Mechanisms for AAA and QoS Interaction

Antonio Cuevas [1], José Ignacio Moreno [1], Rui Aguiar [2], Victor Marques [3], Carlos García [1], Ignacio Soto [1]

1. Ing. Telemática, Universidad Carlos III de Madrid Madrid, Spain
{acuevas;jmoreno;cgarcia;isoto}@it.uc3m.es
2. Dpt. Electronica e Telecomunicações, Universidade de Aveiro, Aveiro, Portugal
ruilaa@det.ua.pt
3. Portugal Telecom Inovação, Aveiro, Portugal
victor-m-marques@ptinovacao.pt

Abstract

The interaction between Authentication, Authorization and Accounting (AAA) systems and the Quality of Service (QoS) infrastructure is to become a must in the near future. This interaction will allow rich control and management of both users and networks. DIAMETER and DiffServ are likely to turn into the future standards in AAA and QoS systems, but they are not designed to interact with each other. To face this, we propose a new Diameter-Diffserv interaction model and describe the Application Specific Module (ASM) implemented to allow this interaction. The ASM has been implemented and tested in a complete AAA-QoS IPv6 scenario.

1 INTRODUCTION

Nowadays, most services offered by IP networks are considered as having low added value. For the end user, internet access is free, he is just aware of paying the cost of a, generally local, call and all users receive the same (best-effort) service. But, as offered services increase their variety, the need for more advanced user control and differentiation will rise in order to offer users tailored services according to the price paid. At present, this issue is tackled thanks to RADIUS, an AAA (Authentication Authorization Accounting) system widely adopted. Radius does have well-known shortcomings, such as scalability and no roaming capabilities, and DIAMETER ([2]) is foreseen to replace it. DIAMETER defines several entities (AAA clients, AAA servers, etc.), their interactions and the communication protocol amongst them.

On the other hand, something very important when setting differences and tariffs for services is the so called QoS (Quality of Service) existing on the network. Quality of service is not inherent to IP networks, so novel concepts had to be added to them. There is no widely implemented IP-based QoS system at the moment. Nevertheless, differentiated services (DiffServ) [1] approaches are very likely to be globally adopted, as they are being tested and seem able to solve the drawbacks of alternate solutions. DiffServ does not provide QoS per flow (as older QoS models like IntServ did) but aggregates similar flows using a DSCP field code in the packets. The Diffserv architecture [1] defines two kinds of entities: edge (ingress and egress) routers and core routers. All traffic entering or leaving a DiffServ domain must go through an edge router. A Bandwidth Broker will control these edge routers.

The Diffserv framework is particularly suitable to Policy Based Management [7] strategies. A control entity (PDP, Policy Decision Point) can be used to control the routers -mainly edge routers- (PEPs, Policy Enforcement Points). PEPs and PDPs can then communicate using COPS protocol [4]. In this paper, we will consider that Bandwidth Brokers (BB) act as PDPs in a Policy Based Management structure. Thus PDPs manage the network resources. In addition they decide which packets are prioritary, and thus perform user control, seeing which priority is to be assigned to a given user's packet. But identifying users is a responsibility of the AAA system. We can see the need for an interface between AAA (DIAMETER) and DiffServ. DIAMETER will focus on users control and DiffServ (PDPs) in network resource management, splitting and thus solving a very big and crucial problem, the capability of controlling users, taking differentiated QoS aspects into account.

Although some proposals for DIAMETER-based QoS control have been proposed (see [3]) they presented implementation problems. As a consequence, AAA and QoS do not easily interact nowadays. That is why PDP themselves perform user control. But this user control can only be very rudimentary: it is not their job and neither are they prepared for it. In the future, when QoS usage is widespread, user Data Bases (DB) will be very large and will only be housed by AAA systems. Moreover, user control policies will be complex and only AAA will be able to centralize them and give them coherence. PDPs will not be able to assume these functions since they must control a network that is to become very complex. It is obvious that existing solutions will not work in future.

What we are proposing to face this problem, is defining adequate interactions between AAA systems and DiffServ-based networks, supported by interconnecting AAA servers and BBs through an ASM. This ASM will act as an API to the AAA server and a COPS client to the BB. This paper describes the AAA - DiffServ interaction model and the design, implementation and testing of a prototype ASM.

The paper describes first the application scenario (or framework) of this work in section II, and later the ASM design. In the fourth section we detail the tests performed, and finally we will gather the main conclusions.

2 SCENARIO

2.1 Overall scenario

In the proposed scenario we consider that the interaction between AAA system and QoS will be carried out exclusively by a AAA DIAMETER server and by a DiffServ Bandwidth Broker. Our scenario is then composed of interconnected domains. Within each domain:

1. The AAA system is implemented by a DIAMETER-aware system.
2. The QoS system is a DiffServ system with PEP's (ingress, egress or core routers) and a PDP, the Bandwidth Broker. The PDP is responsible for network resource management within its domain.

Naturally the network also has access routers (ingress routers to the domain). The access router (an ingress router) accomplishes functions for both AAA and QoS but no interaction between AAA and QoS takes place at it. The router is both an AAA Client (DIAMETER Client) and a PEP (COPS client).

The routers also act like an AAA Client (proxy) translating URP (a specific User Registration Protocol) messages to DIAMETER and sending these to the domain's AAA server (AAA.l). Then this AAA.l just forwards messages from/to the user's home AAA server (AAA.h), if required.

In terms of QoS, this could be provided in two different ways:

1. deterministic QoS, with end to end (e2e) resource reservation
2. probabilistic QoS, without strict warranties

And we have two possible AAA-QoS interaction scenarios, but we note that they do not directly map to the two approaches above.

1. AAA and QoS interaction in all the domains along the e2e path.
2. AAA and QoS interaction solely in the local domain. In the rest of the path, if needed, only BB's interact.

We have considered the second case in our scenario. This interaction scenario may apply in both deterministic and probabilistic QoS provisioning modes.

In our scenario the user (M) interacts with the AAA system (A) and this with the QoS system (Q) (see Figure 1). We call this interaction mode MAQ. The local AAA.l server informs the local BB of the presence of a new registered user in the domain and of his QoS profile. The AAA server acts like a client of the BB. An alternative approach (nicknamed MQA) changes client-server roles of the BB and the AAA server. There is no final consensus on which option (MAQ or MQA) is better (for a detailed discussion see [8]).

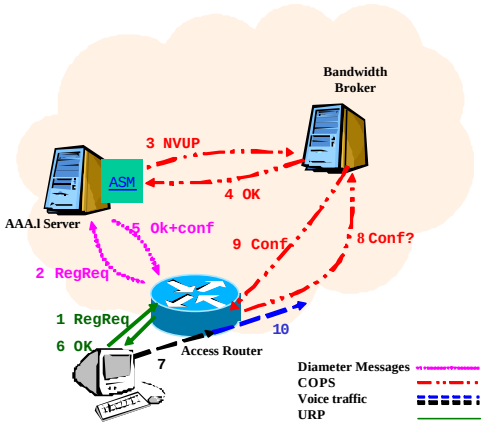

Figure 1. Overall system interaction

2.2 Overall interaction

We shall see now how all the above mentioned components interact at flow authorization during a "session setup", following the messages in Figure 1. We must note that, for simplicity, the scenario depicted is a non roaming scenario. In such a scenario the local AAA.l server is also the user's AAA.h home server.

1. User registers using URP. The AAA client (residing in the AR) transforms this request
2. to an appropriate DIAMETER message and sends it to the local AAA.l server.
3. The AAA.l server (that, in this scenario, is also the AAA.h server) authorises the user and then, using the ASM, dumps the user's QoS profile into the BB
4. BB acknowledges the AAA.l server.
5. AAA.l sends a positive authorisation message to the AR along with the needed configuration parameters (such has keys)
6. AR communicates the positive decision to the user.
7. The user sends the packet
8. the AR detects a new IPv6 Source Address, and DSCP combination and requests the BB the rules to apply to these packets.
9. as the QoS profile was previously dumped by the AAA.l to the BB, the B.B. sends the needed configuration parameters to the AR.
10. AR applies this configuration to all the packets with the same IPv6 Source Addrress, and DSCP combination.

It can be seen that the proposed interaction allows splitting the "session setup" in two parts: i) first registering the user in the AAA system and the AAA server sending QoS permissions to the BB (points 1 to 6); ii) and second using the resources (sending the flow) (points 7 to 10). AAA QoS interaction is only required in the first part.

To implement the above scenario with Mobile IP scenario, it suffices to change the basic DIAMETER system into a DIAMETER mobile-aware IPv6 system, as described in [6]. In this case, the IPv6 Source Address would be the Care of Address (CoA) of the Mobile Node. In a mobile IPv6 system the CoA changes when the Mobile Node performs handovers. The BB must be informed of those changes so that it can move the resources assigned to the old CoA to the new CoA. Further discussion can be found in [9].

2.3 ASM interaction

We proposed that AAA-QoS interaction takes place at "session setup". No interaction is performed at "session end", as detailed later. We must note that the term "session" was employed here only for easy understanding, and will be formalized later.

As it was stated in the previous section, when the user registers, the AAA.l provides the BB with the "QoS profile" of the user registering. The QoS system identifies the user by the IPv6 address of the terminal he is employing and this address is sent, along with the "QoS profile", by the AAA.l to the BB. In a Mobile IPv6 scenario this address would be the CoA. Latter on, when the user's starts to send packets to the core network, the access router will gather the IPv6 source address of these packets and will query the BB if packets with this source address have the "QoS Permissions" to reach the core network. This QoS process is called policing and it's performed following the COPS outsourcing model: access router out sources the policing decision to the BB.

The "QoS Permissions" are simply a list of DSCPs that correspond to priority and bandwidth allocation. Actually the access router gathers the IPv6 Source and the DSCP code of the packets and then queries the BB. If the answer is positive, the packet is given the appropriate QoS treatment (identified by its DSCP) if it is not positive, all the packets with the same profile are blocked.

This list of DSCPs has a time to live. This time to live has the same value as the time to live of the "diameter session", called the Authorization Life Time. After this time to live expires, according to DIAMETER, the user has to register again and the "QoS profile" is sent again to the BB.

The IPv6 address of the terminal the user is employing (or the CoA), the list of DSCPs and the time to live of this list is what we call the Network View of the User Profile (NVUP). The list of DSCPs and the time to live are stored in a Data Base inside the AAA.h. The CoA is sent to the AAA.h when the user registers. Note that the purpose of sending the CoA is that, as defined in DIAMETER mobile ipv6, the AAA.h server can do the binding update in the user's Home Agent. The NVUP and other parts of the user profile are sent by the AAA.h to the AAA.l when the user registers, as shown in ARA message (AA Registration Answer) in Figure 2. The NVUP is the only information exchanged between the AAA system and the QoS system. In our AAA-QoS scenario it is exchanged between the AAA.l server and the BB.

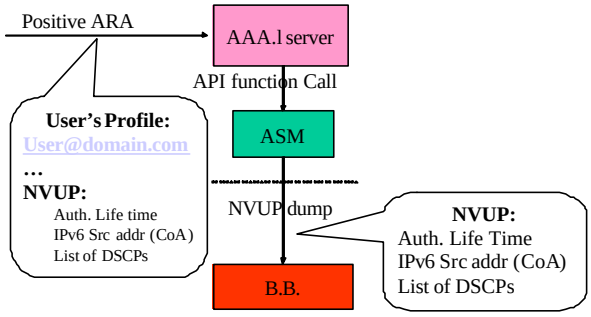

Figure 2. Information interchanged between AAA and QoS

The AAA.l server does not maintain the state of roaming users, as defined by DIAMETER. But the BB is state full, as required by DiffServ and COPS. This

does not pose a problem, since the AAA.l is the one who performs queries to the BB (it can be seen as the AAA.l being a client of the BB). Moreover, as the AAA.l does not maintain roaming users' state it cannot make changes to the NVUP housed in the BB, and in particular it cannot delete it. For this reason, the NVUP has a time to live.

This proposed AAA QoS interaction seems good enough to fulfil the requirements outlined in section 1. We must say that the whole AAA and QoS scenario has been successfully tested in a MIPv6 environment, as described in section V. Our scenario specialises the AAA system on user control and the DiffServ System on network control and defines the needed interaction between AAA and DiffServ. An important aspect of the proposed scenario is that it splits the user presence in the network from the consumption of network resources, just like things actually happen.

Note: when the AAA.l server powers up it will send the BB a "file" indicating which QoS parameters (B.W., priority and max. burst size) each DSCP represents. This has also been implemented and tested, but not in a complete scenario. This "file" would be part of the Service Level Agreements (SLAs) signed between different domains. We foresee that these SLAs will be handled by AAA entities, and thus was located in the AAA.l server.

3 ASM DESIGN

3.1 ASM's Architecture

Profiting the experience from previous projects (see [5]), we developed a simple and easy to use ASM. As depicted in Figure 3. , the proposed ASM has two interfaces:

1. For the AAA.l it's an API with 5 C-style functions that can be called within the AAA server source code.
2. For the BB it is a COPS client using IPv6.

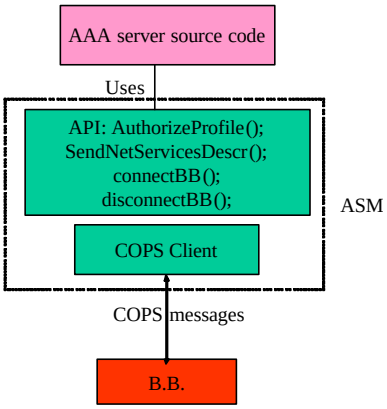

Figure 3. ASM architecture

A similar architecture (but for other scenarios) is proposed in [11] and [5]. From the API point of view, the COPS Client behaves as a library (.lib file) implementing the API (.h file). In our case, this module has been specifically adapted to the AAA system, and has even to perform internal protocol translation.

COPS was designed to connect PDPs (a Bandwith Broker in our scenario) with PEPs (access routers in our scenario). Several authors propose to use COPS to communicate "agents" with PDPs with agents as H323 gatekeepers [5] [11]. We used COPS to communicate the AAA.l with the BB. Since the BB already uses COPS to interact with the Access Routers, choosing COPS to communicate with the COPS-client part of our ASM eases the implementation of the BB.

Essentially the ASM makes the AAA server look like a COPS Client. Any API function called will issue a REQuest message to the Bandwidth Broker, by the COPS Client part of the ASM. When the DECision message arrives, the function returns. This model uses the MAQ model where the AAA.l makes requests (sends the NVUP) to the Bandwidth Broker. This scenario is a client-server model and our ASM is well suited for this case. In case the Bandwidth Broker needed to sent unsolicited messages to the AAA server (and not just answers (DECision) to previous queries (REQquest)) the COPS Client ASM should be able to receive these unsolicited messages. This could be implemented in the ASM API using call back functions.

3.2 API overview

We defined two structures and four functions.

The first structure is called NVUP, Network View of User Profle. It contains the user's CoA, the Time To Live of the NVUP and the list of allowed DSCPs. The second one is termed NetService, it contains the DSCPs and the bandwidth, burst size and priority they stand for.

AuthorizeProfile(), is a function used upon a positive authorization from the AAA.h to dump the NVUP (part of the user profile) to the Bandwidth Broker. One input parameter of this function is the NVUP structure.

SendNetServicesDesc() is a function that must be called once before any operation. It sends the Bandwidth Broker the meaning (in terms of bandwidth, burst size and priorty) of the DSCPs employed within the domain. Reads this data from a given file and uses the NetService structure to handle it.

ConnectBB(). This function must be called once and before any call to any other function.

DisconnectBB(). This function ends the COPS connection to the bandwidth Broker. Called at the end of the program in the AAA.l.

4 TESTS

Our ASM has been implemented and successfully tested in a trial network, an IPv6 test bed. Because of the lack of DIAMETER IPv6 products, our test scenario had to employ RADIUS as AAA system. But as we will briefly explain in the end

of this section, our ASM has also been successfully integrated in an IPv4 DIAMETER AAA.l server.

For our test purposes RADIUS is similar to DIAMETER – with the restriction of the AAA.l and AAA.h having to be the same, and therefore the user cannot move to foreign domains, i.e. he is not allowed to roam. This is precisely the scenario depicted in Figure 1. , a non roaming scenario.

We have added the API functions of our ASM to the Radius server source code. In a similar way, they could be added to the source code of a DIAMETER server, and in fact this flexibility is one of its advantages. At the time the tests were made, BB's functionality was minimal.

We depict here the test bed. We had two private IPv6 subnetworks; the first (down in the figure) represents the core network of the local domain, the second is an access network used by the clients willing to access the domain.

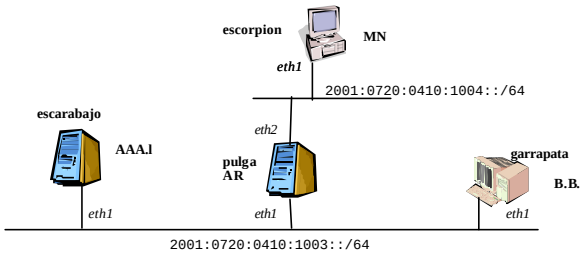

Figure 4. Test Bed

All machines run Linux 2.4.16. The domain ipv6.it.uc3m.es must be added to the machines to obtain their fully qualified domain name.

Pulga is the access router to the sub network standing for the local domain. It has an AAA Client translating URP messages to Radius messages. It is also a PEP querying (via COPS) the BB what to do with the packets willing to access the core network.

Escorpion has installed a registration module capable to register the user via URP sending the user's identity, user's password and the Ipv6 source address of the machine employed by the user.

Garrapata is the BB.

Escarabajo is the AAA.l server. It runs a radius server whose source code was modified adding the API functions of our ASM so that it could interact with the BB. This AAA.l server plays also the role of the AAA.h.

Tests were very simple and divided in 3 main parts:

First we tested the ASM writing a simple program calling all the API functions. We used *ethereal* to capture the COPS packets sent to the BB. In a first stage the only task of the COPS server in the BB was just answer these messages.

These tests allowed us to verify the correct working of the ASM: correct COPS message format and correct API behavior.

Then we modified the radius server source code adding it the API functions. We developed a basic BB and we integrated on it the above mentioned COPS server. The Radius server was tested, and so was the BB. Then the ASM, this time working between a AAA.l and a BB, was again tested and we reached the same results than in the above tests.

Finally, we implemented the PEP in the access router, its QoS functionality and its COPS client functionality. We enhanced the COPS server in the BB so that it could answer the PEP requesting the configuration parameters (just to let the flow go through or not). At this stage we could test the functionality of the overall system. We again used ethereal capturing packets in *pulga's* eth1 interface. We present below an illustration of the results:

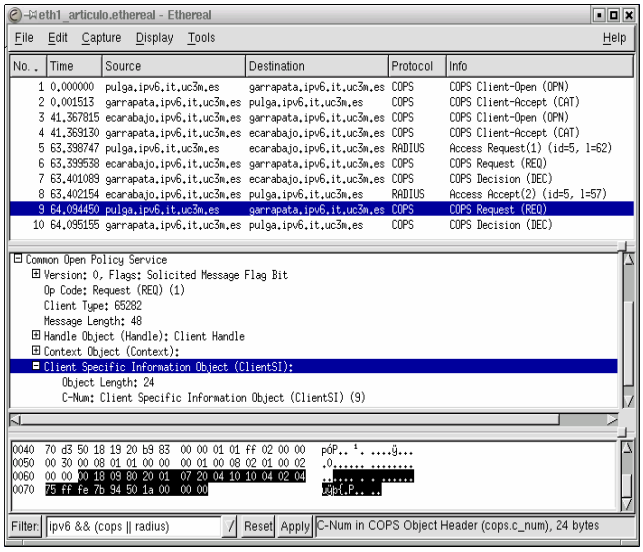

Figure 5. Overall scenario test results

Messages #5, 6, 7 and 8 correspond to messages 2, 3, 4 and 5 in Figure 1. Messages #9 and 10 correspond to messages 8 and 9 in the same figure. We can see in message #9 how the AR gathered the IP Src. Addr. (2001:0720…9450) and the DSCP code (1a) of a packet sent by *escorpion*.

As we stated in the beginning of the section, we also integrated our ASM in a DIAMETER AAA.l server. We built a roaming scenario with the AAA.h located in Stuttgart's Moby Dick Test Bed. Since the DIAMETER system used IPv4 pulga, the AR housing the DIAMETER Client, had to be a dual stack machine: it used IPv4 to communicate with the AAA.l and IPv6 to communicate with the BB. *escarabajo*, the AAA.l, had also to be dual stack because the ASM employs IPv6 to communicate with the BB.

A Stuttgart-based roaming user registered in Madrid's domain. The associated AAA Register Request was forwarded by Madrid's AAA.l Server to Stuttgart's

AAA.h. When the AAA.l received the positive AAA Register Answer (see Figure 2.) it dumped the NVUP to the Bandwidth Broker using our ASM and forwarded the AAA answer to *pulga*, the DIAMETER Client.

5 CONCLUSIONS

In this paper we have proposed an interaction between a DIAMETER system and a DiffServ network in an IPv6 scenario, and presented the ASM that makes this interaction possible. Our scenario specialises the AAA system in user control and the QoS DiffServ System on network control, and defines the adequate interaction between AAA and DiffServ systems. The interaction between AAA and QoS infrastructures is necessary to link the two levels of authentication and authorization, i.e., on user level and network level. This same type of interaction can also be applied to RADIUS systems. The whole scenario and the ASM was implemented and tested with RADIUS in an IPv6 environment, and with DIAMETER in a dual IPv6-IPv4 scenario.

The ASM is easy to use and to integrate in the AAA.l code, and showed to provide high performance. Its functionality is basic, but it could be enriched easily since our ASM model is quite adequate to the specified MAQ-model approach.

6 ACKNOWLEDGMENT

The authors would like to thank the European Commission for partially founding this work through the Moby Dick Project (IST-2000-25394). The code employed for the tests is part of this project and the ASM developed will be employed in Moby Dick.

REFERENCES

[1] S. Blake, "An Architecture for Differentiated Services", RFC 2475, December 1998

[2] Pat R. Calhoun, "Diameter Base Protocol", <draft-ietf-aaa-diameter-10.txt>, April 2002

[3] Pat R. Calhoun, "DIAMETER QOS Extension", <draft-calhoun-diameter-qos-00.txt>, May 1998

[4] D. Durham, Ed., "The COPS (Common Open Policy Service) Protocol" RFC 2748, January 2000

[5] J. Escribano Salazar, "Diseño e Implementación de un entorno DiffServ", Proyecto de Fin de Carrera. December 2001. In Spanish.

[6] Stefano M. Faccin, "Diameter Mobile IPv6 Application", <draft-le-aaa-diameter-mobileipv6-00.txt>, 2001

[7] INTAP, "Survey on Policy-Based Networking", 2001.

[8] I.S.T. Moby Dick Project, WP1, "Moby Dick Framework Specification (D0101)" December 2001.

[9] Victor Marques et al., AN IP-BASED QOS ARCHITECTURE FOR 4G OPERATOR SCENARIOS. IEEE Wireless Communications. June 2003

[10] Stefano Salsano, "COPS Usage for Diffserv Resource Allocation" <draft-salsano-cops-dra-00.txt>, October 2001.

[11] Stefano Salsano, "QoS Control by Means of COPS to Support SIP-Based Applications" IEEE Network, March/April 2002, Vol. 16 No 2.