Authentication, Authorization, Admission and Accounting for QoS applications

Carlos Rabadão[1, 2], Edmundo Monteiro[2]

[1] Superior School of Technology and Management
Polytechnic Institute of Leiria
Morro do Lena – Alto do Vieiro, 2411-901 Leiria, Portugal
crab@estg.ipleiria.pt
[2] Laboratory of Communications and Telematics
CISUC / DEI, University of Coimbra
Polo II, Pinhal de Marrocos, 3030-290 Coimbra Portugal
{crab, edmundo}@dei.uc.pt

Abstract. The main objective of the IETF Differentiated Services (DiffServ) model is to allow the support on the Internet of different levels of service to different sessions and information flows, aggregated in a few number of traffic classes. This differentiated treatment will motivate some users to get better Quality de Service (QoS) for their flows however without assuming the associated costs, leading to the theft of resources that, in extreme situations, will have as consequence the denial of QoS (DQoS) contracted by the users. In the DiffServ model the authentication of flows is carried out on a per packet basis, at the entrance of each domain. The flow classification is supported by some of the IP packet header fields. This approach shows some security limitations that are inherent to the DiffServ model. Being the edge routers (ER) the responsible for the admission and marking of packets, according to the class of service, they are the most vulnerable element to attacks and a security hole in ERs could be propagated to the entire domain. To overcome these limitations, this paper proposes an architecture for Authentication, Authorization, Admission control and Accounting (AAAA) of QoS client applications with dynamic identification of sessions and flows. The proposal functionality is described and analyzed in some detail, focusing the main modules and message exchange among modules. The paper ends with the discussion of the main advantages of the proposal over existing solutions.

1 Introduction

In communication systems, the expression "Quality of Service" (QoS) is used to characterize the capacity of the system to support data flows with service guaranteed parameters (e.g. bandwidth, delay, jitter, losses) in a more or less strict way. The QoS mechanisms impose priorities and restrictions in the access of flows to available communication system resources. In the case of the DiffServ model [1] this traffic prioritization is supported by the identification of Classes of Service (CoS) done according specific fields of the header of IP packets [2]. As discussed in [3, 4] this approach has some security limitations, namely authentication and authorization.

The IETF DiffServ working group has considered some methods to reduce the inherent security limitations of the DiffServ model [4]. These include auditing and IPSec [5, 6]. However the vulnerabilities to security attacks, such as man-in-the-middle and Denial of QoS (DQoS), remain open issues [7].

To overcome the security limitations of Diffserv model, this paper proposes an architecture for Authentication, Authorization, Admission control and Accounting (AAAA) of Quality of Service (QoS) client applications with dynamic identification of sessions and flows. Our proposal address the issues related with the secure negotiation of QoS, in an intra-domain scope, namely admission control at the edge devices of DiffServ domains and the processes of authentication of the customers and authorization of reflows associated with the source reservation procedures. The issues related with the confidentiality and integrity of information flows are relegated to other modules of the communication system.

Besides the present section, the paper has the following structure. Section 2 discusses relevant research work related with the improvement of security and management of QoS networks. Section 3 describes our proposal for dynamic QoS negotiation with authentication, authorization, admission control and accounting of sessions and flows. Section 4 will be dedicated to the evaluation of the proposed architecture. Finally, Section 5 will be devoted to conclusions and directions for future work.

2 Related work

Data communication infrastructures are frequently exposed to attacks to the confidentiality, integrity and availability of the information in transit, and to the authenticity of the origin or the destination of this information.

In communication infrastructures based in the DiffServ IETF model this situation is even more serious because they assure different levels of service to different flows of information, over TCP/IP communication infrastructures, contributing to increase the potential of occurrence of some of these attacks. On DiffServ networks the differentiation of the quality of the service provided to different customers is based on the value of the DSCP (DiffServ Code Point) field [2] included in the ToS (Type of Service) field of the IP header [8]. This approach presents some security limitations [3, 4] that could be explored by less scrupulous users trying to get better quality of service for its flows without however assuming the associated costs, leading to the theft of resources that, in extreme situations, may result on Denial of QoS (DQoS) of the active flows. The IETF DiffServ working group considered some methods to reduce the inherent security limitations of the DiffServ model, such as auditing and the use of IPSec, but vulnerabilities to the attacks such as man-in-the-middle and DQoS remain unsolved [7].

The ARQoS project [9] is one of the initiatives to improve the security in QoS networks, preventing and detecting control and data flow attacks on QoS mechanisms. The work *Preventing Denial of Service Attacks on Quality of Service* [10] addresses the use of the pricing paradigm in the process of resource allocation, that is, the price of the resources increases as the occupation of the resources increases and vice-versa. The process of allocation of network resources is always preceded by authentication and authorization procedures. For such, the proposal uses the *POLICY_DATA* object of RSVP (Resource ReSerVation Protocol) [11] proposed in the RFC2750 – *RSVP Extensions for Policy Control* [12], to allow the use of the PBN (Policy Based Networks) model [13] and RSVP together. The authors also suggest the use of an au-

thorization server, based on the SIP (Session Initiator Protocol) [14], to generate signed policy objects that grant the ability of users to pay a specific price for specific resources. Later, these objects will be carried to the admission control system, through RSVP messages.

A similar approach is proposed by the 3GPP (3rd Generation Partnership Project) to the UMTS *end-to-end* QoS architecture [15], to interoperate with external Int-Serv/RSVP networks. When external networks use RSVP, the signalling messages must contain one authentication token, when available, and the flow identification, issued by the SIP protocol. These elements are carried in the *POLICY_DATA* object.

The RSVP admission control mechanisms are often based on user or application identities [16]. The RFC3520 [17] specifies a new object aimed to add to the RSVP a mechanism for admission control, per session. This new object, called *Session Authorization Policy Element*, will transport the authorization of use of a set of specific resources, for a specific session. This object can include the information of the authorized resources (e.g. QoS parameters), identification of the flow and session, duration of the session and the identification of authorization entity.

Despite the above describe RSVP extensions that provided additional versatility to the protocol, it's origin as a reservation setup protocol designed specifically to support end-to-end QoS signalling in the Internet, originally targeted to multicast receiver oriented applications resulted in a heavy message processing within nodes. Moreover, there are many applications demanding different signalling services not supported by RSVP.

The IETF NSIS workgroup [18] is considering the decomposition of the overall signalling protocol suite into a generic lower layer, with separate upper layers for each specific signalling applications [19]. A proposal for an upper layer protocol was made by NSIS named *NSLP for Quality of Service Signalling* [20]. This protocol is similar to RSVP and uses soft-state peer-to-peer refresh messages as the primary state management mechanism. However, it supports both sender and receiver initiated reservations between arbitrary nodes, e.g. edge-to-edge, end-to-access, etc., and it doesn't support IP multicasting, making it a more flexible and light than RSVP.

A set of other analyzed works [21, 22, 23, 24] addresses the problematic of the security in infrastructures with QoS support, mainly regarding issues related to admission control, considering the adoption of the PBN architectures, and security aspects of inter-domain signalling between Bandwidth Brokers (BB). However, issues related with the security in the peripheral networks, such as user authentication and dynamic authorization of resources, remain open.

3 AAAA architecture for QoS applications

In the DiffServ model flow authentication is carried out on a per packet basis, at the entrance of each domain. Flow classification is supported by some of the IP packet header fields. As said before this approach has some security limitations that are inherent to the DiffServ model.

Being the edge routers (ER) the responsible for the admission and packet marking according to flow's quality of service, they are the most vulnerable element to attacks

and security holes.

To overcome these limitations, this paper proposes an architecture for QoS negotiation with authentication, authorization, admission control and accounting of client applications in a dynamic way, at the entrance DiffServ domains. The proposed architecture will basically focus in questions of secure negotiation of QoS, in an intra-domain scope, addressing the questions related to admission control at the edge devices of DiffServ domains and with the procedures for customer authentication and resources reservation authorization.

3.1 Architecture overview

The proposed architecture, shown in Figure 1, has seven main modules: Policy Repository, QoS Client, Authentication, Authorization, Admission Control, Accounting and Router PEP (Policy Enforcement Point). The AAAA modules interact with the Policy Repository module of the domain and with the QoS Client component located at each client with QoS capabilities. The Router PEP and the DiffServ Edge Router (of the network provider) are the responsible for the enforcement of the QoS definitions in the border between the user domain and the network provider.

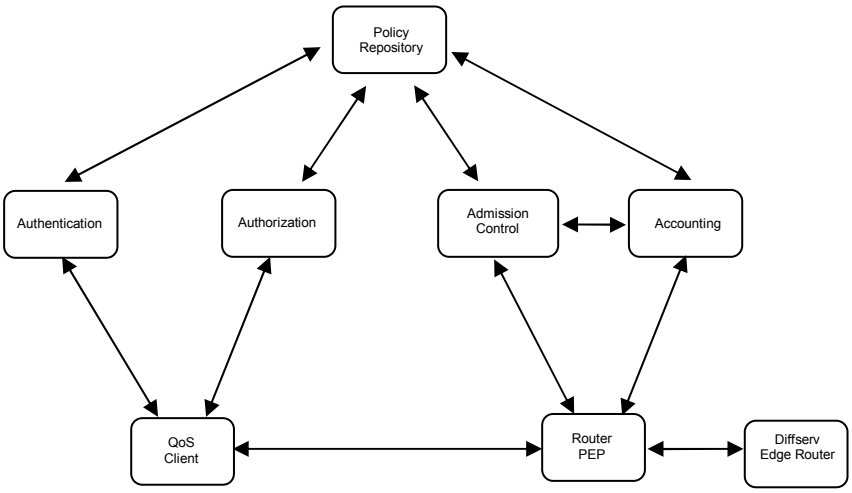

Figure 1. AAAA Architecture for QoS applications

The QoS client will be initially authenticated in an Authentication module. This module will issue credentials to enable the client to contact the Authorization module. Subsequently, the client will request to the Authorization module the allocation of network resources with the characteristics needed for its new session. After that and according to the domain QoS policies the Authentication module will decide to authorize (or not) the new session, sending a *ticket* that contains, among other data, the resources approved for the new session and the identification of the session flows.

After the above procedures, the client has an authorization to use network resources, emitted by the Authorization module, based in QoS policies, but the needed resources are not yet reserved. To make this reservation, the client needs to issue a request to the Admission Control module, the entity that controls the network resources of the domain. This is achieved sending the *ticket* to the Admission Control module with the authorized resources and protected against authenticity treads. This module, based in the domain resources availability and in the *ticket* information, will reserve the required resources, configuring the Edge Router and sending the DSCP to be used by the client to update the session description. After this, the client will mark packets and sends it to the network media. The authorization (*ticket*) has limited time so, before time-out, the QoS Client must refresh the ticket and send it to the Admission Control module.

Figure 2 shows a simplified QoS Client flow diagram for the session (or flow) set-up and teardown procedures. In both cases the QoS client sends a request for a ticket. In the case of a session set-up, the Authorization module issues a *ticket* with information of the session identification, the associated flows and the authorized resources for each one. If the QoS Client requests to add a new flow to the session, the Authorization module will issue a new token with the same session identification and with information about all the flows belonging to this session (already authorized and the new ones). In the case of a flow teardown, the issued *ticket* will include the session identifier, and all the flows to be supported. Finally, in the case of a session teardown, the issued *ticket* will include only the session identifier, being removed all the flows information.

3.2 Architecture elements

As said before, there are seven fundamental entities in this architecture: Policy Repository, QoS Client, Authentication, Authorization, Admission control, Accounting and Router PEP. The functionalities of each of the modules are described bellow.

QoS Client - This module intercepts all the session/flow set-up and teardown requests, implicitly made by the application or on the behalf of an external protocol (like SIP signalling, for instance), and issues a resource request to the Authentication module indicating, among other information, the identification of the user, sessions and flows. The client must be authenticated before the request can be issued. If the resource request is authorized by the Authentication module, a *ticket* will be received, with similar capabilities of the gate [25], ticket [26] and token [27] proposals. With this *ticket* the QoS Client will request resources reservation to the Admission Control, sending the token inside a *POLICY_DATA* object [11, 12]. This object will be carried by a signalling protocol like RSVP or NSLP-QoS. This module is also responsible by the DSCP packet marking.

Authentication - This module is the responsible for the authentication of the QoS Client, enabling I the access to the Authorization entity. The module will consult the Policy Repository in order to determine if user has administrative permission to access to the system.

Authorization - This module decides whether or not the client has administrative permissions to make an issued reservation, and to generate the associated *ticket*, based on the resource request, issued by the QoS Client, and on the domain QoS policies, defined by the network administrator and stored in the Policy Repository.

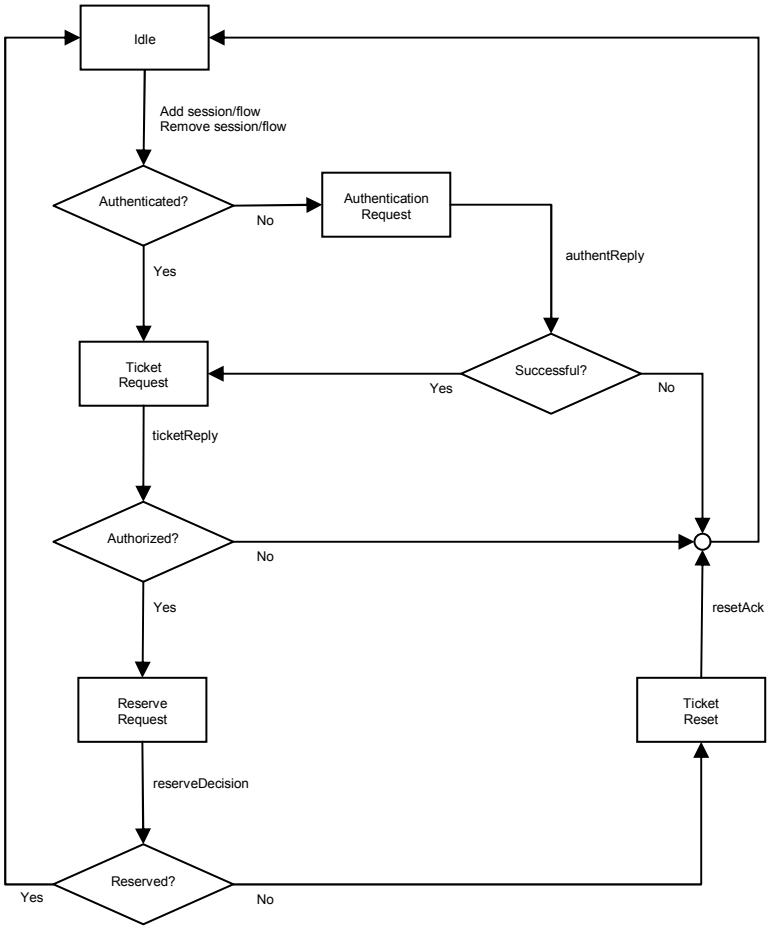

Figure 2. QoS Client flow diagram for a session or flow set-up/teardown

Router PEP - This module is the responsible for the analysis of the resource request messages issued by the QoS Client. It will extract the relevant information from the received messages and forward this information to the Admission Control entity (e.g. COPS REQ message) [28, 29]. It should also configure the router interfaces according to the configurations received from the Admission Control entity, enabling the flows belonging to authorized sessions to be routed to its destinations, outside the local domain.

Admission Control - This module has as principal task the reception and analyse of the resources request issued by the QoS clients. After consulting the domain resources availability, and according to the resources requested by the QoS client, via *ticket*, this module will issue the decision to the client (for instance with a COPS DEC message [28, 29]) and will ask the PEP Router to configure the router interfaces. The module could also send *tickets* to the Accounting entity for accounting purposes.

Accounting - This module is the responsible to collect information concerning the characteristics and duration of the QoS resources used by the clients, using *tickets* received from the Admission Control entity and/or information received from the PEP Router.

Policy Repository - This module is the responsible to store all the domain policies, including security policies, QoS policies and accounting policies. These policies are defined by the network administrator and could be accessed by the Authentication and Authorization module, in a per session/flow set-up/teardown basis, and sporadically by the Accounting and Admission Control. The policies will reflect the treatment to give to each flow, according to its owner, user group it belongs and privileges, and according to the type of application/flow and its QoS requirements.

3.3 Example of a session QoS set-up

In this section we will present and discuss an example of the use of the proposed architecture. The example will show the message exchange between the several networks module involved at the set-up of a session QoS reservation. Only the originating side flows are described for simplicity. The same concepts apply to the terminating side. The example is illustrated in Figure 3.

The QoS Client intercepts all the session/flow set-up requests, implicitly made by the application or on the behalf of an external protocol like SIP. After this, the module issues a resource request (`ticketRequest`) to the Authorization module, indicating, among other information, the identification of the user and flows to be authorized. As part of this step, the QoS Client must get credentials (`authenRequest`), from the Authentication module, to authenticate itself to the Authorization Server.

The Authorization Server queries the Policy Repository (`policyRequest`) in order to know if user has administrative permissions to make the reservation and to decide which the resources should be authorized for each flow, according to the application and user privileges. After receiving and analysing the list of Policies (`policyList`), the Authorization entity sends a response to the QoS Client (`ticketReply`), possibly after modifying the parameters of the resources to be used. This response is personated by a *ticket* including, among other elements, the session identifier, the authorized flows and the approved resources.

After receiving the ticket, the QoS Client issues a request for reservation of the resources authorized for the new session (`resourceRequest`). the ticket provided by the Authorization module is included in this request.

The Router PEP intercepts the reservation request message and sends a policy decision request (e.g., COPS REQ message [12, 28, 29]) to the Admission Control en-

tity (`provisionRequest`) in order to know if the resource reservation request should be allowed to proceed. The *ticket* is included in this above described request.

The Admission Control module uses the *ticket* to extract information about the user and about the services authorized by the Authorization module. The Admission Control, after examining the information, checks the availability of resources to supply the requested QoS and allows or rejects the reservation request. After this, it sends this decision (for instance with COPS DEC message [28, 29, 12]) to the Router PEP (`provisionDecision`). The parameters to configure the edge router interfaces and the DSCP values to apply to the flows of the new session are included in this message. The Admission Control sends the ticket (*accountInforme*) to the Accounting entity, for accounting of the network resources used by this session.

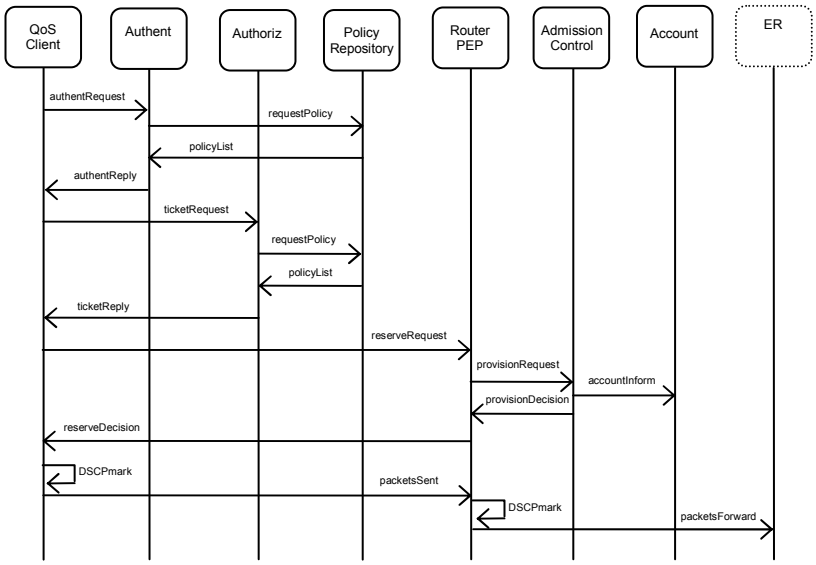

Figure 3. Sequence diagram for the session QoS set-up

The Router PEP, after receiving the message, configures the router interfaces to allow the flows of the new session and sends a response (`reserveDecision`) to the QoS Client indicating that resource reservation is complete and the DSCP values to mark the packets of flows belonging to the authorized session.

After receiving the response, the QoS Client, allows the flow of the new session to proceed. All the packets of authorized sessions will be marked.

4 Evaluation

In the architecture proposed in this paper for QoS negotiation with AAAA, instead of resource allocation based on static rules, the resources are allocated based on the client and session identification, as well as in accordance with the defined policies, in

a dynamic way, e.g. when new flows or sessions are requested. The requests are authorized by an authorization module issuing a *ticket* according to the security and QoS policies of the local domain. After the authentication, the admission control entity will analyse the *ticket* and, if resources are available, the edge router will be configured to enable the flow or the set of flows belonging to a session. With this procedure the contracts established between the users and network providers will not be violated and the resource allocation will be done in accordance with the domain policies.

Besides the above advantages, when the customer gets a *ticket* authorizing him to use a set of resources with QoS and the admission control module denies this reservation, an auditing process could be triggered to identify the causes of the fault (e.g. bad use of QoS policies, insufficient resources) enabling the system to take decisions to solve the malfunctioning of the system (e.g. to eliminate the cause of bad application of policies, to renegotiates the resources near the network provider).

The use of tickets enable the system to add and to remove new flows in an expedite way and to reduce the vulnerabilities to the attacks of theft of resources and consequent denial of QoS, because only clients who acquire one ticket could request resources reservation and only authenticated users with expressed permission will be able to acquire its. To reduce the vulnerabilities to theft or corruption of tickets, all the issued tickets will have a limited time of life and will be authenticated by the authorization entity. These mechanisms reduce the vulnerabilities to man-in-the-middle attacks, aimed to adulterate ticket and consequently deny the QoS of the flows.

5 Conclusions and Future Work

In this paper, he proposed an architecture for QoS negotiation with Authentication, Authorization, Admission control and Accounting (AAAA), for client application session with QoS needs. The proposal seeks to overcome the security limitations of current DiffServ model and to manage the admission control of sessions in a dynamic way, according to the client profile and to the available resources.

The proposal addresses the issues of secure negotiation of QoS in an intra-domain scope, the issues related to admission control in the edge devices of DiffServ domains and the procedures of authentication of the clients and authorization of resources reservation for sessions establishment.

Future work (already ongoing) will address experimentation and evaluation of scalability and performance behaviour of the proposed architecture.

Acknowledgements

This work was partially financed by the PRODEP program supported by the Portuguese Government and the European Union FSE Programme.

References

[1] S. Blake *et al*, *An Architecture for Differentiated Services*, RFC 2475, IETF, Dec. 1998.

[2] K. Nichols *et al*, *Definition of the Differentiated Services Fields (DS Fields) in the IPv4 and IPv6 Headers*, RFC 2474, IETF, Dec. 1998.
[3] C. Rabadão, E. Monteiro, *Segurança e QoS no Modelo DiffServ (Security and QoS in the DiffServ Model)*, 5th Conference on Computer Networks (CRC2002), Faro, Portugal, University of Algarve, 26-27 Sep. 2002.
[4] Zhi Fu *et al*, *Security Issues for Differentiated Service Framework*, Internet Draft (expired), Oct. 1999.
[5] S. Kent, R. Atkinson, *IP Encapsulating Security Payload (ESP)*, RFC 2406, Nov. 1998.
[6] R. Atkinson, *IP Authentication Header*, RFC 1826, IETF, Aug. 1995.
[7] A. Striegel, *Security Issues in a Differentiated Services Internet*, Proc. of Trusted Internet Workshop - HiPC, Bangalore, India, Dec. 2002.
[8] J. Postel, Editor, *Internet Protocol*, RFC 791, IETF, Sep. 1981.
[9] D. Maughan *et al*, *The ARQoS Project: Protection of Network Quality of Service Against Denial of Service Attacks*, http://arqos.csc.ncsu.edu/, State University of North Carolina, University of California and MCNC.
[10] E. Fulp *et al*, *Preventing Denial of Service Attacks on Quality of Service*, Proc. of DARPA Information Survivability Conference and Exposition (DISCEXII'01), IEEE Computer Society, 2001.
[11] R. Braden *et al*, *Resource ReSerVation Protocol (RSVP) - Version 1 Functional Specification*, RFC2205, IETF, Sep. 1997.
[12] S. Herzog, *RSVP extensions for policy control*, RFC2750, IETF, Jan. 2000.
[13] S. Hahn *et al*, *Resource Allocation Protocol*, IETF, http://www.ietf.org/html.charters/rap-charter.html.
[14] J. Rosenberg *et al*, *SIP: Session Initiation Protocol*, RFC 3261, IETF, Jun. 2002.
[15] *Access Security for IP-based Services*, Technical Specification 3GPP TS 33.203, Version 6.1.0, 3rd Generation Partnership Project, Dec. 2003.
[16] S. Yadav *et al*, *Identity Representation for RSVP*, RFC 3182, IETF, Oct. 2001.
[17] L-N. Hamer *et al*, *Session Authorization Policy Element*, RFC3520, IETF, Apr. 2003.
[18] J. Loughney *et al*, *Next Steps in Signaling (NSIS)*, IETF, http://www.ietf.org/html.charters /nsis-charter.html.
[19] R. Hancock *et al*, *Next Steps in Signaling: Framework*, Internet Draft (work in progress), IETF, Oct. 2003.
[20] S. Van den Bosch, G. Karagiannis and A. McDonald, *NSLP for Quality-of-Service Signaling*, Internet Draft (work in progress), IETF, Feb. 2004.
[21] G. Pujolle, H. Chaouchi, *QoS, Security, and Mobility Management for Fixed and Wireless Networks under Policy-based Techniques*, IFIP World Computer Congress 2002.
[22] E. Mykoniati *et al*, *Admission Control for Providing QoS in DiffServ IP Networks: The TEQUILA Approach*, IEEE Communications Magazine, Jan. 2003, pp. 38-44.
[23] A. Ponnappan *et al*, *A Policy Based QoS Management System for the IntServ/DiffServ Based Internet* , Proc. of 3rd International Workshop on Policies for Distributed Systems and Networks, POLICY'02, Monterey-California, 5-7 Jun. 2002.
[24] V. Sander *et al*, *End-to-End Provision of Policy Information for Networks QoS*, Proc. of 10th IEEE International Symposium of High Performance Distributed Computing, São Francisco-Califórnia, 07-09 Aug. 2001.
[25] *PacketCable Dynamic Quality of Service Specification*, CableLabs, Dec. 1999.
[26] J. Vollbrecht *et al*, *AAA Authorization Framework*, RFC 2904, IETF, August 2000.
[27] L-N. Hamer, B. Gage and H. Shieh, *Session Authorization Policy Element*, RFC3521, IETF, Apr. 2003.
[28] D. Durham, *The COPS (Common Open Policy Service) Protocol*, RFC2748, IETF, Jan. 2000.
[29] J. Boyle *et al*, *COPS usage for RSVP*, RFC2749, IETF, January 2000.