Available online at www.sciencedirect.com

SCIENCE @ DIRECT®

ELSEVIER

Ad Hoc Networks 3 (2005) 389–401

Ad Hoc Networks

www.elsevier.com/locate/adhoc

Design and demonstration of policy-based management in a multi-hop ad hoc network

Kaustubh S. Phanse *, Luiz A. DaSilva, Scott F. Midkiff

Bradley Department of Electrical and Computer Engineering, Virginia Polytechnic Institute and State University, Alexandria, VA 22314, USA

Received 16 May 2003; received in revised form 4 September 2003; accepted 24 September 2003
Available online 29 November 2003

Abstract

In this paper, we propose a policy-based framework for the management of wireless ad hoc networks and briefly describe a characteristics-based taxonomy that provides a platform to analyze and compare different architectural choices. We develop a solution suite that helps achieve our goal of a self-organizing, robust and efficient management system. One of the main contributions of this work is the prototype implementation and testing of the mechanisms and protocols comprising our framework in a multi-hop ad hoc network environment. Experiments are conducted using both an emulated ad hoc network testbed and a true wireless testbed. Degradation in management system performance is observed as the number of hops between a policy server and client increases. Our proposed k-hop clustering algorithm alleviates this problem by limiting the number of hops between a server and client. We demonstrate the operation of our prototype implementation, illustrating QoS management in a multi-domain ad hoc network environment using the proposed cluster management, redirection, and policy negotiation mechanisms.

Keywords: Policy-based networking; Resource management; Service discovery; Ad hoc networks

1. Introduction

Mobile ad hoc networks are characterized by dynamic topologies, 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 such networks.

Policy-Based Network Management (PBNM) is one network management approach that has met with considerable interest in the networking community [1,2]. 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 simplified, logically centralized and automated control over the entire network. PBNM can simplify administration of complex operational characteristics of a network, including Quality of Service (QoS), access control, network security, and IP address allocation. However, so far, the work on policy-based network management [1–4] has focused on large fixed networks such as enterprise networks,

* Corresponding author.
E-mail address: kphanse@vt.edu (K.S. Phanse).

1570-8705/$ - see front matter © 2003 Elsevier B.V. All rights reserved.
doi:10.1016/j.adhoc.2003.09.013

content provider networks, and Internet service provider (ISP) networks.

In this paper, we discuss how to extend and apply the policy-based approach for managing QoS in ad hoc networks. QoS requires mechanisms to support service differentiation as well as mechanisms for QoS management. The former concerns itself with *how* to achieve performance objectives of heterogeneous flows; the latter concerns itself with *who* should be entitled to preferential treatment by the network. QoS architectures such as SWAN [5], INSIGNIA [6] and dRSVP [7] address the means for service differentiation, including marking, classification, scheduling and others, and often assume some external mechanism such as pricing will determine who should have access to what level of service. However, the control structure (in support of authentication, authorization, and dynamically changing policies) required for QoS provisioning and management in ad hoc networks is not yet well understood, and is the focus of our research. The dynamic nature of ad hoc networks calls for a control structure that allows automated QoS management and one that supports dynamic admission control or bandwidth allocation based on different policies such as bandwidth availability, traffic ownership (e.g., the identity of the user, application, or organization from which the traffic originates), and temporal elements (e.g., time of day).

A policy-based approach addresses most of the key requirements of an ad hoc network management system, namely automation, self-organizing capability, robustness and efficiency (for a survey of related work and detailed discussion of how these requirements are met by a policy-based management system, the reader is referred to [8]). The fundamental challenge in extending the policy-based approach to ad hoc networks is to adapt this conceptually centralized approach to a distributed, infrastructure-independent environment.

In this paper, we describe a PBNM framework for ad hoc networks consisting of a suite of solutions that together help meet the challenges outlined in [8]. The key components of this solution suite are *automated service discovery, k-hop clustering*, and *Dynamic Service Redundancy (DynaSeR)*.

We also address policy inter-operability issues in a multi-domain ad hoc inter-network—formed by a multi-organization consortium, such as the US Navy's Coalition Wide Area Network (C-WAN) [9]. We propose and demonstrate a signaling mechanism for inter-domain policy negotiation that makes seamless QoS feasible in such networks.

Unlike many research efforts in the field of ad hoc networks that are based solely on simulations, we implement and test our management scheme in a multi-hop ad hoc network testbed. We believe that the challenges of operating in an actual ad hoc network environment are not always exposed in a simulation environment or through theoretical analysis; experimental evaluation of proposed solutions is critical. Prior experiences of researchers (e.g., [10,11]) support our concerns. We implement a prototype of our PBNM system and illustrate its operation in a multi-hop ad hoc network testbed.

2. Policy-based management framework for wireless ad hoc networks

An automated, intelligent, efficient and robust management structure is needed to manage ad hoc networks. In this section, we describe the various modules that constitute the framework. In particular, we focus on the policy architectural and distribution, resource discovery and policy provisioning aspects of the framework; the underlying techniques we propose—*k*-hop cluster management, the DynaSeR solution, service discovery and signaling for inter-domain policy negotiation—are presented.

Using a systems approach, we decompose the framework into functional blocks. This approach highlights the inter-dependencies among the various components and the complex functional tasks that need to be carried out by a management system. A comprehensive representation of a management system is lacking in most prior published research (which focuses mainly on network monitoring, e.g., [12,13]) and is one of the contributions of this work.

The seven key modules that constitute the framework are as follows.

2.1. Policy specification

The policy specification is a mapping of the overall network goals (e.g., QoS specification) into network-wide policies. Typically, the high-level policies are reasonably static, while lower-level policies may change according to network utilization or time of day.

2.2. Policy architecture and distribution

2.2.1. Types of architectures

We have proposed a characteristics-based taxonomy of policy architectures in [14]. Here we provide a brief overview of the taxonomy. The taxonomy is based on four characteristics: *locus of control*, *locus of information*, *policy distribution mode*, and *tiers of control*. The policy architecture taxonomy is summarized in Table 1.

The taxonomy broadly classifies the various architectures into three categories based on the policy distribution model used. These categories are: *outsourced* (all policy decisions are outsourced by a client to a remote server), *provisioned* (clients are configured to make policy decisions locally), and *hybrid* (combination of the outsourcing and provisioning models) architectures, which are then further classified as shown in Table 1.

The taxonomy provides a systematic way to analyze and compare the applicability of one or more architectures to the network environment of interest. We have identified the hybrid architecture as the most promising for implementing our management framework in an ad hoc network environment [14]. The provisioning of clients allows them to make decisions locally minimizing overhead, while outsourcing provides support for dynamic policies and inter-domain mobility.

2.2.2. Protocol for policy distribution

Several mechanisms exist for policy distribution in a network [1]: 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), or Lightweight Directory Access Protocol (LDAP).

We choose the COPS for PRovisioning (COPS-PR) [15], an extension of the COPS protocol [16], for policy distribution. COPS-PR integrates the outsourcing and provisioning models, thus allowing the flexibility to support a hybrid architecture.

Some of the features that make COPS-PR a promising choice are: event-driven control (i.e., there is no polling) and asynchronous notification, structured row-level access and atomic transactional model, support for fault tolerance and security mechanisms, and reliable transport using

Table 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
Provisioned	DDO	Distributed	Distributed	Outsourcing	1
	DDP	Distributed	Distributed	Provisioning	2
	DDP-hierarchical	Distributed	Distributed	Provisioning	>2
Hybrid	DDOP	Distributed	Distributed	Outsourcing and provisioning	2
	DDOP-hierarchical	Distributed	Distributed	Outsourcing and provisioning	>2

persistent TCP connections. COPS-PR may also co-exist and interact with other management protocols such as SNMP.

2.2.3. Automated and self-organizing control structure

A self-organizing and automated management system is the key for effective management in an ad hoc network environment. We propose a suite of solutions and techniques that address these requirements. The components of this solution suite include *k*-hop cluster management, service discovery, and dynamic service redundancy.

During initial deployment of a wireless ad hoc network, we assume that a certain number of policy servers are present in the network and initially serve as cluster heads. Other nodes can eventually become PDPs (and, hence, cluster heads) through the process of delegation, described below. All the policy clients within *k*-hops from a server are eligible for service from that policy server. Each policy server along with its clients forms a cluster, as shown in Fig. 1.

Whenever a proactive ad hoc routing protocol is adopted, the *k*-hop cluster management can be implemented by taking advantage of the routing information available with the routing daemon. Using this topology information, the policy servers can track nodes moving in and out of the clusters with minimal additional clustering protocol overhead. We implement the proposed method by interfacing our PBNM application with the underlying Optimized Link State Routing (OLSR) protocol daemon (see Section 4).

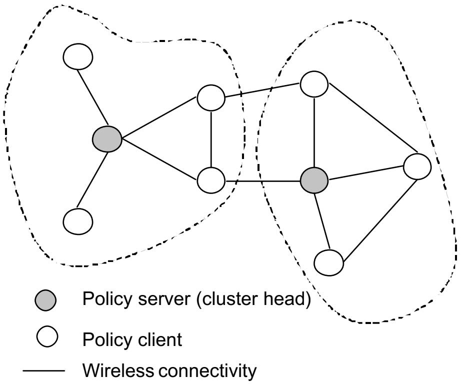


Fig. 1. *k*-hop cluster management; 1-hop ($k = 1$) clusters indicated by dotted lines.

Due to deployment of an insufficient number of policy servers or due to node mobility, it is possible that one or more nodes may not be within *k*-hops of any of the policy servers. To increase policy-based service availability and to allow policy servers to efficiently keep track of network nodes as they move, we further enhance our clustering algorithm using what we call *Dynamic Service Redundancy (DynaSeR)*. The DynaSeR solution consists of two techniques: *redirection* and *delegation*. When a client moves out of a cluster, the server managing the client gathers relevant topology information (from the routing daemon) to detect whether the client is within *k*-hops of any other servers. If it is, then it *redirects* the client to the appropriate server. The client is now a part of the new cluster managed by the "Redirected PDP." This can be termed as a *server-centric* way of implementing cluster management. If a policy client is not within *k* hops from any of the existing servers, an existing policy server *delegates* an appropriate network node to assume serving responsibilities for that client. For both delegation and redirection, the decision of which node should act as a server may be based on factors such as connectivity, processing load, bandwidth availability, and remaining battery life. Such "delegated" servers or policy decision points (PDPs) form another tier of control below the super-PDPs, creating a hierarchical control architecture. Simulation results reported by us in [17] indicate that this delegation scheme considerably improves the service coverage of the PBNM system, while allowing the use of smaller cluster sizes.

We propose and implement a lightweight service discovery mechanism to facilitate automated discovery of policy servers in the network. Two types of messages are used: Service Advertisement (SA) and Client Service Request (CSRQ). A policy server periodically advertises itself via a limited *k*-hop broadcast of the SA message. A client that does not receive an SA message within a certain time interval broadcasts a CSRQ message. The server that may have moved within *k*-hops of the client responds with a unicast SA message. Alternatively, a client node that is currently being serviced, upon hearing a CSRQ message, may volunteer to act as a delegated server.

The main motivation for this type of clustering is to limit the number of wireless hops between a client and a server. As will be seen from our results (Sections 3.1 and 3.2), this considerably improves the performance of the management system—keeping the policy response time low and reducing the unpredictability in the performance. Further, fewer hops between a client and a server means fewer resources (e.g., bandwidth and battery life) are used at intermediate nodes for forwarding control messages. The trade-off is generally in the poorer service coverage for smaller cluster sizes k [17].

2.3. Resource discovery

A policy-based management system translates high-level policies (Policy Specification) into device-specific configuration to dictate the use of network resources. To achieve this, the management framework must be aware of the available network resources. This includes active network devices and their capabilities, network topology, bandwidth utilization, etc. This is even more critical in ad hoc networks, where it is crucial for the policy system to keep updated knowledge about the dynamic network topology. Resource discovery calls for additional signaling and/or computation; thus, the trade-off is generally between efficiency (minimal signaling) and accuracy of the information maintained by the management system.

2.4. Policy provisioning

Policy provisioning can be viewed as the phase after policies are distributed, consisting of installing and implementing the policies using device specific mechanisms (e.g., marking, classification and queuing). Thus, policy provisioning directly affects the way in which the various traffic flows in the network are treated. In our implementation, we use a DiffServ-like architecture to provision QoS policies.

2.5. Policy-based routing

A routing approach that honors the defined network policies is called policy-based routing [18,19]. These policies may involve end-users, temporal policies, access control, resource allocation, etc. While policy-based routing has been studied and deployed extensively in wireline networks, its applicability to ad hoc networks is open for further investigation.

2.6. Policy monitoring

To provide robust management of a network, an independent policy monitoring process must ensure that the network meets the high-level goals or specifications. Policy monitoring can be achieved using active (e.g., dummy transactions or sending probe packets) or passive (e.g., measurement-based estimation) methods [1].

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. Using feedback (e.g., policy monitoring) and resource discovery mechanisms, a management system can make intelligent decisions and adapt to the changing network environment. Further discussion of this module is outside the scope of this paper.

3. Experimental evaluation

In [8], we presented our preliminary experimental results to validate our qualitative analysis of policy architectures based on our taxonomy. However, those experiments involved single-hop communications. In an ad hoc network, the number of hops between a policy client and a policy server may change over time and it is important to characterize the performance of the management system as a function of the number of hops.

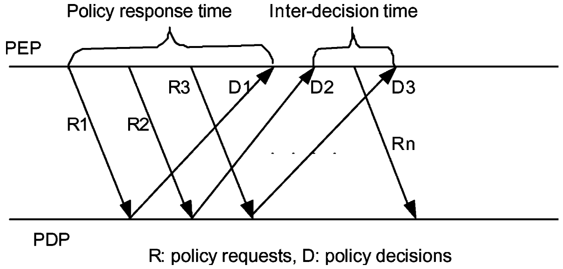


Fig. 2. Illustration of network management metrics—policy response time and inter-decision time—being considered.

In this section, we describe the experiments conducted in a multi-hop ad hoc network testbed, and present our results. We are interested in characterizing the effect of available bandwidth on management system performance, in particular under constrained bandwidth availability typical of many wireless ad hoc networks.

We used the Intel® COPS client software development kit (SDK) [20] to implement a COPS-based management system. Dummy policy requests were generated by running a script at each policy enforcement point (PEP).

The metrics used in this set of experiments are the policy response time and the inter-decision time. The policy response time is 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. The inter-decision time is the difference between consecutive decisions received by the PEP. These are illustrated in Fig. 2.

3.1. Multi-hop ad hoc network (wired testbed)

We emulate different ad hoc network topologies using a software developed at Virginia Tech [21] that allows emulation of dynamic topologies, variable packet drop rates and low bandwidth.

We implemented a CCO architecture (see Table 1). Intermediate nodes used the Optimized Link State Routing protocol daemon (olsrd) [22,23] to route packets between the policy server (PDP) and client (PEP). A low end-to-end bandwidth of 64 kb/s was emulated using the Diffserv on Linux tool [24]. We measured the policy response time and the inter-decision time as a function of the number of hops between the PEP and PDP. Results were collected using multiple iterations for each case; 95% confidence intervals were calculated using the method of batch means [25]. As seen in Fig. 3, the policy response time and inter-decision time increase exponentially with the number of hops, indicating the desirability of 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 2) that controls the number of hops between a PEP and PDP. The confidence intervals for the two metrics are shown in Table 2.

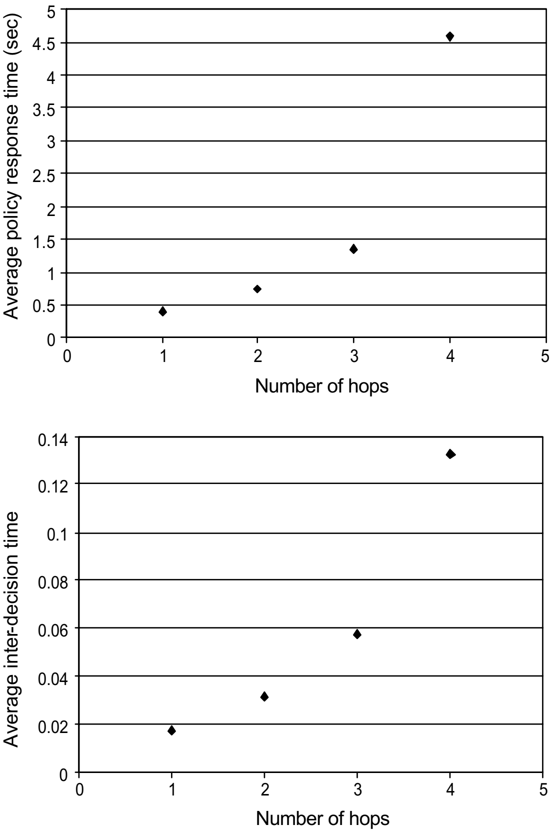


Fig. 3. Policy response time and inter-decision time (in seconds) plotted as a function of the number of hops between the policy client and server.

Table 2
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

3.2. Multi-hop ad hoc network (wireless testbed)

To gain insight into the 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 a wireless testbed. Laptops with IEEE 802.11b wireless PC cards were used in this set of experiments. The wireless cards are based on the Intersil Prism chipset, which supports transmitter power control. Fig. 4 shows the placement of laptops in our work area. A 4-hop wireless network topology is illustrated.

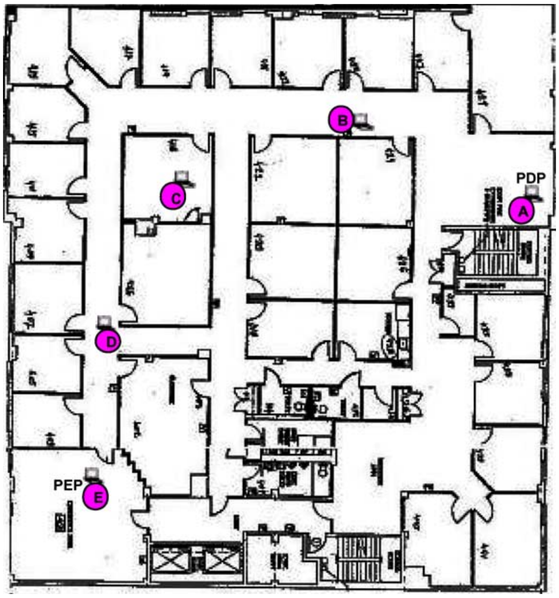


Fig. 4. Layout of the area where we conduct our wireless experiments. Placement of nodes for a 4-hop wireless ad hoc network topology is shown. Nodes A and E are the policy server and client, respectively.

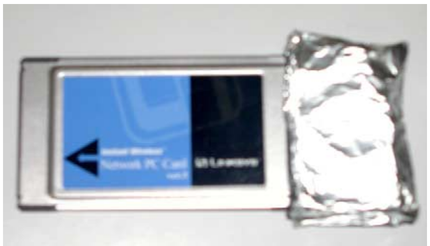


Fig. 5. Antenna portion of the wireless PC card wrapped with aluminum foil "attenuator."

To set up multi-hop topologies in a small work area, we reduced the transmitter power of the wireless cards using the *Wireless Tools for Linux* [26] package. In addition, we used a crude, but effective, way to further reduce the transmitter power. We wrapped the antenna portion of the wireless cards with aluminum foil acting as an attenuator as shown in Fig. 5.

In an otherwise static setup, intermittent loss of routes between end nodes was observed, especially with the increase in the number of hops. Clausen et al. [10] 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 [22]) by modifying the current OLSR daemon.

Results from this set of experiments are shown in Table 3; the policy response time with the confidence intervals is plotted in Fig. 6. Increasing the number of hops resulted in considerable increase in

Table 3
Policy response time and inter-decision time vs. number of hops (wireless testbed)

Number of hops	95% confidence interval	
	Policy response time (ms)	Inter-decision time (ms)
1	425.026 ± 1.342	15.452 ± 0.0008
2	938.346 ± 12.725	32.629 ± 0.0399
3	1374.268 ± 763.494	58.817 ± 5.6528
4	5181.985 ± 17221.85	165.466 ± 20.717

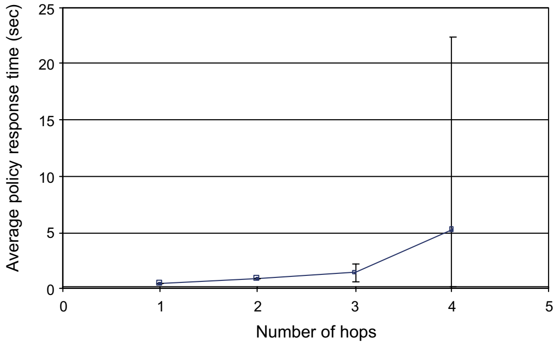


Fig. 6. Policy response time plotted as a function of the number of hops between the policy client and server (wireless testbed); the dotted lines indicate the confidence interval.

the response time and inter-decision time. The policy response time results collected after numerous runs exhibited high variance for the 3-hop and 4-hop scenarios. This indicates that the average values in these two cases are not a good indicator of system performance. A look at the instantaneous values revealed intermittent occurrence of large spikes. For example, values as high as 5 and 15 s were observed for policy response time in the 3-hop and 4-hop scenarios, respectively.

The increase in the response times and the unpredictability of the system with an increase in the number of hops bolsters our proposal of using *k*-hop cluster management to control the number of hops between a policy server and client in a wireless ad hoc environment.

4. Policy-based QoS management

In the previous section, we discussed our experimental results that characterized the policy management architecture performance in a static multi-hop ad hoc network environment. In this section, we briefly describe implementation of our proposed schemes and illustrate our PBNM system at work in a multi-domain mobile ad hoc network.

4.1. Implementation

To implement our proposed schemes, we used the open source COPS API made available by Telia Research [27]. We implemented a policy server and client to incorporate our proposed cluster management, redirection, delegation and service discovery mechanisms. In addition, we propose and implement an extension to the COPS-PR protocol, for inter-PDP communication and policy negotiation. This is of particular importance in a multi-domain network environment [9].

- *Inter-domain policy negotiation:* In a wireless mobile ad hoc network formed by a consortium of different organizations, nodes may move across domains [1] administered by the policies of each individual organization. In general, a node's movement into a foreign domain may have several implications on its operation and performance. 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 applications may be rendered impractical.

  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 [16]. A client embeds both these objects in the COPS *OPEN* message sent to the server for establishing a new COPS connection. The policy negotiation signaling is shown in Fig. 7. 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.

[1] We define the administrative domain to which a node belongs as its "home" domain, while any other domain is referred to as a "foreign" domain.

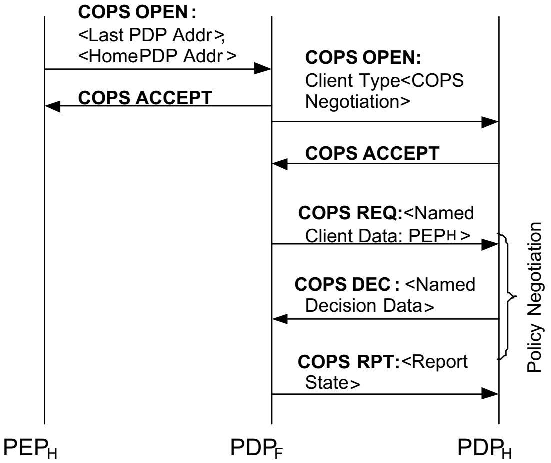


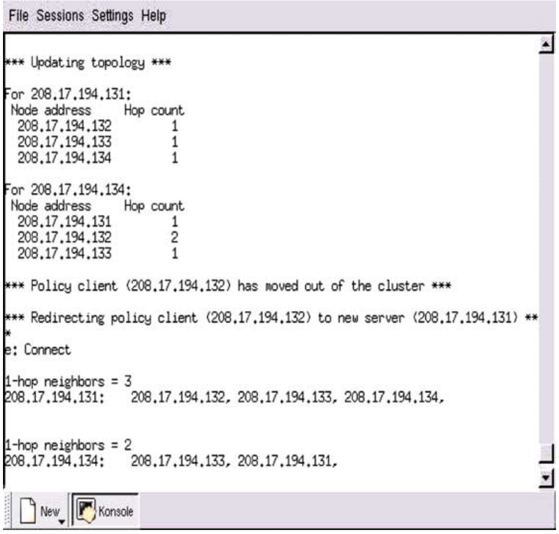
Fig. 7. Signaling for inter-domain policy negotiation.

It then sends a COPS request (*REQ*) message to $\mathrm{PDP_H}$ to download policies relevant to the "alien" client ($\mathrm{PEP_H}$). $\mathrm{PDP_F}$ then adapts its policies to reflect the service level agreement between the domains.

For policy provisioning, we use a partial implementation of the Differentiated Services PIB [28] that provides a simple *if-then* mapping between the node/domain addresses and application flows based on source and destination ports and the corresponding policies for bandwidth allocation.

- *Integration with OLSR:* The *olsrquery* tool, a part of the INRIA OLSR implementation [23], 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. Every 10 s, the thread 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. Fig. 8 shows a snapshot of a policy server exhibiting updated topology information maintained by it.



```
*** 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,
```

Fig. 8. User-interface of a policy server showing topology information gathered from underlying OLSR routing daemon, and implementation of 1-hop cluster management.

- *Cluster management:* We discussed our *k*-hop clustering algorithm in Section 2. Here, we describe its implementation and deployment using the COPS protocol. As mentioned earlier, a policy server maintains topology information relevant to its clients.

  When "significant" topology changes occur, i.e., one or more clients move out of the *k*-hop cluster, the policy server is alerted about these changes, and functions for accessing the topology information (from the linked list) and, hence, for cluster management are invoked. Currently we have implemented the *redirection* mechanism assuming that a client that moves out of the *k*-hop cluster of its policy server, moves within *k*-hops of at least one other policy server. The *delegation* mechanism, applicable to a more general case of node movements, is part of our ongoing research.

  The COPS protocol has some inherent support for the *redirection* mechanism. We take advantage of the "PDP Redirect Address" object in the COPS "Client-Close" message defined in the COPS protocol standard [16]. Whenever a policy server determines that a client has moved out of its *k*-hop cluster, it looks for another policy server within *k*-hops of that client. If more than one policy server is available, it randomly chooses one of the policy servers. It includes the address of that policy server in the "PDP Redirect Address" object and sends it to the client using the "Client-Close" message. Upon receiving this message, the client closes its connection with the current policy server and establishes a new connection with the "redirected" policy server. The user interface of a policy server exhibiting the *redirection* technique is shown in Fig. 8. The corresponding COPS-based signaling, captured using Ethereal [29], is shown in Fig. 9.

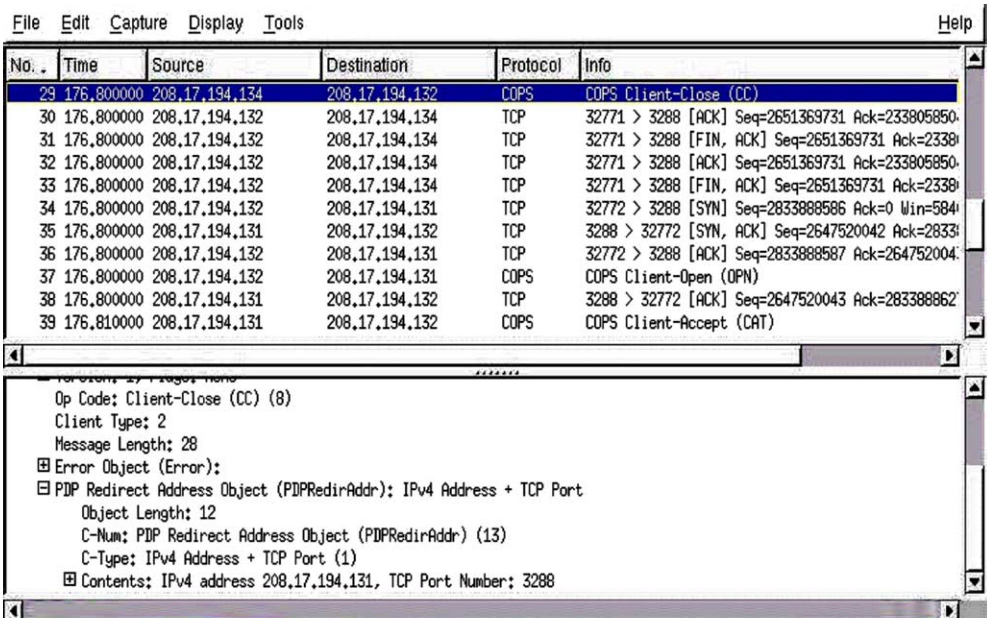
- *Quality of service mechanisms:* In our experiments, we use the Hierarchical Token Bucket (HTB) [30] packet scheduler available in the *Diffserv on Linux* or *tc* tool for QoS provisioning. The HTB scheduler allows us to perform traffic shaping (for low bandwidth emulation) in its *parent* class and then to define hierarchical *child* classes for bandwidth allocation and management.

4.2. Illustration: QoS management in a multi-domain ad hoc network

The demonstration network consists of four laptops: A–D, as shown in Fig. 10. The laptops communicate using IEEE 802.11b wireless cards. The OLSR protocol is used for routing. Nodes B and C represent two policy servers, each controlling a separate organizational network domain. COPS-PR is used for policy distribution.

Node D is a mobile node, currently in its "home" domain administered by policy server B. Node D is transmitting real time video traffic to node A, using the *vic* video conferencing tool [31]. In its home domain, a minimum bandwidth of 64 kb/s is allocated to node D for its video transmission.

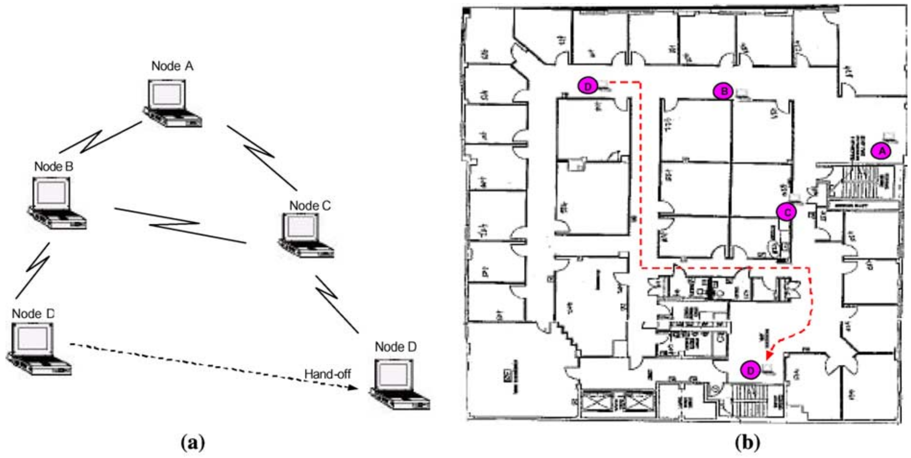
For illustration purposes, we configure our policy-based management system with 1-hop clustering. Policy servers B and C are assumed to have knowledge of each other's existence. A hybrid architecture is implemented. Policy servers are initially loaded with relevant policies and provision the clients with policies. Whenever a node does not have relevant policies, it out-



```
No. . Time       Source          Destination     Protocol Info
   29 176.800000 208.17.194.134  208.17.194.132  COPS     COPS Client-Close (CC)
   30 176.800000 208.17.194.132  208.17.194.134  TCP      32771 > 3288 [ACK] Seq=2651369731 Ack=233805850
   31 176.800000 208.17.194.132  208.17.194.134  TCP      32771 > 3288 [FIN, ACK] Seq=2651369731 Ack=2338
   32 176.800000 208.17.194.132  208.17.194.134  TCP      32771 > 3288 [ACK] Seq=2651369731 Ack=233805850
   33 176.800000 208.17.194.132  208.17.194.134  TCP      32771 > 3288 [FIN, ACK] Seq=2651369731 Ack=2338
   34 176.800000 208.17.194.132  208.17.194.131  TCP      32772 > 3288 [SYN] Seq=2833888586 Ack=0 Win=584
   35 176.800000 208.17.194.131  208.17.194.132  TCP      3288 > 32772 [SYN, ACK] Seq=2647520042 Ack=2833
   36 176.800000 208.17.194.132  208.17.194.131  TCP      32772 > 3288 [ACK] Seq=2833888587 Ack=264752004
   37 176.800000 208.17.194.132  208.17.194.131  COPS     COPS Client-Open (OPN)
   38 176.800000 208.17.194.131  208.17.194.132  TCP      3288 > 32772 [ACK] Seq=2647520043 Ack=283388862
   39 176.810000 208.17.194.131  208.17.194.132  COPS     COPS Client-Accept (CAT)
```

```
    Op Code: Client-Close (CC) (8)
    Client Type: 2
    Message Length: 28
  ⊞ Error Object (Error):
  ⊟ PDP Redirect Address Object (PDPRedirAddr): IPv4 Address + TCP Port
        Object Length: 12
        C-Num: PDP Redirect Address Object (PDPRedirAddr) (13)
        C-Type: IPv4 Address + TCP Port (1)
      ⊞ Contents: IPv4 address 208.17.194.131, TCP Port Number: 3288
```

Fig. 9. 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.



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

Fig. 10. Demonstration scenario depicting a multi-domain wireless ad hoc network. Hosts B and C are policy servers in distinct administrative domains.

sources decisions to the appropriate remote policy servers.

The layout of our work area and illustration of the demonstration are shown in Fig. 10(b), with the movement of node D away from its "home" domain shown as a dotted line. Node D loses direct connection with node B and, instead, connects via a foreign domain, specifically 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 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 (as shown in Fig. 8). Following this redirection, 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. This triggers the policy negotiation phase, where server C sets up a COPS connection with node B and outsources a COPS request (*REQ*) message to obtain policies for node D (the signaling sequence is shown in Fig. 7). After downloading relevant policies, policy server C is now able to allocate a bandwidth in the range of 64–128 kb/s for node D's video application, thus resulting in acceptable video performance. The difference in the received video quality (with and without policy negotiation) is illustrated in Fig. 11.

5. Concluding remarks and future work

The PBNM framework proposed in this paper formalizes the complex functional tasks that need to be carried out by a management system for managing ad hoc networks.

We propose a solution suite composed of four key mechanisms—*automated service discovery*, *cluster management*, *dynamic service redundancy*

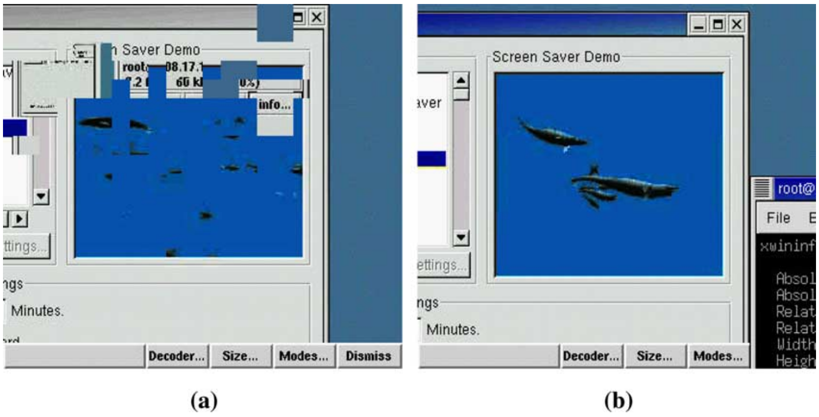


Fig. 11. (a) Degraded video quality without policy negotiation (allocated bandwidth is 12 kb/s); (b) Acceptable video quality (bandwidth in the range 64–128 kb/s allocated after automated policy negotiation).

and policy negotiation—that helps achieve the goal of a self-organizing, efficient, and robust ad hoc network management system.

We focus on prototype implementation and experimental analysis of our proposed schemes. We present our experiences in working in a multi-hop ad hoc network testbed, and demonstrate our PBNM system implementation at work for managing QoS in a multi-domain ad hoc network.

Our ongoing efforts include evaluation of our proposed mechanisms in larger mobile ad hoc networks in presence of random mobility models [14]. In addition, we are currently investigating ways to mathematically model the behavior our PBNM system.

Acknowledgements

This research is supported in part by the Office of Naval Research (ONR) through the "Navy Collaborative Integrated Information Technology Initiative" (NAVCIITI).

References

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

[2] D. Kosiur, Understanding Policy-Based Networking, Wiley, New York, 2001.

[3] D. Verma, M. Beigi, R. Jennings, Policy based SLA management in enterprise networks, Workshop on Policies for Distributed Systems and Networks, January 2001, pp. 137–152.

[4] D. Verma, S. Calo, K. Amiri, Policy based management of content distribution networks, IEEE Network Magazine 16 (2) (2002) 34–39.

[5] G. Ahn, A. Campbell, A. Veres, L. Sun, Supporting service differentiation for real-time and best-effort traffic in stateless wireless ad hoc networks (SWAN), IEEE Transactions on Mobile Computing 1 (3) (2002) 192–207.

[6] S. Lee, G. Ahn, X. Zhang, A. Campbell, INSIGNIA: an IP-based quality of service framework for mobile ad hoc networks, Journal of Parallel and Distributed Computing 60 (4) (2000) 374–406.

[7] M. Mirhakkak, N. Schult, D. Thomson, Dynamic quality of service for mobile ad hoc networks, in: Proceedings of the 1st Annual Workshop on Mobile and Ad Hoc Networking and Computing, August 2000, pp. 137–138.

[8] K. Phanse, L. DaSilva, Addressing the requirements of QoS management in wireless ad hoc networks, Computer Communications 26 (12) (2003) 1263–1273.

[9] D. Kidston, J. Robinson, Distributed network management for coalition deployments, Proceedings of the IEEE Military Communications Conference, vol. 1, 2000, pp. 460–464.

[10] T. Clausen, G. Hansen, L. Christensen, G. Behrmann, The optimized link state routing protocol, evaluation through experiments and simulation, Proceedings of IEEE Symposium on Wireless Personal Mobile Communications, September 2001.

[11] P. Gupta, R. Gray, P. Kumar, An experimental scaling law for ad hoc networks, University of Illinois at Urbana-Champaign, May 2001, Available from <http://decision.csl.uiuc.edu/~prkumar/publications.html>.

[12] W. Chen, N. Jain, S. Singh, ANMP: ad hoc network management protocol, IEEE Journal on Selected Areas in Communications 17 (8) (1999) 1506–1531.

[13] C. Shen, C. Srisathapornphat, C. Jaikaeo, An adaptive management architecture for ad hoc networks, IEEE Communications Magazine 41 (2) (2003) 108–115.

[14] K. Phanse, L. DaSilva, S. Midkiff, A taxonomy and experimental evaluation of policy architectures for bandwidth-constrained networks, Internal Report, Virginia Tech, 2002, Available from <http://www.ee.vt.edu/~kphanse/unpublished.html>.

[15] K. Chan et al., COPS usage for Policy Provisioning (COPS-PR), IETF RFC 3084, March 2001.

[16] D. Durham et al., The COPS (Common Open Policy Service) Protocol, IETF RFC 2748, January 2000.

[17] K. Phanse, L. DaSilva, Protocol support for policy-based management of mobile ad hoc networks, 2004 IEEE/IFIP Network Operations and Management Symposium (NOMS 2004), to appear.

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

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

[20] Intel COPS Client Software Development Kit, Available from <http://www.intel.com/labs/manage/cops/>.

[21] T. Lin, S.F. Midkiff, J.S. Park, A dynamic topology switch for the emulations of wireless mobile ad hoc networks, Proceedings of the IEEE Conference on Local Computer Networks, November 2002, pp. 791–798.

[22] T. Clausen, P. Jacquet, Optimized link state routing protocol, IETF Internet-draft, draft-ietf-manet-olsr-11.txt, work in progress, July 2003.

[23] Optimized Link State Routing (OLSR) Protocol Implementation, Available from <http://menetou.inria.fr/olsr/>.

[24] Differentiated Services on Linux Tool, Available from <http://diffserv.sourceforge.net/>.

[25] A. Leon-Garcia, Probability and Random Processes for Electrical Engineering, second ed., Addison-Wesley, Reading, MA, 1993.

[26] Wireless Tools for Linux, Available from <http://www.hpl.hp.com/personal/Jean_Tourrilhes/Linux/Tools.html>.

[27] E. Liden, 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.

[28] K. Chan, R. Sahita, S. Hahn, K. McCloghrie, Differentiated Services Quality of Service Policy Information Base, IETF RFC 3317, March 2003.

[29] The Ethereal Network Analyzer, Available from <http://www.ethereal.com/>.

[30] Hierarchical Token Bucket (HTB) Packet Scheduler, Available from <http://luxik.cdi.cz/~devik/qos/htb/>.

[31] Video Conferencing Tool, Available from <http://www-nrg.ee.lbl.gov/vic/>.

Kaustubh S. Phanse is a postdoctoral associate in the Bradley Department of Electrical and Computer Engineering at Virginia Tech. He received the B.E. in Electronics and Telecommunications from University of Mumbai, India in 1998, and the M.S. and Ph.D. in Electrical Engineering from Virginia Tech in 2000 and 2003 respectively. His research interests are in wireless mobile networks, quality of service, and policy-based network management. He is a member of the IEEE and ASEE.

Luiz A. DaSilva joined Virginia Tech as an Assistant Professor at the Bradley Department of Electrical and Computer Engineering in 1998, after receiving his Ph.D. in Electrical Engineering at the University of Kansas. He has previously worked for IBM for six years. His research interests focus on performance and resource management in wireless mobile networks and Quality of Service (QoS) issues. Current and recent research sponsors include NSF, the Office for Naval Research, the US Customs Services, Intel, and Microsoft Research, among others. He is a member of the Center for Wireless Communications (CWT), associated faculty at the Mobile and Personal Radio Research Group (MPRG), and a member of the Governing Board of the NSF-funded Integrated Research and Education in Advanced Networking (IREAN) program at Virginia Tech. Dr. DaSilva is a senior member of IEEE and a member of ASEE.

Scott F. Midkiff is a Professor of Electrical and Computer Engineering at Virginia Tech where he teaches courses in networking and telecommunications and conducts research in wireless networks and mobile systems. He received the B.S.E. from Duke University in 1979, the M.S.E.E. from Stanford University in 1980, and the Ph.D. from Duke University in 1985. He is a Senior Member of the IEEE and a Member of the ACM and ASEE.