacity Links

Wireless links are typically more bandwidth-constrained than their wired counterparts. Fading, interference, jamming etc., may cause intermittent link failures or considerable variation in the channel error rate. In addition, the diverse nature of nodes (e.g., with different transmission power levels), and communication technologies (e.g., IEEE 802.11, direct line of sight UHF/VHF links, satellite links, etc.) being used may lead to links of varying capacity in multi-hop wireless networks (Figure 2.2).

(2) Dynamic Topology

The topology of an ad hoc network can change dynamically for various reasons. In a wireless network, two nodes are said to be “connected” when they are within communication range of each other. In mobile ad hoc networks, the topology changes as nodes move out of range of one

or more nodes with which they were connected, and move closer and connect to other nodes. In addition, even in fixed wireless ad hoc networks (e.g., wireless sensor fields), due to limited survivability of wireless links (subject to fading or jamming) or of the nodes themselves (e.g., damaged due to hostile conditions, or discharged battery), the logical topology of the network may change. Finally, one or more nodes may enter or leave an existing ad hoc network during its operation, leading to a change in topology.

Keeping current knowledge of the network topology is an important requirement in any network management system. In fixed wired networks, this is a relatively simple task since the changes in topology (mainly due to node or link failure, or addition/removal of a node) are infrequent. In a wireless ad hoc environment, it is crucial that the management system keep up with the frequent topology changes. However, the frequent exchange of topology information may lead to considerable signaling overhead, congesting low bandwidth wireless links, and possibly depleting the limited battery life of the nodes involved. Hence, the choice of mechanism used to collect or manage topology information is critical.

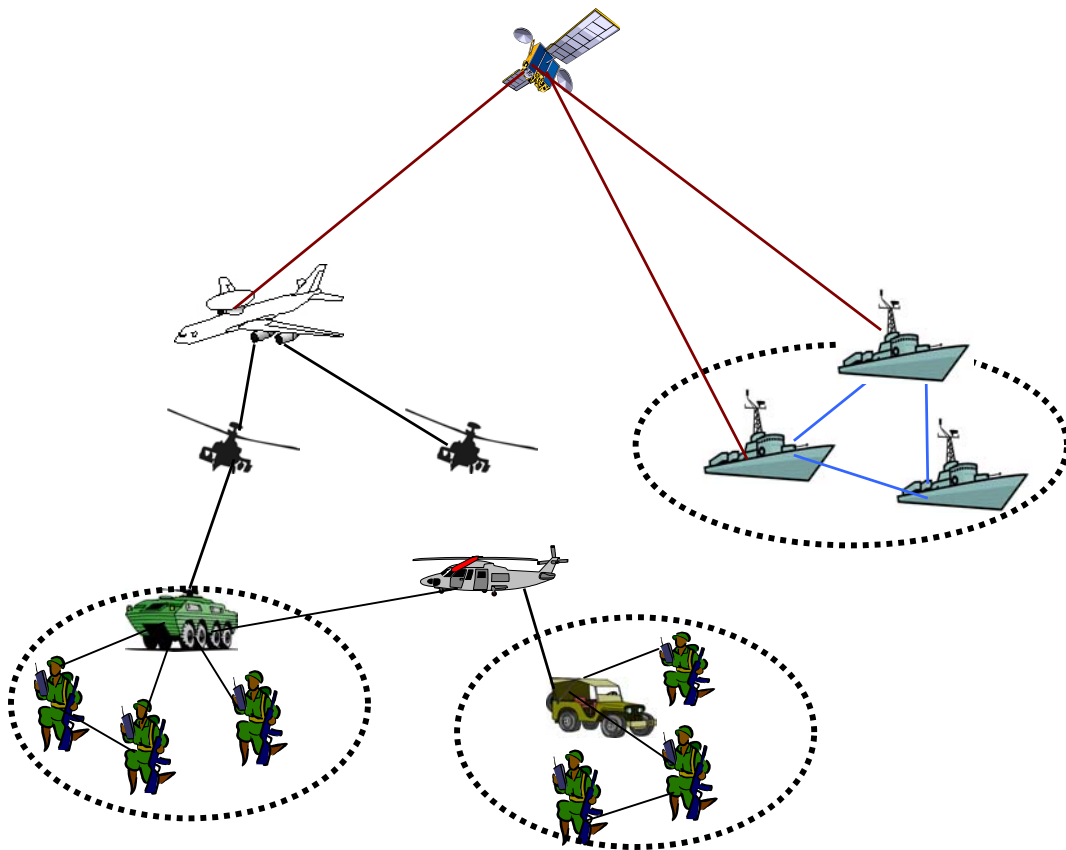


Figure 2.2 Sample scenario showing deployment of wireless ad hoc networks in a military environment.



Figure 2.3 Heterogeneous wireless devices. (source: COMPAQ)

(3) Limited Resources

Battery is the only source of power for nodes in many ad hoc network environments, and the need to keep these nodes compact, light and even wearable, imposes limitation on their storage and processing capabilities. This is again very different from conventional wired networks, wherein the network nodes seldom depend on batteries as their sole source of energy, and typically have significant storage and processing capacity.

(4) Network Nodes Play Multiple Roles (source/destination vs. routers)

In most wired networks, network nodes play distinct roles, e.g., end-hosts (source or destination for an application flow) versus routers (intermediate nodes that route traffic toward its destination). Also, nodes are typically dedicated to specific network operations and their characteristics well suited to the role they play. For example, machines are specifically designed to operate as servers, and dedicated high-end routers and switches are used to handle network traffic. On the other hand, in ad hoc networks, most nodes are expected to route packets for other nodes in the network, while they themselves may also be a source or destination for one or more application flows. A management framework must account for the multiple roles played by network nodes.

(5) Heterogeneity

Heterogeneity is inherent to most ad hoc networks due to the diverse nature of communication technologies (IEEE 802.11, line of sight UHF/VHF, etc.) that may be used and the different types

of nodes (Figures 2.2 and 2.3) – ranging from sensors, palmtops, laptops to mobile networks hosted on a ship, a tank or an airplane – that may form the network. The heterogeneity of nodes can be a criterion to assign roles (e.g., management server versus client) to the various nodes. An ad hoc network may also be a result of a multi-organization consortium (e.g., a coalition network or C-WAN [KR00]) and present additional interoperability challenges for a management system.

(6) Limited Survivability

One of the major challenges in using ad hoc networks is their limited survivability and vulnerability to security attacks [ZH99]. The use of a wireless medium for communication opens another venue for initiating link level attacks ranging from passive eavesdropping to message replay and message distortion. In addition, deployment of wireless ad hoc networks in diverse and often hostile environments (rapidly deployed military battle-site network, sensor fields used to collect sensitive data in remote, unmanned locations) makes these networks even more prone to network security attacks leading to failure of network elements.

(7) Temporary and Mission-centric Deployment

Unlike infrastructure-based networks, the deployment of ad hoc networks is mostly temporary and frequently tends to be mission-centric. Ad hoc networks provide an efficient and economical alternative in setting up the means for communication and networking in situations where the use of a wired or infrastructure-based network is either too expensive or impractical. Further, ad hoc networks are typically used to carry out one or more specific missions, e.g., in military or disaster management and relief operations.

The above discussion helped us gain a better understanding of the constraints imposed by typical ad hoc networking environments that any management system needs to address. In our research, we propose and implement a management framework for configuration or provisioning of wireless ad hoc networks, with primary application to QoS management. In the following section, we discuss recent research in the area and how it relates to our work.

2.2 Related Work

Extensive research dealing either with network management or Quality of Service (QoS) in ad hoc networks exists. However, it is noteworthy that a comprehensive approach to QoS

management in ad hoc networks, i.e., the study of a network management system with QoS in the main text, is still lacking. We believe that such a comprehensive approach is necessary to improve network survivability and to ensure QoS robustness – two major challenges in ad hoc networks.

2.2.1 Ad Hoc Network Management

Leinwand and Fang [LF93] define network management as “the process of controlling a complex data network so as to maximize its efficiency and productivity.” A network management framework typically involves two main tasks: monitoring and provisioning. These tasks in effect contribute to five functional areas [LF93] defined by the International Standards Organization (ISO): fault management, configuration management, security management, performance management, and accounting management. Monitoring allows a management system to discover the capabilities of the network elements, maintain up-to-date knowledge about the network topology, collect information about applications/users involved, keep track of network utilization, discover problems such as failure of network elements, etc. Provisioning allows the management system to configure the various network elements and determine relationships among them. Clearly, the monitoring and configuration tasks are inter-dependent. Hence, while our focus is on robust QoS provisioning and management in wireless ad hoc networks, and the mechanisms involved therein, we also reflect on the other relevant aspects of network management that co-exist in a management framework.

The literature in the field of network management, in general, indicates two main approaches that have been adopted: one using a *client-server or manager-agent model*, and a second using the concept of *mobile agents*. Both of these models have been extended to the management of ad hoc networks.

In [CJS99], an Ad hoc Network Management Protocol (ANMP), essentially an extension to the widely used Simple Network Management Protocol (SNMP) [Sta93], has been proposed. ANMP uses an enhanced SNMP management information base (MIB-II) that provides it with the improved flexibility required to deal with some of the problems identified in managing ad hoc networks. Chen *et al.* [CJS99] focus on three areas of network management: data collection, fault management, and security management. The evaluation of ANMP involves comparison of two

clustering algorithms with respect to the message overhead generated. However, it does not reflect its performance over severely bandwidth-constrained wireless links, one of the main challenges in ad hoc networks. Further, ANMP, being an extension of SNMP, still lacks certain essential features required in a protocol for Quality of Service (QoS) provisioning [LT00, Rap-1, Rap-2] (see Section 3.2.2 for details). Yong-xin *et al.* [YLG01] have extended the work in [CJS99]; a new clustering algorithm is proposed that takes into consideration existence of unidirectional wireless links in mobile ad hoc networks and the relative mobility among the network nodes.

A hierarchical SNMP-based management architecture, the *Spreadsheet-based Hierarchical Architecture for MANagement (SHAMAN)*, is presented in [SZH+01]. The focus of this work is again on monitoring and data collection. Using a hierarchical approach, Sethi *et al.* [SZH+01] demonstrate how SHAMAN can be used to achieve effective location management in a battlefield ad hoc network. A new Spreadsheet Scripting Language is used to facilitate procedural management functionalities; its utility is demonstrated using simulations as well as a prototype implementation.

Experiences in managing a real-life ad hoc networking environment that involves a multi-national naval ad hoc network known as Coalition Wide Area Network (C-WAN) are described in [KR00]. This work addresses problems such as heterogeneity, low bandwidth, scalability and interoperability encountered in the C-WAN. The inefficiency of existing *off-the-shelf* SNMP management system in providing an easy-to-use, lightweight and flexible management framework, especially to manage QoS and security sensitive traffic, is highlighted, motivating the need for a policy-based automated management system. An alternative distributed-object architecture based on the Common Object Request Broker Architecture (CORBA) and SNMP, to mitigate some of the problems mentioned above, is described in [KR00]. However, no information on the performance evaluation of such architecture is available in the literature.

The work in [BH01] also deals with network management of military ad hoc networks, providing a preliminary discussion of a C2 data model. The C2 data model is a database for storing logical and situational awareness data. The primary objective of this data model is to facilitate network planning, operation and reconfigurations in low bandwidth military tactical networks. Although the inception of data into the schema was automated based on certain feedback mechanisms, the

use of this database to configure devices (assigning nodes to a network and manipulating the address structure of multiple linked networks) was done manually.

The work in [MR01] motivates the need for adaptive and automated management for future data networks, including ad hoc networks. An interesting approach for automated network management has been proposed based on bacterial colony and genetic algorithms. A colony of software agents is simulated and is shown to perform tasks such as reacting to increase in network load, load balancing and adoption of new services, simple payment-based service quality, and ability to handle realistic traffic streams robustly and efficiently.

Several examples of using mobile agents for network management in the Internet exist [GGG+02]. Recently, the concept of mobile agents has also been extended to manage mobile ad hoc networks [SB00, SSJ03]. The basic concept underlying these systems is to use portable code (e.g., in Java or Tcl) for executing different types of agents on different hosts, in effect completing tasks near the source of data to reduce the need for large data transfers across a network. These include agents for data collection (discover available resources such as devices, bandwidth etc.), caching, configuration etc. The *Guerilla Management Architecture* proposed in [SSJ03] focuses largely on the resource discovery and monitoring aspects of ad hoc network management. Available network resources and operating conditions such as battery power usage, processing load, and node isolation probability, as well as certain management objectives are modeled as time-varying utility functions. These functions are then used to make management decisions.

Although the mobile agent-based approach to automated network management seems promising, considerable work is still required. Most of the previous works describe the management architecture; however, no quantitative performance analysis is presented. Also, it is not well understood whether such agents will be able to support complex network policies, e.g., pertaining to QoS, network security, etc. At this time, we are not aware of any evaluation of such mobile agent systems used to provision and manage QoS in mobile ad hoc networks. The Mobile Agent Runtime Environment (MARE) approach in [SB00] provides a very preliminary discussion of an agent API supporting QoS. Also, further research is required in the area of security and interoperability of such agent-based systems. Techniques to ensure secure execution of agents, and to allow seamless integration of such agents with existing management systems (e.g., SNMP) are needed.

We know that considerable work is currently being done in ad hoc network management. However, literature on the provisioning of devices and protocol support for ad hoc network configuration is scarce. Most previous research efforts concentrate mainly on network monitoring and data collection aspects of network management. There is a clear need to investigate the means to enable and implement ad hoc network provisioning, in support of complex network operational characteristics such as QoS. This is the primary motivation for this dissertation. We propose and implement a provisioning mechanism that suits the needs of ad hoc networking. In doing this, we adopt a comprehensive approach in studying the various crucial components that need to work together in such a management system.

2.2.2 Quality of Service (QoS) in Ad Hoc Networks

As mentioned earlier, QoS management in ad hoc networks is still largely uncharted territory. The existing literature on QoS in ad hoc networks can be broadly categorized into two groups: one dealing with *QoS architectures and signaling* (e.g., [LAZ+00, MST00, XSL+00]), and a second focusing on *QoS-aware routing* (e.g., [Che99, CN99, EHA01, Lin01, LG01, LL99, SSB99]). Most of the work on QoS routing deals with QoS constraints such as bandwidth, delay or jitter bound, and routing cost. Further, with the exception of [Che99] to some extent, none of these proposals deal with the aspects of QoS robustness [CM01], graceful degradation of service guarantees or preemption of low priority flows in the wake of network congestion or failure.

One approach for QoS provisioning in the Internet that has met with considerable interest in the networking community is that of Policy-Based Network Management (PBNM) [Kos01, Pol, Ver00]. Unlike legacy network management, which generally involves configuring and managing each network entity individually, PBNM configures and controls the various operational characteristics (such as network security and quality of service) of a network as a whole, providing the network operator with a simplified, logically centralized and automated control over the entire network. So far, the work on PBNM has mainly focused on large fixed networks [Kos01, Ver00, VBJ01, VCA02], e.g., enterprise, content provider, and Internet Service Provider (ISP) networks; policy-based networking has shown a lot of promise for managing QoS in such networks.

In recent times, there has been growing interest in extending policy-based networking to mobile and nomadic computing. Munaretto et al. [MAF02] discuss the potential of adopting policy-based management to manage mobile users within an enterprise network, while Harroud et al. [HAK03] propose and demonstrate an agent-based architecture to provide policy-driven personalized multimedia services for nomadic users. However, both the works focus on infrastructure-based (wireless LANs, UMTS/3G) mobile networks.

To our knowledge, PBNM has not been extensively studied in the context of ad hoc networks; only a very preliminary discussion of how PBNM could be useful in managing ad hoc networks is presented in an unpublished report [Lin00]. The promise PBNM has shown and a clear need [KR00, PD03] for a management framework that facilitates deployment of complex policies in ad hoc networks in an automated fashion has motivated us to choose policy-based approach for our management framework. In our research efforts, we extend and study the policy-based approach, for the first time, to manage mobile ad hoc networks, with focus on QoS provisioning and management. In the next section, we provide a brief overview and the state of the art in the field of Policy-Based Network Management (PBNM) and the developments in the field of policy-based QoS.

2.3 Policy-Based Networking

2.3.1 Overview

Over the past few years, the growing interest in the field of policy-based networking is evidenced by several research and development efforts in both academia and industry, working groups leading standardization efforts, new technical conferences, and new commercial products supporting policy-based management. However, the idea of using policies in network management is not new; the original idea is known to have evolved in the early 1970s [Lew96], to monitor and control the access rights of resources in large distributed systems. Since then, with the evolution of the Internet, the increasing complexities and heterogeneity of modern networking technology, and the increase in the number of resources to be managed, it is not surprising that the policy-based approach to automating network management has become so popular. The policy-based approach can be used to manage different aspects of a network, commonly known

as policy disciplines [Ver00]. Some examples of policy disciplines are Quality of Service (QoS), network security, and IP address allocation. Policy-based networking configures and controls the various operational characteristics of a network as a whole, providing the network operator with a simplified, logically centralized, and automated control over the entire network.

In [WSS+01], a policy is defined as “a definite goal, course or method of action to guide and determine present and future decisions.” In general, policies can be seen as plans of an organization to achieve its objectives. This may involve a set of rules to govern the behavior of its network and its components (resources, users, applications, etc.), and the specification of a set of actions to be performed. Policies can be classified [RVS+99, Ver00] into different levels in a hierarchy allowing simplified abstraction of complex low-level policies to simple high-level policies that do not use networking jargon.

Business-level or *high-level policies* are those that express the overall goals of an organization. *Network level policies* are essentially business level policies mapped and expressed into networking terminology. These are defined and entered by a network operator with a high-level perspective of the network topology, objectives and network-wide utilization. *Node-level policies* are those that correspond to the objectives and requirements at the different network nodes, and *device-level policies* are device-specific instructions that facilitate implementation of algorithms, for example, for classification, scheduling, buffer management, etc. The node and device level policies typically constitute the *low-level policies*. In order to successfully deploy policies in a network, the policies need to satisfy certain requirements [Ver00]. They should be precisely defined and specified to be understood and enforced at a network element. The policies must be compatible with the capabilities of the network element where they may be enforced. Furthermore, policies must be mutually consistent to avoid conflicts and ambiguous decision-making. Finally, the policies should be simple, intuitive and easily understood at a higher-level by human operators, and the network operator should be able to specify them with ease.

The Internet Engineering Task Force (IETF) and the Distributed Management Task Force (DMTF) have been working together to define a policy framework [Dis, Pol]. The IETF Policy Framework working group provides guidelines for defining a policy framework, and an information model and schemata to define, store and retrieve policies [Pol, WSS+01]. The architectural elements typically found in a policy-based system are as shown in Figure 2.4.

A *Policy Management Tool (PMT)* provides the network administrator with an interface to interact with the network. A network administrator uses the policy management tool to define the various policies or policy groups. It is typically the function of the PMT to validate the syntactic and semantic correctness of the administrator input, to ensure consistency among the high-level policies and to check for compatibility of the various policies. Further, the PMT typically determines the association between the policies and the various network elements where these policies are to be enforced, determines which low-level policies can be used to support the specified high-level policies and ensures that the specified policies are comprehensive enough to cover all the relevant scenarios. The policies specified at the PMT are then stored in a *policy repository*. A policy repository can be defined as a data store or a model abstraction that holds policy rules, their conditions and actions, and related policy data [WSS+01]. A Policy Information Base (PIB) can be considered as a type of policy repository. The concept of PIB is based on the Structure of Management Information (SMI) [MPS+99] to leverage the experience with the Simple Network Management Protocol (SNMP) [Sta93] and related Management Information Bases (MIBs). A typical PIB structure is shown in Figure 2.5. It can be thought of as a tree, with branches representing Policy Rule Classes (PRCs) and the leaves representing Policy Rule Instances (PRIs).

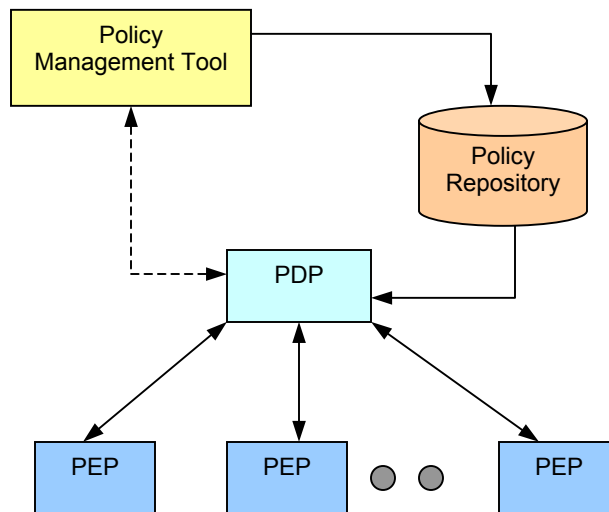


Figure 2.4 Key architectural elements of a policy-based management system.

The *Policy Decision Point (PDP)*¹ or the *policy server* generally retrieves the policies from the policy repository and performs complex policy interpretation and translation into a format that can then be used to configure one or more *Policy Enforcement Points (PEPs)* or policy clients. The PDP also needs to monitor any changes in the policies that might occur at the policy management tool (shown in Figure 2.4 as a dashed line) or repository. A policy management tool may not detect policy conflicts at a lower level, and such conflicts may have to be handled by the PDP. The PEP is a network device (e.g., end-host or router) where the policies are actually executed or enforced. The PEP is also responsible for monitoring any relevant information (such as installation/removal of policies, updates about its current status, etc.) and reporting it to the PDP to facilitate automated efficient network management.

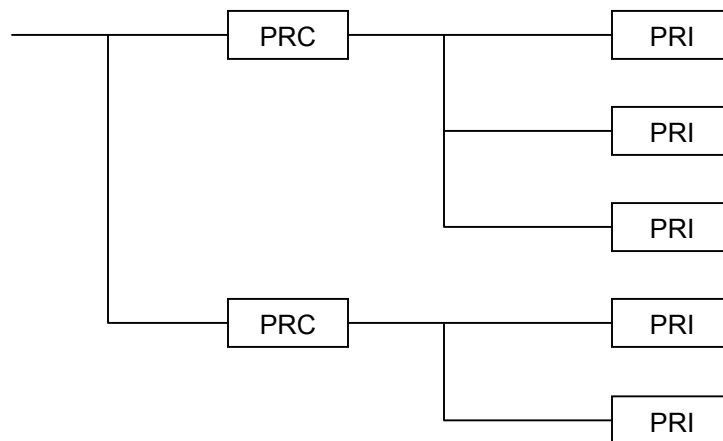


Figure 2.5 Policy information base tree structure.

2.3.2 Policy-Based QoS

The IETF Resource Allocation Protocol (RAP) [Rap] working group is active in the field of QoS policy. It has defined, among other standards, the policy-based admission control framework [YPG00], and the Common Open Policy Service (COPS) protocol [DBC+00] and its extension –

¹ It is noteworthy that a policy decision point (PDP) is typically a sub-component of a policy server [Stra99]. However, often the terms PDP and policy server are used synonymously [Kos01, Ver00, WSS+01]. To keep our discussion generalized, we will follow the latter approach.

COPS for PRovisioning (COPS-PR) [CSD+01]. COPS is a simple query protocol that facilitates communication between the policy clients and remote policy server(s).

Two policy control models have been defined: outsourcing and provisioning, illustrated in Figure 2.6. While COPS supports the outsourcing model, its extension COPS-PR integrates both the outsourcing and provisioning models. The outsourcing model is tailored to signaling protocols such as the resource ReSerVation Protocol (RSVP) [BZB+97, HBC+00], which requires traffic management on a per-flow basis. On the other hand, the provisioning or configuration model is used to control aggregate traffic-handling mechanisms such as the Differentiated Services (DiffServ) architecture [BBC+98]. In the outsourcing model, when the PEP receives an *event* (e.g. RSVP reservation request) that requires a new policy decision it sends a request (REQ) message to the remote Policy Decision Point (PDP). The PDP then makes a decision and sends a decision (DEC) message (e.g. *accept* or *reject*) back to the PEP. The outsourcing model is thus PEP-driven and involves a direct 1:1 relation between PEP events and PDP decisions.

On the other hand, the provisioning or configurations model [CSD+01] makes no assumptions of such direct 1:1 correlation between PEP events and PDP decisions. The PDP may proactively provision the PEP reacting to external events, PEP events, and any combination thereof (N:M correlation). Provisioning thus tends to be PDP-driven and may be performed in bulk (e.g., entire router QoS configuration) or in portions (e.g., updating a DiffServ marking filter [CSH+03]).

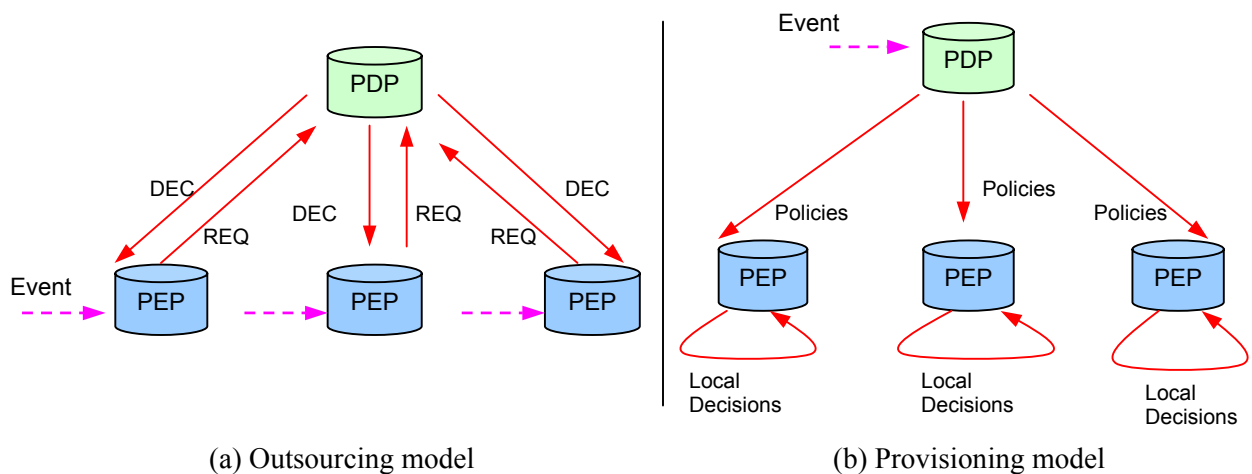


Figure 2.6 Policy distribution models.

2.4 Summary

We started this chapter by introducing the reader to the constraints typically imposed by a wireless ad hoc network environment. In Section 2.2, we presented a survey of related work that has influenced our research. We provided an overview on policy-based networking in Section 2.3.

From our literature review, we found that much work still needs to be done in the area of policy-based QoS provisioning and management in ad hoc networks, the focus of our research. The discussion in this chapter sets the stage for rest of our work. In the next chapter, we characterize the salient features sought in an ad hoc network provisioning and management system; we then propose a novel framework to address the requirements of such a management system.

Honesty is the best policy.

-Anonymous

*I never had a policy; I have just tried to do my very best
each and every day.*

- Abraham Lincoln

Chapter 3

Policy-Based Management Framework for Wireless Ad Hoc Networks

Our aim is to propose and implement a policy-based management framework for wireless ad hoc networks with focus on QoS management.

Before designing and deploying any system, it is important to thoroughly understand the characteristics of the environment in which the system is to be deployed; and as a result of these characteristics and their implications, to formulate the desirable operational properties of the system. In Chapter 2, we enumerated and studied the salient features that characterize most ad hoc network environments. In this chapter, we will take the next step – formulation of the key requirements or features sought in our management framework (Section 3.1.1). This approach will provide us with a set of requirements, essentially a benchmark to evaluate candidate solutions, and will help us in making our design decisions.

In addition, we also look at a fundamental challenge in adapting the policy-based approach to ad hoc network environments (Section 3.1.2). Following this we describe in detail our proposed policy-based management framework (Section 3.2). In doing this, we adopt a systems approach and represent our framework as an arrangement of its key modules.

3.1 Requirements for Management of Wireless Ad Hoc Networks

3.1.1 Desirable Features

We studied the constraints typically imposed by ad hoc networks in Chapter 2: *low and variable bandwidth, dynamic topology, resource-limited nodes, multiple roles played by network nodes, heterogeneity, limited survivability and temporary, mission centric deployment*. With this in mind, we now discuss the desirable operational features sought in a management system for deployment in a wireless ad hoc network environment.

(1) Efficient Signaling Mechanism

Any network management system involves a certain amount of additional control traffic to regulate the various operational characteristics of the network. In bandwidth-constrained wireless networks, it is extremely important to minimize this signaling overhead, ensuring that the links are not congested with management traffic. Thus, the constrained bandwidth in wireless ad hoc networks greatly influences the choice of the mechanisms or protocols used for the various managerial tasks (e.g., monitoring and configuration).

(2) Lightweight

Ad hoc networks generally have nodes with limited battery life, and may have limited storage and/or processing capabilities. Hence, we need a management system that does not burden the resource-limited network nodes with undue storage and processing requirements. Efficient signaling and minimal computation requirements will substantially alleviate the demand on the limited battery power [Toh01].

(3) Automated, Intelligent and Self-organizing

The ability for “self-organization” is one of the key aspects in the successful deployment of any application in an ad hoc network environment. Given the dynamic nature of most ad hoc networks, an adaptive management framework that automatically reacts to changes in network conditions is required. For example, the system, upon being alerted of failure of one or more network elements, should adjust to the depletion of resources and allow graceful degradation of

performance. In order to accomplish this, the management system should be able to automatically learn about the diverse capabilities of the nodes involved, and use this information as one of the criteria to assign appropriate roles (e.g., policy server versus client) to the different types of nodes. Dynamic policies need to be supported for automated network control based on dynamic re-evaluation of communication capabilities and assets of an ad hoc network and/or changes in its mission requirements. Further, the diverse and hostile environments in which ad hoc networks are frequently deployed also calls for a management framework that requires minimal human intervention.

(4) Secure and Robust

Finally, an ad hoc network management system should be secure and robust. It should allow secure exchange of management data among authorized hosts, and enhance the overall survivability of the network. This may require the means to authenticate and authorize users/hosts, and support encryption capabilities. In addition, the system should allow reliable transmission of management data with means to detect failure and provide fault-tolerance.

3.1.2 Adapting to the Decentralized Paradigm

From our discussion in Chapter 2, we know that policy-based networking lends itself to automated network management, simplified abstraction of network-wide policies, and centralized administration or control over the network. Automation is indeed one of the primary features required for a management system in an ad hoc network environment. However, there is still a fundamental challenge in extending the policy-based approach to ad hoc networks.

The policy-based approach can be classified under the server-client or manager-agent model for network management systems. Policy-Based Network Management (PBNM) is conceptually an idea about centralized configuration and administration of a network as a whole. This is contrary to the distributed, decentralized paradigm on which ad hoc networks are based. Hence, the challenge is to adapt this traditionally centralized service to ad hoc networks – to take advantage of the automation and simplified abstraction of the policy-based approach, but at the same time make its deployment feasible in an ad hoc network environment.

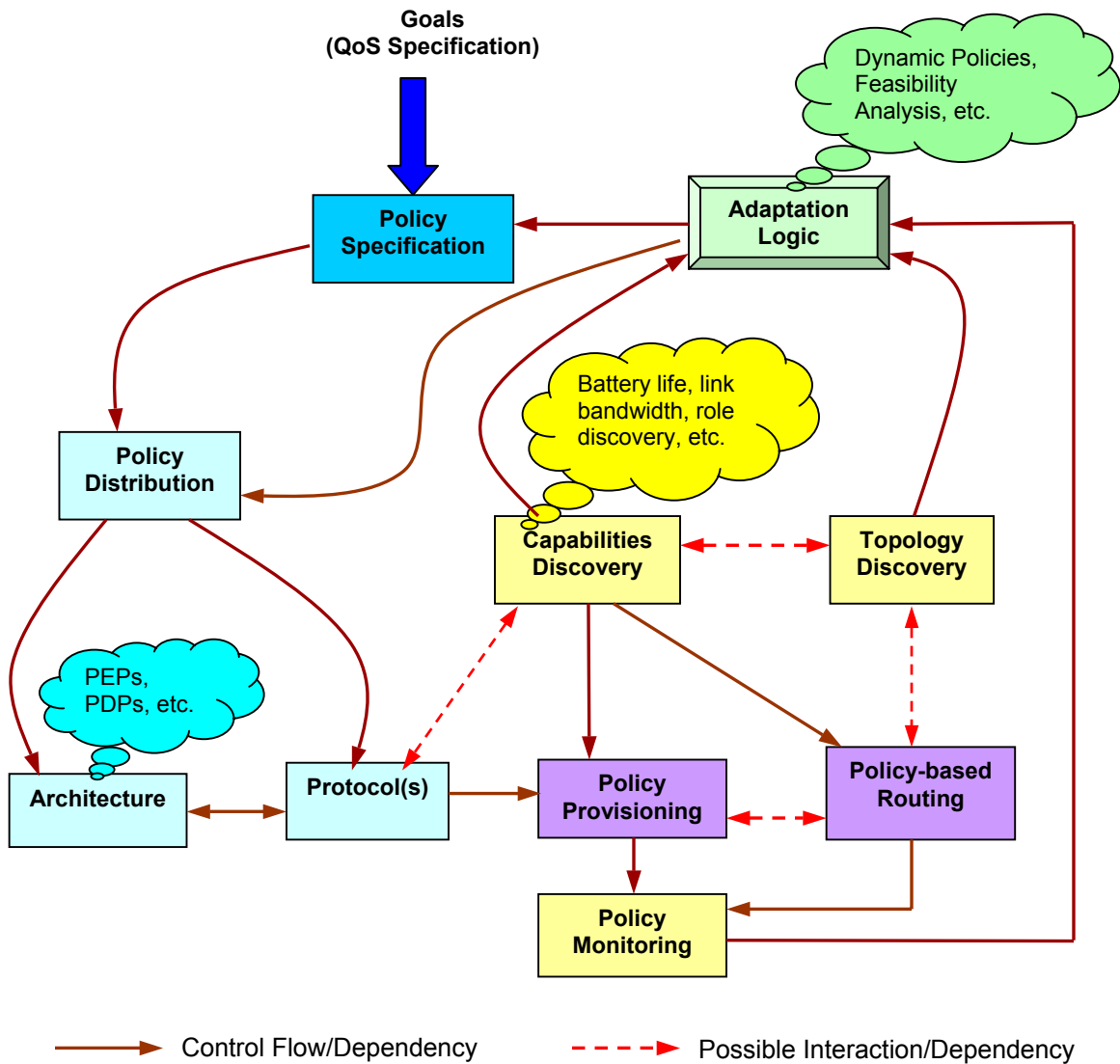


Figure 3.1 Depiction of our proposed policy-based framework for wireless ad hoc networks, and its main components.

In the following section, we propose a policy-based management framework and identify its major components. While presenting the big picture, we also point out the key components of the framework that will be the focus of this research. In doing this, we discuss certain design choices we made (e.g., the signaling protocol used for management) and preview the solutions we propose to meet the research challenges outlined above.

3.2 Systems Approach

We present our policy-based management framework (Figure 3.1) using a systems approach to gain insight into the crucial components that constitute the framework and their interdependencies. We have identified seven components (Sections 3.2.1 through 3.2.7) crucial for a Policy-Based Network Management (PBNM) system. While some of these components are important for any policy-based system (including wireline networks), we highlight the importance of certain others in a wireless ad hoc environment.

3.2.1 Policy Specification

The policy specification is a mapping of the overall goals specified for a network (e.g., QoS specification) into network-wide policies. Typically, the high-level policies are specified by a network administrator in a logically centralized fashion, and are expected to be static [Ber01]. The lower-level policies may be more dynamic based on factors such as network utilization, time of day, etc.

However, given the mission-centric and dynamic nature of ad hoc networks, even higher-level policies may need to change during network operation; policies may need to reflect unexpected and significant changes in network resources, changes in certain sub-modules of the overall mission based on feasibility analysis [Ste02], etc. Furthermore, in a multi-domain ad hoc network such as the Coalition Wide Area Network (C-WAN) [KR00] or the Virtual Operations Network (VON) [Nav98], the higher-level policies may be specified by multiple administrative domains and reconciled for overall network operation.

3.2.2 Policy Architecture and Distribution

This module is sub-divided into three major components. We provide a brief description of each of these components.

(1) Types of Architectures

The choice of policy architecture in the deployment of a policy-based system is one of the key factors in the success of such a system. We propose a taxonomy (Chapter 5) of policy architectures to provide a systematic way to explore the applicability of the various architectures to a given network environment. We use the taxonomy to indicate which architectures seem promising for low bandwidth mobile wireless networks and validate our qualitative assessment with experimental evaluation using a network testbed. Our results (Section 5.3) indicate that a distributed, hybrid architecture that combines the outsourcing and provisioning models provides an efficient and flexible solution for policy distribution in wireless ad hoc networks. A detailed discussion on this topic is provided in Chapter 5 *Taxonomy and Experimental Evaluation of Policy Architectures*.

(2) Protocol for Policy Distribution

An important factor influencing the effectiveness of a policy architecture is the mechanism used for policy distribution. Several mechanisms for policy distribution in a network exist [Ver00]: using a command-line script, using management frameworks (e.g., based on CORBA), using web servers, and using protocols such as Common Open Policy Service (COPS), Simple Network Management Protocol (SNMP), and Lightweight Directory Access Protocol (LDAP). In evaluating these different options for policy distribution, we have two goals in mind:

- (i) We require an efficient, lightweight, automated and robust mechanism for QoS provisioning in an ad hoc network environment;
- (ii) The policy distribution mechanism must facilitate interoperability between our PBNM system for ad hoc networks and the standard PBNM mechanisms that are in place in the Internet. This is important since an ad hoc network may act as a “stub network” connected to the Internet or other existing infrastructure-based network.

Based on our study of the existing literature [LT00, Ver00] and the discussions on the IETF standards working group mailing lists [Rap-1, Rap-2], the COPS for PRovisioning (COPS-PR) protocol [CSD+01] seems to be a strong candidate for QoS provisioning and management in ad hoc networks. COPS-PR is an extension of the Common Open Policy Service (COPS) protocol [DBC+00].

COPS-PR was proposed with the aim of providing an efficient and reliable means of provisioning multiple network devices. We briefly describe some salient features of COPS-PR.

- COPS-PR integrates the outsourcing and provisioning models allowing the flexibility to support a hybrid architecture [PDM02], which we believe is suitable for ad hoc network management.
- Control using COPS-PR is completely event-driven, i.e., there is no polling (unlike SNMP) between policy decision points (PDPs) and policy enforcement points (PEPs). Instead COPS-PR allows asynchronous communication between the PDPs and PEPs, with notifications (reports, changes in policies, etc.) conveyed only when required.
- COPS-PR supports structured row-level access and a completely transactional model, in that messages represent atomic transactions and a given transaction is received as a single message. This considerably improves its efficiency when handling large transactions.
- It incorporates certain fault-tolerance mechanisms. All parts of a transaction either succeed or fail. On failure, the device automatically rolls-back to its last operational state. On resuming communication, a PDP can quickly synchronize its state with the PEP. Further, if a primary policy server is inaccessible, then a PEP can switch to a back-up PDP. Alternatively, a PDP may redirect its PEP(s) to another PDP, if required.
- COPS-PR uses persistent TCP connection for reliable transfer of messages between a policy server (PDP) and client (PEP), to reduce the overhead incurred in establishing and tearing down TCP connections.
- It supports message level security using Keyed-Hash Message Authentication Code (HMAC) and Message Digest (MD5) [KBC97, Riv92] authentication, and can also be used over other levels of authentication and security mechanisms (e.g. IPsec [Atk95] and TLS [DA99, WK02]).
- COPS-PR supports control of a policy client by multiple servers without the danger of data corruption. Each PDP has its own data instance space on a PEP that cannot be manipulated by other PDPs or servers. The isolation of data instances is achieved by the message-level “Client-type” field.
- COPS-PR incorporates feedback from the PEP to the PDP(s) allowing it to report successful installation of the Policy Information Base (PIB), or failures or errors. It also allows capability discovery, wherein a PEP can indicate to its policy server the type of policies it can support.
- Finally, COPS is an extensible protocol. New COPS “client types” can be defined, and existing PIBs can be extended or new PIBs can be defined to support the requirements of a given policy discipline or networking environment.

It must be noted that, although we propose COPS-PR be used for provisioning and distribution of policies, it may co-exist and interact with other management protocols (e.g., SNMP used to monitor bandwidth utilization).

(3) Automated and Self-organizing Control Structure

In our earlier discussion, we highlighted the importance of the management framework being self-organizing and automated. We recognize five mechanisms as key to achieving this goal: service discovery, clustering, adaptivity, and topology management. We briefly enumerate these mechanisms below.

- **Clustering**

The idea of *clustering* in ad hoc networks is not new. Clustering basically transforms a physical network into a virtual network of interconnected clusters or groups of nodes. Researchers have used the concept of clusters for different purposes [Ste00] – to facilitate management [CJS99], to improve routing efficiency [LG97], to support QoS [RS98], to name a few.

In this dissertation, we propose a k -hop clustering scheme solely from a management viewpoint. The basic idea behind our cluster management scheme is to limit the service coverage of each policy server, i.e., to limit the number of wireless hops between a policy server and its clients. The motivation behind our clustering mechanism is two-fold:

- (i) By maintaining fewer hops between a policy server and its clients, i.e., by keeping the management traffic in the vicinity of a policy server, fewer resources (e.g., bandwidth and battery life) are required at intermediate nodes for management. This considerably improves the efficiency of the management system.
- (ii) Further, this leads to considerable improvement in system performance (in terms of response time and predictability) as shown by our preliminary experimental results in Section 5.3.

We present our k -hop cluster management scheme in Section 6.1.

- **Service Adaptivity**

Given that service coverage (cluster size) of each policy server in the network is limited, it is possible that any existing server may not cover one or more clients in the network. To address such cases, and to adapt to mobility of network nodes and thereby increase policy-based service availability, we further enhance our clustering algorithm using what we call *Dynamic*

Service Redundancy (DynaSeR). It is noteworthy that using the DynaSeR solution, we can also tackle the problem described in Section 3.1.2, namely, adapting a traditionally centralized service (PBNM) to a decentralized paradigm (ad hoc networks). The DynaSeR solution is described in detail in Section 6.2.

Earlier, we mentioned that we are interested in addressing policy-based management in multi-domain ad hoc networks. The main challenge in such an environment is to provide means to reconcile or negotiate policies (between different domains or organizations) *on-the-fly* with minimal or no human intervention. We propose a policy negotiation mechanism (see Section 6.3) to make this possible and demonstrate its operation for QoS management (Section 7.3).

- **Service Location Discovery**

The *Service Location Discovery* could be considered as one of the modules to appear under Section 3.2.3: *Resource Discovery*. However, since service location also forms an integral part of the automation and self-organizing process in our framework, we found it appropriate to discuss it here.

For any service (e.g., printer server or information service such a central database) deployed in a network to be useful it is important that the other nodes in the network be able to discover the service locations (one or more nodes offering the service). In fixed, wireline networks, this is sometimes taken care of by the network administrator who configures the nodes at initial setup, and then propagates any relevant changes that occur. But, this is not a scalable approach for larger networks, or for mobile wireless networks.

Some solutions (proprietary and standards) for automated service discovery exist. However, most of these solutions [Toh01] are aimed primarily at infrastructure-based networks (e.g., Service Location Protocol [VGP+97, GPV+99], Jini [Sun00], Salutation Protocol [Sal99]) or single-hop Bluetooth piconets (Service Discovery Protocol [Blu01]).

Recently, some work [GT01, KF02] is being done for service location discovery in ad hoc networks. In [GT01], Guichal and Toh evaluate the use of centralized and distributed service location protocols for MANETs. Analysis of service location protocol overhead in ad hoc networks has been provided in [KF02].

We describe in detail our proposed heuristic scheme for service discovery and its implementation in Section 6.4. Our primary goal is to facilitate automated discovery of policy servers in our framework for ad hoc networks.

- **Topology Management**

Topology management [BG03] in ad hoc networks is also an important aspect of self-organization. So far, topology management has been effectively used to facilitate routing, improve energy-efficiency, for load balancing and to efficiently schedule channel access. A relevant problem of particular importance to an ad hoc network management system is that of network partitioning. Network partitioning is of concern especially in presence of group mobility [CBD02]. Using ways (such as prediction techniques of [WL02]) to estimate and proactively adapt to network partitioning can play an important role in improving service coverage and robustness of the management system.

While the first three mechanisms – clustering, service adaptivity, and service discovery – form the core of our proposed solution suite, topology management is out of scope of this dissertation, and will not be discussed any further.

3.2.3 Resource Discovery

A PBNM system translates the high-level policies (Policy Specification) into device-specific configuration to dictate the use of network resources. In order to achieve this, the management framework must be aware of the various resources available in the system. These include the devices active in the network and their capabilities, network topology, bandwidth utilization, etc. This issue is particularly critical in ad hoc networks. For example, in a fixed wireline network, the network topology changes very infrequently, mainly due to a node being removed or added, or a link becoming temporarily disconnected. On the other hand, given the dynamic topology in a wireless mobile ad hoc network, it is crucial for the policy system to keep updated knowledge about the network topology.

In this research, we demonstrate how topology discovery can be accomplished by interfacing our management system with two existing ad hoc routing protocols, namely the Optimized Link State Routing (OLSR) protocol [CHC+01], and the OSPF-MCDS protocol [Lin03, LMP03] being implemented at Virginia Tech. Both protocols are proactive or table-driven; while OLSR makes partial topology information locally available, OSPF-MCDS provides global topology information. In chapter 6, we describe our implementation – integration of our policy server

software with OLSR and OSPF-MCDS routing daemons – in the Linux operating system, and demonstrate its operation.

Similarly, resource discovery also involves keeping updated knowledge of factors such as available bandwidth, battery power, etc., in an ad hoc network. However, this typically calls for additional signaling and/or computation. In resource discovery, the trade-off is generally between efficiency (minimal signaling) and accuracy (based on how current is the information maintained by the management system).

3.2.4 Policy Provisioning

Policy provisioning occurs after policies are distributed, and consists of installing and implementing the policies using device specific mechanisms (e.g. marking, classification, queuing, policing, etc.). Thus, policy provisioning directly affects the way in which quality of service mechanisms at a device are configured, and the way in which the various traffic flows in the network are treated.

As mentioned earlier, we intend to demonstrate the usability of our policy-based management framework with Quality of Service (QoS) as the primary application. In our work, we use a Diffserv-like [BBC98] approach for provisioning QoS policies to govern per-hop behaviors for the various traffic flows in the network.

3.2.5 Policy-based Routing

One of the key functions of a policy-based framework is to control the flow of data traffic in the network based on pre-determined policies. Integrating these policies into the routing functionality seems to be one of the most effective ways of achieving such control. A routing approach that honors the defined network policies is termed as policy-based routing [Bra89, Cla89]. These policies may involve end-users or hosts, temporal policies, access control, resource allocation or QoS policies (QoS-routing can be considered a subset of policy-based routing), etc.

Policy-based routing has been studied and deployed extensively in wireline networks [Avr92, CS98]. In the literature on ad hoc networks, two types of routing schemes can be grouped under the category of policy-based routing: QoS-aware routing (e.g., [Che99]) and routing based on power management policies – to increase longevity of network nodes by conserving battery life (e.g., [Toh01-1]). We believe that policy-routing approach could be further extended to deploy complex policies (e.g., preemption priorities, access control, inter-domain policies, etc.), to provide the much desired flexibility in the dynamic ad hoc network environment, and help enhance network survivability and robustness. For example, in ad hoc networks, the wireless links tend to be unreliable, causing routes between two nodes to intermittently go down. To mitigate the unreliability, policy-based routing could be used to route high-priority traffic (mission critical, loss intolerant data) over more stable links.

Policy-based routing and its applicability to ad hoc network environments still largely remain open for further investigation.

3.2.6 Policy Monitoring

Policy distribution and provisioning are the first essential tasks in ensuring that devices are configured consistently with the defined policy specification. However, to provide robust management of the network, it is desirable to have an independent policy monitoring process to ensure that the network in fact meets the high-level goals or specifications. Policy monitoring can be achieved via active (using dummy transactions and sending probe packets) or passive (running probes to estimate performance of network flows) methods [Ver00]. In general, policy monitoring, including the mechanisms to deploy it in a policy-based system, is also an open area for research.

3.2.7 Adaptation Logic

Given the dynamic nature of ad hoc networks, it is necessary for a policy system to incorporate dynamic and state-dependent policies that allow the control structure to adapt to the current state of the network. For example, a certain set of policies may be functional as long as the network

utilization is under a certain threshold value. However, if the threshold value is exceeded, then another set of policies may be used. Another example of adaptive policies is one based on temporal parameters – certain policies may be triggered at certain time of the day, or may remain active only for certain duration of time. Such adaptive policies need to be supported in ad hoc networks.

The adaptation logic is essentially a representation of the “intelligence” that sits in the management framework and allows it to choose the appropriate policies. Using the feedback (e.g., policy monitoring) and resource discovery mechanisms, a management system can make intelligent decisions and adapt to the changing network environment. Furthermore, such intelligence can allow for feasibility analysis beforehand. For example, [Ste02] describes a scenario wherein the command and control center in a naval environment (an ad hoc network consisting of a fleet of ships, and other military platforms such as fighter planes and unmanned air vehicles), needs intelligent feedback from the network management system for a priori feasibility assessment of future missions.

3.3 Summary

In this chapter, we presented the main requirements and research challenges in managing ad hoc networks. In doing this we highlighted the fundamental challenge of adapting the conceptually centralized PBNM idea to the decentralized paradigm on which ad hoc networks are based.

We proposed a policy-based management framework for wireless ad hoc networks that we believe addresses the key desirable features sought in an ad hoc network management system. We presented the framework using a systems approach that allowed us to breakdown the framework into its crucial components and gain better understanding of their functional tasks, while providing insight into their interdependencies.

We provided the reader with a preview of the core elements of this work: the policy architecture taxonomy; COPS-PR as a candidate protocol for policy provisioning; and our proposed solution suite (clustering, DynaSeR, service discovery, and policy negotiation).

There is no method but to be very intelligent.

- T. S. Eliot

No amount of experimentation can ever prove me right;

a single experiment can prove me wrong.

- Albert Einstein

Chapter 4

Research Methodology

In this chapter, we discuss the methodology and tools used to perform the research reported in this document. Our research methodology consists of two main components: experimental measurements and evaluation, and simulation-based study. While the former allowed us to test our implementation in a real network and helped us gain insight into some of the practical issues, the latter enabled a scalability study of our proposed management scheme in the presence of random mobility. Section 4.1 describes our experimental testbed network. In Section 4.2, we describe the various software tools used in our implementation and for performance measurement and analysis. We then provide a brief overview of the QualNet™ network simulator that was used to conduct the simulation study. Finally, we discuss the metrics and factors used for performance evaluation and demonstration of our policy-based management system implementation.

4.1 Testbed Network

Most of the experimental work reported in this dissertation was done using a Linux-based testbed network. This testbed incorporates both wired and wireless networking capabilities, as described below.

4.1.1 Wired Testbed: Mobility Emulation using the Dynamic Switch

The wired testbed is comprised of ten computers as listed in Table 4.1. Up to nine of these machines can be used to form a mobile network, while the remaining computer operates as the Dynamic Switch [LMP02]. The operating system used is Red Hat Linux (versions 7.0, 7.2 and 9.0) [Red].

The Dynamic Switch was implemented at Virginia Tech as a part of the NAVCIITI project (Task 3.1). The dynamic switch is a software application that allows emulation of a mobile wireless network environment over a wired network; it allows the creation of dynamic topologies, supports variable packet drop rates, and allows low bandwidth emulation by shaping traffic. Thus, the dynamic switch makes the experimental evaluation of a real implementation simpler – experiments can be run in a much more controlled (wired) environment as compared to using an actual wireless network, and dynamic network topologies can be accurately replicated for multiple iterations of the same experiment.

Table 4.1 Machines Used in the Testbed Network

Machine	RAM (MB)	Processor (MHz)
Desktop 1	128	Celeron 600
Desktop 2	128	Celeron 600
Desktop 3	128	Celeron 600
Desktop 4	128	Celeron 600
Laptop 1	256	Pentium III 1130
Laptop 2	256	Pentium III 1130
Laptop 3	256	Pentium III 1130
Laptop 4	256	Pentium III 1130
Laptop 5	256	Pentium III 833
Dynamic Switch	256	Celeron 600

4.1.2 Wireless Testbed

The five laptops listed in Table 4.1 constitute the wireless testbed. The laptops use Linksys wireless LAN (IEEE 802.11b) PC Cards (Figure 4.1), which operate in the unlicensed Industrial,

Scientific and Medical (ISM) band in the 2.4 GHz frequency range, and support a maximum bandwidth of up to 11 Mbps. The cards use the Intersil Prism chipset and support variable transmitter power in the range -44dBm to 20dBm . Transmitter power can be controlled from command line interface using the *Wireless Tools for Linux* [Wir]. This feature was important for us; all our wireless experiments involved operation in the ad hoc mode, and reducing transmitter power allowed designing multi-hop wireless topologies over our small work area (approximately $25\text{m} \times 25\text{m}$) in an indoor environment. With the minimum transmitter power of -44dBm , a single wireless hop was formed over a distance of 6-7 meters. Using a simple but effective idea allowed us to further reduce the transmitter power. We used aluminum foil as an “attenuator” by wrapping it around the antenna portion of the wireless cards as shown in Figure 4.2. This reduced the distance of a single wireless hop to about 3 meters, and made deployment of larger number of wireless hops and more complex network topologies possible.



Figure 4.1 Laptop with wireless interface PC Card.



Figure 4.2 Antenna portion of the wireless PC Card wrapped with aluminum foil “attenuator.”

4.2 Software components

In this section, we introduce the software applications and tools used in our experiments. A detailed description of our implementation and how these tools were used is given in Chapter 6.

4.2.1 Intel® COPS Client SDK

The Intel® COPS Client Software Development Kit (SDK) [Int] provides Common Open Policy Service (COPS) and COPS-PR protocol implementations for policy clients. It also provides a COPS server simulator, which can be used to test with the client program. However, the source code for the Intel COPS server is not available in the public domain. We needed access to the server-side code to implement our proposed solutions. Thus, as an alternative, we used the COPS distribution implemented by researchers at Telia Research AB and Lulea University of Technology.

4.2.2 Telia Research COPS API

Researchers at Telia Research AB and Lulea University of Technology, Sweden, made a freeware implementation of the COPS and COPS-PR API available in the public domain². Apart from using the Intel COPS SDK for some of our preliminary experiments, we used the Telia Research COPS distribution for implementing our policy-based management framework. The distribution provides API support for implementing a policy client as well as a policy server.

4.2.3 OLSR Routing Daemon

The Optimized Link State Routing (OLSR) was proposed and implemented by researchers at INRIA, France [Opt-1]. It is a proactive protocol that maintains local link state and routing information. In addition to the OLSR daemon (*olsrd*), the software package includes two utilities

² At the time of writing this dissertation, the COPS distribution was no longer available on the external webpage of Telia Research.

namely *olsrquery* and *olsrtrace*. While *olsrquery* allows a user to access the OLSR routing table, *olsrtrace* can be used to generate routing traffic trace files for a given network node. In most of our experiments, we used the INRIA OLSR implementation for Linux. A ported version of OLSR for Windows 2000 and Pocket PC has been developed and made available by researchers at Polytechnic University of Valencia [Opt-2].

4.2.4 OSPF-MCDS Routing Daemon

OSPF-MCDS [Lin03, LMP03] is a mobile ad hoc routing protocol proposed and developed at Virginia Tech. It is a link state routing protocol that maintains global topology information, simplifying topology discovery. We integrated our PBNM software with the OSPF-MCDS routing daemon and demonstrated its operation.

4.2.5 DiffServ on Linux Tool

We use the *Differentiated Services on Linux* tool [Dif], also popularly known as *Linux Traffic Control (tc)* tool, to implement IP QoS. Different algorithms for queuing, policing, shaping, metering, etc., are available in the *tc* tool. We have interfaced our PBNM software with the *tc* tool to allow automated, dynamic QoS provisioning; policy definitions are expressed as QoS configurations (essentially, *tc* commands) using scripts. In this dissertation, we do not address QoS issues at layers below the network layer.

4.2.6 Real-Time Application and Middleware

In this section, we provide a brief overview on real-time systems and describe the real-time software we used in our experiments. For details on real-time systems, readers are encouraged to refer to [SM01].

Distributed real-time systems for real-time control are emerging in many domains such as defense, space communication, industrial automation, and financial market. There is an

increasing interest in using these applications in multi-hop networks. In our research, we are interested in *soft* real-time applications – those that tolerate some unpredictability and deadline misses. Utility functions or benefit functions, such as those shown in Figure 4.3, are generally used to precisely specify the timing constraints of soft real-time applications. These functions define the amount of utility accrued on completion of an application activity as a function of the activity completion time. Given the performance-sensitive nature of these applications, Quality of Service (QoS) support from the underlying network is very important.

In the NAVCIITI project, we were interested in demonstrating policy-based management support for such applications in a multi-domain ad hoc network. Hence, we integrated our PBNM software and the *tc* tool with a real-time application software developed at Virginia Tech [CPB+03].

The real-time software includes a traffic generator and a middleware API. The client application is a traffic generator, which can be configured to source multiple real-time traffic flows with different characteristics and requirements, e.g., defined by utility functions. The client invokes the middleware API functions to register the application characteristics and send data packets across the network to the server. The server program receives the application packets and evaluates the performance and also displays the real-time performance plots on the choirGUI as shown in Figure 4.4.

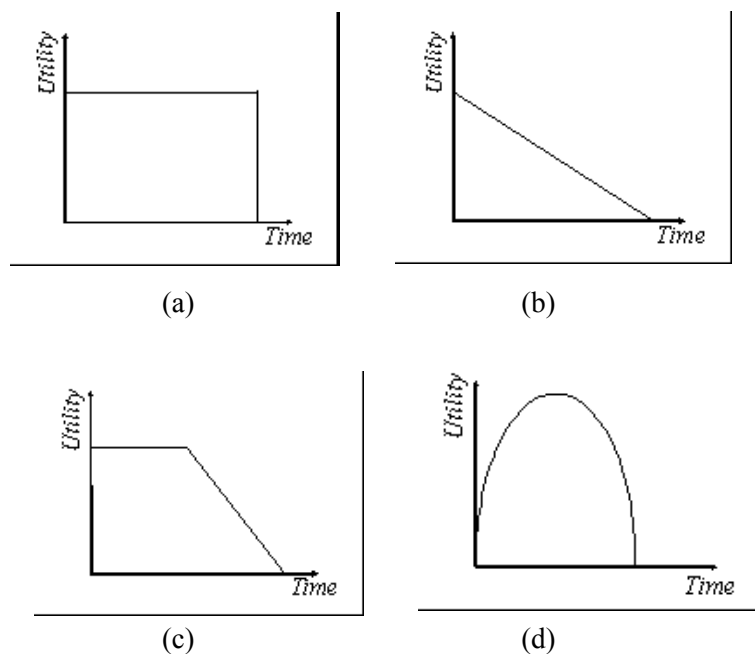


Figure 4.3 Sample soft timing constraints described using utility functions.

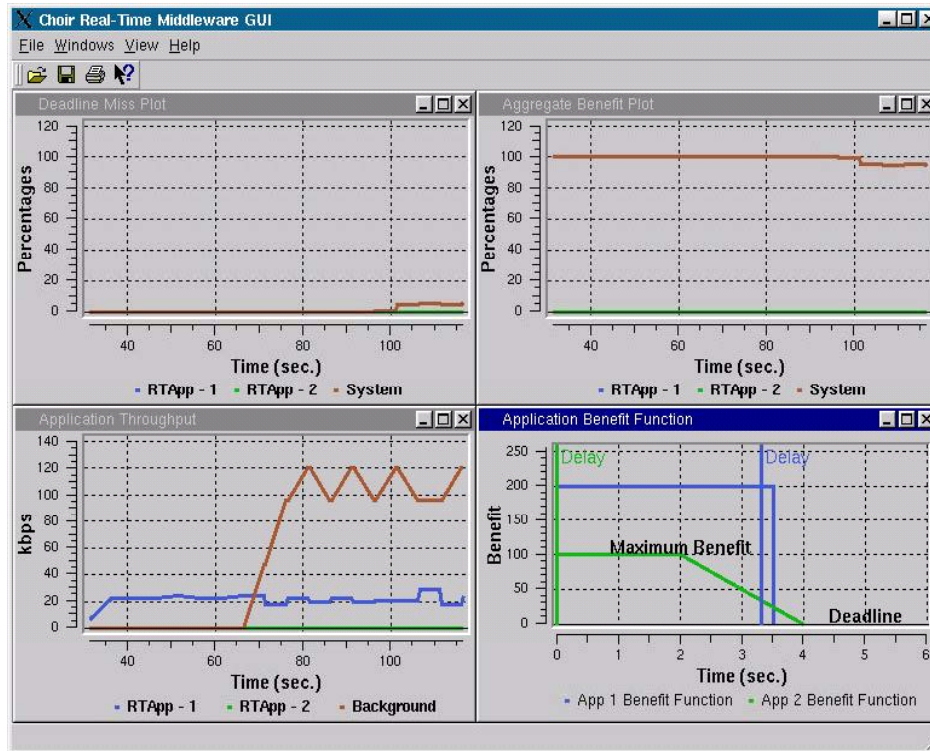


Figure 4.4 GUI of the performance monitoring tool.

4.2.7 Video Conferencing (VIC) Tool

We also used the *Video Conferencing (vic)* tool [Vic], developed by the Network Research Group of Lawrence Berkeley National Laboratory (LBNL), for qualitative performance analysis and to demonstrate QoS management using our PBNM system. VIC can be used in two modes: unicast and multicast; it can be used to stream video captured from a camera or alternatively, video captured from the desktop (screenshot).

4.2.8 BonnMotion and Supplementary Programs

BonnMotion [Bon] is a Java-based software developed and distributed by the University of Bonn. It generates mobility trace files to simulate and analyze different mobility models. It is known to support at least four mobility models [CBD02]: Random Waypoint, Gauss-Markov, Manhattan

Grid and Reference Point Group Mobility (RPGM). The generated trace files can be processed using certain utilities in-built in BonnMotion to study the mobility models themselves, or to generate trace files for network simulators – NS-2 [Net] and QualNet/GloMoSim [Glo, Qua].

We were interested in generating mobility files that can be used to conduct experiments with the Dynamic Switch. Hence, we wrote supplementary programs³ to convert the NS-2 mobility trace files generated by BonnMotion into mobility files readable by the Dynamic Switch, as shown in Figure 4.5.

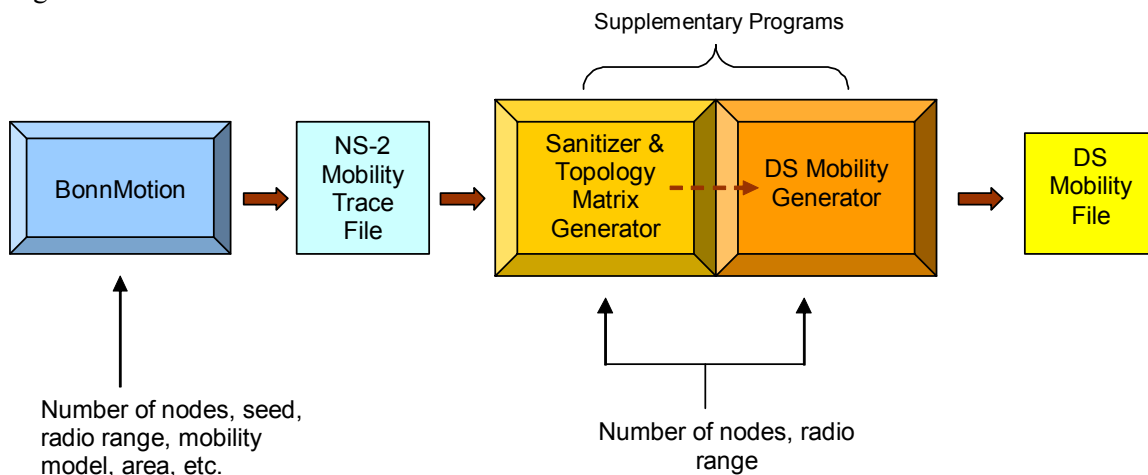


Figure 4.5 BonnMotion and supplementary programs to generate mobility files for the Dynamic Switch.

4.2.9 Analysis and Other Utility Tools

In addition to the software packages discussed above, which formed the core of our implementation and experiments, we used some other tools for traffic generation, to capture network traffic, to measure and analyze network performance and hence to debug experiments. We briefly describe these utilities below.

Traffic generators such as *Iperf* [Ipe] and *Mtools* [Mto] were used mainly to test the traffic shaping and low bandwidth emulation using the *tc* tool. They were also useful in setting up the wireless experiments and to do pre-data-collection testing.

³ The Sanitizer and Topology Matrix Generator programs are developed by Tao Lin, Virginia Tech; the DS Mobility Generator program is implemented by the author.

Tcpdump [Tcp] was used to capture network traffic, and graphs were plotted using the *trpr* tool [Trp], which supports *tcpdump* and NS-2 traffic trace files. The *trpr* tool, with its ability to plot graphs in real-time during an experiment, was especially useful for analysis and debugging.

We used *Ethereal* [Eth] to measure, debug and demonstrate our implementation. *Ethereal* is a *tcpdump*-like tool, but with an impressive graphical user interface. It was very useful in debugging and verifying our implementation, as it captured and displayed the desired network traffic. *Ethereal* played an important role in our wireless experiments. It was used to analyze, in real time, the ad hoc routing control traffic being received as we moved the nodes around. This greatly facilitated setting up of different network topologies.

Finally, we wrote some utility programs to measure and process the metrics in our experimental study. These utilities are designed to facilitate automated repetition of experimental runs; they process the data obtained from multiple such iterations and output the average value for the metrics for each set of experiments. This feature greatly eased the work required in running and analyzing several iterations of an experiment; at the same time it allowed us to present statistically significant results.

4.3 Simulation Environment

One limitation of our experimental study was the maximum number of nodes (not more than ten machines) available in the testbed. To overcome this limitation and to supplement our experimental study – to address scalability and get a better perspective of the management system performance in larger networks – we conducted numerous simulations. This exercise proved to be invaluable, since the effect of varying some of the system parameters (e.g., cluster size) on network behavior under wider variety of conditions was visible only in simulations.

We used the QualNet™ network simulator, a commercial version of the popular GloMoSim simulator, to conduct our simulation study. QualNet is a C-based simulator with a Graphical User Interface (GUI) as shown in Figure 4.6. It has a substantial library of in-built or pre-defined functions that makes modeling of new protocols easier. QualNet runs on different Microsoft

Windows and UNIX (Solaris SPARC and Linux) platforms. It is based on the PARSEC parallel simulation technology [BMT+98] that allows it to take advantage of multi-processor systems.

QualNet is comprised of four main components.

- *QualNet Animator* (Figure 4.6) allows the user to graphically define a simulation scenario. It provides control buttons and tools to configure the various network (node placement, protocol properties, statistics collection, etc.) and simulation (animation filters, simulation time, batch experiments, etc.) parameters.
- *QualNet Analyzer* is a tool to graphically plot and analyze the various metrics for which data is collected during a simulation run. It allows the user to interactively view the statistics of interest and compare statistics from different simulation runs of a batch experiment.
- *QualNet Designer* provides the user with a graphical platform to develop new protocol models in the form of Finite State Machines (FSMs), as shown in Figure 4.7. The FSM models are saved as XML files.

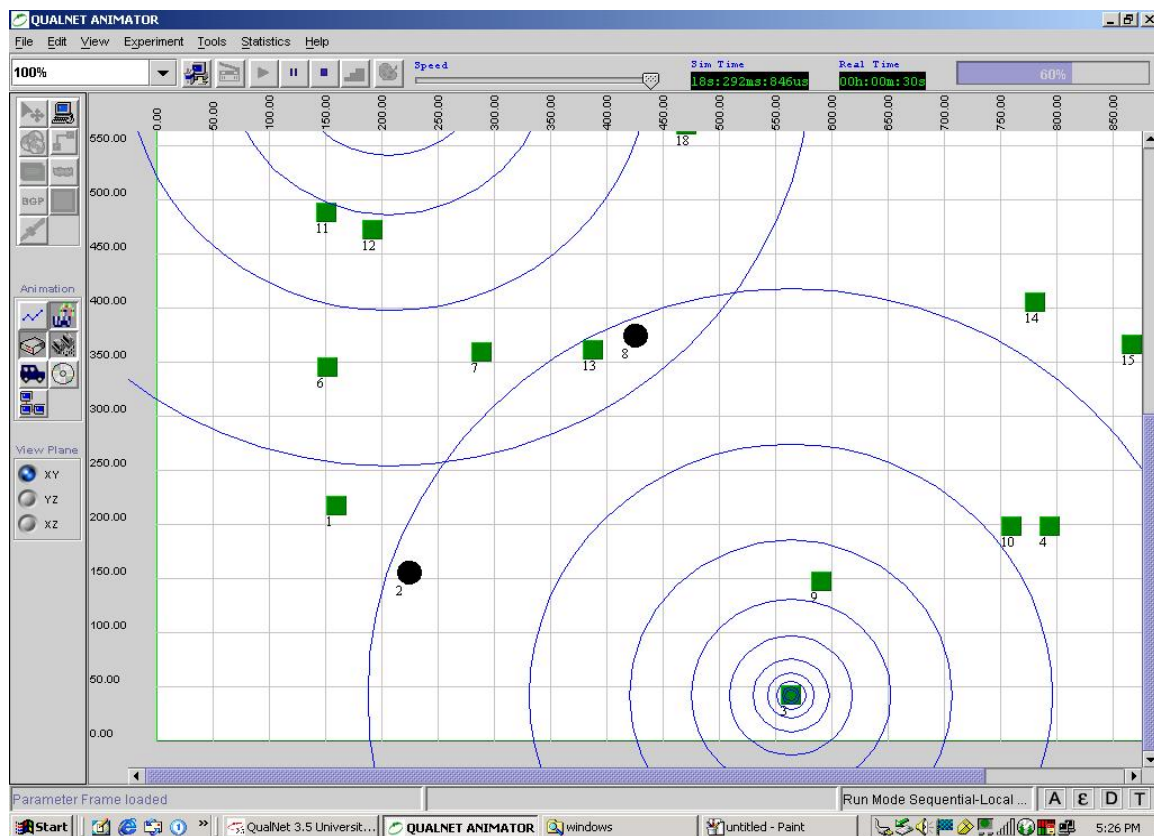


Figure 4.6 QualNet Animator GUI.

A useful feature in the Designer is that of Automated Code Generation (ACG). After an FSM is defined (with appropriate transitions and custom code entered into appropriate states), the user clicks on the “Generate Code” option. The simulator then *intelligently* coalesces the custom user code along with the *required* functionalities (such as other existing library functions and interaction of the new protocol code with higher and lower protocol layers as suitable). Following this, the model is *added* to QualNet using the “Add to QualNet” option. Using the automated code generation has a trade-off. The code written by the developer during the initial phases of the development may need to be modified at a later stage. It is during these later stages of the development that we found the Designer to be inflexible in the way it handles the code. An alternative approach, which we followed, was to use the ACG facility to initially build a code framework via the QualNet Designer. Subsequent development can then be done by directly modifying the .c and .h files. The code framework created by ACG makes subsequent code development less daunting, while directly working with the .c and .h files not only provides more flexibility, but also allows better understanding of the simulator itself.

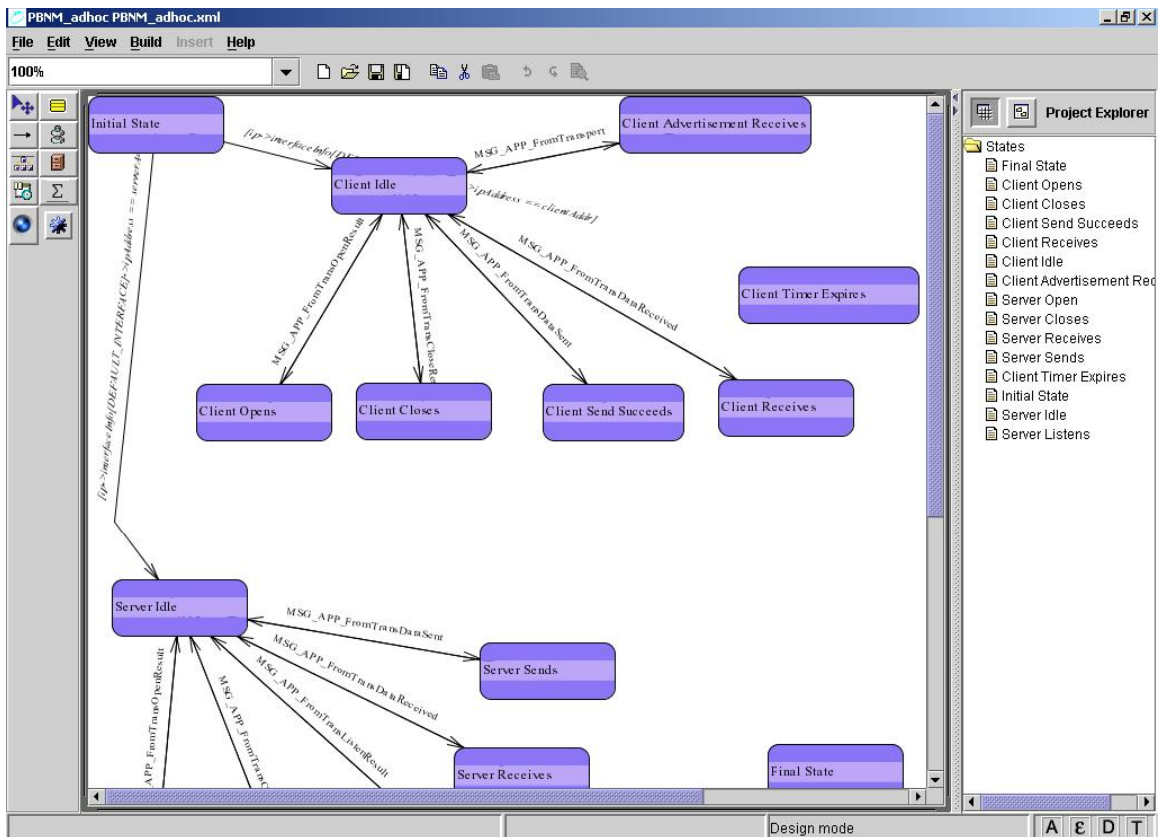


Figure 4.7 Network protocols modeled as a Finite State Machine in the QualNet Designer.

- *QualNet Tracer* allows the user to capture and trace packets belonging to specific protocols of interest. Using the Tracer GUI, the user can dissect a packet header and corresponding hexadecimal data. In other words, the Tracer is the simulator equivalent of the Ethereal tool.

In general, QualNet proved to be a very effective tool for simulating mobile ad hoc networks (MANETs). It offers a reasonably robust implementation of the 802.11b MAC protocol and wireless physical layer. It has a very comprehensive suite of MANET routing protocols. It has in-built support for the Random Waypoint mobility model. Alternatively, custom-generated mobility files (e.g., using BonnMotion) can be used to simulate node mobility.

4.4 Measurements and Evaluation

In this section, we discuss the performance metrics and factors used in this research.

4.4.1 Performance Metrics

Following are the metrics we used to evaluate performance of our PBNM system. In our results, we represent each metric as a value averaged over the number of nodes involved and over the number of simulation runs conducted.

- **Policy Response Time**

Policy response time is defined as the difference between the time at which a policy client (PEP) sends a policy request, and the time at which the corresponding policy decision is received back.

- **Inter-Decision Time**

Inter-decision time is the time difference between consecutive decisions received by the PEP. The policy response time and inter-decision time are illustrated in Figure 3.4. From our results in Chapter 5, we will see that while the policy response time quantifies the round-trip delay, the inter-decision time provides insight into the processing time spent behind every decision.

- **Management Signaling Overhead**

A management framework generally results in additional control traffic or signaling. Measuring the signaling overhead allows us to estimate how efficiently our management system operates.

We measured the management overhead in terms of two metrics: *Number of COPS Connections Handled Per Server* and *Control Message Overhead (bytes)*.

- **Percentage Service Availability**

It is the percent of the total network operation time that a policy client is able to receive service from the policy server. This metric helps characterize the service coverage of the management servers.

- **Number of COPS Connection Timeouts**

This is the number of COPS connections that timeout, i.e., a COPS client or a server shuts down the connection since a COPS Keep-Alive (KA) message was not received before the KA timer timed out. This metric provides insight into how COPS performs in our network environment of interest, i.e., in a MANET environment.

- **Subjective Testing and Qualitative Assessment**

In addition to quantitative analysis using the above mentioned performance metrics, we also use subjective testing or the quality perceived by the user(s) (e.g., quality of the received video stream) as one of our metrics. This is useful in demonstrating the network performance as perceived by the end-user.

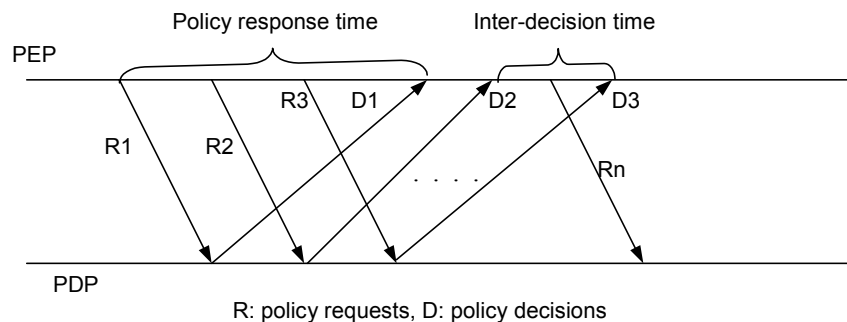


Figure 4.8 Illustration of policy response time and inter-decision time.

4.4.2 Factors

In our study, we considered the following factors:

- **Bandwidth**

Constrained bandwidth availability is one of the key characteristics of wireless ad hoc networks. Hence, we use available bandwidth as one of the factors in our experiments. Specifically, the policy response time and inter-decision time are measured for different values of link bandwidth.

- **Load (number of simultaneous policy requests)**

The management-related traffic load, essentially the number of policy requests sent simultaneously by policy client(s) to a policy server, is another factor used in our preliminary experiments to compare different policy architectures.

- **Cluster Size (k)**

This is an important parameter, which dictates the coverage of each policy server. It defines the size of the clusters in our k -hop clustering algorithm. While it seems intuitive that increasing the cluster size would increase service coverage, therein lie certain trade-offs as will be seen from our results and discussion in Chapter 7.

- **Node Mobility or Speed**

Studying the PBNM system performance as a function of node mobility allows us to reflect on whether our proposed approach is suitable for mobile ad hoc networks, and how effective it is over a spectrum of mobility.

- **Network Density**

We evaluated our system for different network densities, i.e., average number of nodes per unit area. We achieved this by keeping the simulation area constant, and varying the number of nodes in the network. Doing this also allowed us to assess the scalability of the management framework in larger networks.

4.5 Summary

This chapter described our research methodology, which includes an implementation and experimental study using a testbed network, as well as simulation-based protocol modeling and assessment. We first discussed the hardware and software components of our testbed network. We then briefly described the QualNet simulator used in this research. Finally, we defined the metrics and factors used for performance evaluation in the rest of this dissertation.

*Inanimate objects can be classified scientifically into three major categories;
those that don't work, those that break down and those that get lost.*
- Russell Baker

Creativity is the ability to introduce order into the randomness of nature.
- Eric Hoffer

Chapter 5

Taxonomy and Experimental Evaluation of Policy Architectures

In this chapter, we propose a characteristics-based taxonomy of policy architectures. The taxonomy provides a systematic way to evaluate the various policy architectures for deployment in a network environment of interest. Using the experimental testbed, we validate our initial taxonomy-based assessment of different policy architectures for low bandwidth networks, with applicability to wireless ad hoc networks. We also report experimental results obtained using our wireless testbed.

5.1 Introduction

The choice of architecture for a policy-based management system is one of the key factors in the success of such a system. An architecture advantageous in managing a particular network environment (e.g., high bandwidth enterprise network) may not be an appropriate choice for managing another network (e.g., low bandwidth mobile wireless network). Hence, it is important to study the performance trade-offs involved and choose an architecture (or combination of architectures) that suits our requirements.

As a first step in this direction, we propose a characteristics-based taxonomy of policy architectures. The main advantages of having such a taxonomy are:

- (i) By systematically classifying policy architectures into distinct categories, the taxonomy will lead to a better understanding of the various features that characterize different architectures.
- (ii) The taxonomy provides a platform to determine and analyze the strengths and weaknesses of the various architectures.
- (iii) The choice of policy architecture must consider the constraints (e.g., bandwidth availability, total number of nodes/users to be managed, mobility, processing and storage capability of nodes, etc.) that a networking environment may impose. The proposed taxonomy will help determine the applicability of one or more architectures to a given environment.

5.2 Taxonomy of Policy Architectures

5.2.1 Characteristics of a Policy Architecture

As mentioned earlier, our proposed taxonomy is based on a set of characteristics, which when linked together define the various elements in the taxonomy. These characteristics are:

(1) Locus of Control

The locus of control represents one or more policy servers or policy decision points (PDPs) in a network, capable of making policy-based decisions. The locus of control is classified as: *centralized* or *distributed*. The former corresponds to a central policy server controlling other client nodes, also known as policy enforcement points (PEPs), in the network. If this capability of making policy decisions is distributed over multiple policy servers it leads to a distributed locus of control.

(2) Locus of Information

The locus of information refers to the location of the policy information storage module or repository used to store policies and accessed by a policy server to make decisions. Typical

examples are: a policy information base (PIB) such as the differentiated services (Diffserv) PIB [CSH+03] or a directory used to store policies in directory-enabled networks [Str99]. The repository may or may not be co-located with a policy server. The locus of information is classified into two types: *centralized* or *distributed*. In the former, all the policy information is concentrated in, and accessed from, a central information base, while in the latter the policy data is stored in multiple information modules, providing redundancy.

(3) Policy Distribution Model

The distribution component of a policy system addresses the transfer of policy information between different points in the system. Policy distribution mechanisms include [Ver00]: command-line scripts, configuration files, and protocols such as the Common Open Policy Service (COPS) protocol [DBC+00]. The functionality of most of these mechanisms can be classified into two models: *outsourcing* and *provisioning*. In outsourcing, a PEP outsources its policy decisions to a remote policy server. In the provisioning model, a policy server configures, in advance (and also, if required, during network operation), one or more PEPs that it controls, thus enabling local decision-making at the PEPs. It must be noted that outsourcing and provisioning mechanisms are not mutually exclusive [WSS+01]. Combining the two mechanisms yields hybrid architectures in the taxonomy, as will be discussed in Section 5.2.2. Many times the applications supported by a network also greatly influence the choice of the policy distribution model to be used. For example, outsourcing is more appropriate for signal-oriented mechanisms such as the resource reservation protocol (RSVP) [BZB+97, HBC+00] that require more dynamic policies, while provisioning [CSD+01] is appropriate in architectures such as Diffserv [BBC+98] that involve aggregate flow handling.

(4) Tiers of Control

We define the tiers⁴ of policy control as the levels in a network where policy decisions are made. Currently most application instances [Ber01, Kos01, Ver00] of policy-based systems fit in the one and two tiered categories. In single-tiered policy control, policy servers (acting as peers) handle all the policy decisions for the various PEPs in the network, i.e., all the policy decisions are outsourced to these servers. However, if the policy servers in the upper tier configure the PEPs with relevant policies, then the PEPs can make some decisions independently. This leads to a two-tiered policy control structure, since policy decisions are being made at two levels. We can

⁴ Our definition of tiers is with regards to policy control, and is different from the concept of tiered architectures discussed in [Kos01, RVS+99].

extend this idea to more than two tiers, wherein nodes at level $n+1$ handle policy control for the nodes at level n , which in turn control the nodes at level $n-1$ and so on. Thus, nodes at tier n can be looked upon as PEPs controlled by PDPs at tier $n+1$, and at the same time as PDPs controlling PEPs at tier $n-1$. This leads to a hierarchical policy architecture or control structure.

In addition to the above-mentioned characteristics, there are certain other important features that are necessary for a practical implementation of any policy architecture. These features (presented earlier as part of our policy-based management framework in Chapter 4) are briefly enumerated below.

- ***Abstraction and Compilation of Policies***

A policy-based system provides the network administrator with a user-friendly interface and a simple high-level view or abstraction of the complex low-level policies.

- ***Resource Discovery***

A policy system must deploy policies consistent with the available network resources. Hence, a resource discovery mechanism [Ver00] is required to allow the system to discover and audit the underlying network resources (e.g., network topology, capability of PEPs and/or PDPs, etc.).

- ***Policy Monitoring and Feedback***

It is important for a policy server to have knowledge of the current policies installed in a network. This requires a feedback mechanism between the various PEPs and the PDP.

5.2.2 Types of Policy Architectures

The various possible combinations of characteristics and design choices mentioned earlier lead to a policy taxonomy tree, with the leaves representing feasible architecture types. The policy architecture taxonomy is summarized in Table 5.1.

To begin with, the policy taxonomy broadly classifies the various policy architectures into three categories based on the policy distribution model used. These categories are: *outsourced*, *provisioned*, and *hybrid* (combination of the outsourcing and provisioning models) *architectures*. These categories are further classified as follows.

Table 5.1 Policy Architecture Taxonomy Matrix

	Features Architectures	Locus of Control	Locus of Information	Policy Distribution	Tiers of Control
Outsourced	CCO	Centralized	Centralized	Outsourcing	1
	DCO	Distributed	Centralized	Outsourcing	1
	DDO	Distributed	Distributed	Outsourcing	1
Provisioned	DDP	Distributed	Distributed	Provisioning	2
	DDP-H	Distributed	Distributed	Provisioning	> 2
	DDOP	Distributed	Distributed	Outsourcing and Provisioning	2
Hybrid	DDOP-H	Distributed	Distributed	Outsourcing and Provisioning	> 2

(1) Outsourced Architectures

All the policy decisions are made at a single control tier, controlling the underlying nodes or PEPs. Three possible variations of the outsourced architecture are noted.

- **CCO architecture**

The centralized-centralized-outsourcing (CCO) architecture deploys a centralized locus of control and information. All the PEPs in the network outsource their policy decisions to the central policy server. Centralization considerably simplifies policy conflict check, ensuring co-existence of multiple non-conflicting policies. Further, since the PEPs depend on the central server for policy decisions, the PEPs can be designed to be relatively simple, as they are required to handle minimal policy-related processing.

However, a major disadvantage of the centralized approach is that the smooth functioning of the entire policy framework is largely dependent on the central policy server. In the event of the policy server becoming non-functional (e.g., shutting down due to a security attack), the entire policy system can break down. In addition to the security and integrity requirements, a centralized architecture requires redundancy through one or more back-up policy

servers. Further, this approach suffers from poor scalability, as the number of PEPs that can be handled by a single policy server is limited, and outsourcing all policy decisions to the central server entails significant amount of signaling. Thus, a CCO architecture seems applicable to relatively small and considerably secure networks, with enough bandwidth to handle the potentially high signaling overhead.

- **DCO architecture**

The distributed-centralized-outsourcing (DCO) architecture involves multiple distributed policy servers and a centralized locus of information. Note that, in this case, the distributed property of the locus of control is limited to a single tier, i.e., the policy servers operate as peers at the same single tier of control, and access the policies from the central repository. The set of policy enforcement points (PEPs) controlled by each of these servers may be disjoint or overlapping. An example of the former is when each server handles all the policy decisions for a distinct group of PEPs. An example of the latter case is a network with one server designated to each policy discipline (e.g., QoS, network security) and multiple such servers controlling each PEP. In either case, all the PEPs outsource the policy decisions to the corresponding policy server(s).

Using multiple policy servers, the DCO approach helps mitigate the scalability problem associated with the CCO approach, and hence can be used to control a larger network. Also, in the event of a policy server failure, one or more functional servers may be used to temporarily serve the set of unmanaged PEPs, providing implicit redundancy. With regards to the centralized locus of information, the DCO architecture experiences the same advantages (ease of storage and policy consistency) and disadvantages (single point of failure making redundancy of policy repository necessary) as the CCO architecture.

An example scenario where such an architecture could be deployed is a small corporate or university network comprised of several departmental sub-networks situated in a building or over a small geographical area. One or more policy servers serve each department, and all the policy servers access the information stored in a central repository or directory.

- **DDO architecture**

The distributed-distributed-outsourcing (DDO) architecture uses distributed policy control (limited to a single tier), as well as distributed storage of policy data. For instance, one can employ a central repository or a directory where all the policy information is stored, and the physically distributed policy servers access the policy information via directory servers that are

typically in close proximity to the policy servers. The directory servers cache relevant policy information for local storage and use, through replication [Ber01].

Another example of the DDO architecture could be a large network comprised of several sub-networks that are distributed over a large region (e.g., a corporate network with facilities in several parts of a country). Each sub-network has a local policy repository and is controlled by one or more local policy servers. The policy repositories may contain both policies specific to the sub-network and certain network-wide policies.

The DDO architecture provides improved scalability and fault tolerance. It also alleviates the need for policy servers to maintain WAN connections to access policy data. The policy response time tends to be lower than the DCO architecture. However, a distributed locus of information may cause inconsistent policies for transient periods of time, since replication of policy data is not instantaneous. It also makes the task of policy consistency check more challenging. Further, implementing network-wide policies may require inter-PDP communication.

(2) Provisioned Architectures

These architectures use provisioning to push the policy information to different nodes in the network. Unlike the outsourced approach, wherein the policy distribution is mainly PEP-driven, the provisioned approach involves a PDP-driven policy distribution. At least two tiers of policy control are involved, i.e., the locus of control and the locus of information are distributed across two or more tiers. In our discussion of the outsourced architectures we focused on one tier of control. That tier now forms the upper tier in the provisioned architectures, while the lower tier consists of PEPs configured by the upper tier. Two types of provisioned architectures are noted.

- **DDP architecture**

In distributed-distributed-provisioning (DDP) architecture, the policy servers in the upper tier push relevant policy information to the PEPs in the lower tier. Installing this information, the PEPs are now capable of making policy-based decisions for themselves, thus operating semi-autonomously.

Configuring the PEPs in advance helps to considerably reduce policy signaling overhead during network operation, since the PEPs can make policy decisions locally. Further, this may help reduce the policy response time, which is now independent of bandwidth availability. Both factors motivate the use of such an architecture in bandwidth-constrained networks.

The main disadvantage of this approach is that the distributed nature of locus of information and control tends to make the task of policy consistency check, and, in general, the co-ordination among the various PDPs, more complex. Further, only using provisioning to deploy policies may not be adequate to support dynamic policies needed to handle network dynamics e.g., frequent changes in network conditions (bandwidth, topology, etc.) in a MANET.

- **DDP-H architecture**

The hierarchical DDP (DDP-H) architecture extends the DDP approach to more than two tiers of policy control. The highest level of policy control configures the next lower level of policy servers, which in turn push the configuration information down to the next lower level and so on. Most of the advantages and disadvantages of the DDP architecture are shared by its hierarchical counterpart. In addition, the DDP-H architecture tends to improve scalability. The number of levels in the hierarchy may be driven by factors such as the network size or number of nodes involved, processing and storage capability of the nodes, bandwidth availability, etc.

(3) Hybrid Architecture

A hybrid approach combines features of the centralized and distributed architectures. The idea is to take advantage of the outsourcing mechanism to support more dynamic policies, and also allow configuration of PEPs through provisioning, in turn alleviating the problem of excessive signaling overhead. Hybrid architectures are classified as follows.

- **DDOP architecture**

The distributed-distributed-outsourced and provisioned (DDOP) architecture involves use of both outsourcing and provisioning. Policy servers push configuration information to the PEPs. In addition, if a policy (relevant to certain node or application) is unavailable at a PEP, the PEP outsources the decision to its policy server in the upper tier. In reply, the policy server may choose to send only the decision for the particular request, or may also push additional policy information in anticipation of future events. The DDOP approach not only provides the advantages of the DDP architecture, but it also improves flexibility and adaptivity.

- **DDOP-H architecture**

The hierarchical DDOP (DDOP-H) architecture extends the DDOP architecture to more than two levels of policy control. Such a hierarchical control structure tends to break down the complexity in managing the entire network as a whole into smaller, more manageable modules, thus

providing more flexibility and scalability. Also, it may help alleviate the processing and/or storage requirements at the various PDPs/PEPs involved. However, such a control structure calls for sophisticated interaction among the various policy elements of the network.

5.2.3 Summary

In this section, we discussed the various characteristics that define the policy architecture taxonomy and the resulting architectures. By categorizing the possible combinations of the design features, the taxonomy provides a framework for qualitative and quantitative comparisons among the different architectures. It is noteworthy that a combination of the architectures discussed above may be deployed in a network, given the heterogeneous scope of policy application instances and the performance trade-offs involved. Even in such a case, the contribution of the taxonomy lies in the fact that it would play a key role in helping us choose the “right” combination of architectures that suits our requirements.

As the area of policy-based networking matures and its applicability to newer services and networking environments is explored, additional characteristics or features may emerge. This taxonomy can then be extended to include these characteristics for a finer-grained classification of policy architectures.

5.3 Preliminary Experimental Evaluation

We have conducted experiments to validate our qualitative analysis of the policy architectures described in the taxonomy and to gain further insight into the various performance metrics. The experiments in Section 5.3.1 involve a single-hop topology, i.e., all the clients are one hop from the central policy server; we present some preliminary experimental results to illustrate the performance of select policy architectures. We are interested in characterizing the effect of available bandwidth on management system performance. Given the constrained bandwidth availability in wireless ad hoc networks, this evaluation is important, but something that has been lacking in most prior research works (e.g., [CJS99], [MR01], [YLG01]) in the area of ad hoc network management.

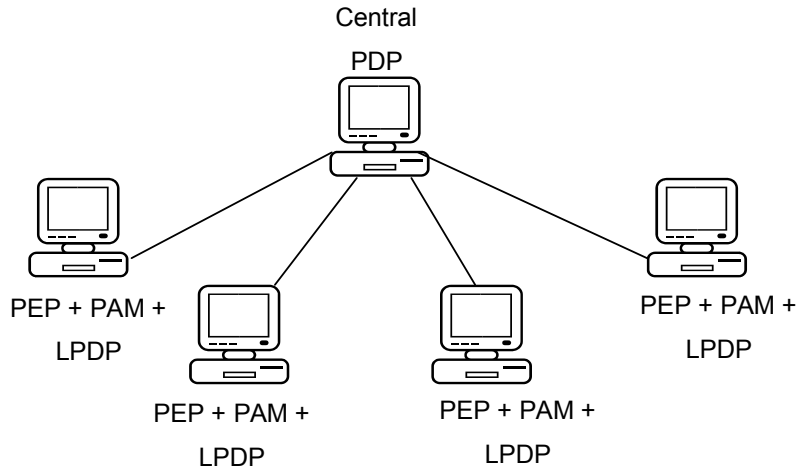


Figure. 5.1 Testbed setup for single-hop experiments.

In Sections 5.3.2 and 5.3.3, we extend our experiments to multi-hop ad hoc network scenarios. Using the dynamic switch we emulate different topologies using the wired testbed. For proof of concept, we then port our experiments to a wireless testbed, and discuss our observations and experiences while working in a wireless ad hoc environment.

5.3.1 Comparison of Policy Architectures

The experimental setup (Figure 5.1) for our preliminary experiments involved five of the desktops on our wired testbed (Section 4.1). We used the Intel® COPS client software development kit (SDK) [Int] to deploy the Common Open Policy Service (COPS) protocol supporting the outsourcing model for RSVP [BZB+97]. Policy requests were implemented as dummy RSVP bandwidth reservation requests by running a script file at each PEP. Our experiments involved the outsourced (CCO type), provisioned (DDP type) and hybrid (DDOP type) architectures. We modified the Intel COPS SDK user interface to allow a user to choose the type of architecture to be configured and tested.

To implement CCO architecture, one of the machines was configured as a central Policy Decision Point (PDP), while four other machines were configured as Policy Enforcement Points (PEPs). A DDP type of provisioned architecture was implemented with all the decisions made locally using the local PDP (LPDP) module at the four PEPs. We extended the COPS SDK implementation with a Policy Advisor Module (PAM) interface to realize the hybrid (DDOP) architecture. The

DDOP architecture comprises a central PDP, and the LPDP and PAM modules at the individual PEPs. The PAM interface allows a local PDP to detect whether local policies are available to service a given request; if available, the LPDP makes decision locally, else it directs the PEP to connect to the central PDP and outsource its decision.

In a bandwidth-constrained network, it is important to keep the signaling overhead to a minimum. In our earlier discussion, we noted that a CCO policy architecture might result in large amount of signaling overhead. In the first set of experiments we characterize this signaling. The plot of the signaling overhead as a function of the number of policy requests is shown in Figure 5.2. The experiments involve two cases: (1) a single PEP communicating with the PDP, and (2) four PEPs simultaneously communicating with the PDP. In the latter case, the PEPs were synchronized using the Network Time Protocol (NTP) [Mil92, Net-2].

As we would expect, for a given load (number of simultaneous requests sent to the PDP), the presence of four separate connections with the policy server in case of four PEPs results in greater signaling overhead as compared to a single PEP. However, the difference in the overhead for the two cases considerably reduces with increasing load. This is attributed to the fact that the COPS protocol uses TCP as the underlying transport protocol. For a large number of simultaneous policy requests, several COPS requests are encapsulated and sent as a single TCP segment, reducing the amount of TCP overhead per COPS request per connection.

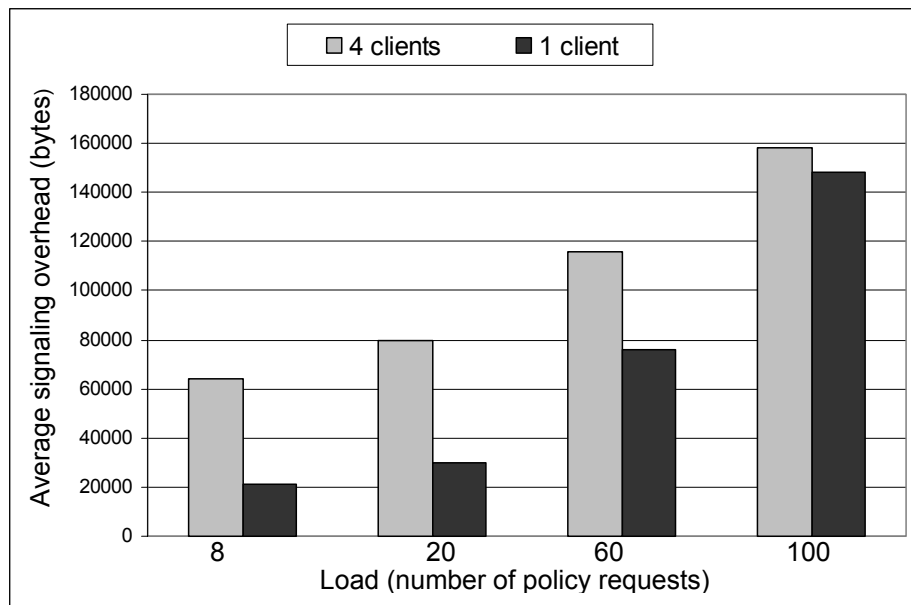


Figure 5.2 Policy signaling overhead for outsourced (CCO) architecture.

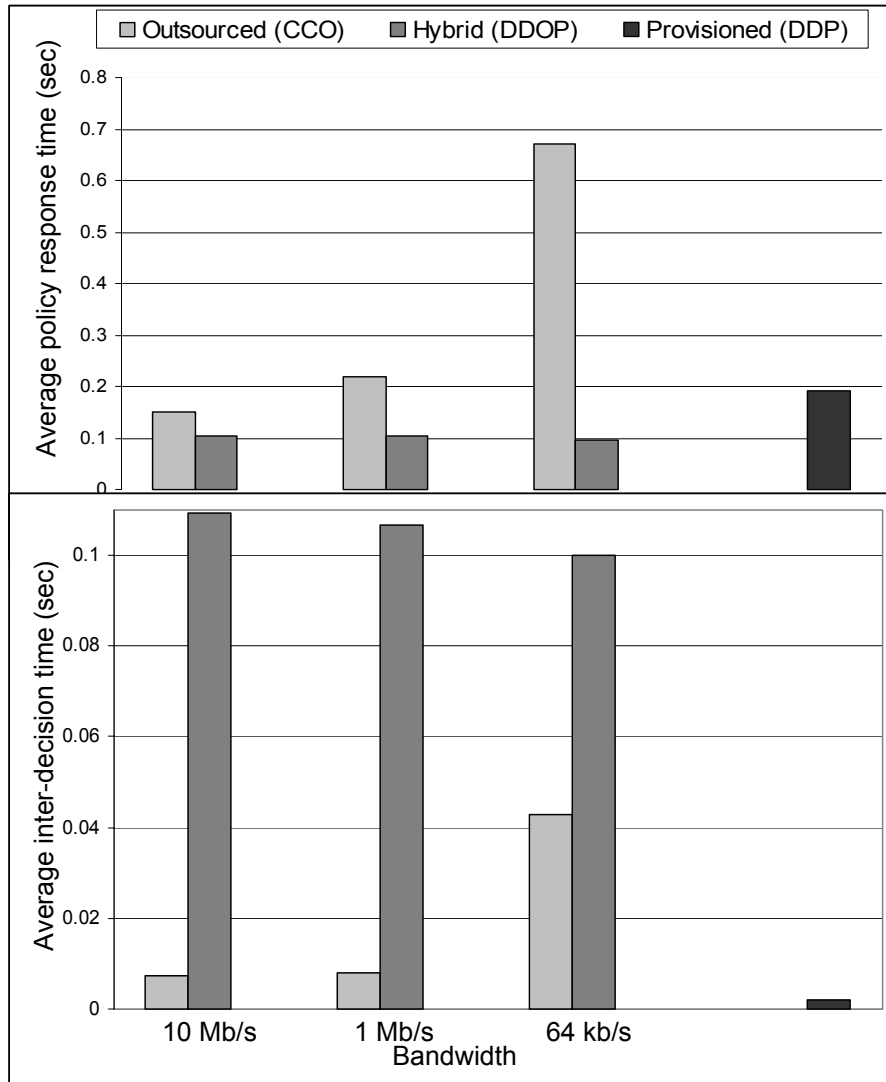


Figure 5.3 Policy response time and inter-decision time (in seconds) plotted as a function of bandwidth (Load = 25 policy requests/PEP; for the hybrid architecture about 60% of these requests are processed locally).

A second set of experiments compares the three architectures (CCO, DDP and DDOP) being considered. The metrics used for this set of experiments are: the policy response time (the difference between the time at which the policy request was sent by a PEP and the time at which the corresponding policy decision was received at the PEP); and the inter-decision time (the time difference between consecutive decisions received by the PEP). The metrics were illustrated in Chapter 3 (Figure 3.4).

Table 5.2 Policy Response Time and Inter-decision Time (for different bandwidths)

Architectures	Bandwidth	95% Confidence Interval	
		Policy response time (ms)	Inter-decision time (ms)
Outsourced (CCO)	10 Mb/s	151.0 ± 1.0464	7.4 ± 0.0016
	1 Mb/s	219.6 ± 2.412	8.1 ± 0.0025
	64 kb/s	672.0 ± 6.134	42.8 ± 0.0816
Provisioned (DDP)	-	191.9 ± 3.38	2 ± 0.0004
Hybrid (DDOP)	10 Mb/s	104.4 ± 0.0164	109.3 ± 0.0155
	1 Mb/s	103.0 ± 0.0499	106.5 ± 0.0281
	64 kb/s	97.0 ± 0.0307	100.0 ± 0.0323

Figure 5.3 shows the average policy response time and the inter-decision time as a function of the available bandwidth and for a load of 25 policy requests per PEP (i.e., in the CCO approach, the central PDP services $25 \times 4 = 100$ simultaneous requests). The constrained-bandwidth links were implemented over the wired (Ethernet) network by using a token bucket filter implementation supported by the *tc* tool [Dif]. For illustrative purposes, the hybrid architecture in these experiments was configured to have about 60% of the requests processed locally, while the remaining 40% of the requests are outsourced to the central PDP. The results were collected from multiple iterations of each experiment to achieve 95% confidence intervals. The confidence intervals were computed using the “method of batch means” [Leo93] and are shown in Table 5.2. From Figure 5.3, we note that the average policy response time per request for the CCO architecture increases considerably in the case of low bandwidth of 64 kb/s. For an outsourced approach the constrained bandwidth is the major bottleneck and in this case renders the approach impractical. The response time for the provisioned (DDP) approach is essentially independent of the available bandwidth. Even the performance of the hybrid architecture is not much affected by available bandwidth, resulting in considerably lower policy response time as compared to the outsourced case for low values of bandwidth. This illustrates the performance advantage of distribution of policy information and control for networks with constrained bandwidth.

In addition, we also recorded the average inter-decision time for the three architectures, reflecting the average service rate of the policy requests. Due to the additional processing of interacting with the remote PDP interspersed with local decision-making, the average inter-decision time is greater in case of the hybrid architecture than the CCO and DDP architectures for all three scenarios. The inter-decision time is found to be lowest for the completely distributed (DDP) case, while it increases for the centralized (CCO) case as bandwidth decreases. Thus, for the management traffic model we used, we can conclude that the additional processing required in the hybrid architecture seems to be the main bottleneck, while bandwidth is the major bottleneck for the outsourced approach.

In this section, we discussed some initial results from our experimental evaluation of the CCO, DDP and DDOP policy architectures. The hybrid category of architectures emerges as a promising candidate for low bandwidth mobile wireless networks.

5.3.2 Multi-hop Ad Hoc Network (Wired Testbed)

In an ad hoc network, the number of hops between a policy client and a policy server often changes over time. Hence, it is important to measure and analyze our metrics – policy response time and inter-decision time – as a function of the number of hops that separates a policy client from the server.

In this section, we describe our experiments involving multi-hop ad hoc network topologies. Using the Dynamic Switch we emulated multi-hop topologies; a CCO type of architecture was deployed. Intermediate nodes used the Optimized Link State Routing protocol daemon (*olsrd*) to route packets between the PDP and PEP. A low bandwidth of 64 kb/s was emulated end-to-end using the *tc* tool. We measured the policy response time and the inter-decision time for four scenarios: 1, 2, 3, and 4 hops between the PEP and PDP. Results were collected using multiple iterations for each case to obtain 95% confidence intervals. As seen from Figure 5.4, the policy response time and the inter-decision time increases with the number of hops, as expected. However, it is noteworthy that for both metrics, the increase is exponential, indicating the need for an upper bound on the number of hops between a policy client and server. These results largely motivated our proposal of a k -hop clustering algorithm (Section 6.1) that controls the number of hops between a PEP and PDP.

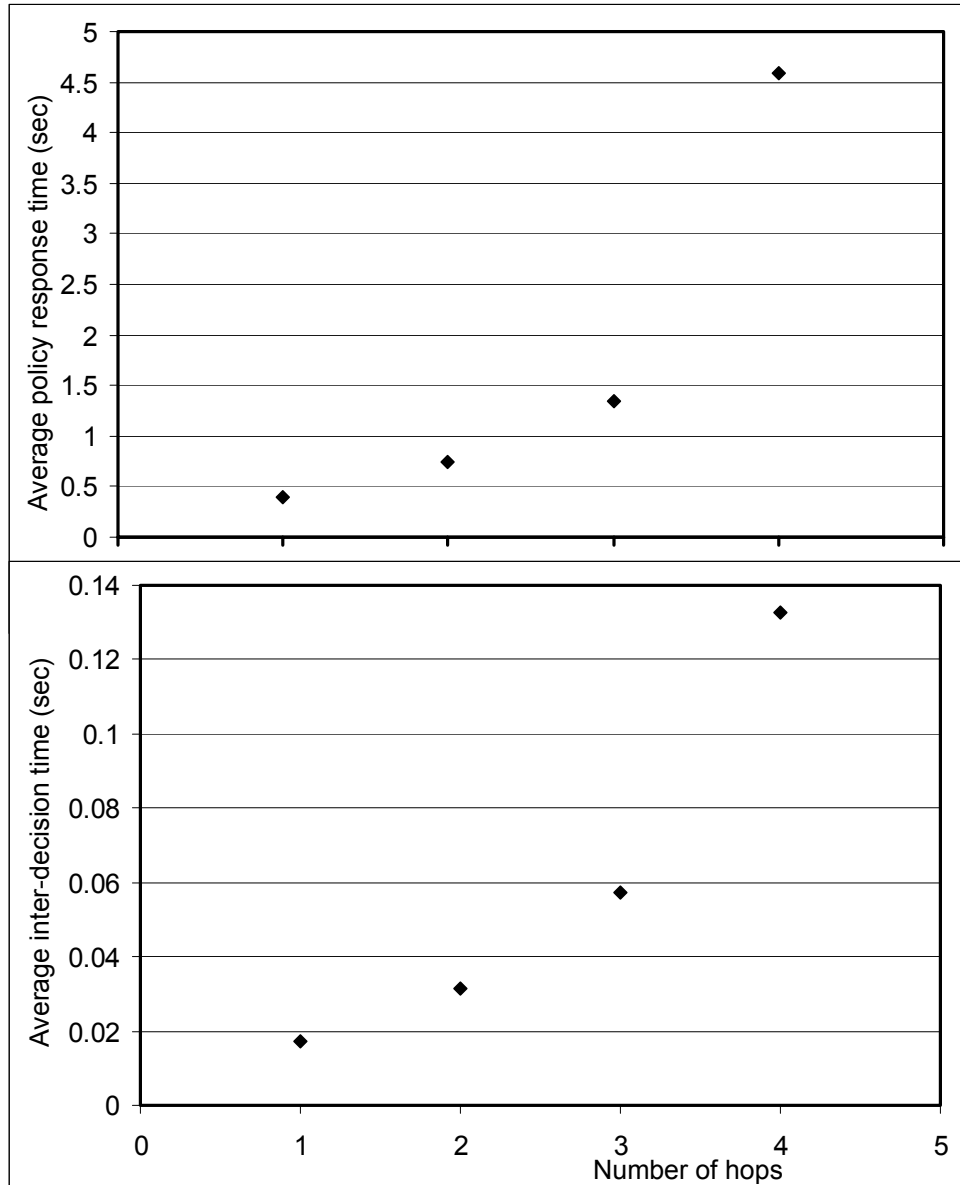


Figure 5.4 Policy response time and inter-decision time plotted as a function of the number of hops between the policy client and server (wired testbed).

The confidence intervals for the two metrics are shown in Table 5.3. For the same number of iterations in the four scenarios, we also observe increased variance for both the metrics with increase in the number of hops. This is attributed to the switched Ethernet environment (using the Dynamic Switch). Nodes cannot “sense” the transmissions from other nodes, and may transmit simultaneously. This leads to contention at the switch, which increases with increase in the number of nodes. Further, it must be noted that since the switch emulates a broadcast wireless medium, it transmits a packet received on any of its network interface onto all other interfaces.

Table 5.3 Policy Response Time and Inter-decision Time vs. Number of Hops (wired testbed)

Number of hops	95% Confidence Interval	
	Policy response time (ms)	Inter-decision time (ms)
1	399.082 ± 0.1674	17.156 ± 0.0003
2	743.591 ± 1.798	31.308 ± 0.0032
3	1340.473 ± 15.584	57.503 ± 0.0347
4	4595.428 ± 601.453	132.537 ± 0.7254

5.3.3 Multi-hop Ad Hoc Network (Wireless Testbed): Proof of Concept

In order to gain insight into performance of a policy-based management system in an actual multi-hop wireless ad hoc network, and as a proof of concept, we ported our experiments to the wireless testbed.

Again, we considered four experimental scenarios: with 1, 2, 3, and 4 hops between the policy server and client. Figure 5.5 shows the placement of laptops in our work area; a 4-hop wireless network topology is illustrated. As mentioned in Chapter 3, the laptops use IEEE 802.11b wireless cards with Intersil Prism chipset supporting transmitter power control. Reducing the transmitter power using the *Wireless Tools for Linux* [Wir] package and with the aluminum foil acting as an “attenuator”, we were able to setup the topology shown in Figure 5.5.

Below, we discuss some observations, and also describe our experiences, in general, in conducting wireless experiments.

- In our indoor work environment, the wireless links were found to be irregular and unpredictable during regular work hours (owing to factors such as people moving around, doors being opened, and use of the microwave oven). Since our motive in these experiments was to characterize the system performance over the wireless network, a more controlled environment was desirable. Hence, we conducted most of the wireless experiments either on weeknights or weekends.

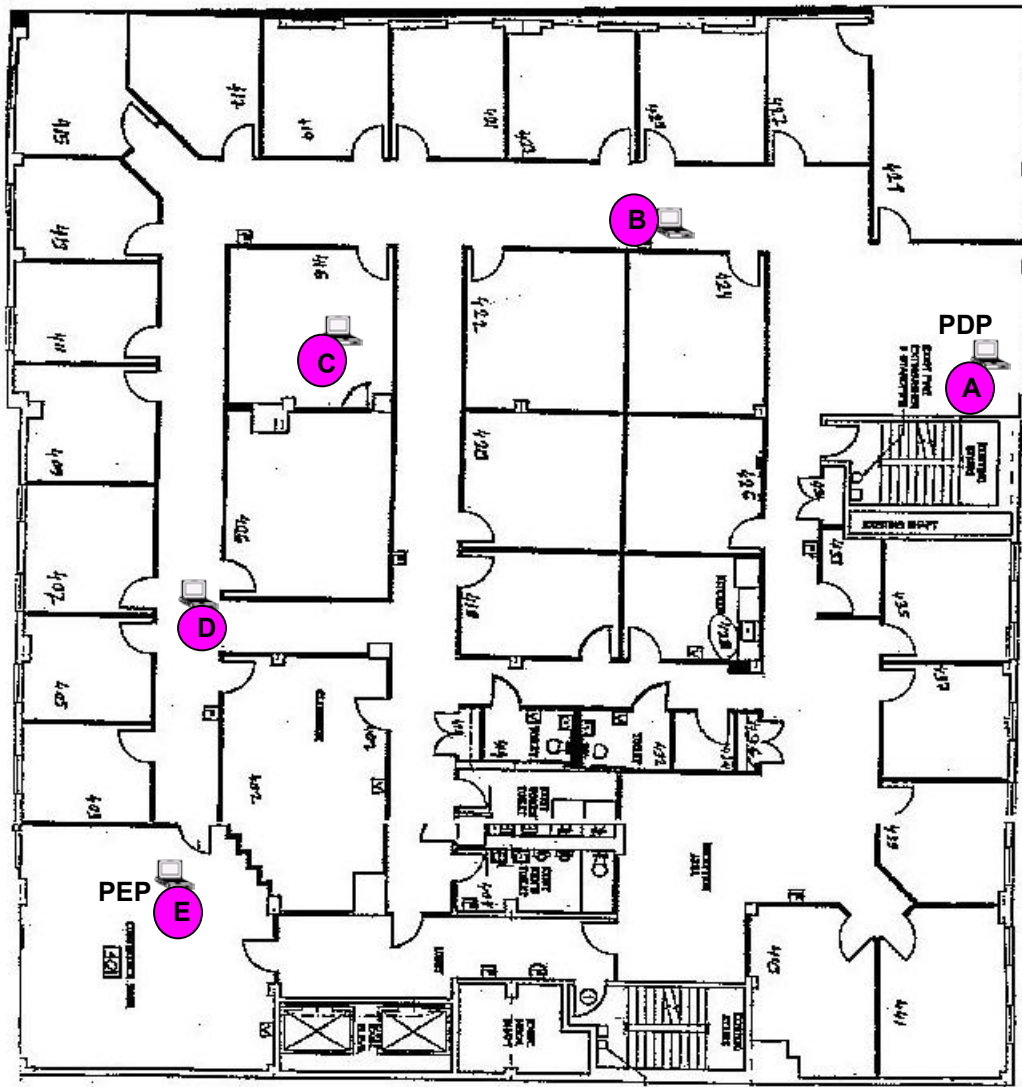


Figure 5.5 Layout of our work area; placement of nodes for a 4-hop wireless ad hoc network topology is shown. Nodes A and E are the policy server and client, respectively.

- In an otherwise static setup, we observed intermittent loss of routes between end nodes, especially with the increase in the number of hops. Clausen et al. [CHC+01] reported a similar problem when evaluating the OLSR protocol using an experimental testbed. Further investigation indicated that the transmissions of control packets by the OLSR daemon at two or more nodes became synchronized resulting in packet collisions. To alleviate this problem, we have implemented random jitter (in the range suggested in [CJ03]) by modifying the current OLSR daemon.

In addition, we found that the hidden terminal problem [Gas02] also occasionally led to temporary loss of routing packets; this is attributed to the somewhat serial placement of nodes.

- We were able to run multiple iterations in an automated fashion for 1-hop and 2-hop experiments. However, our automation mechanism broke down in case of 3-hop and 4-hop scenarios. The percentage of *successful*⁵ experiments decreased considerably from almost 100% for 1 and 2-hop scenarios to around 75% and 50% for 3-hop and 4-hop scenarios respectively. Given this unpredictability, we gathered data by running some of the 3-hop and 4-hop experiments manually.
- Results for the wireless experiments are shown in Figures 5.6 and 5.7. The general trend for the policy-response time and inter-decision time is similar to that obtained in our experiments with the wired tested – exponential increase in the metrics with increase in the number of hops. However, a very high variance in the policy response time was observed for the 3 and 4-hop scenarios, which indicates that the average values for these two cases are not a good indicator of system performance. Detailed study of the instantaneous values revealed intermittent occurrence of large spikes (e.g., as high as 5 and 15 seconds) for policy response time in the 3-hop and 4-hop scenarios, respectively. All in all, increased response times and the unpredictability of the system with increase in the number of wireless hops bolsters our proposal of using k -hop cluster management.

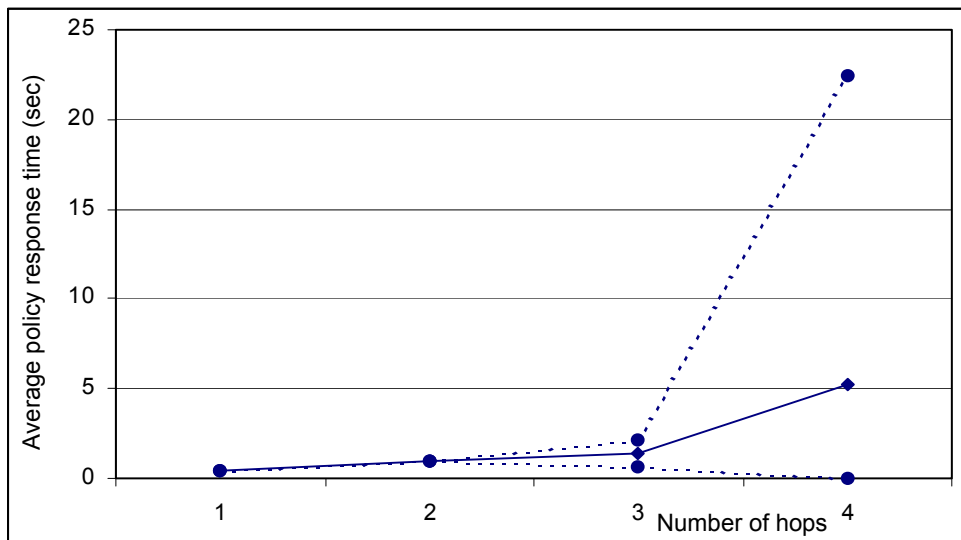


Figure 5.6 Policy response time plotted as a function of the number of hops between the policy client and server (wireless testbed); the dotted lines indicate the confidence interval.

⁵ An experiment is referred to have been successful if all the policy requests and corresponding decisions are successfully transmitted between the end-nodes.

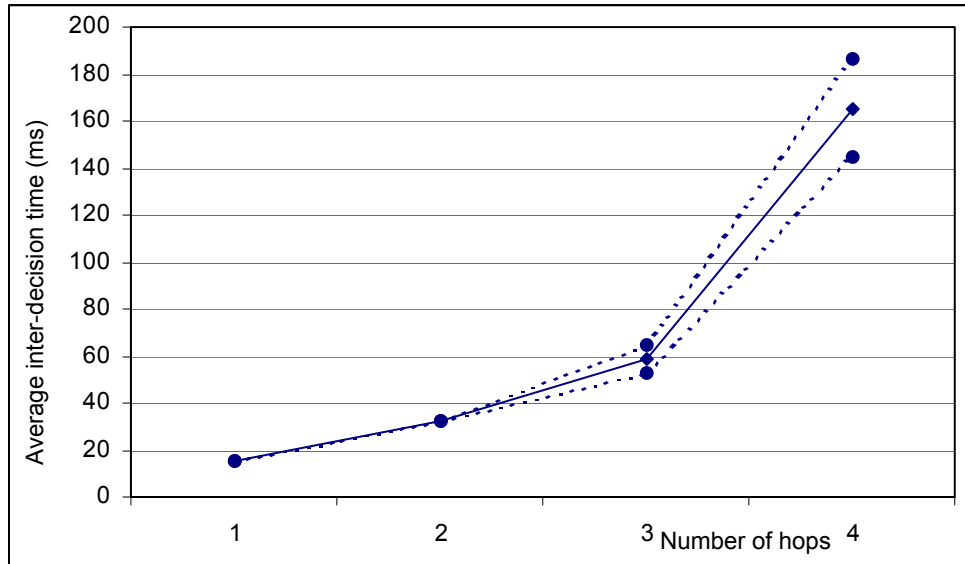


Figure 5.7 Inter-decision time plotted as a function of the number of hops between the policy client and server (wireless testbed); the dotted lines indicate the confidence intervals.

5.4 Summary

In this chapter, we proposed a characteristics-based taxonomy to classify the different policy architectures into distinct categories. The taxonomy allows better understanding of the operational characteristics of the various policy architectures and provides a common ground to compare them. This in turn facilitates the choice of the best-suited architecture for the network environment of interest.

To validate our assessment (based on the taxonomy) of certain policy architectures and to study the performance of the COPS in a multi-hop network topology we conducted experiments using our testbed network. Preliminary results indicated that the hybrid architectures are a promising choice for low bandwidth wireless networks. The experiments in multi-hop ad hoc network topologies, both using the wired and wireless testbeds, indicated an exponential increase in the policy response time and inter-decision time. In addition, we reported our experiences and general lessons learned from conducting experiments in an actual wireless ad hoc environment.

Based on our understanding of the requirements of an ad hoc network management system (outlined in Chapter 3) and the assessment made in this chapter, we present our solution suite and discuss its implementation in our testbed network and in QualNet, in the following chapter.

CATP (Caffeine Access Transport Protocol) -- Common method of moving caffeine across Wide Area Networks such as the Internet.
- Anonymous

It is what we do after we make the decision to implement and execute it that makes it a good decision.
- William Pollard

Chapter 6

Solution Suite: Protocol Support and Implementation

This chapter describes our proposed suite of solutions (Sections 6.1-6.4) – clustering, Dynamic Service Redundancy (DynaSeR), inter-domain policy negotiation, and service discovery – designed to facilitate deployment of policy-based management in mobile ad hoc networks (MANETs). The solution suite plays a major role in achieving our goal of an automated, self-organizing, efficient, and robust management system. In Section 6.5, we describe our efforts in real network implementation and prototyping of our PBNM system, and highlight the integration efforts involved in making the various applications in the framework work together. Finally, we discuss the simulation models we develop using QualNet.

6.1 Cluster Management

In our work, we propose a simple k -hop clustering scheme solely from a management viewpoint, using an approach somewhat similar to graphical clustering mentioned in [CJS99]. However, the algorithms we use to form and maintain clusters are different from those proposed in [CJS99].

Most previous works propose distributed algorithms for cluster-head election and hence for cluster maintenance to optimize certain parameters such as bandwidth efficiency, spatial frequency channel reuse, etc. However, in policy-based management, typically the governing body or network operator (e.g., a command and control center that deploys a military ad hoc network) needs to decide, *a priori*, specific nodes that will operate as policy servers. Such an approach allows the network administrator to take advantage of the centralized idea fundamental to policy-based management at the highest level of abstraction. At the same time, the network would be expected to adapt to the decentralized paradigm underlying ad hoc networking. Our cluster management and the following schemes in the solution suite provide this adaptation.

In our clustering algorithm, we assume that during initial deployment of a wireless ad hoc network, a certain number of policy servers is present in the network. Each cluster is characterized by a policy server, which acts as a cluster-head. All the policy clients within k hops from the server are eligible for service from that policy server. Each policy server along with its clients forms a cluster, as shown in Figure 6.1. The main motivation for this type of clustering is to limit the number of wireless hops between a client and a server.

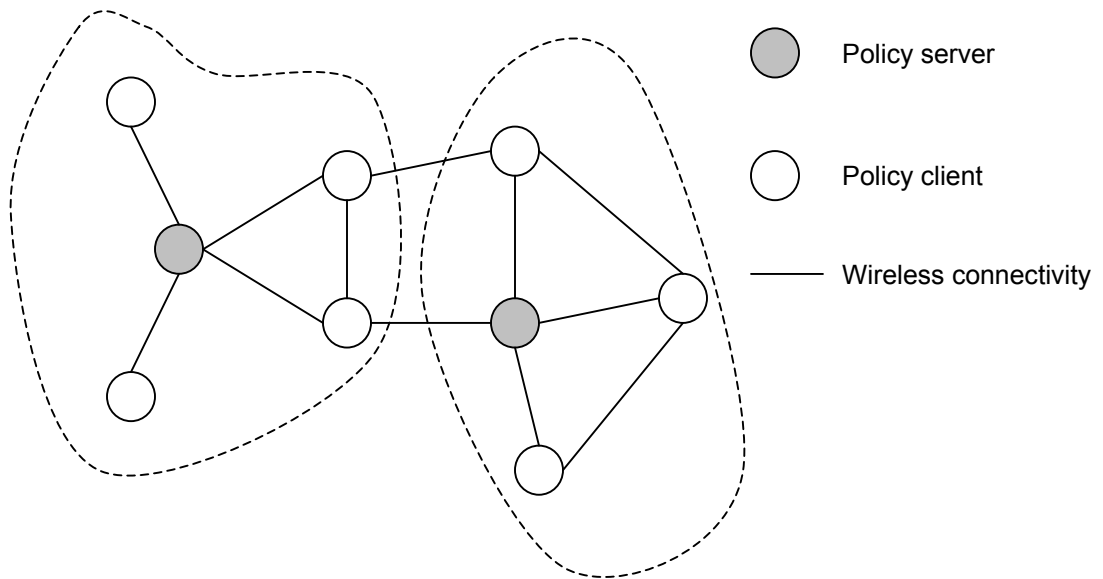


Figure 6.1 k -hop cluster management; 1-hop ($k = 1$) clusters indicated by dotted lines.

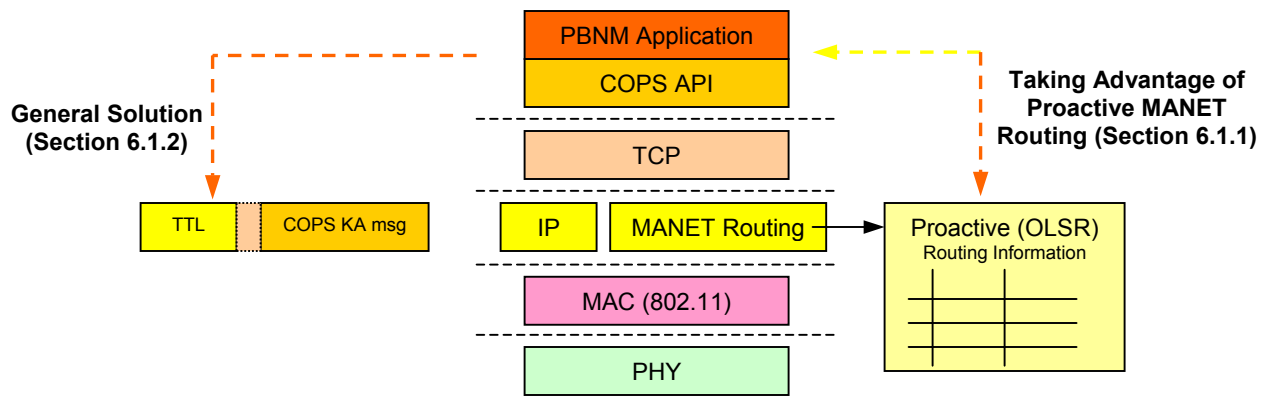


Figure 6.2 Cross-layer interaction for k -hop cluster management.

Our experimental results in Sections 5.3.2 and 5.3.3 showed that the PBNM system performance degrades considerably (increased policy response time and unpredictability) with an increase in the number of hops between a policy client and server. Furthermore, limiting the hops means fewer resources (e.g., bandwidth and battery life) are spent at intermediate nodes for relaying management messages.

Local k -hop topology information is needed for policy servers to implement k -hop cluster management. We propose two ways of achieving this efficiently via cross-layer interaction between the COPS-based PBNM application layer and the network layer. Given the resource-limited nature of ad hoc networks, our aim is to keep the management control overhead to a minimum. The proposed cluster management methods, one taking advantage of routing information available from OLSR and the other a more general method independent of the underlying routing protocol, are depicted in Figure 6.2. We discuss these next.

6.1.1 Taking Advantage of Proactive MANET Routing

If a proactive MANET routing protocol, such as the Optimized Link State Routing (OLSR) protocol [CJ03], is being used, we can take advantage of the topology information that is available locally from the routing daemon. This method allows a policy server to quickly (bound by the granularity of routing updates) and efficiently detect the movement of client nodes in and out of its k -hop cluster. Further, by querying other policy servers for specific local topology

information, a policy server can efficiently track node movements in all the k -hop clusters in the network.

In our work, we have implemented an interface between our PBNM software and two proactive MANET routing protocol daemons – OLSR and OSPF-MCDS – in a Linux-based testbed, and successfully demonstrated the operation of this method. Details of our implementation are provided in Section 6.5.1.2.

6.1.2 General Approach

While the method outlined in the earlier section allows policy servers to quickly track node movements between clusters, it assumes that a proactive MANET routing protocol is being used. As an alternative, we propose a general solution – one that does not make any assumptions about the underlying ad hoc routing protocol – to implement k -hop clustering.

A COPS server and client periodically (defined by a Keep-Alive timer) exchange Keep-Alive (KA) messages. The COPS client sends a KA message, which is then echoed back by the server, indicating the COPS connection is still alive. In k -hop clustering, a policy server maintains COPS connections only with clients within k hops. The server may advertise the value of k during *Service Advertisement* (see Section 6.4) or may choose to send it to the client during COPS connection establishment. Once a COPS connection is established, a client sets the Time-To-Live (TTL) field in the IP header of its KA message to $2k$. On receiving a KA message, if the server finds that the value in the TTL field of the received KA message is less than k , it concludes that the particular client is more than k hops from itself and closes the connection.

6.2 Dynamic Service Redundancy (DynaSeR)

It is possible that during initial network deployment, the existing number of policy servers in the network is not sufficient to serve all network nodes. In other words, one or more nodes may not be within k hops of any of the policy servers. Alternatively, during network operation, a client may move out of its current cluster and lose service from its current server. To allow the PBNM

system to adapt to network mobility, and to increase policy-based service availability, we further enhance our clustering algorithm using what we call *Dynamic Service Redundancy (DynaSeR)*. It is noteworthy that using the DynaSeR solution, we can also tackle the problem described in Section 3.1.2, namely, adapting a traditionally centralized service (policy-based management) to a decentralized paradigm (ad hoc networks). The two mechanisms – *Redirection* and *Delegation* – that comprise the DynaSeR solution are described below.

6.2.1 Redirection

A policy server may use *redirection* to help a client to *hand-off* to another appropriate server as necessary. For example, consider the case where a client moves out of a k -hop cluster. Once the client's current server detects this, it checks its topology information (if available) to see whether the client has moved into the k -hop cluster of another server. If such is the case, the server redirects the client to the “new” or “redirected” server.

The COPS standard [DBC+00] has inherent support for the redirection mechanism. An optional “Redirected PDP Address” field is defined in the COPS Client-Close message. A COPS policy server can include the address of the new server in this field and notify the client during the COPS connection closure.

This method can be termed as a “server-centric” way of implementing hand-offs. An alternative way is to allow clients to discover the near-by servers in a distributed fashion (as illustrated in Section 6.4).

6.2.2 Delegation

The concept of distributed Management by Delegation (MbD) was introduced Goldszmidt and Yemini [GY95]. The idea is to achieve decentralized management through dynamic distributed computing. MbD has been widely studied, especially with regards to SNMP monitoring. Instead of retrieving collected data from a remote location for processing, the analysis program is dispatched to where the data is.

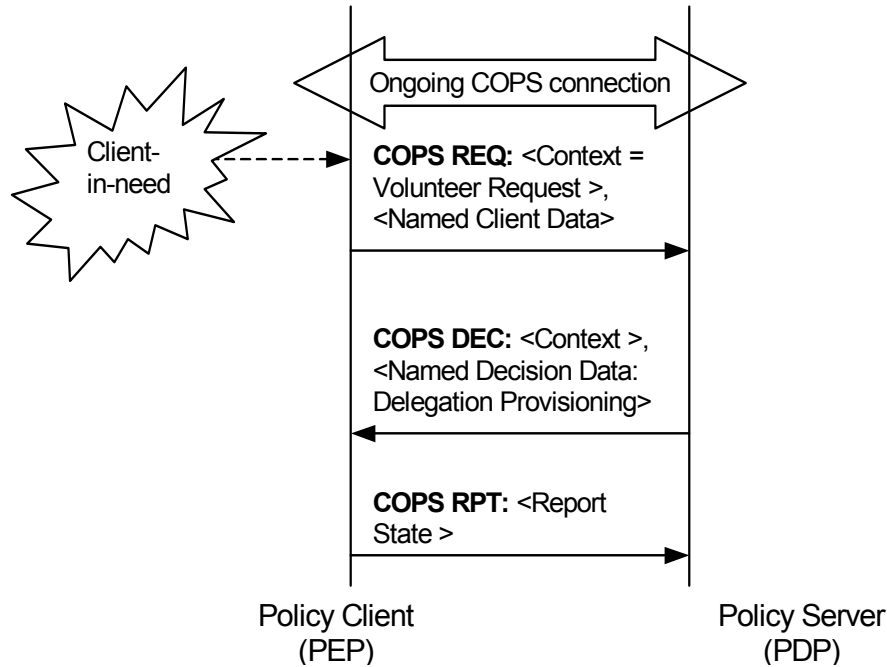


Figure 6.3 Delegation of policy-based service.

In the context of policy-based management, provisioning network nodes with relevant policies is conceptually similar: intelligence (policies) is dispatched to distributed locations (clients) so that policy-based decisions can be made locally. Such provisioned client nodes may still need to intermittently outsource policy decisions to remote servers. The likelihood of this happening in a MANET environment is even higher due to varying network conditions, nodes entering and leaving a network, and node failures. Furthermore, policies may change during network operation. For example, in a military battle-site network, the command and control center may send policy updates to select policy servers in the network. These policy servers are then responsible for disseminating policy updates in the network. To implement such policy disseminations in a scalable and efficient manner, and to dynamically increase the service coverage of the PBNM system as required, we propose to use *delegation – dynamic invocation of policy server instances*.

We propose two extensions to COPS-PR to implement delegation: a new *Context type* and a new *Named Decision Data type*. The signaling involved is shown in Figure 6.3. Whenever a client node, already being served by a policy server, detects one or more clients in need of service (for

example by means outlined in Section 6.4), it may *volunteer*⁶ to act as a server for those clients. To accomplish this, we define a new “Volunteer Request” COPS Context. The volunteering node embeds the “Volunteer Request” Context in a COPS Request message to its server. Based on the policies defined, the *parent* server sends a COPS Decision back to the volunteering client. If the delegation is approved, the parent server includes relevant policies in the Decision message using the proposed *Delegation Provisioning* Named Decision Data Type. Finally, the volunteer node sends a COPS Report message to the parent server to indicate success (the policies are installed and the service is activated) or failure of the delegation.

6.3 Policy Negotiation

The COPS protocol traditionally supports communication between a policy server or Policy Decision Point (PDP) and one or more policy clients or Policy Enforcement Points (PEPs). We propose a signaling procedure, using the COPS protocol, for inter-PDP communication and policy negotiation [PDM03]. This is of particular importance in a multi-domain network environment [KR00]. In a mobile ad hoc network formed by a consortium of different organizations, nodes may move across domains administered by policies of distinct organizations. In general, a node’s movement into a *foreign* domain may have several implications to its operation and performance. For example, from a QoS perspective, if the foreign domain does not have policies to handle a particular “visiting” node, the service guarantees enjoyed by that node may degrade considerably. Specifically, time-sensitive mission critical data and real-time audio/video applications may be rendered impractical.

We make a distinction between our proposed method and the mechanism for inter-domain policy negotiation proposed by Nguyen, *et al.* in [NBM03]. We aim to provide seamless QoS, based on some predetermined Service Level Agreements (SLAs) between the organizations, to mobile nodes moving across different administrative domains. Nguyen, *et al.* use COPS for inter-domain negotiation of Service Level Specification (SLS) mainly with a fixed Internet model in mind – policy negotiation is defined for traffic flowing across multiple domains (e.g., different ISPs), but the application end-points are in their respective *home* domains.

⁶ A node may use different parameters, e.g., remaining battery power, connectivity, and processing load, to decide whether it should volunteer. Similarly, a parent server may use different criteria to delegate to the most appropriate volunteer. The study of these criteria is outside the scope of this dissertation.

To account for mobility of nodes across domains, we define a new object called the “Home PDP Address” in the COPS protocol. The format of the “Home PDP Address” object is similar to the “Last PDP Address” object defined in the COPS protocol standard [DBC+00]. A client embeds both these objects in the COPS *OPEN* message sent to the server in the establishment of a new COPS connection.

The policy negotiation signaling is shown in Figure 6.4. When a “visiting” client (PEP_H) establishes a COPS connection with a policy server (PDP_F) in a foreign domain, the server searches for policies for that client. If it does not find any relevant policies in its Policy Information Base (PIB), it gathers the address of the client’s home domain policy server (PDP_H) from the “Home PDP Address” object. PDP_F then acts as a client (with a new “COPS Negotiation” client-type) and establishes a COPS connection with PDP_H. It then sends a COPS request (*REQ*) message to PDP_H to download policies relevant to the “visiting” client (PEP_H). PDP_F adapts its policies to reflect existing SLAs between the domains.

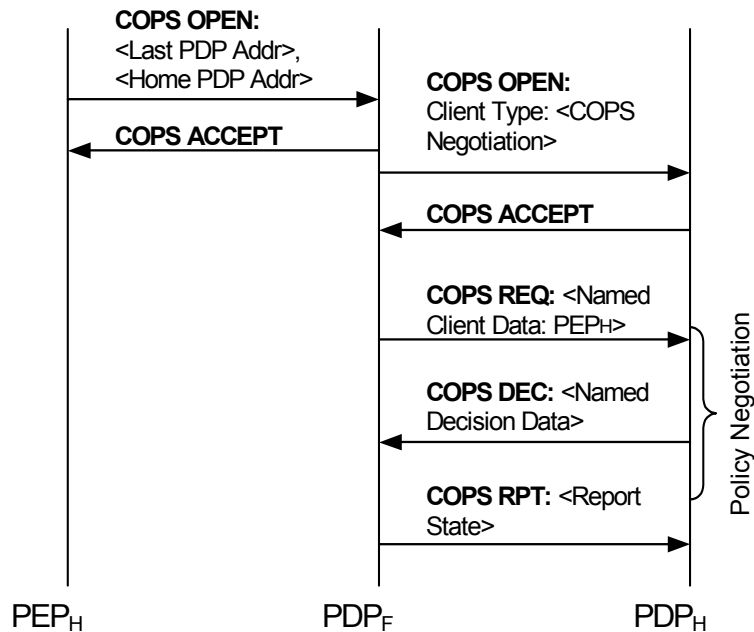


Figure 6.4 Signaling for inter-domain policy negotiation.

6.4 Service Discovery

We propose a service discovery mechanism with the primary goal of automated discovery of policy server(s) in a PBNM system in MANETs.

6.4.1 Proposed Mechanism

The service discovery mechanism we propose uses two types of messages: *Service Advertisement (SA)* and *Client Service Request (CSRQ)*. Both messages use the packet format shown in Figure 6.5.

A policy server periodically broadcasts an SA message to advertise itself. The SA message broadcast is limited to k hops by setting the Time-To-Live (TTL) value to k at the server. The TTL value is decremented by one at each hop. A client caches the address of all the policy servers for which it receives an SA message. The client chooses a server and sends a COPS Client-Open message to establish a COPS connection. The client cache expires every few time units.

A client that does not receive an SA message within a certain time interval broadcasts a CSRQ message. To begin with, the CSRQ broadcast is limited to k hops. If a server (that may have moved within k hops of the client) receives the CSRQ message, it responds with a unicast SA message. On receiving the SA message, the client can then establish a COPS connection. Alternatively, a client node in the k -hop neighborhood of the “client-in-need” *volunteers* to act as a *delegated server* and follows the signaling protocol outlined in Section 6.2.2. If delegation is successful, the delegated server serves the client-in-need. If the client-in-need does not receive any response within a certain time interval it may choose to increase the hop-length of its CSRQ broadcast.

SA message broadcasts from two or more servers that are operating in the vicinity (within $2k$ hops) of one another and are even somewhat time synchronized may experience collisions. In fact, this was observed in the pre-data collection phase of our simulations, wherein a sudden flurry of SA messages from different clusters led to the messages getting lost at nodes within k hops of multiple servers. Hence, we suggest and implement the interval T between consecutive SA messages for a server to be randomly chosen from a range $[T_{min}, T_{max}]$.

Originator Address	Message Type	Cluster Size	TTL
-----------------------	-----------------	-----------------	-----

Figure 6.5 Service discovery message format.

6.4.2 Time-Based Heuristic to Minimize Broadcast Overhead

In MANETs, the problem of blind broadcast flooding is well known [TNC+02], and has been studied mainly with regards to ad hoc routing protocols. Several solutions have been proposed to minimize redundant broadcasts. Minimizing service discovery broadcast overhead is extremely important for low bandwidth and low energy MANETs and sensor networks.

One popular category of solutions to the blind-broadcast problem uses the concept of Minimal Connected Dominating Set (MCDS) [LMP03, Wes01]. These solutions assume the knowledge of global or local topology to compute the MCDS. In an actual MANET, topology information is typically gathered by exchanging routing information. Secondly, computation of MCDS is a NP-hard problem. Hence, most solutions use some approximation to arrive at a quasi-optimal solution. Even in such cases, MCDS computation can be a major bottleneck in large networks.

We are interested in finding a solution that is completely distributed, does not assume availability of topology information, requires minimal computation, and involves no extra control overhead. Given the periodic nature of the SA and CSRQ message broadcasts, we propose a heuristic time-based mechanism to reduce unnecessary broadcasts of these messages. When a node broadcasts a message (SA or CSRQ), two cases may occur:

- (i) A node receives one or more re-broadcasts of its own message, in which case it disregards the message and does not re-broadcast it.
- (ii) A node receives a broadcast message originated by another node (originator address). In this case, for each originator address, the node re-broadcasts the message and sets a timer with expiration interval τ ($0 < \tau \leq T$, where T is the interval between consecutive message transmissions of the same type – SA or CSRQ). It does not re-broadcast any message (of a given type) from that originator address until the timer expires. As mentioned earlier, since T is chosen from a range $[T_{min}, T_{max}]$, τ can be set equal to T_{min} .

Although this method does not provide an optimal solution, it considerably reduces broadcast overhead (as will be seen from the simulation results in Section 7.1.3), especially for larger cluster sizes (k).

6.5 Implementation

In the earlier sections, we described our proposed schemes to deploy a PBNM system in a MANET environment. In this section, we describe our prototype implementation of the various schemes in a testbed network as well as simulation models developed using the QualNet network simulator.

6.5.1 Prototype Implementation

6.5.1.1 Policy-Based Management Application

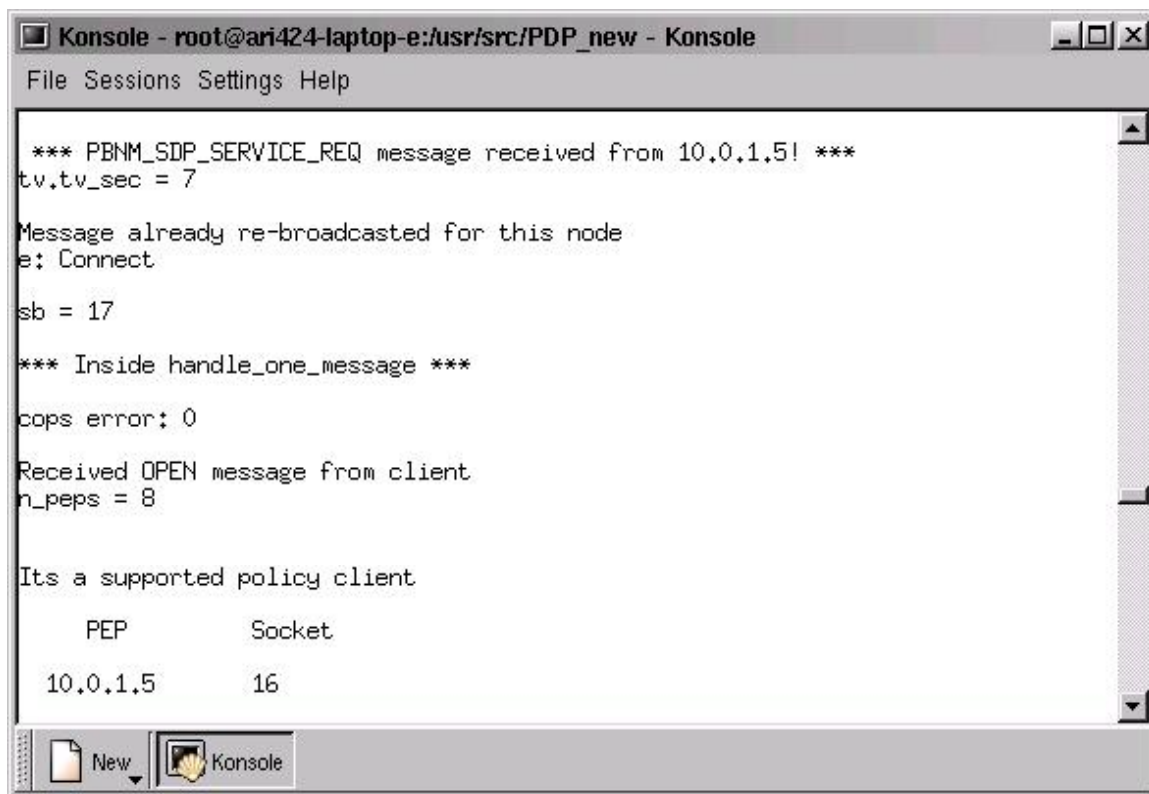
We implemented our policy-based management application using the COPS API made available by Telia Research. The COPS API implements a combination of asynchronous I/O and POSIX multi-threaded techniques for the policy server. A pool of threads is defined during initialization. A *select()* function call [CS00] is run by one thread in the pool that listens on all connected sockets for arriving data and for new incoming connections. The use of multiple threads allows the implementation to take advantage of multi-processors whenever possible. Also, the combination of multiplexing and multi-threaded techniques tends to yield higher efficiency by reducing the thread-switching overhead present in a purely threaded approach. The disadvantage of this approach, however, is increased implementation complexity.

Since some of the work at LTU was done before COPS and COPS-PR were standardized by the Internet Engineering Task Force (IETF) [Int-1], we modified their implementation to better conform to current standards. We also carried out interoperability tests using the LTU implementation and the Intel COPS SDK. This helped us fine-tune our implementation, and also to discover and report some significant bugs in the Intel COPS SDK implementation (version 3.1), e.g., missing “Context” object in the COPS-PR Request (*REQ*) message, and incorrect format of the “Last PDP Address” object in the COPS OPEN (*OPN*) message.

Our current implementation of the policy server and client includes support for our proposed schemes described in Sections 6.1-6.4. A snapshot of the policy server and client user interface is shown in Figures 6.6 and 6.7 respectively. The operation of the distributed service discovery

mechanism allowing a client to discover a policy server in an automated fashion can be seen from these figures. At initialization, a user can configure the various client and server settings via a configuration file. The current implementation of the policy server allows the user to control the policy clients to be supported by the server. Since our analysis involves both wired and wireless experiments, we also let the user notify the server about the specific type of network interface to be used in an experiment. This is important for the server's interaction with the *Diffserv on Linux* tool used for Quality of Service (QoS) provisioning; it determines the network interface on which bandwidth management is to be deployed. The server and client programs also read other system parameters such as the value of k or the maximum cluster size in our k -hop clustering algorithm, and timer values for service discovery and COPS KA messages, from the configuration file.

Both the client and server implementations use distinct threads to carry out different tasks such as handling the TCP socket, handling UDP socket for service discovery, running the COPS keep-alive timer, implementing k -hop cluster management and implementing the time-based heuristic to reduce service discovery overhead.



```
Konsole - root@ari424-laptop-e:/usr/src/PDP_new - Konsole
File Sessions Settings Help

*** PBNM_SDP_SERVICE_REQ message received from 10.0.1.5! ***
tv.tv_sec = 7

Message already re-broadcasted for this node
e: Connect

sb = 17

*** Inside handle_one_message ***

cops error: 0

Received OPEN message from client
n_peps = 8

Its a supported policy client

      PEP      Socket
10.0.1.5      16
```

Figure 6.6 Policy server user-interface.

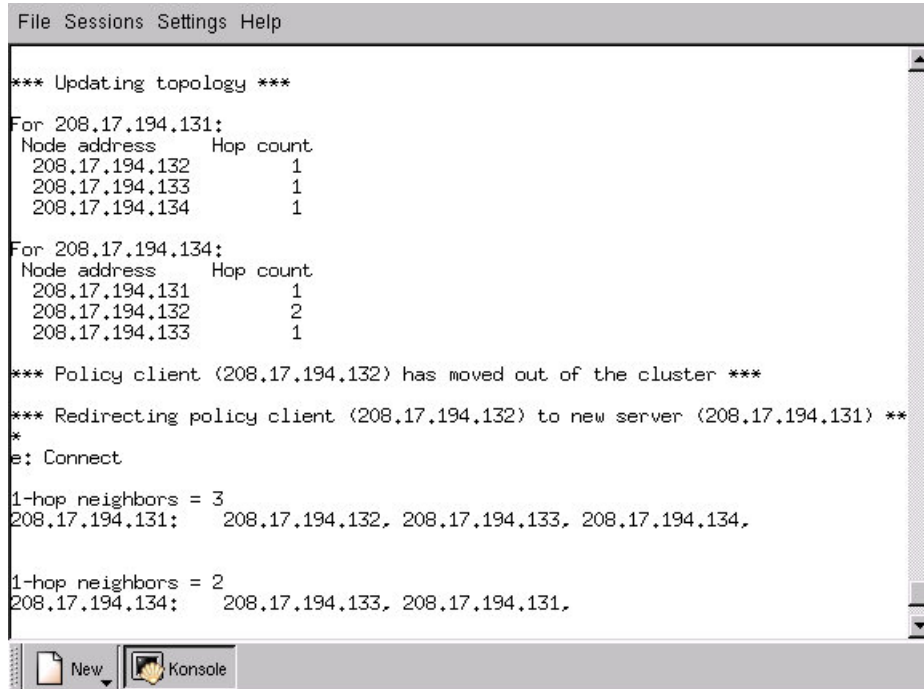
```
root@ari424-laptop-b:/usr/src/PDP_new/COPS_API/src
File Edit Settings Help
*** PBNM_SDP_SA message received from 10.0.1.1! ***
CONNECTED = 0
hostname: 10.0.1.1
Inside handle_connection
Sending OPEN message...
cops_error = 0
Inside parse_message
m->op_code = 7
Originator Address
10.0.1.1
ACCEPT message received from server
Connection successfully established!
*** Rebroadcasting SDP message ***
Originator address: 10.0.1.1
TTL = 1
Type = 0
K = 2
*****
```

Figure 6.7 Policy client user-interface.

6.5.1.2 Integration with Ad Hoc Routing Protocols

We interfaced our PBNM software with two proactive MANET routing protocols: OLSR and OSPF-MCDS, and demonstrated its operation on top of these routing protocols. As mentioned in Chapter 3, the *olsrquery* tool, a part of the INRIA OLSR implementation [Opt], allows a user to access the OLSR routing table maintained at the various network nodes. We modified the *olsrquery* tool to direct its output to a text file in a desired format. When a policy server is initialized, a separate thread is created and dedicated to interact with the underlying *olsrd* routing daemon. The thread periodically calls a function to execute the *olsrquery* command that generates an output file containing the required topology information (routing information gathered from the policy servers in the network). The policy server then stores (or updates) the information from the file in a linked list, and uses it for cluster management. Figure 6.8 shows a snapshot of a

policy server exhibiting updated topology information maintained by it. A similar architecture was implemented to interact with the OSPF-MCDS routing daemon.



```
File Sessions Settings Help

*** Updating topology ***

For 208.17.194.131:
Node address Hop count
208.17.194.132 1
208.17.194.133 1
208.17.194.134 1

For 208.17.194.134:
Node address Hop count
208.17.194.131 1
208.17.194.132 2
208.17.194.133 1

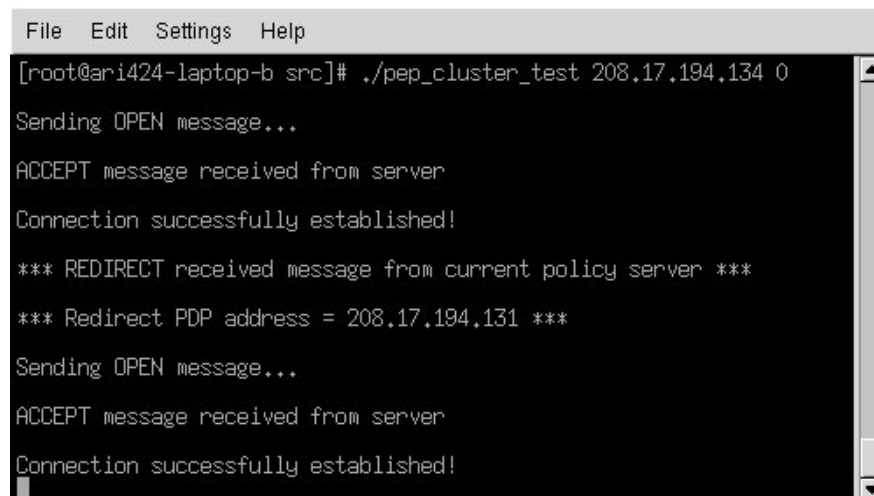
*** Policy client (208.17.194.132) has moved out of the cluster ***

*** Redirecting policy client (208.17.194.132) to new server (208.17.194.131) **
*
e: Connect

1-hop neighbors = 3
208.17.194.131: 208.17.194.132, 208.17.194.133, 208.17.194.134,

1-hop neighbors = 2
208.17.194.134: 208.17.194.133, 208.17.194.131,
```

Figure 6.8 User-interface of a policy server showing topology information gathered from underlying OLSR routing daemon, and implementation of 1-hop cluster management.



```
File Edit Settings Help

[root@ari424-laptop-b src]# ./pep_cluster_test 208.17.194.134 0

Sending OPEN message...
ACCEPT message received from server
Connection successfully established!

*** REDIRECT received message from current policy server ***

*** Redirect PDP address = 208.17.194.131 ***

Sending OPEN message...
ACCEPT message received from server
Connection successfully established!
```

Figure 6.9 User-interface of the policy client; as the client moves out of the 1-hop cluster of its original server, the client is redirected to another policy server.

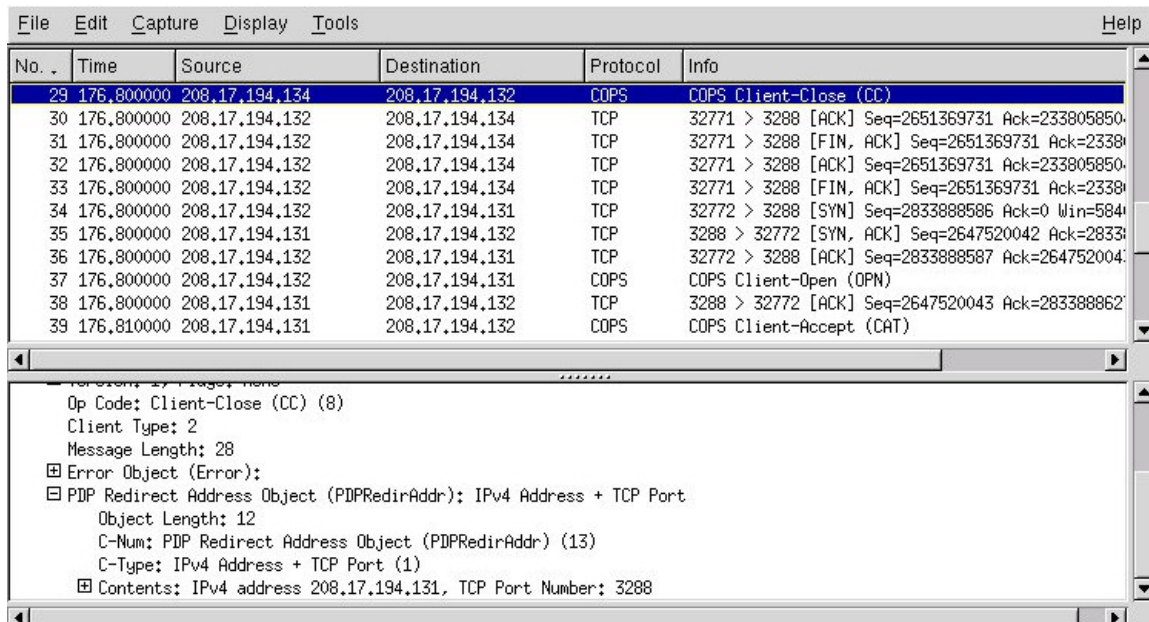


Figure 6.10 Redirection of a policy client (208.17.194.132) by one policy server (208.17.194.134) to another policy server (208.17.194.131), as a part of k -hop cluster management.

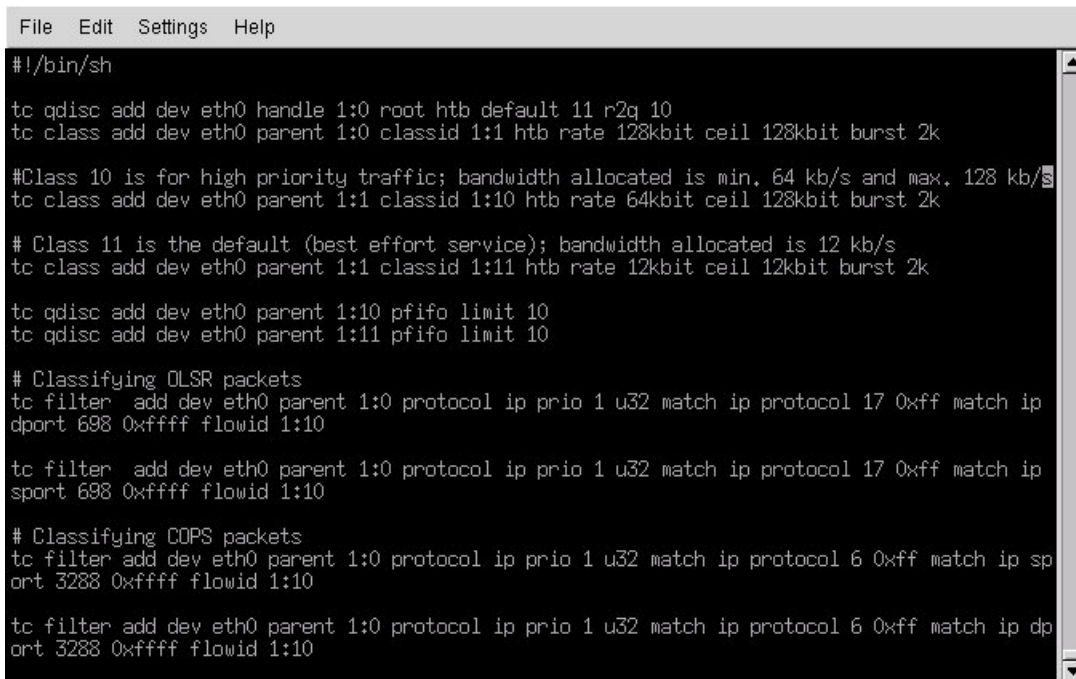
By using a dedicated thread, we have segregated the functions for obtaining topology information and the ones for cluster management. Only if there is a topology change of significance, i.e., one or more clients moving out of the k -hop cluster, the policy server is alerted about it, and functions for accessing the topology information and hence for cluster management are invoked.

As described in Section 6.1.1, the topology information obtained from the routing daemon is used to implement cluster management. The user interface of a policy client exhibiting the *redirection* technique is shown in Figure 6.9. The corresponding COPS-based signaling, captured using Ethereal [Eth], is shown in Figure 6.10.

6.5.1.3 Quality of Service Mechanisms

In our experiments, we use the *Diffserv on Linux* or *tc* tool for Quality of Service (QoS) provisioning. Among the various scheduling techniques available in the *tc* tool, we found the Hierarchical Token Bucket (HTB) [Hie] packet scheduler to be the most useful for our purposes. The HTB scheduler, a variant of the Class-Based Queuing (CBQ) scheduler [SV95], allows us to do traffic shaping (for low bandwidth emulation) in its *parent* class, and then to define

hierarchical *child* classes for bandwidth allocation and management. We use shell scripting to execute the various *tc* commands. An example script is shown in Figure 6.11.



```
File Edit Settings Help
#!/bin/sh

tc qdisc add dev eth0 handle 1:0 root htb default 11 r2q 10
tc class add dev eth0 parent 1:0 classid 1:1 htb rate 128kbit ceil 128kbit burst 2k

#Class 10 is for high priority traffic; bandwidth allocated is min. 64 kb/s and max. 128 kb/s
tc class add dev eth0 parent 1:1 classid 1:10 htb rate 64kbit ceil 128kbit burst 2k

# Class 11 is the default (best effort service); bandwidth allocated is 12 kb/s
tc class add dev eth0 parent 1:1 classid 1:11 htb rate 12kbit ceil 12kbit burst 2k

tc qdisc add dev eth0 parent 1:10 pfifo limit 10
tc qdisc add dev eth0 parent 1:11 pfifo limit 10

# Classifying OLSR packets
tc filter add dev eth0 parent 1:0 protocol ip prio 1 u32 match ip protocol 17 0xff match ip
dport 698 0xffff flowid 1:10

tc filter add dev eth0 parent 1:0 protocol ip prio 1 u32 match ip protocol 17 0xff match ip
sport 698 0xffff flowid 1:10

# Classifying COPS packets
tc filter add dev eth0 parent 1:0 protocol ip prio 1 u32 match ip protocol 6 0xff match ip sp
ort 3288 0xffff flowid 1:10

tc filter add dev eth0 parent 1:0 protocol ip prio 1 u32 match ip protocol 6 0xff match ip dp
ort 3288 0xffff flowid 1:10
```

Figure 6.11 Sample shell script executing commands for the *DiffServ on Linux* tool.

6.5.2 Simulation Models

One of the contributions of this research is the simulation models we developed using QualNet. While QualNet provided support for the underlying protocol stack, it did not have any pre-existing models of interest for policy-based management at the application layer. We implemented the COPS/COPS-PR protocol model as well as protocol support for our solution suite.

6.5.2.1 COPS and COPS-PR

We implemented most of the COPS and COPS-PR protocol functionalities conformant with the IETF standards [CSD+01, DBC+00]. The highlights of this implementation are as follows.

- Support for the different COPS Message Content types: Client-Open (OPN), Client-Accept (CAT), Client-Close (CC), Request (REQ), Decision (DEC), Keep-Alive (KA), Report State (RPT), and Synchronize State (SSQ, and SSC).
- COPS Specific Object Formats such as Handle, Reason, Decision, Client Specific Information (ClientSI), Keep-Alive Timer, PEP Identification (PEPID), Report-Type, and PDP Redirect Address.
- Common COPS signaling and operation as specified in [DBC+00].
- Support for outsourcing and configuration operations.
- Support for select COPS-PR Objects and COPS-PR Client Specific Data Formats (Named Decision Data and ClientSI Request Data).
- Proposed extensions to COPS-PR to implement Volunteer Request Context Type and Delegation Provisioning Named Decision Data Type.
- Interaction with the underlying TCP protocol model in QualNet to handle COPS connections.

6.5.2.2 Solution Suite

In addition to the COPS/COPS-PR protocol model, we implemented our PBNM suite of solutions proposed in the earlier parts of this chapter.

The k -hop cluster management was implemented to incorporate the general method outlined in Section 6.1.2. The existing IP protocol module in QualNet was modified to allow cross-layer interaction between the COPS protocol and IP. At the client side, this was used in setting the appropriate TTL value (equal to $2k$) for the COPS Keep-Alive (KA) messages. At the server side, the interface was used to pass the TTL value from the IP header of the received COPS KA message to the application layer.

The service discovery mechanism including the time-based heuristic algorithm was also implemented. One of the goals was to run and compare simulations with and without this algorithm.

6.5.2.3 Model Details

Three core QualNet program files that comprise our PBNM simulation model are as follows:

- *pbnm_adhoc.c* combines all the functions used in our model, which includes COPS/COPS-PR protocol functionalities along with our proposed extensions, *k*-hop clustering, and service discovery.
- *cops.h* includes all the functions, structures and variable definitions that comprise the COPS/COPS-PR protocol model.
- *pbnm_adhoc.h* combines all other functions, structures and variable definitions used in our simulation model. It includes the structure *struct_pbnm_adhoc_str*, shown in Figure 6.12, which maintains all the information for a node in a simulation.

```

struct struct_pbnm_adhoc_str
{
    int state;                //FSM state
    int  connectionId;       //TCP connection ID
    int  uniqueId;          //Client requesting to establish TCP connection
    short sourcePort;       //TCP port
    int  protocol;          //Application layer protocol ID
    clocktype sessionStart; //Time at which COPS connection is established
    clocktype sessionFinish; //Time at which COPS connection is
    BOOL  sessionIsClosed;  //Flag to indicate if COPS connection is ongoing
    NodeAddress clientAddr; //Policy client address
    NodeAddress serverAddr; //Policy server address

    int cluster_size;        //Value of k in k-hop clustering
    double PBNM_SDP_numBytesSent; //Number of service discovery bytes sent
    double PBNM_SDP_numBytesRecvd; //Number of service discovery bytes recvd
    double COPS_numBytesSent; //Number of COPS bytes sent
    double COPS_numBytesRecvd; //Number of COPS bytes recvd

    BOOL PBNM_ADHOC_SERVER; //Flag to indicate if a node is a PBNM server
    BOOL DELEGATED_SERVER; //Flag to indicate if a node is a delegated server
    int KA_wait_time; //Max. KA message interval
    PBNM_SDP_INFO *addr_list; /* List of node addresses used for the time-based
                               heuristic */

    /* Timers to implement the various timing functionalities as self-interrupts */
    Message *PBNM_TimerMsg[5]; /*PBNM_TimerMsg[0]: COPS KA msg timer
                               [1]: connection timeout timer
                               [2]: PBNM_SDP timer
                               [3]: statistics timer
                               [4]: service discovery message forwarding timer */

    /* Statistics */
    clocktype serviceAvailability; //service availability
    clocktype SimTime; //To store current time
    clocktype COPS_Connection_Timer; //To compute the timer value for COPS KA
    int numCOPSconnections; //Number of COPS connection established
    int sessionTimeout; //Number of COPS connections timedout
    BOOL COLLECT_DATA; //To indicate when to start collecting statistics
};

```

Figure 6.12 Snapshot of the dataPtr structure (of type *struct_pbnm_adhoc_str*) that maintains all the information for each node in a simulation.

To incorporate an application layer model into QualNet, few other files need to be handled. These are:

- *application.c* and *application.h* play a key role of interfacing all the application models in QualNet to the user environment. They provide an interface between the simulator and the configuration files used to define the various protocol and other network settings for a simulation scenario. For example, the functions (*APP_Initialize()*, *APP_InitializeApplication()*) defined in the *application.c* file help read the *.APP* file that stores all the information (input by the user) relevant to the applications being used in a simulation. For example, in our simulations we define the nodes that act as clients and servers, the cluster size (*k*), the COPS KA message interval, and the service discovery interval in a *.APP* file. This information is then passed onto to the appropriate protocol models (e.g., in our case to *pbnm_adhoc.c*) within the simulator. Also, during a simulation all the events related to the application layer are handled (*APP_ProcessEvent()*) by the *application* module.
- *app_util.c* and *app_util.h* define the functions to let an application layer protocol to *register* itself (*APP_RegisterNewApp()*) with the simulator and to support interaction between the application layer and the underlying transport layer. For example, it has functions that allow the COPS protocol to indicate to the TCP layer whenever a connection needs to be established or closed (e.g., *APP_TcpOpenConnection()*, *APP_TcpCloseConnection()*) and allow data delivery (e.g., *APP_TcpSendData()*) from the application layer.

It is noteworthy that our implementation of the PBNM system simulation model is somewhat unconventional as compared to existing application models in QualNet. All existing QualNet applications such as FTP, HTTP, CBR, Telnet, etc., require the user to stipulate beforehand the specific nodes that act as a client and server for each application session. In other words, the mapping between an application session and its end-points (source and destination) is predetermined. This mode of operation is inappropriate for a MANET management system, wherein we require a client to be able to discover a server and establish a COPS connection with it *on the fly*. Furthermore, during network operation a client may receive service from different servers at different times. QualNet currently (upto version 3.6) does not have in-built support for configuring such *global* applications using the *.APP* file. Hence, we had to adopt a slightly different approach in defining the simulation scenario settings for our application (client nodes, server nodes, and other parameters such as COPS KA timer and cluster size) in the *.APP* file. Typical *.APP* file formats for our PBNM application and other QualNet applications are shown in Figure 6.13 below.

```

PBNM_MAX_CLUSTER <Max. cluster size>
PBNM_COPS_KATIMER 1 <COPS KA timer value in seconds>
PBNM_SDP_SERVER_TIMER 1 <Service advertise timer value in
seconds>
PBNM_SDP_CLIENT_TIMER 1 <Service request timer value in seconds>
PBNM_ADHOC_SERVER <Server nodeID>
PBNM_ADHOC_SERVER <Server nodeID>
.
.
PBNM_ADHOC_SERVER <Server nodeID>
PBNM_ADHOC_CLIENT <Client nodeID>
PBNM_ADHOC_CLIENT <Client nodeID>
.
.
PBNM_ADHOC_CLIENT <Client nodeID>

```

(a)

```

FTP <Client nodeID> <Server nodeID> <Number of packets> <start time>

```

(b)

Figure 6.13 Snapshot of the format used to define the (a) PBNM application as compared to (b) existing QualNet applications such as FTP.

6.6 Summary

In this chapter, we presented our solution suite – *k*-hop clustering, Dynamic Service Redundancy (DynaSeR), automated service discovery, and policy negotiation – that facilitates deployment of policy-based management in MANETs. We described the design and implementation of the various schemes involved. Our realization efforts include a prototype PBNM client-server software as well as a simulation model of the PBNM system we implemented in QualNet.

Working with the Intel COPS SDK during preliminary experiments (Chapter 5) helped us gain useful practical experience. Using the COPS API made available by Telia Research, we implemented our policy server and client; interoperability tests with the Intel COPS SDK increased the confidence in our implementation.

Integration with two proactive routing protocols: OLSR and OSPF-MCDS, allowed us to demonstrate *k*-hop cluster management using the topology information made available by these

protocols. Two methods were presented to allow mobile policy clients obtain service from different policy servers as the clients move across different clusters – a server-centric method using redirection and a distributed scheme using the proposed service discovery mechanism. Extensions to the COPS-PR were proposed to implement delegation – dynamic invocation of policy server instances as and when required. We addressed the issue of clients moving across different administrative domains in a multi-domain ad hoc network, and presented a new COPS-PR based policy negotiation method to provide seamless Quality of Service (QoS) to such clients. We interfaced our policy-based management software with the *Diffserv on Linux* or *tc* tool for QoS provisioning; scripts were used to support enforcement of QoS policies.

Finally, we described the simulation models we developed to implement our PBNM system using QualNet network simulator. Our models include implementation of the COPS and COPS-PR protocols along with our proposed suite of solutions.

Our prototype implementation serves as a proof of concept to verify the operation of the PBNM framework in a real network testbed. To address scalability of the framework and to study its behavior under different ad hoc networking conditions we simulated the entire PBNM system in the QualNet network simulator.

In the next chapter, we present a comprehensive performance evaluation of the policy-based management system using simulations. In addition, we describe our experimental results obtained by emulating the Random Waypoint model using the Dynamic Switch on our testbed network and compare them with our simulation results. Finally, we demonstrate policy negotiation and QoS provisioning in a multi-domain ad hoc network using our testbed. We use the Video Conferencing (VIC) tool to conduct subjective analysis of the quality of service as perceived by the end-user. In addition, we use the real-time application and middleware, implemented by researchers at Virginia Tech, for quantitative assessment of real-time mission-critical applications.

Measurements are not to provide numbers but insight.

- Ingrid Bucher

Chapter 7

Performance Evaluation

This research, to our knowledge, is the first attempt to study policy-based management in the context of mobile ad hoc networks. Hence, our objective was first to delineate methods for adapting the policy-based approach to manage ad hoc networks, and, second, to conduct a comprehensive study of a PBNM system in an ad hoc network environment – assess the effectiveness of our proposed solutions and provide directions for deploying such systems in general.

In the previous chapters, we presented a framework (Chapter 3) outlining the important components that should comprise a policy-based ad hoc network management system. Our preliminary experiments (Chapter 5) using the wired and wireless testbeds helped us identify some of the challenges in meeting our goal of designing a self-organizing, robust, and efficient PBNM system. These initial results, along with our qualitative assessment of the problem, led to our proposed solutions (Chapter 6) for achieving this goal.

In this chapter, we assess the impact of our proposed schemes in provisioning and managing mobile ad hoc networks. We characterize the PBNM system behavior under different network conditions, e.g., cluster size (k), node mobility, and network density, and gain insight into the trade-offs involved. We hope that our assessment will help network designers and administrators in making important design choices for deploying *policy-based MANETs* in the future.

The rest of this chapter is organized as follows. In Section 7.1, we present the simulation environment used in our study. We then discuss our simulation results in Section 7.2. We

describe our experimental testbed set-up in Section 7.3 and compare our experimental and simulation results. Finally, we demonstrate the application of our PBNM system for dynamic QoS provisioning and management in a multi-domain ad hoc network with two illustrations in Section 7.4.

7.1 Simulation Environment

The simulation environment used in this study consisted of a 1000 x 1000 sq. meter flat area. All simulations used the 802.11b MAC layer, while Optimized Link State Routing (OLSR) was used for MANET routing. The 802.11b radios are modeled after the 802.11b WaveLAN radios, which support a data rate of 11 Mbps and transmission power of 15 dBm. A statistical propagation model and two-ray path-loss model was used, leading to a radio range of about 370 meters. The maximum COPS KA timer interval is set to 50 seconds and the service discovery timer interval is set to 20 seconds.

The commonly used Random Waypoint mobility model [CBD02] was employed to simulate node mobility. In the Random Waypoint model, a node is initially stationary for a fixed *pause time*. It then randomly chooses a destination and starts moving towards it with a speed chosen from a uniform distribution $[V_{min}, V_{max}]$, where V_{min} and V_{max} are the minimum and maximum node speeds. When the node reaches its destination, it again waits for an interval equal to the pause time, before choosing its next destination and speed. This sequence is repeated. Following the recent study [YLN03] on the anomalies in the Random Waypoint mobility model, the minimum speed (V_{min}) parameter was set to around 90% of the maximum speed (V_{max}) for all simulations. This avoided the speed-decay problem in the model, and also helped the network reach steady state in an acceptable time (less than 500 seconds). Pause time for the mobility model was set to 10 seconds.

Each simulation scenario was run for 1500 seconds; results were collected over 1000 seconds by disregarding data in the first 500 seconds of “warm-up” period. Nodes were uniformly distributed. During initialization, 10% of the nodes were randomly chosen to act as policy servers. Several runs of each simulation scenario were conducted (each run representing a random initial placement of nodes) to obtain statistically confident averages. Our aim was to keep the

confidence interval, computed using the method of batch means [Leo93], acceptably tight (within $\pm 5\%$ of the reported average value). The confidence intervals are plotted in the graphs.

All simulations incorporate our proposed general approach for cluster management (Section 6.1.2). Client nodes deploy the proposed service discovery scheme (Section 6.4) for automated discovery of policy servers. PBNM systems with and without our proposed delegation mechanism (Section 6.2.2) are simulated. The metrics used in this study are: percentage service availability, COPS signaling overhead (in terms of the number of COPS connections handled by a policy server), number of COPS connection timeouts, and service discovery overhead (average number of messages received per node).

7.2 Simulation Results

In this section, we describe our simulation results. First, we characterize the PBNM system performance as a function of cluster size, network mobility and network density (Sections 7.2.1 and 7.2.2), and show the effectiveness of our proposed time-based heuristic algorithm in minimizing service discovery broadcast overhead (Section 7.2.3); these simulations do not incorporate our proposed delegation mechanism. In Section 7.2.4, we then compare the performance of the PBNM system with and without delegation.

7.2.1 Effect of Cluster Size and Mobility

In the first set of simulations, we consider a network of 20 nodes (with 2 nodes acting as policy servers). The results are plotted as a function of the cluster size k , for three different mobility scenarios ($V_{max} = 5$ m/s, 10 m/s, and 20 m/s). Figure 7.1 shows the average number of new COPS connections handled by a policy server. As cluster size (k) increases, the probability of a client moving in and out of a cluster decreases, thus reducing the number of new COPS connections being established and then being torn down due to the client moving out of the k -hop cluster. This phenomenon is evident at lower speeds (5 m/s). As mobility increases, the probability of a client moving in and out of a cluster is high even for larger values of k . Hence, the number of new COPS connections being established does not reduce significantly with increase in k .

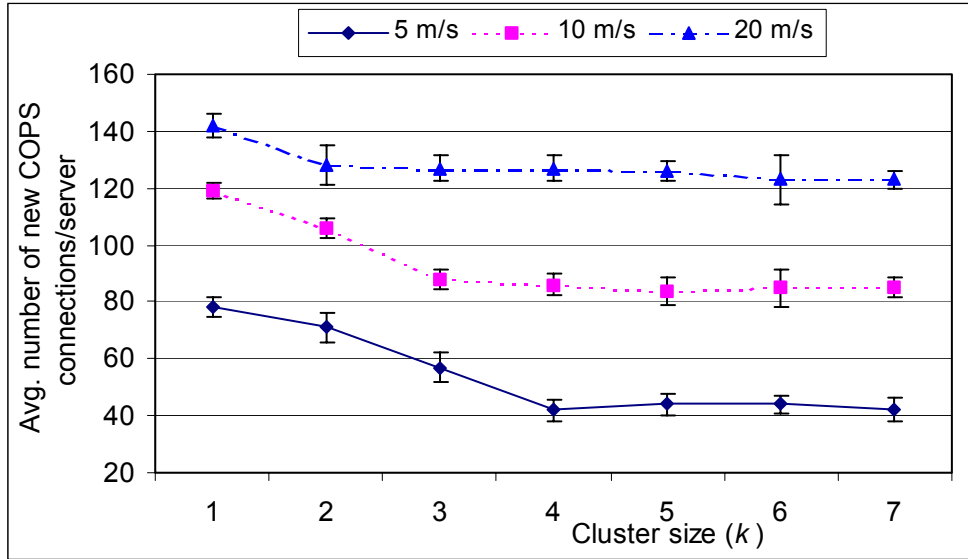


Figure 7.1 Average number of COPS connections established per server as a function of mobility and cluster size.

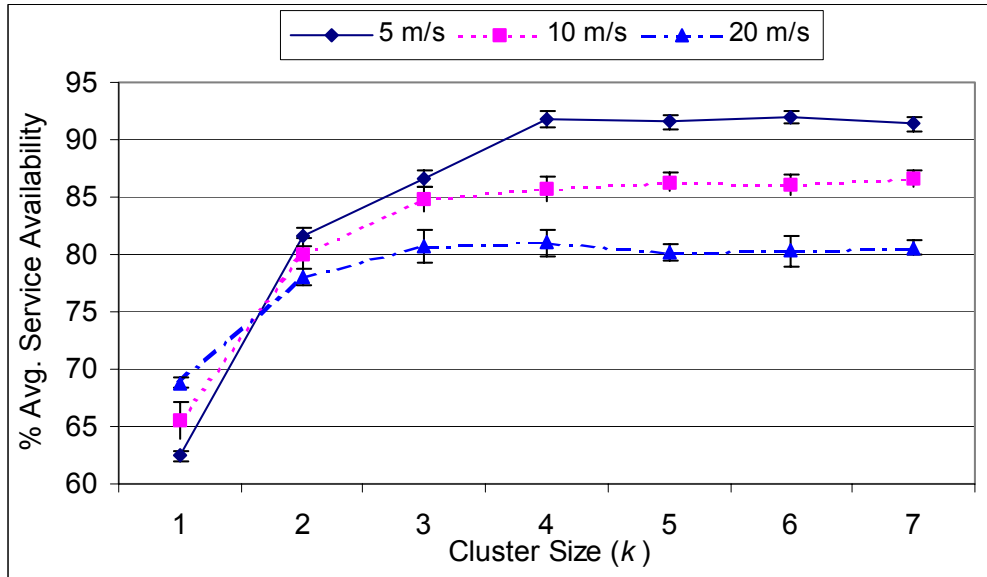


Figure 7.2 Percentage average service availability as a function of cluster size and mobility.

In Figure 7.2, we plot the average percent service availability for different values of k again for three different speeds (5, 10, and 20 m/s). As the cluster size increases, the coverage area of each server increases, thus resulting in an increase in the service availability. The system typically performs much better at low mobility, except for $k = 1$, wherein the service availability actually improves with speed! We decided to study this interesting behavior in some more detail. We considered different mobility scenarios ($V_{max} = 0, 2, 5, 10, 20, 50,$ and 100 m/s) for $k = 1$. The

results are shown in Figure 7.3. The service availability was found to be the least when the network is static, since clients that are initially placed outside the k -hop clusters never get serviced. The service availability then increases with mobility, allowing clients to spend more time inside a k -hop cluster. However, beyond a certain threshold speed (at 50 m/s and 100 m/s), mobility hampers the system performance, as clients move very quickly in and out of a cluster, thus resulting in a decrease in the overall service availability.

In general, we observe that increasing the cluster size improves the service availability and reduces the overhead (in terms of number of COPS connections). However, we also need to address the trade-off involved in increasing the cluster size. In our preliminary experiments (Sections 5.3.2, 5.3.3), we observed that with increase in k , there is considerable unpredictability in addition to an exponential increase in the policy response time (end-to-end delay). Further, increasing k means that a greater number of intermediate nodes are involved in forwarding management traffic, leading to expenditure of expensive resources such as bandwidth and energy. Here, we characterize how COPS performs with increase in the cluster size.

As shown in Figure 7.4, it is found that increasing number of COPS connections are timed out as k increases, due to the COPS Keep-Alive (KA) message sent by a client not reaching the server, or the server's response not reaching the client in a timely fashion. Comparing Figures 7.1 and 7.4, we observe that for larger values of k , close to 40% of the new COPS connections are due to connection timeouts.

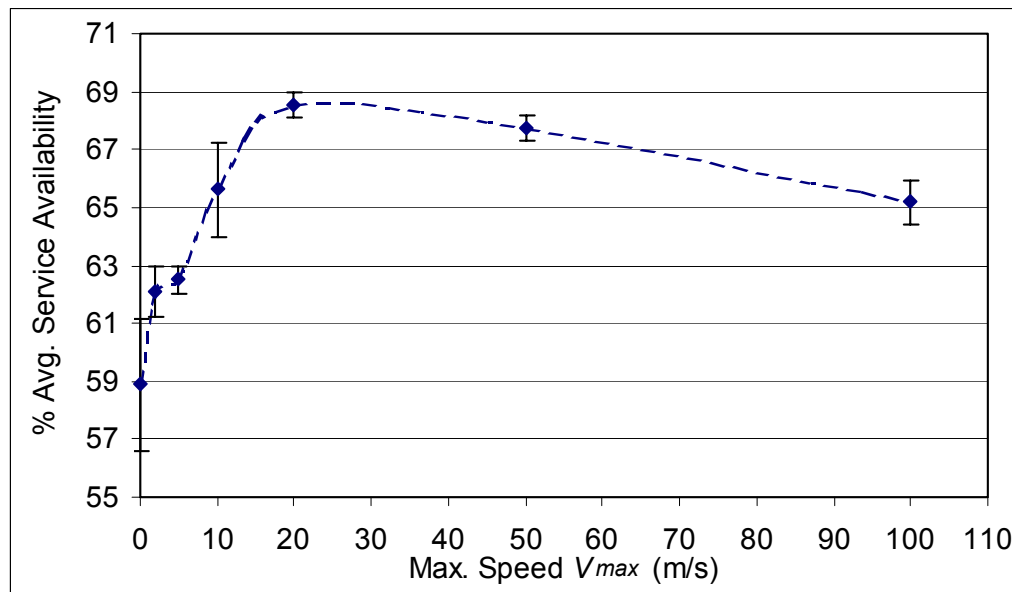


Figure 7.3 Percentage average service availability at different speeds for cluster size $k = 1$.

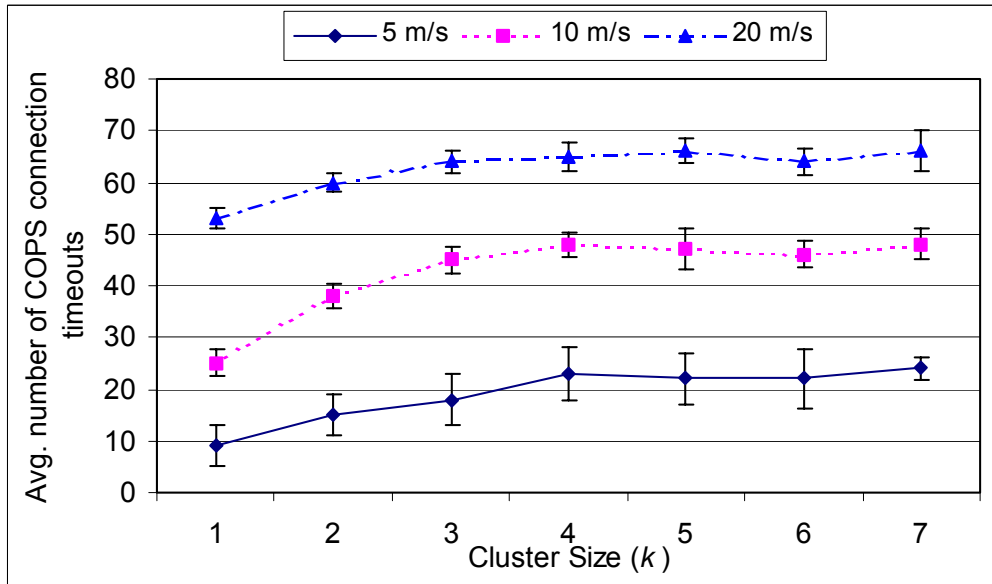


Figure 7.4 COPS connection timeouts as a function of cluster size and mobility.

In general, for all the metrics, we observe that the average values “settle down” close to $k = 4$. This is attributed to the fact that with the simulation area, network density, and transmission range considered in our simulations, most clients are less than 5 hops away from the server.

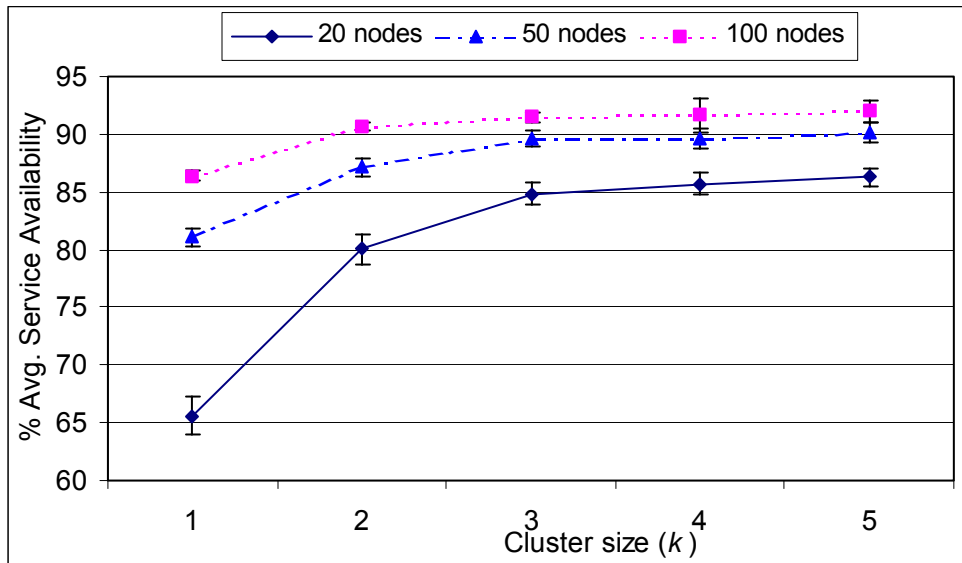


Figure 7.5 Percentage average service availability as a function of cluster size and network density.

7.2.2 Effect of Network Density

In our second set of simulations, we vary the network density – changing the number of nodes and keeping the simulation area constant. The aim is to address scalability of the management system in larger networks, and to study the effect of network density on system performance. Three different scenarios were considered: 20, 50, and 100-nodes/simulation area; V_{max} was set to 10 m/s. The PBNM service availability is plotted in Figure 7.5. The service availability is found to improve with increase in network density, attributed to the improved connectivity. In general, the PBNM system was found to scale very well over larger MANETs.

7.2.3 Service Discovery Overhead Minimization

In this section, we characterize the improvement obtained by using our proposed time-based heuristic algorithm to minimize service discovery broadcast overhead. As shown in Figure 7.6, the proposed mechanism considerably reduces the broadcast overhead by minimizing unnecessary broadcasts, as compared to the case with blind k -hop limited broadcasts. The benefit becomes significant as we increase the cluster size, as more nodes are involved in re-broadcasting service discovery messages.

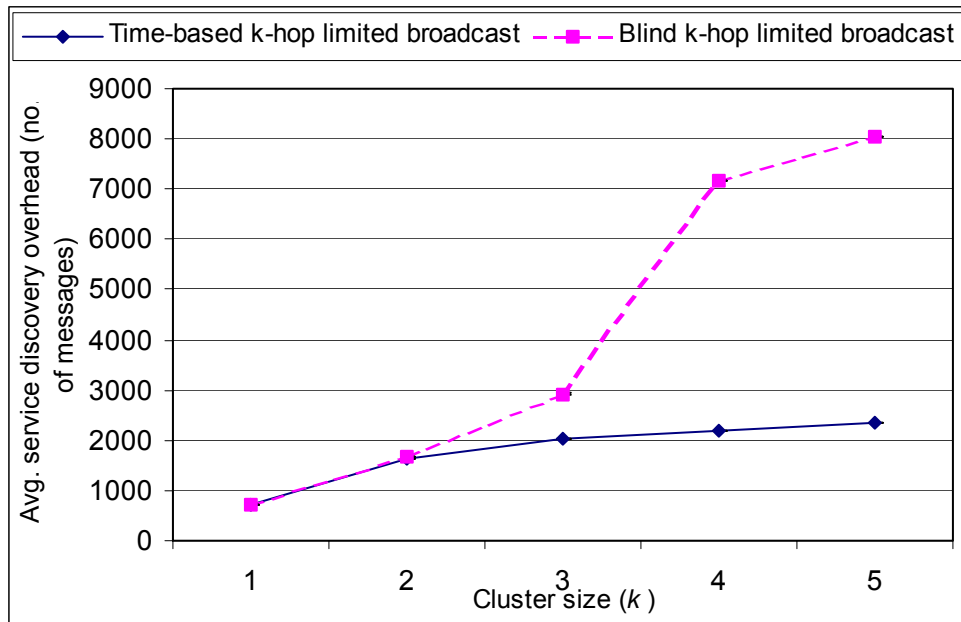


Figure 7.6 Average service discovery broadcast overhead as a function of cluster size, in presence and absence of our proposed time-based heuristic.

7.2.4 Delegation

The advantages of our proposed delegation mechanism were qualitatively discussed in Section 6.2.2. Here, we quantify the improvement in system performance (% service availability) through delegation. We consider a 20-node network with, initially, two policy servers present. Three nodes are randomly chosen to be capable of acting as delegated servers. As mentioned in Chapter 6, in an actual management system, a node may volunteer to act as a delegated server based on certain resource management criteria such as processing load, battery life, and connectivity; however, the study of the various criteria is outside the scope of this dissertation.

With two policy servers initially deployed, and three servers that may get elected as delegated servers, there are a maximum of 5 servers operating in the network at any given time. The delegation procedure follows the signaling outlined in Section 6.2.2. Considerable increase in service availability is observed with delegation, as shown in Figure 7.7. It is noteworthy that in general such dynamic invocation of policy server instances helps improve the service coverage of a PBNM system *on-demand* while allowing it to maintain cluster sizes as small as possible. The trade-off lies primarily in the overhead in delegating policies, the exact overhead depending on the specific policy traffic model (size of policy transactions) for a given network.

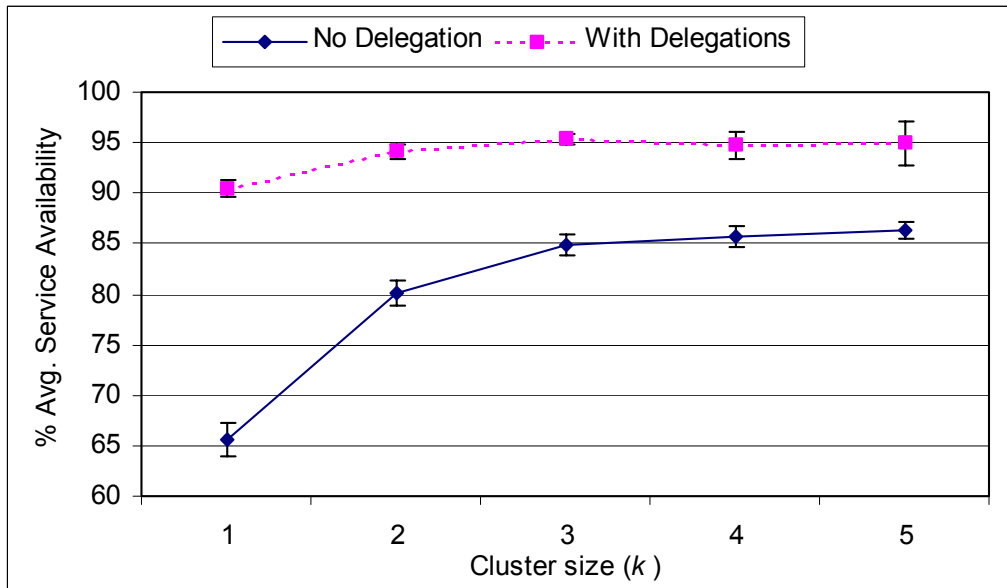


Figure 7.7 Improvement in service availability using our proposed delegation scheme.

7.3 Experiments

7.3.1 Testbed Setup

To verify the trends exhibited in our simulation results, we conducted experiments using our testbed network. Nine machines were connected via the Dynamic Switch. Using our utility programs (Sanitizer + Topology Matrix Generator + DS Mobility Generator), we generated mobility files for the Dynamic Switch emulating the Random Waypoint mobility model. The system parameters of concern for the experiments (used by the Dynamic Switch to emulate connectivity) are the area and radio range. Parameter settings identical to those in the simulations were used; OLSR was used for MANET routing. Our experimental testbed network size was limited to nine nodes; hence, we ran simulations with a network of nine nodes to compare the results. In both cases, one node was chosen to act as a policy server. The noteworthy characteristics that were different in the experimental setup as compared to the simulation environment were the absence of the 802.11 MAC protocol and the lack of the wireless propagation and pathloss models.

7.3.2 Results

Here we present a comparison of the results obtained from our experiments and simulations. Plots for the percentage average service availability and the number of COPS connections per server as a function of cluster size are shown in Figure 7.8 and 7.9 respectively. The general trend in the average service availability obtained from the experiments was similar to that observed in the simulations. Both exhibited increase in service availability with increase in the cluster size as expected.

An interesting observation was made in the plot for the average number of COPS connections established during the duration of the simulation. In our earlier simulation results for network size of 20 nodes (Figure 7.1), we saw that the number of COPS connections handled per server decreased with increase in the cluster size. As discussed earlier, two factors play a role in this metric – first, the number of COPS connections that are closed due to the client node moving out of a cluster, and secondly, the number of COPS connections that time out. The first factor plays a

major role for smaller cluster size while the latter plays a role in larger clusters as shown in Figure 7.4.

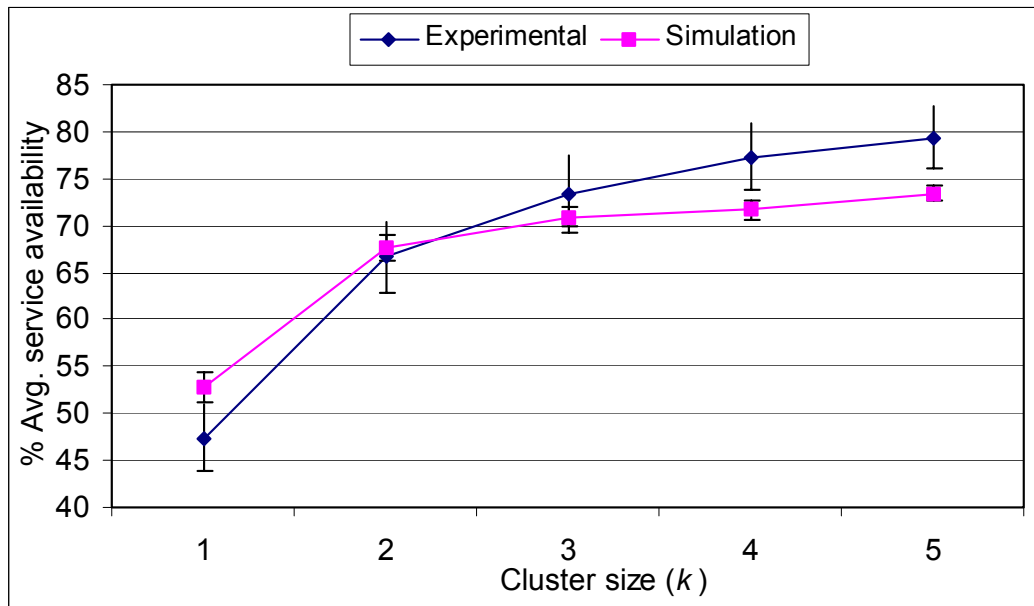


Figure 7.8 Comparison of average service availability obtained from the experiments and simulations.

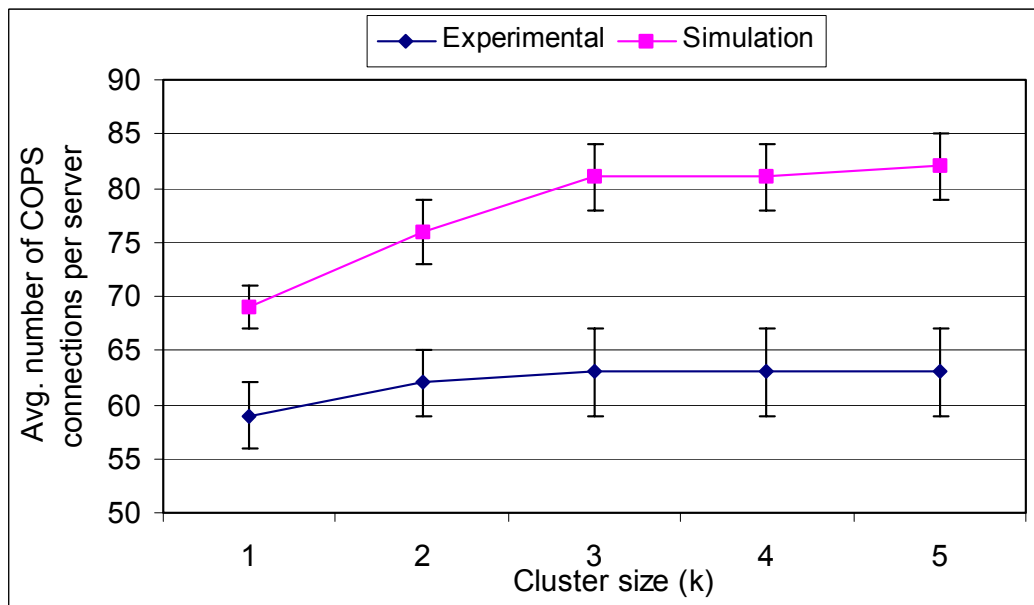


Figure 7.9 Comparison of average number of COPS connections obtained from the experiments and simulations.

Unlike the modest to high network densities we considered earlier, a much sparser network with nine nodes was considered in the set of simulations discussed in this section. Due to the sparse connectivity, for smaller cluster sizes, e.g., for $k = 1$, only a small percentage of nodes (that moved within one hop from the policy server) got a chance to establish a COPS connection with the policy server. As a result, the number of COPS connections that were established was small. As the cluster size increased, more clients had the opportunity to establish a COPS connection with the server; the poor connectivity also led to many of these connections to timeout, in turn resulting in increase in the number of COPS connections established. This resulted in an increasing trend of the number of COPS connections with increase in cluster size.

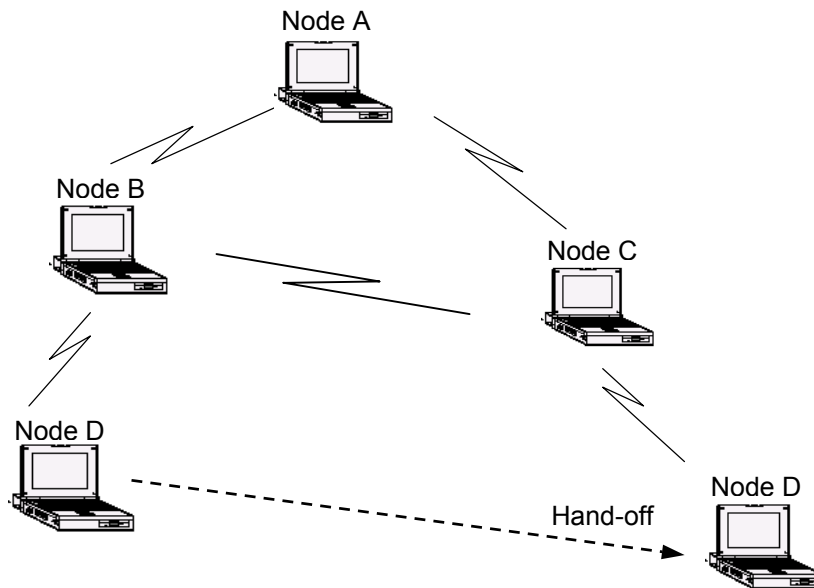
In the experiments, only a slight increase in the average number of COPS connections was observed with increasing cluster size, the number remaining almost constant for larger cluster size. In general, much fewer COPS connections were observed in the experimental results as compared to the simulations, while the resulting service availability was equal to or greater (except for $k = 1$) than that obtained from the simulations. This indicated that on an average each COPS connection lasted for a longer time than was the case in the simulations. This can be attributed to the fact that the wired experimental environment, in the absence of the wireless channel model and the 802.11 MAC protocol behavior, was more *stable* as compared to the simulation environment.

7.4 QoS Management in Multi-domain Ad Hoc Networks

In the previous sections, we characterized the performance of the PBNM system under different mobile ad hoc networking conditions. In this section, we demonstrate the PBNM system at work for provisioning and managing Quality of Service (QoS) in multi-domain ad hoc networks. The goal is to illustrate the use of our proposed COPS-based policy negotiation mechanism to guarantee seamless QoS in such networks. Two illustrations are used – subjective assessment of the received video streaming application as perceived by the end-user is shown in Section 7.4.1; and a quantitative analysis of real-time mission critical applications is presented in Section 7.4.2.

7.4.1 Illustration 1: Video Streaming

Here, we illustrate the effectiveness of our proposed policy negotiation scheme in provisioning QoS for a video-streaming application. Our demonstration network consists of four laptops: A, B, C, and D, as shown in Figure 7.10. The laptops communicate using IEEE 802.11b wireless cards. The OLSR protocol is used for routing purposes. Nodes B and C represent two policy servers, each controlling a separate organizational network domain. Node D is a mobile node, currently in its “home” domain administered by policy server B. D is transmitting video traffic to node A, using the VIC tool [Vic]. In its home domain, a minimum bandwidth of 64 kb/s is allocated to node D for its video transmission.



Node	Description
A	Video stream application sink
B	Policy server for domain 1
C	Policy server for domain 2
D	Video stream application source

Figure 7.10 Demonstration scenario depicting a multi-domain wireless ad hoc network; hosts B and C are policy servers in distinct administrative domains.

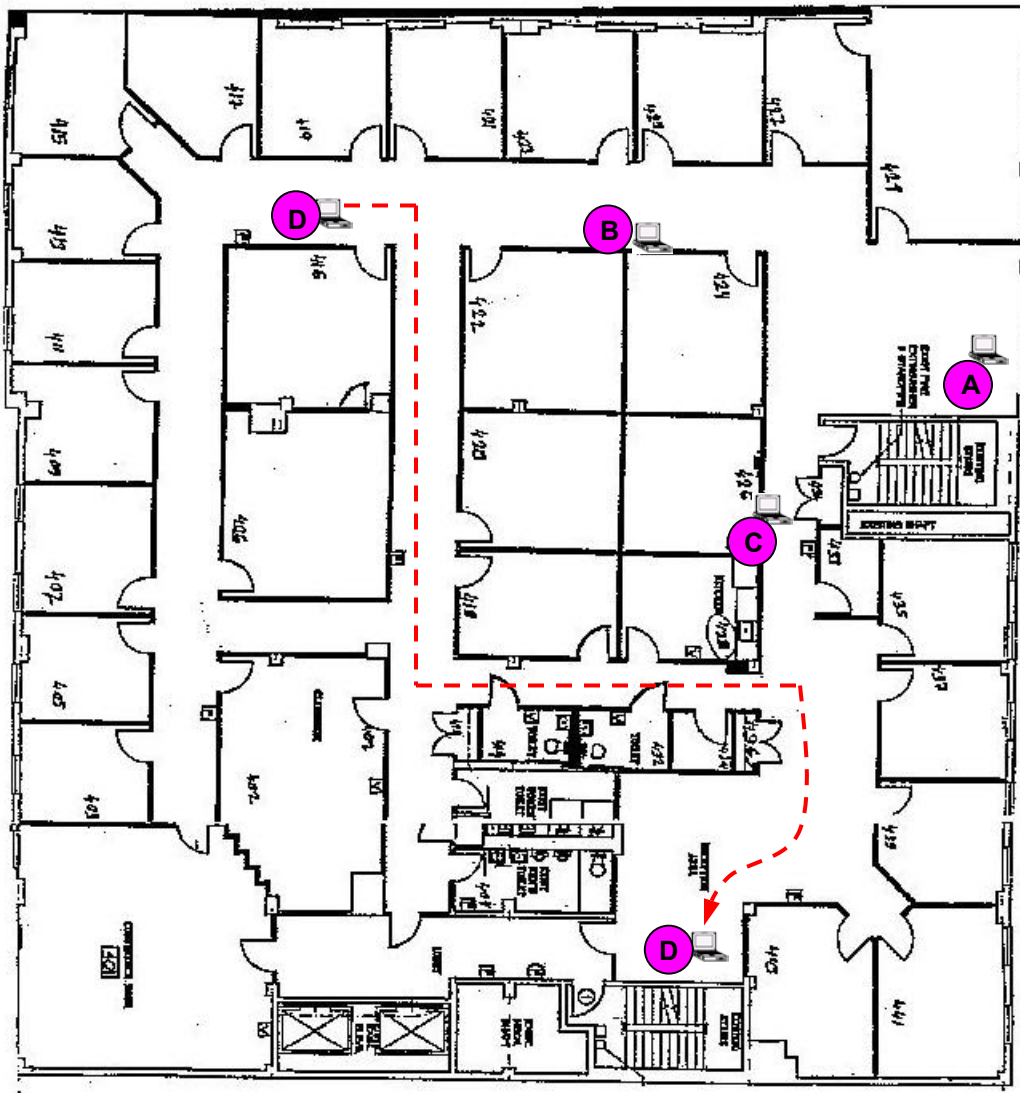


Figure 7.11 Layout of the area where we conducted the demonstration; nodes A, B and C are static, while the movement of node D is shown with dotted line. Node D is the policy client that “hands-off” from policy server B to policy server C.

For illustration purposes, we configure our policy-based management system to realize 1-hop clustering. The layout of our work area and illustration of the experiment are shown in Figure 7.11.

The movement of node D away from its “home” domain (shown as the dotted line in Figure 7.11) changes its point of connectivity in the network; it loses direct connection with node B and instead connects via a foreign domain (through node C). Based on the routing information

gathered from the underlying OLSR daemon, policy server B detects the change in topology – that policy client D is now two hops away from it, and at a one hop distance from policy server C. Hence, policy server B closes the COPS connection with node D and in doing so, it *redirects* node D to policy server C. Node D establishes a COPS connection with policy server C. In the COPS open (*OPN*) message, client D indicates to server C the address of its original policy server, namely node B. Server C does not have the relevant policies for node D. Hence, it sets up a COPS connection with node B and sends a COPS request (*REQ*) message to obtain policies for node D (this signaling sequence was previously illustrated in Figure 6.4). After downloading relevant policies, policy server C is now able to allocate bandwidth in the range of 64 kb/s to 128 kb/s for node D’s video application, thus resulting in acceptable video performance, as shown in Figure 7.12.

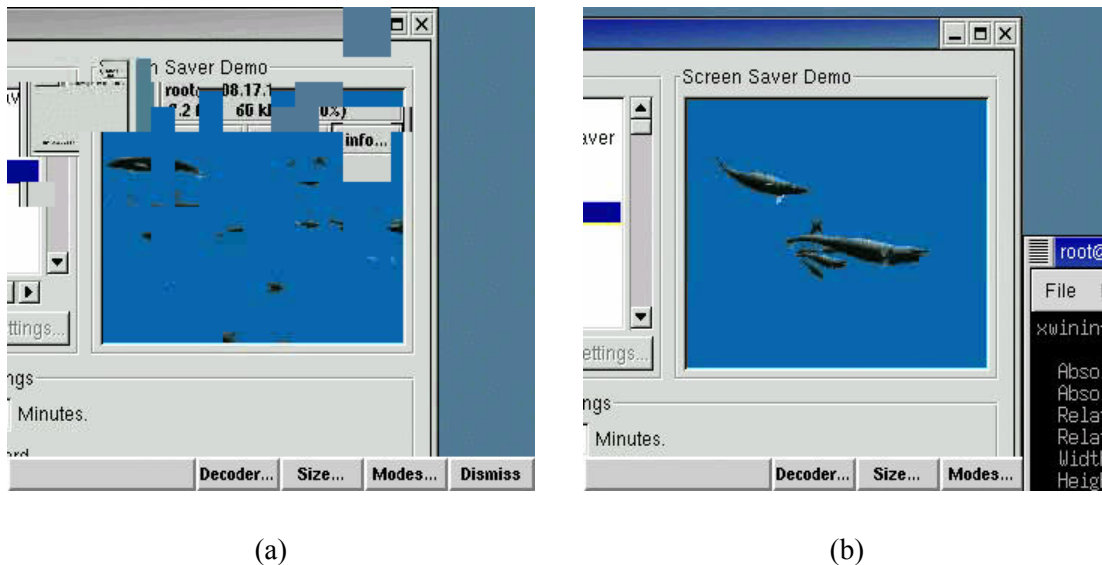


Figure 7.12 (a) Degraded video quality without policy negotiation (allocated bandwidth is 12 kb/s); (b) Acceptable video quality (bandwidth in the range 64 kb/s to 128 kb/s allocated) after policy negotiation.

7.4.2 Illustration 2: Real-Time Application

The second illustration was conducted using the wired testbed. We again considered a mobile ad hoc internetwork; however, this time four different administrative domains were setup. The real-time application software implemented by researchers at Virginia Tech was used to generate real-time traffic. Random node mobility was emulated using the Dynamic Switch. In order to

characterize the effectiveness of our proposed policy negotiation mechanism, we ensured multi-hop connectivity between the application endpoints during the experiment. Transmission of three traffic flows (from the source node) was considered: two real-time applications with distinct utility functions (each application transmitting at the rate of 48 kb/s) and one flow representing background traffic (data rate of about 32 kb/s). The source node moved randomly across the various mobile domains.

Three metrics were considered: percentage accrued utility, percentage missed deadlines and average throughput. The performance of the two real-time applications was captured using the monitoring tool and choirGUI (part of the real-time software package). As seen in Figure 7.13, the GUI consists of four windows. The window on the bottom-right shows the utility functions for the two real-time applications. The X-axis represents the end-to-end delay and the Y-axis represents the utility. The end-to-end delay for each received packet is recorded and indicated by vertical bars appearing in this window (shifting along the X-axis as the delay changes). Whenever a packet is received after the deadline – the threshold delay beyond which the utility for that application is zero – the corresponding bars appear red indicating a missed deadline. The top-right window shows the total accrued utility as percentage of the maximum utility. The window on the top-left displays the percentage of missed deadlines, while the window on the bottom-left shows the average throughput for the real-time applications as well as the background traffic.

Figure 7.13 shows the case without policy negotiation. Initially, the source node is resident in its “home domain” and obtains the desired QoS for the real-time applications. As the source node moves into “foreign” domains, the real-time traffic is classified along with the background into the default best-effort class (due to lack of appropriate QoS policies), leading to considerable decrease in throughput. Soon the end-to-end packet delay increases beyond the threshold, and packets start missing deadlines as shown by the red bars in the bottom-right window. The percentage accrued utility (seen in top-right window) eventually drops to zero.

In presence of the policy negotiation mechanism, the various domains are able to negotiate policies and the node receives the desired QoS for its real-time applications as it moves into domains administered by other organizational policies. The resulting plot captured using the choirGUI is shown in Figure 7.14. Except for the few temporary glitches in the performance (e.g., around the 100, 170, 210 and 250 second marks) when the node moves from one domain to another, the real-time applications experience almost seamless QoS.

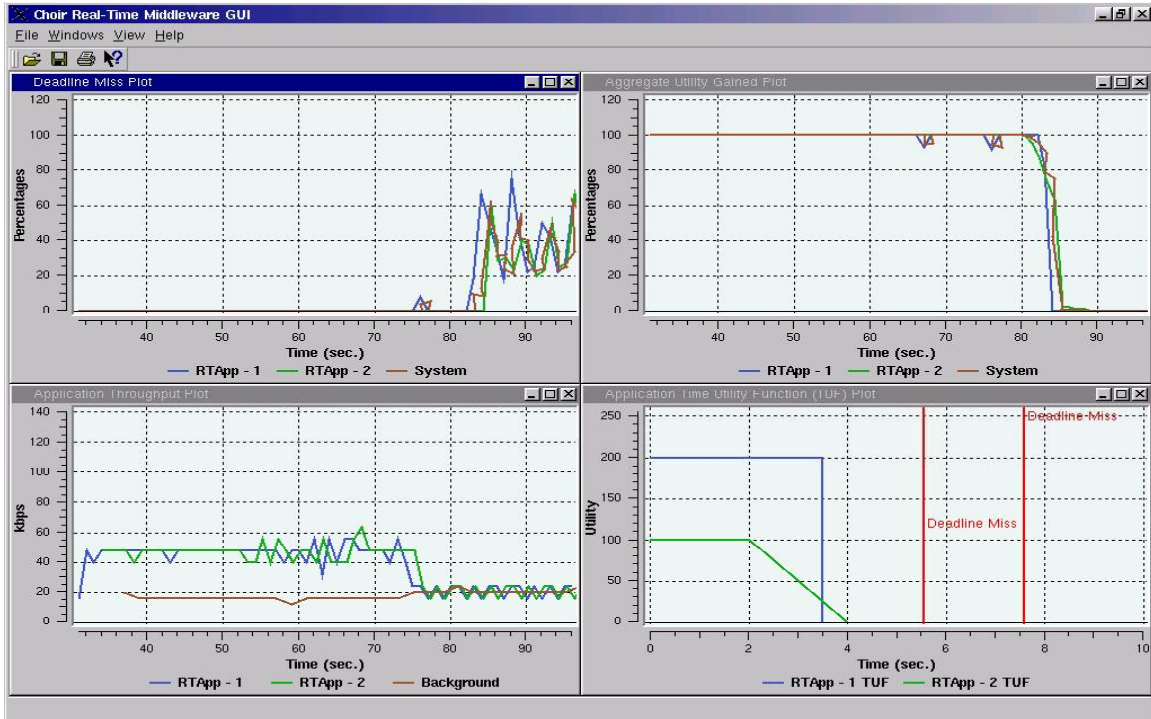


Figure 7.13 Without policy negotiation, the real-time mission critical applications are treated as best-effort along with background traffic as the source node moves into foreign domains.

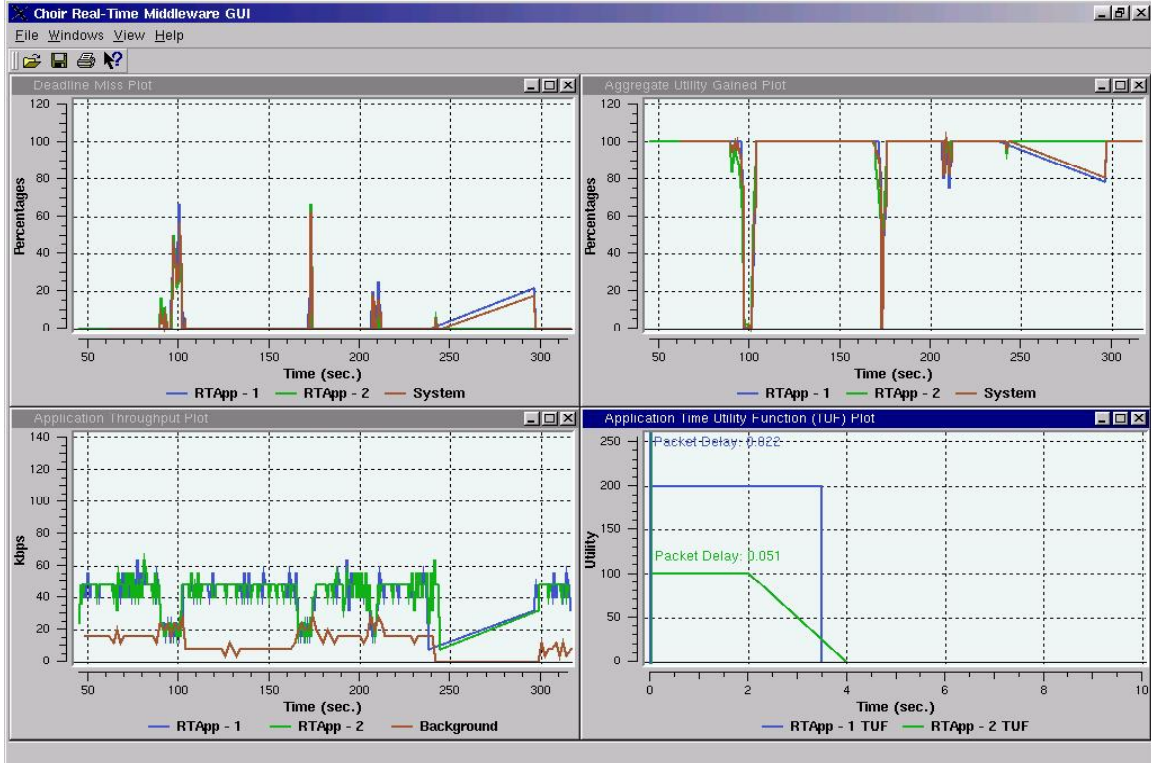


Figure 7.14 Almost seamless QoS is achieved for real-time mission critical applications in presence of policy negotiation even as the source node moves across different network domains.

7.5 Summary

In this chapter, we presented the performance evaluation of our PBNM system implementation studied under varied networking conditions – as a function of the cluster size k , node mobility, and network density. In general, the average service availability of the PBNM system improved with the increase of the cluster size. The trade-off lies in the fact that larger cluster size results in greater unpredictability in the performance (more COPS connection timeouts) and more resources are expended at intermediate nodes to forward management-related traffic. The PBNM implementation was found to scale very well over larger networks (up to 100 nodes were considered), and the system performance improved with increase in network density due to improved network connectivity. Our proposed delegation mechanism showed great promise in improving the service coverage even for smaller cluster sizes.

We verified the general trend of our simulation results by comparing the performance of the PBNM system in a testbed network. The experiments were conducted using a wired network with the Dynamic Switch emulating the Random Waypoint mobility model.

Finally, we illustrated the effectiveness of the proposed policy negotiation technique, implemented as an extension to the COPS-PR protocol, via two concrete applications. The performance of video streaming and mission critical real-time applications was assessed via subjective and quantitative measurements, respectively.

I think and think for months and years. Ninety-nine times, the conclusion is false. The hundredth time I am right.
- Albert Einstein

*And in the end it's not the years in your life that count.
It's the life in your years.*
- Abraham Lincoln

Chapter 8

Conclusion and Future Work

We conclude this dissertation by summarizing the contributions of this research and providing directions for future work.

8.1 Summary

Ad hoc networks provide wireless and mobile computing capability, in situations where efficient, economical and rapid means of communication is required, and where the use of a wired or an infrastructure-based wireless network is either too expensive or impractical. The various constraints imposed by ad hoc network environments make Quality of Service (QoS) provisioning and management in such networks a challenging task. While a variety of research dealing with network management in ad hoc networks exists, QoS management in wireless ad hoc networks remains largely an uncharted territory and forms the focus of this work.

We identify policy-based management as a promising approach for managing QoS in ad hoc networks. Policy-Based Network Management (PBNM) provides a logically centralized, simplified and automated control of the network as a whole, making management of complex

network operational characteristics such as QoS, access control, and network security easier. We propose and implement an automated, intelligent, efficient, and robust policy-based management framework for MANETs, and demonstrate its application for Quality of Service (QoS) management. The main components and contributions of this dissertation are summarized below.

- **PBNM framework**

Based on our assessment of the salient features that characterize most ad hoc networks, we outline the key requirements sought in an ad hoc network management system, and hence identify the components that should comprise a policy-based framework for provisioning ad hoc networks. The framework provides insight into the interdependencies among the various components, and helps formalize the complex functional tasks that need to be carried out. To our knowledge, such a comprehensive representation of a management system is lacking in prior research works, and thus, it is one of the contributions of this work.

- **Policy Architecture Taxonomy**

In this work, we propose a characteristics-based policy architecture taxonomy that provides a systematic platform for qualitative assessment of the various architectures. The benefits of the taxonomy are not limited to this research; we hope that the taxonomy-based study of the various policy architectures will help network operators to weigh the different architectural choices, and to deploy their policy-based management solutions in a resourceful manner.

Using the taxonomy and some preliminary experimental evaluation, we suggest the use of hybrid types of architectures for ad hoc network management – ones that combine the outsourcing and provisioning techniques. This improves the efficiency of the PBNM system while allowing it to support dynamic policies.

- **Solution Suite**

While the policy-based approach offers certain attractive characteristics, we confront the challenge of adapting and extending the fundamentally centralized network management idea to a decentralized ad hoc networking paradigm. We propose a suite of solutions to achieve a self-organizing, robust and efficient PBNM system. The four components of the solution suite are:

- ***k*-hop Cluster Management**

Our proposed *k*-hop clustering helps limit the number of hops between a policy client and server. Doing this considerably improves the predictability of the PBNM system, reduces the number of COPS connection timeouts, and bounds the policy response time. Furthermore,

localizing the management signaling avoids expending scarce resources at intermediate nodes, which otherwise would have to forward the control traffic. However, from our experimental and simulation-based evaluation, we observed that service coverage improves considerably with an increase in cluster size. Thus, while deploying a PBNM system, study of this trade-off and determining an optimal value or a range for the cluster size is crucial. As seen from our results, this optimal value or range depends on the network size, density and average node mobility; $k = 3$ was found to be a good choice for most scenarios we considered.

We proposed and demonstrated two approaches to implement k -hop cluster management – one method takes advantage of the topology information available with the underlying MANET routing daemon and the second method employs a more general approach (does not assume availability of topology information) and achieves cluster management via cross-layer interaction between the COPS-based application layer and the IP layer.

- **Dynamic Service Redundancy (DynaSeR)**

We proposed the DynaSeR solution to adapt to clients that are outside or have moved out of a k -hop cluster and thus improve the service coverage of the system. It consists of two techniques: the server-centric *redirection* technique, which allows a server to redirect a client to another policy server, and *delegation*, which involves dynamic invocation of policy server instances on demand. Delegation was achieved by implementing two of our proposed extensions to the COPS-PR protocol. For the network set-up considered, the delegation mechanism resulted in 10 to 25% improvement in the service availability.

- **Policy Negotiation**

We proposed and implemented another extension to the COPS-PR protocol to facilitate inter-PDP communication and hence to allow negotiation of policies between network domains administered by different organizations. The impact of this enhancement was demonstrated with two illustrations using our testbed network.

- **Service Discovery**

Automated service discovery – determining the existence of nearby policy servers – was crucial to impart the desired self-organizing characteristic to the PBNM system. We proposed and implemented a lightweight service discovery mechanism to facilitate this. A time-based heuristic was developed to reduce the service discovery broadcast overhead and its effectiveness was verified using simulations; 50 to 400% savings were obtained for larger cluster sizes.

- **Real-Network Implementation and Experimental Analysis**

We focused on the real network implementation of our policy-based management framework. Experiments using the wired and wireless testbed networks served as a proof of concept and allowed us to demonstrate the practicality of our proposed schemes, and to some extent validate our simulation results. On a personal note, the prototype implementation and experiments had a good learning curve and the “real world” experience gained from this hands-on work was invaluable. We reported our experience in working with the wired and wireless network testbeds, and the general lessons learnt in the implementation process in this thesis, and we hope that it will provide useful guidance to other researchers to take up such an endeavor.

- **Simulation-based Study**

The primary motivation for conducting a simulation study was to address scalability of our proposed PBNM framework, and to understand its behavior under different networking conditions. While QualNet supported the underlying protocol stack, it did not have any pre-existing models of interest for policy-based management at the application layer. Our implementation of simulation models for the COPS and COPS-PR protocol and our proposed solution suite serves as a significant contribution of this research. We hope that it will provide impetus to other researchers to further this work, and leverage simulation-based study of network management systems in general.

8.2 Directions for Future Work

To our knowledge, this research is the first attempt to apply the concept of policy-based management to ad hoc networks, and we believe there is considerable scope to advance this research in the future.

- **Optimization and Enhancement of Proposed Solutions**

Future work should investigate ways to improve some of our proposed solutions. For example, there may be scope to enhance the k -hop clustering algorithm to optimize the PBNM system performance. The existing k -hop clustering can be extended to incorporate adaptive clustering – one that adapts the cluster size k to changing network conditions, and resource-based clustering – one that facilitates load balancing, adapts cluster membership to available resources at a cluster-

head (policy server), and allows clients to choose the most appropriate policy server (based on criteria such as stability of links) whenever possible. In case of the adaptive clustering, a mathematical expression could be formulated to compute the optimal value of k based on the *cost saved* and *cost incurred* in expansion or reduction of the cluster size k . The definition of the *cost* will depend on various factors discussed earlier, such as bandwidth, battery life, delay or response time, and service availability.

Also, as mentioned earlier, topology control and management techniques such as prediction of network partitions leading to proactive delegations and hand-offs may be investigated to make the management system more robust. It will be interesting to study the behavior of the PBNM system in presence of other peculiar mobility models such as group mobility and grid mobility wherein the network dynamics are very different [CBD02] as compared to the Random Waypoint model used in this research.

- **Policy Monitoring**

Our proposed PBNM framework included policy monitoring as one its components; however, it was outside the scope of this dissertation. It is desirable to have an independent policy monitoring process to ensure robustness of the PBNM application and to verify whether the network in fact meets the high-level goals or specifications. Policy monitoring can be achieved using active (dummy transactions or sending probe packets) or passive (monitoring to estimate performance of network flows) methods [Ver00]. Policy monitoring and the use of efficient feedback mechanisms is an open area for research.

- **Application of the Policy Framework to Other Policy Disciplines**

We demonstrated the effectiveness of our PBNM implementation by illustrating its impact on QoS management in ad hoc networks. The implementation could be extended for managing other policy disciplines such as policy-based routing, e.g., in support of access control and to improve fault tolerance; network security, e.g., for dynamic key management in multi-domain networks; and to control physical layer properties of cognitive or software radios, e.g., dynamic spectrum allocation facilitating efficient use of the available frequency band and spatial frequency reuse.

- **Feasibility Analysis**

In our earlier discussion, we pointed out that the feedback information provided by a policy-based management system is crucial given the dynamic nature of ad hoc networks. The feedback can be used for feasibility analysis and planning of future missions in mission-centric ad hoc networks.

Implementation of *policy-aware* applications and their integration with the policy-based management framework to take advantage of the feedback information is required. Some relevant work in this area is being done by researchers at Virginia Tech [Ste02].

- **Alternative Mechanisms for Policy-Based Management**

It may be useful to explore alternative mechanisms for implementing policy-based management. One possible alternative is the use of mobile agents. While mobile agents have shown great promise in monitoring and data collection in wireline networks as well as MANETs, their use to distribute complex policies and configure network devices is still unexplored.

- **Analytical Modeling of Client-Server Systems (such as our PBNM architecture) in MANETs**

Client-server behavior in ad hoc networks is not yet well understood, and we believe that mathematical modeling of client-server systems in MANETs is a challenging area of research with considerable scope for contribution. Our PBNM system is a prime example of a client-server architecture, and analytical modeling of our PBNM system is in fact a part of our ongoing research efforts.

A preliminary representation of the policy client behavior in our PBNM system modeled as a network of closed queues is shown in Figure 8.1 In the queuing model, λ_1 represents the rate of client departure from a k -hop cluster; π_1 represents the probability that a client moving out of its current cluster moves immediately into another k -hop cluster; and, for the case where a client moving out of a cluster does not move immediately into another cluster, λ_2 represents the client arrival rate into a k -hop cluster. λ_1 and λ_2 are assumed to be Poisson distributed. Using the Gordon-Newall convolution algorithm [Rob94], we solve the Markov chain for this closed network of queues and obtain the service availability $S.A. = \lambda_2 / [\lambda_2 + (1 - \pi_1)*\lambda_1]$. Results using heuristic values for λ_1 , λ_2 , and π_1 have shown promise.

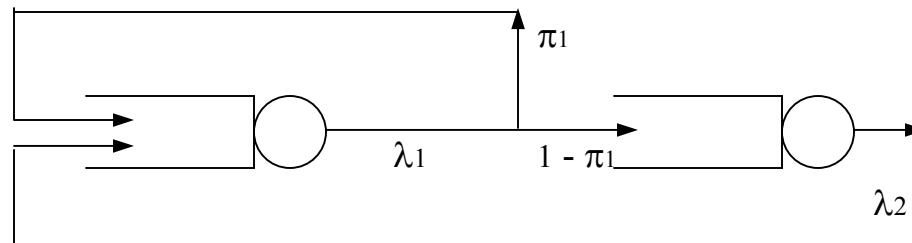


Figure 8.1 Representation of the client behavior as a closed network of queues to model service availability.

Parameterization of the model is very challenging. One solution is to develop an empirical model, wherein parameters such as λ_1 , λ_2 , and π_1 are obtained from simulations. To mathematically model these parameters accurately, in-depth investigation of the various underlying components (such as the mobility model, routing protocol, and clustering) that influence the system performance is necessary. Furthermore, the empirical model can be used to determine whether the Poisson assumption for λ_1 and λ_2 is valid.

The next step is to enhance the existing model to include certain other complexities of the system, and to model the server behavior with multiple clients. Eventually, the entire PBNM system can then be modeled by combining the client and server models and taking care of the interaction involved.

While dealing directly with Markov chains in the preliminary stages of modeling is feasible, it may become cumbersome or even impractical to represent the entire PBNM system or a general client-server system using Markov chains due to the sheer complexity (large number of states and transitions) involved. One analytical modeling methodology that seems promising to this end is the use of Stochastic Petri Nets (SPNs) [Ger02, Pet81]. SPNs generally provide a relatively succinct representation of complex systems as compared to their Markov chain counterpart, and allow better visualization of the dynamics of a system.

*Cutting through the acronyms and argot that littered
the hearing testimony, the Internet may fairly be regarded
as a never-ending worldwide conversation.*

-- Judge Dalzel

Glossary

ACG: Automated Code Generation
ANMP: Ad hoc Network Management Protocol
API: Application Programming Interface
ARI: Alexandria Research Institute
CBQ: Class-Based Queuing
CCO: Centralized-Centralized-Outsourcing
COPS: Common Open Policy Service
COPS-PR: COPS for PRovisioning
CORBA: Common Object Request Broker Architecture
CSRQ: Client Service Request
C-WAN: Coalition Wide Area Network
DCO: Distributed-Centralized-Outsourcing
DDO: Distributed-Distributed-Outsourcing
DDOP: Distributed-Distributed-Outsourcing-Provisioning
DDOP-H: Hierarchical DDOP
DDP: Distributed-Distributed-Provisioning
DDP-H: Hierarchical DDP
DiffServ: Differentiate Services architecture
DMTF: Distributed Management Task Force
DynaSeR: Dynamic Service Redundancy
FSM: Finite State Machine
GUI: Graphical User Interface
HMAC: key-Hashed Message Authentication Code
HTB: Hierarchical Token Bucket

IEEE: The Institute of Electrical and Electronics Engineers
IETF: Internet Engineering Task Force
IP: Internet Protocol
IPSec: IP Security protocol
ISM: Industrial Scientific and Medical
ISO: International Standards Organization
ISP: Internet Service Provider
KA: Keep-Alive
LAN: Local Area Network
LARTC: Linux Advanced Routing and Traffic Control
LBNL: Lawrence Berkeley National Laboratory
LDAP: Lightweight Directory Access Protocol
LPDP: Local Policy Decision Point
LTU: Lulea University of Technology
MANET: Mobile Ad hoc NETwork
MARE: Mobile Agent Runtime Environment
MbD: Management by Delegation
MCDS: Minimal Connected Dominating Set
MD: Message Digest
MIB: Management Information Base
NAVCIITI: Navy Collaborative Integrated Information Technology Initiative
NTP: Network Time Protocol
OLSR: Optimized Link State Routing
ONR: Office of Naval Research
OSPF-MCDS: Open Shortest Path First routing protocol using MCDS
PAM: Policy Advisor Module
PBNM: Policy-Based Network Management
PDP: Policy Decision Point
PEP: Policy Enforcement Point
PIB: Policy Information Base
PMT: Policy Management Tool
PRC: Policy Rule Class
PRI: Policy Rule Instance
QoS: Quality of Service

RAP: IETF Resource Allocation Protocol working group
RPGM: Reference Point Group Mobility
RSVP: resource ReSerVation Protocol
SA: Service Advertisement
SHAMAN: Spreadsheet-based Hierarchical Architecture for MANagement
SDK: Software Development Kit
SLA: Service Level Agreement
SLS: Service Level Specification
SMI: Structure of Management Information
SNMP: Simple Network Management Protocol
tc: Traffic Control
TCP: Transmission Control Protocol
TLS: Transport Layer Security
TTL: Time To Live
UMTS: Universal Mobile Telecommunications System
VIC: Video Conferencing tool

If I have seen further it is by standing on the shoulders of Giants.

- Sir Isaac Newton

*Knowledge is of two kinds. We know a subject ourselves,
or we know where we can find information upon it.*

- Samuel Johnson

Bibliography

[Atk95] R. Atkinson, "Security Architecture for the Internet Protocol", IETF RFC 2401, August 1995.

[Avr92] Z. Avramovic, "Policy based routing in the Defense Information System Network," *Proceedings of the IEEE Military Communications Conference*, vol. 3, pp. 1210-1214, 1992.

[Ber01] Y. Bernet, *Networking Quality of Service and Windows Operating Systems*, New Riders Publishing, November 2001.

[Blu01] Bluetooth Consortium, "Bluetooth Specification Version 1.1," February 2001.

[Bon] BonnMotion tool.

Available at <http://web.informatik.uni-bonn.de/IV/Mitarbeiter/dewaal/BonnMotion/>.

[Bra89] H-W. Braun, "Models of Policy Based Routing," IETF RFC 1104, June 1989.

[BBC+98] S. Blake *et al.*, "An Architecture for Differentiated Services," IETF RFC 2475, December 1998.

[BG03] L. Bao and J. Garcia-Luna-Aceves, "Topology Management in Ad Hoc Networks," *Proceedings of the ACM International Symposium on Mobile Ad Hoc Networking and Computing*, pp. 129-140, June 2003.

[BH01] J. Brand and G. Hartwig, "Ad Hoc Network Management with C2 Data Models," *Proceedings of the IEEE Southeast Conference*, pp. 100-104, March 2001.

[BMT+98] R. Bagrodia *et al.*, "ParSec: A Parallel Simulation Environment for Complex Systems," *IEEE Computer Magazine*, vol. 31, no. 10, pp. 77-85, October 1998.

[BZB+97] R. Braden, L. Zhang, S. Berson, S. Herzog and S. Jamin, "Resource ReReservation Protocol (RSVP) – Version 1 Functional Specification," IETF RFC 2205, September 1997.

[Che99] S. Chen, "Routing Support For Providing Guaranteed End-To-End Quality-Of-Service," PhD thesis, University of Illinois, Urbana-Champaign, 1999.

Available at: <http://cairo.cs.uiuc.edu>.

[Cis] Cisco QoS Policy Manager.

Available at <http://www.cisco.com/warp/public/cc/pd/wr2k/qoppmn/index.shtml>.

[Cla89] D. Clark, "Policy Routing in Internet Protocols," IETF RFC 1102, May 1989.

[CBD02] T. Camp, J. Boleng and V. Davies, "A Survey of Mobility Models for Ad Hoc Network Research," *Wireless Communication and Mobile Computing: Special issue on Mobile Ad Hoc Networking: Research, Trends and Applications*, vol. 2, no. 5, pp. 483-502, 2002.

[CHC+01] T. Clausen, G. Hansen, L. Christensen and G. Behrmann, "The Optimized Link State Routing Protocol, Evaluation through Experiments and Simulation," *Proceedings of the IEEE Symposium on Wireless Personal Mobile Communications*, September 2001.

[CJ03] T. Clausen and P. Jacquet, "Optimized Link State Routing Protocol," IETF Internet-draft, draft-ietf-manet-olsr-11.txt, work in progress, July 2003.

[CJS99] W. Chen, N. Jain and S. Singh, "ANMP: Ad Hoc Network Management Protocol," *IEEE Journal on Selected Areas in Communications*, vol. 17, no. 8, pp. 1506-1531, August 1999.

[CM99] S. Corson and J. Macker, "Mobile Ad Hoc Networking (MANET): Routing Protocol Performance Issues and Evaluation Considerations," IETF RFC 2501, January 1999.

- [CM01] S. Chakrabarti and A. Mishra, "QoS Issues in Ad Hoc Wireless Networks," *IEEE Communications Magazine*, vol. 39, no. 2, pp. 142-148, February 2001.
- [CN99] S. Chen and K. Nahrstedt, "Distributed Quality-of-Service Routing in Ad Hoc Networks," *IEEE Journal on Selected Areas in Communications*, vol. 17, no. 8, pp. 1488-1505, August 1999.
- [CPB+03] K. Channakeshava, K. Phanse, B. Ravindran, L. DaSilva and S. Midkiff, "IP Quality of Service Support for Real-Time Systems," work in progress, 2003.
- [CS98] M. Chatzaki and S. Sartzetakis, "Multilevel QoS-Policy based routing management architecture appropriate for heterogeneous network environments," *Proceedings of the SPIE Broadband European Networks and Multimedia Services*, vol. 3408, pp. 128-138, May 1998.
- [CS99] D. Chalmers and M. Sloman, "A Survey of Quality of Service in Mobile Computing Environments," *IEEE Communications Survey*, vol. 2, no. 2, Second Quarter 1999.
- [CS00] D. Comer and D. Stevens, *Internetworking with TCP/IP: Client-Server Programming and Applications, Linux/Posix Sockets Version*, Prentice Hall Publication, September 2000.
- [CSD+01] K. Chan *et al.*, "COPS usage for Policy Provisioning (COPS-PR)," IETF RFC 3084, March 2001.
- [CSH+03] K. Chan, R. Sahita, S. Hahn and K. McCloghrie, "Differentiated Services Quality of Service Policy Information Base," IETF RFC 3317, March 2003.
- [Dif] Differentiated Services on Linux tool. Available at <http://diffserv.sourceforge.net/>.
- [Dis] The Distributed Management Task Force (DMTF). <http://www.dmtf.org>.
- [DA99] T. Dierks and C. Allen, "The TLS Protocol Version 1.0," IETF RFC 2246, January 1999.
- [DBC+00] D. Durham *et al.*, "The COPS (Common Open Policy Service) Protocol," IETF RFC 2748, January 2000.

[DP02] L. DaSilva and K. Phanse, "Prioritizing Access to Scarce Resources: Network Survivability through Policy-Supported Quality of Service," in press, *International Journal of Critical Infrastructures*, 2003.

[Eth] The Ethereal Network Analyzer. Available at <http://www.ethereal.com/>.

[EHA01] E. Elmallah, H. Hassanein and H. AboEIFotoh, "Supporting QoS Routing in Mobile Ad Hoc Networks using Probabilistic Locality and Load Balancing," *Proceedings of the IEEE Global Telecommunications Conference*, vol. 5, pp. 2901-2906, November 2001.

[Gas02] M. Gast, *802.11 Wireless Networks: The Definitive Guide*, O'Reilly & Associates, April 2002.

[Ger00] R. German, *Performance Analysis of Communications Systems: Modeling with Non-Markovian Stochastic Petri Nets*, John Wiley & Sons, May 2000.

[Glo] GloMoSim network simulator. Available at <http://pcl.cs.ucla.edu/projects/glomosim/>.

[GGG+02] D. Gavalas, D. Greenwood, M. Ghanbari and M. O'Mahony, "Hierarchical network management: a scalable and dynamic mobile agent-based approach," *Computer Networks*, vol. 38, no. 6, pp. 693-711, Elsevier Science, April 2002.

[GGK01] P. Gupta, R. Gray, and P. Kumar, "An Experimental Scaling Law for Ad Hoc Networks," University of Illinois at Urbana-Champaign, May 2001. Available at http://black1.csl.uiuc.edu/~prkumar/ps_files/exp.pdf.

[GPV+99] E. Guttman, C. Perkins, J. Veizades and M. Days, "Service Location Protocol version 2," IETF RFC 2608, June 1999.

[GT01] G. Guichal and C. Toh, "An Evaluation of Centralized and Distributed Service Location Protocols for Pervasive Wireless Networks," *Proceedings of the IEEE International Symposium on Personal, Indoor and Mobile Radio Communications*, vol. 2, pp. E55-E61, 2001.

[GY95] G. Goldszmidt and Y. Yemini, "Distributed Management by Delegation," *Proceedings of the International Conference on Distributed Computing Systems*, pp. 333-340, May 1995.

[HAK03] H. Harroud, M. Ahmed and A. Karmouch, "Policy-Driven Personalized Multimedia Services for Mobile Users," *IEEE Transactions on Mobile Computing*, vol. 2, no. 1, January-March 2003.

[Hie] Hierarchical Token Bucket (HTB) Packet Scheduler.

Available at <http://luxik.cdi.cz/~devik/qos/htb/>.

[HBC+00] S. Herzog *et al.*, "COPS usage for RSVP," IETF RFC 2749, January 2000.

[Ily02] M. Ilyas, ed., *The Handbook of Ad Hoc Wireless Networks*, CRC Press, December 2002.

[Int] Intel Policy-Based Network Management.

Available at <http://www.intel.com/labs/manage/pbnm/index.htm>.

[Int-1] Internet Engineering Task Force (IETF): <http://www.ietf.org>.

[Ipe] Iperf TCP/UDP Bandwidth Measurement Tool.

Available at <http://dast.nlanr.net/Projects/Iperf/>.

[Kos01] D. Kosiur, *Understanding Policy-Based Networking*, John Wiley & Sons, Inc., 2001.

[KBC97] Krawczyk, H., Bellare, M. and R. Canetti, "HMAC: Keyed-Hashing for Message Authentication," IETF RFC 2104, February 1997.

[KF02] H. Koubaa and E. Fleury, "Service Location Protocol Overhead in the Random Graph Model for Ad Hoc Networks," *Proceedings of the IEEE Symposium on Computers and Communications*, pp. 49-54, July 2002.

[KR00] D. Kidston and J. Robinson, "Distributed Network Management For Coalition Deployments," *Proceedings of the IEEE Military Communications Conference*, vol. 1, pp. 460-464, October 2000.

- [Leo93] A. Leon-Garcia, *Probability and Random Processes for Electrical Engineering*, Addison-Wesley Publishing, 2nd ed., July 1993.
- [Lew96] L. Lewis, "Implementing Policy in Enterprise Networks," *IEEE Communications Magazine*, vol. 34, no. 1, pp. 50-55, January 1996.
- [Lin00] A. Lindfors, "Policy Based Management in Ad-Hoc Networks," Helsinki University of Technology, technical report, May 26, 2000. Available at http://www.tml.hut.fi/Opinnot/Tik-110.551/2000/papers/policy_management/internetworking.html.
- [Lin01] C. Lin, "On-Demand QoS Routing in Multihop Mobile Networks," *Proceedings of the Annual Joint Conference of the IEEE Computer and Communications Societies*, vol.3, pp. 1735-1744, 2001.
- [Lin03] T. Lin, "A Framework and a Novel Protocol for Routing in Wireless Ad Hoc Networks," PhD Preliminary Examination Research Proposal, Virginia Tech, 2003.
- [LAZ+00] S. Lee, G. Ahn, X. Zhang and A. Campbell, "INSIGNIA: An IP-based Quality of Service Framework for Mobile Ad Hoc Networks," *Journal of Parallel and Distributed Computing*, vol. 60, no. 4, pp. 374-406, April 2000.
- [LF93] A. Leinwand and K. Fang, *Network Management: A Practical Perspective*, Addison-Wesley, 1993.
- [LG97] C. R. Lin and M. Gerla, "Adaptive clustering for mobile wireless networks," *IEEE Journal on Selected Areas of Communications*, vol. 15, no. 7, pp. 1265-1275, September 1997.
- [LG01] S. Lee and M. Gerla, "Split Multipath Routing with Maximally Disjoint Paths in Ad hoc Networks," *Proceedings of the IEEE International Conference on Communications*, vol. 10, pp. 3201-3205, June 2001.
- [LL99] C. Lin and Jai-Shing Lui, "QoS Routing in Ad Hoc Wireless Networks," *IEEE Journal on Selected Areas in Communications*, vol. 17, no.8, pp. 1426-1438, August 1999.

[LMP02] T. Lin, S.F. Midkiff, and J.S. Park, “A Dynamic Topology Switch for the Emulations of Wireless Mobile Ad Hoc Networks,” *Proceedings of the Annual IEEE Conference on Local Computer Networks*, pp. 791-798, November 2002.

[LMP03] T. Lin, S.F. Midkiff, and J.S. Park, “Approximation Algorithms for Minimal Connected Dominating Sets and Application with Routing Protocol in Wireless Ad Hoc Networks,” *Proceedings of the IEEE International Performance Computing and Communications Conference*, pp. 157-164, April 2003.

[LT00] E. Liden and A. Torger, “Implementation and Evaluation of the Common Open Policy Service (COPS) Protocol and its use for Policy Provisioning,” Masters thesis, Department of Computer Science and Electrical Engineering, Lulea University of Technology, January 2000.

[Mil92] D. Mills, “Network Time Protocol (Version 3) Specification, Implementation and Analysis,” IETF RFC 1305, March 1992.

[Mto] Mtools UDP Traffic Generator.

[MAF02] A. Munaretto, N. Agoulmine and M. Fonseca, “Policy-based Management of Ad Hoc Enterprise Networks,” *Proceedings of the Workshop of the HP OpenView University Association*, June 2002.

[MBJ99] D. Maltz, J. Broch and D. Johnson, “Experiences Designing and Building a Multi-Hop Wireless Ad Hoc Network Testbed,” Carnegie Mellon University, March 1999. Available at <http://reports-archive.adm.cs.cmu.edu/anon/1999/CMU-CS-99-116.pdf>.

[MG98] A. Muir and J. Garcia-Luna-Aves, “An Efficient Packet Sensing MAC Protocol for Wireless Networks,” *ACM Journal on Mobile Networks and Applications*, vol. 3, no. 2, pp. 221-234, 1998.

[MPS+99] K. McCloghrie, D. Perkins, J. Schoenwaelder, J. Case, M. Rose and S. Waldbusser, “Structure of Management Information Version 2 (SMIv2)”, IETF RFC 2578, April 1999.

[MR01] I. Marshall and C. Roadknight, "Management of Future Data Networks," *Proceedings of the IEEE International Conference on Systems, Man, and Cybernetics*, vol. 2, pp. 1143-1148, 2001.

[MST00] M. Mirhakkak, N. Schult and D. Thomson, "Dynamic Quality of Service for Mobile Ad hoc Networks," in the *Proceedings of the Annual Workshop on Mobile and Ad Hoc Networking and Computing*, pp. 137-138, August 2000.

[Nav98] "Information Technology Interoperability," Naval Research Advisory Committee Report, NRAC 98-2, November 1998.

[Net] The Network Simulator (NS-2). Available at <http://www.isi.edu/nsnam/ns/>.

[Net-2] Network Time Protocol Project. Available at <http://www.ntp.org/>.

[NBM03] T. Nguyen, N. Boukhatem, and G. Pujolle, "COPS-SLS usage for dynamic policy-based QoS management over heterogeneous IP networks," *IEEE Network Magazine*, vol. 17, no. 3, pp. 44-50, June 2003.

[Opt-1] INRIA Optimized Link State Routing (OLSR) Protocol Implementation. Available at <http://menetou.inria.fr/olsr/>.

[Opt-2] OLSR for Windows 2000 and Pocket PC.
Available at <http://reptar.grc.upv.es/~calafate/olsr/olsr.htm>.

[Per00] C. Perkins, ed., *Ad Hoc Networking*, Addison-Wesley Publishing, December 2000.

[Pet81] J. Peterson, *Petri Net Theory and the Modeling of Systems*, Prentice Hall, June 1981.

[Pol] IETF Policy Framework Working Group: <http://www.ietf.org/html.charters/policy-charter.html>.

[PD03] K. Phanse and L. DaSilva, "Addressing the Requirements of QoS Management for Wireless Ad Hoc Networks," *International Journal on Computer Communications*, vol. 26, no. 12, pp. 1263-1273, July 2003.

[PDM02] K. Phanse, L. DaSilva and S. Midkiff, "A Taxonomy and Experimental Evaluation of Policy Architectures for Bandwidth-constrained Networks," Internal Report, Virginia Tech, May 2002.

[PDM03] K. Phanse, L. DaSilva, and S. Midkiff, "Design and Demonstration of Policy-Based Management in a Multi-Hop Ad Hoc Network Testbed," submitted for publication, 2003.

[Qua] QualNet network simulator. Available at <http://www.scalable-networks.com>.

[Rap] IETF Resource Allocation Protocol (RAP) working group:
<http://www.ietf.org/html.charters/rap-charter.html>

[Rap-1] IETF RAP working group mailing list archive (thread: *COPS vs. SNMP*). Available at <http://www.atm.tut.fi/list-archive/rap/msg00840.html>.

[Rap-2] IETF RAP working group mailing list archive (thread: *why i should like pibs*). Available at <http://www.atm.tut.fi/list-archive/rap/msg00859.html>.

[Riv92] R. Rivest, "The MD5 Message-Digest Algorithm," IETF RFC 1321, April 1992.

[Red] Red Hat Linux: <http://www.redhat.com>.

[Rob94] T. Robertazzi, *Computer Networks and Systems: Queuing Theory and Performance Evaluation*, 2nd ed., Springer-Verlag New York Inc., 1994.

[RH00] R. Ramanathan and R. Hain, "An ad hoc wireless testbed for scalable, adaptive QoS support," *Proceedings of the IEEE Wireless Communications and Networking Conference*, vol. 3, pp. 998-1002, 2000.

- [RS98] R. Ramanathan and M. Steenstrup, "Hierarchically-organized, multihop mobile wireless networks for quality-of-service support," *ACM/Baltzer Mobile Networks and Applications: Special issue on Mobile Multimedia Communications*, vol. 3, no. 1, pp. 101-119, June 1998.
- [RVS+99] R. Rajan, D. Verma, S. Kamat, E. Felstaine and S. Herzog, "A Policy Framework for Integrated and Differentiated Services in the Internet," *IEEE Network Magazine*, vol. 13, no. 5, pp. 36-41, September/October 1999.
- [Sal99] Salutation Consortium, "Salutation Architecture Specification Version 2.1," 1999.
- [Sta93] W. Stallings, *SNMP, SNMPv2, and CMIP: The Practical Guide to Network Management Standards*, 1st edition, Addison-Wesley, 1993.
- [Ste00] M. Steenstrup, "Cluster-based networks," *Ad Hoc Networking*, C. E. Perkins, ed., Ch. 4, pp. 75-138, Addison Wesley, December 2000.
- [Ste02] J. Steele, "Determining Communications Resource Feasibility in a Tactical Communications Network," M. S. thesis, Virginia Tech, April 2002.
- [Str99] J. Strassner, *Directory-Enabled Networks*, New Riders Publishing, 1999.
- [Sun00] Sun Microsystems, "Jini Architecture Specification Version 1.1," October 2000.
- [SB00] M. Storey and G. Blair, "Resource Configuration in Ad Hoc Networks: The MARE Approach," *Proceedings of the IEEE Workshop on Mobile Computing Systems and Applications*, pp. 60-69, 2000.
- [SM01] C. Siva Ram Murthy and G. Manimaran, *Resource Management in Real-Time Systems and Networks*, MIT Press, March 2001.
- [SSB99] P. Sinha, R. Sivakumar and V. Bharghavan, "CEDAR: a Core-Extraction Distributed Ad Hoc Routing algorithm," *IEEE Journal on Selected Areas in Communications*, vol. 17, no. 8, pp. 1454-1465, August 1999.

[SSJ03] C. Shen, C. Srisathapornphat and C. Jaikaeo, "An Adaptive Management Architecture for Ad Hoc Networks," *IEEE Communications Magazine*, vol. 41, no. 2, pp.108-115, February 2003.

[SV95] S. Floyd and V. Jacobson, "Link-sharing and Resource Management Models for Packet Networks," *IEEE/ACM Transactions on Networking*, vol. 3, no. 4, pp. 365-386, August 1995.

[SZH+01] A. Sethi, D. Zhu, V. Hnatyshin and P. Kokati, "Battlefield Network Application of the SHAMAN Management System," *Proceedings of the IEEE Military Communications Conference*, vol. 2, pp. 933-937, October 2001.

[Tcp] Tcpcdump. Available at <http://www.tcpdump.org/>.

[Toh01] C. K. Toh, *Ad Hoc Mobile Wireless Networks: Protocols and Systems*, Prentice Hall, December 2001.

[Toh01-1] C. K. Toh, "Maximum battery life routing to support ubiquitous mobile computing in wireless ad hoc networks," *IEEE Communications Magazine*, vol. 39, no. 6, pp. 138-147, June 2001.

[Trp] TRace Plot Real-time (trpr). Available at <http://manimac.itd.nrl.navy.mil/Tools/trpr.html>.

[TNC+02] Y. Tseng, S. Ni, Y. Chen, and J. Sheu, "The Broadcast Storm Problem in a Mobile Ad Hoc Network," *ACM Wireless Networks*, vol. 8, no. 2/3, pp. 153-167, March-May 2002.

[Ver00] D. Verma, *Policy-Based Networking: Architectures and Algorithms*, New Riders Publishing, November 2000.

[Vic] Video Conferencing Tool. Available at <http://www-nrg.ee.lbl.gov/vic/>.

[VBJ01] D. Verma, M. Beigi and R. Jennings, "Policy based SLA Management in Enterprise Networks," *Proceedings of the Workshop on Policies for Distributed Systems and Networks*, pp. 137-152, January 2001.

[VCA02] D. Verma, S. Calo and K. Amiri, "Policy based Management of Content Distribution Networks," *IEEE Network Magazine*, vol. 16, no. 2, pp. 34-39, March-April 2002.

[VGP+97] J. Verizades, E. Guttman, C. Perkins, and S. Kaplan, "Service Location Protocol," IETF RFC 2165, June 1997.

[Web98] M. Webster, ed., *The Merriam-Webster's Collegiate Dictionary*, Merriam-Webster, Inc., 10th ed., 1998.

[Wes01] D. West, *Introduction to Graph Theory*, Prentice-Hall, 2nd ed., 2001.

[Wir] Wireless Tools for Linux.

Available at http://www.hpl.hp.com/personal/Jean_Tourrilhes/Linux/Tools.html.

[WK02] J. Walker and A. Kulkarni, "COPS Over TLS," IETF Internet-Draft, draft-ietf-rap-cops-tls-03.txt, work in progress, April 2002.

[WL02] K. Wang and L. Baochun, "Efficient and Guaranteed Service Coverage in Partitionable Mobile Ad-Hoc Networks," *Proceedings of the Annual Joint Conference of the IEEE Computer and Communications Societies*, vol. 2, pp. 1089-1098, June 2002.

[WSS+01] A. Westerinen *et al.*, "Terminology for Policy-Based Management," IETF RFC 3198, November 2001.

[XSL+00] H. Xiao, W. Seah, A. Lo and K. Chua, "A Flexible Quality of Service Model for Mobile Ad-Hoc Networks," in the *Proceedings of the IEEE Vehicular Technology Conference*, vol. 1, pp. 445-449, 2000.

[YLG01] F. Yong-xin, Z. Lin-liang and W. Guang-xing, "A Clustering Algorithm applied to the Network Management on Mobile Ad hoc Network," in the *Proceedings of the IEEE International Conference on Info-tech and Info-net*, vol. 2, pp. 626-631, 2001.

[YLN03] J. Yoon, M. Liu, and B. Noble, "Random Waypoint Considered Harmful," *Proceedings of the Annual Joint Conference of the IEEE Computer and Communications Societies*, vol. 2, pp. 1312-1321, March 2003.

[YPG00] R. Yavatkar, D. Pendarakis and R. Guerin, "A Framework for Policy-based Admission Control," IETF RFC 2753, January 2000.

[ZH99] L. Zhou and Z. Haas, "Securing Ad Hoc Networks," *IEEE Network Magazine*, vol. 13, no. 6, pp. 24-30, November/December 1999.

Vita

Kaustubh S. Phanse was born on March 25, 1976 in the city of Mumbai (erstwhile Bombay), India. He earned his Bachelors degree in Electronics and Telecommunications from the Vivekananda Education Society's Institute of Technology, University of Mumbai, in May 1998. He then joined the MSEE program in the Bradley Department of Electrical and Computer Engineering at Virginia Tech in Fall 1998. After spending three semesters in the Blacksburg campus, he moved to Virginia Tech's Alexandria Research Institute (ARI) in Spring 2000, where he spent the remaining years of his graduate student life. After completing his Masters degree in Fall 2000, he joined the Ph.D. program in Electrical Engineering. Kaustubh is a student member of the IEEE, IEEE ComSoc, IEEE Computer and ASEE. After completing his Ph.D., he hopes to pursue a career in the academia.

Publications in Refereed Journals

- K. Phanse and L. DaSilva, "Addressing the Requirements of QoS Management for Wireless Ad Hoc Networks," *International Journal on Computer Communications*, vol. 26, no. 12, pp. 1263-1273, Elsevier Science, July 2003.
- L. DaSilva and K. Phanse, "Prioritizing Access to Scarce Resources: Network Survivability through Policy-Supported Quality of Service," in press, *International Journal of Critical Infrastructures*, 2003.

Publications in Refereed Conferences

- L. DaSilva, V. Dham, K. Phanse and K. Kidambi, "Traffic Management for Voice/Data Convergence in ADSL," *Proceedings of the International Teletraffic Congress*, September 2001.
- K. Phanse, L. DaSilva and K. Kidambi, "Characterization of Performance of TCP/IP over PPP over ATM over Asymmetric Links," *Proceedings of the International IEEE Conference on Communications and Computer Networks*, October 2000.
- K. Phanse, L. DaSilva and K. Kidambi, "Effects of Competing Traffic on TCP/IP Performance over Asymmetric Links," *Proceedings of the Annual IEEE Conference on Local Computer Networks*, October 2000.