Transparent Network Configuration for Internet Telephony Traffic

Antonio Pescapè[1], Maurizio D'Arienzo[2], Giorgio Ventre[1, 3]
[1] Dipartimento di Informatica e Sistemistica, Università di Napoli "Federico II" (Italy)
[2] Dipartimento di Studi Europei e Mediterranei, Seconda Università di Napoli (Italy)
[3] CRIAI Scarl - Consorzio Campano di Ricerca per l'Informatica e l'Automazione Industriale
e-mail : {maudarie, pescape, giorgio}@unina.it

Abstract— **Telephony services represent an increasing class of applications supported by the current Internet. Due to their cheapest cost, many residential users are adopting VoIP solutions. Therefore, Internet Service Providers need to plan and configure their networks to support services with Quality of Service requirements. In this paper we present a transparent and dynamic approach to the network devices configuration according to the "*user traffic profile*". Here, a dynamic configuration of synthetic traffic reproducing real VoIP streams is considered. Our proposal relies on the capabilities of an "*intelligent node*", which is able to observe the network behavior and to configure it in an efficient way.**

I. Introduction

During the last years Internet Telephony has gained an unprecedented success (due to both market and technology evolution): nowadays the main push towards the use of Internet Telephony is the lower cost of IP network connections with respect to traditional networks. A residential user who takes advantage of Internet Telephony for his national and international communication can obtain remarkable savings in the rates applied by telephone companies.
The causes of triggering the "*VoIP (Voice over IP) phenomenon*" have to be found in the increase of both data traffic (which has now exceeded voice traffic) and number of users who get connected to the Internet. This situation has involved a drastic cost reduction of IP network devices. Therefore, an Enterprise or an Home user that decides today to invest in an IP-based integrated network can account for a saving on the overall setup cost that oscillates between 30 and 50%. Yet, IP networks have been planned and constructed in order to support non real-time data applications, like e-mail or file transfer, that are characterized by a discontinuous bandwidth demand (great bursts of data interleaved by long periods of inactivity). Though such networks do not represent a *plug-and-play* solution to voice delivery, they may disclose an unprecedented potential if rearranged in a proper way in order to support several types of traffic on the same infrastructure. One network for all services: traditional phone, advanced phone with messaging services, data transfer, video streaming and TV broadcasting. This integrated environment is nowadays very common in many residential scenarios.

Therefore the main issue to cope with is the integration of several services on the same network infrastructure like the flexible network protocol IP enhanced with the introduction of QoS (*Quality of Service*) features. On the basis of the previous considerations, the insertion of the phone service into an existing network infrastructure can follow two different approaches, each leading to a different architecture: Voice over IP (VoIP) and IP Telephony. By the term VoIP we mean the barebone service, i.e. nude transport with all the problems of forwarding, scheduling, signalling and mapping between traditional telephone protocols (e.g. *SS7*) and IP protocols (e.g. H323 or SIP). Users keep on using standard telephones without even knowing that their conversations go through a data network. In this case there is no relation between telephone constraints and information systems. IP Telephony, on the contrary, means a complete telephone service that looks at the customer terminal as part of the network. The user has a telephone that is an IP device (with an IP address) and exploits the H323 protocol or a proprietary software to implement multimedia functionality on his own PC. In this case there is a full interaction between "*IP data application*" and "*plain telephone networks*".

In this work we present an experimental analysis of transparent configuration of QoS phone calls over IP network. We use an intelligent open platform router to be placed into the network able to hear the traffic and to configure the nodes in an appropriate way.

The rest of the paper is organized as follow. In Section 2 we present a network scenario. In the Section 3 the ARMP (*Available Resource Management Protocol*) actions are presented, while Section 4 presents the open configurable router architecture. In Section 5 a preliminary results analysis is presented. Finally, Section 6 ends the paper with some conclusion and it traces issues for future researches.

II. NETWORK SCENARIO

Nowadays residential services like VoD (Video on Demand) and VoIP are planned and given out in a general framework related to dynamic set up of service on top of a QoS-aware network.
To this purpose a general architecture for the dynamic creation and provisioning of QoS based communication

services can be called *Premium IP* networks [1]. Such an architecture includes key functional blocks at the user-provider interface, within the service provider domain and between the service provider and the network provider. The combined role of these blocks is to manage user's access to the service, to present the portfolio of available services and to appropriately configure and manage the QoS-aware network elements available in the underlying network infrastructure. Their internal operations comprise activities such as authentication, aggregation and a mediation procedure that includes the mapping of user-requested QoS to the appropriate service/network resources, taking into account existing business processes.

In our view, network architectures are expected to be highly heterogeneous in terms of variety of systems and nodes in order to support dynamic service creation and service configuration on top of generic QoS-aware IP networks. Strictly related to this activity there is the management of those resources in the underlying networks that are reserved at registration/subscription stage, as well as those that are used, and maybe subsequently modified, when the service is invoked/configured. Associated with the reservation and usage of resources there is the automated production and presentation of the corresponding SLAs to the user and the translation from the SLA (*Service Level Agreement*) to the corresponding *Service Level Specification(s)* (SLS). Currently network operators can not rely on dynamic and fast re-configuration tools to adapt their networks to the variable requirements of the user. Network interconnections in a network domain are usually established after a service negotiation – this subscription of a contract is also known as SLA (*Service Level Agreement*). Network domains are then configured statically by means of SLSs (*Service Level Specifications*), which represent the technical translation of one or more SLAs. The SLS configuration requires manual intervention to allocate a fixed amount of resource for the entire service lifetime. Unfortunately, this approach to manual (and static) configuration of network domains has many disadvantages, hence a more dynamic model should be adopted. Managing users' SLA/SLS dynamically leads not only to a flexible management of network interconnections, but also to significantly better resource utilization [2].

In this paper, we address the issue of dynamic resource adaptation in network configuration, according to users' (or his organization) individual requirements. Our work stems from the "proactive" computing paradigm [3]; we envision that networked systems will interact to anticipate user requirements in real-time, and, in some cases, take actions on their behalf. We introduce this concept in the management of network interconnections within a network domain, so that manage resources "transparently" and dynamically even after the user SLA subscriptions. Next figure describes the network architecture we envisage. We introduce a new management entity, which we term as AcMe (*Active Mediator*), that is responsible of each network domain. AcMe activities are orchestrated using an innovative protocol, named ARMP (*Available Resource Management Protocol*), that we introduced in such a framework. Stemming from the assumption that a network domain is dimensioned to bear traffic entering in it, AcMe simply interacts with boundary routers of its domain. In particular, an AcMe consistently probes Egress network interfaces of its domain (the nearest to the user) to understand the real traffic profile of the user. In case this profile is significantly different from that previously configured, the AcMe enforces a new network configuration (via SLS) on the Ingress network interface (the nearest to the Service Provider), but still according to the user's SLA saved in a centralized *SLA repository*.

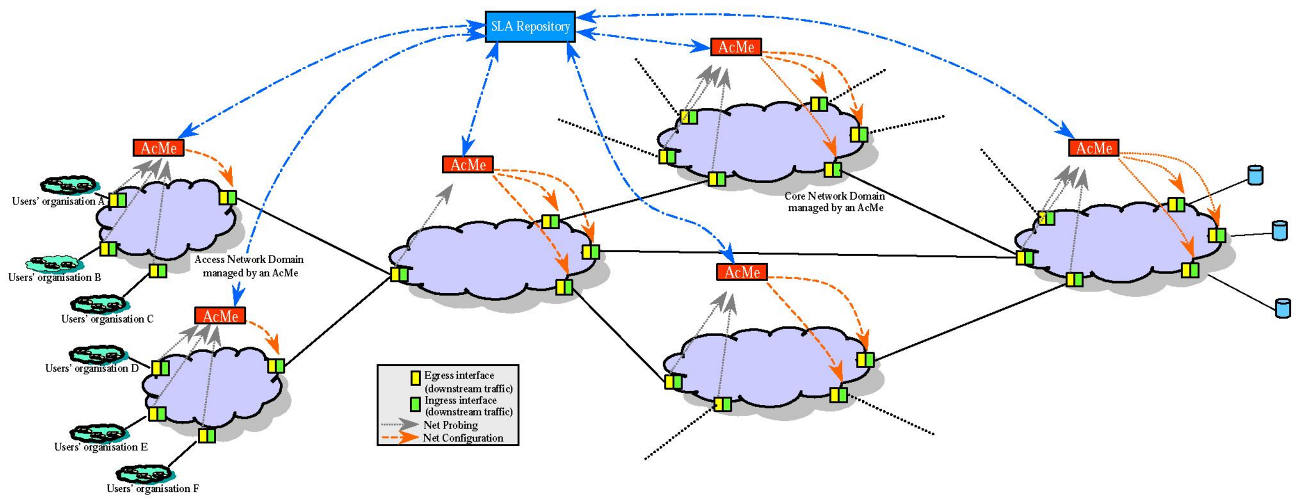

Figure 1 : AcMe activities

III. ARMP

In the scenario described in previous sections, users have to subscribe SLA to make services requests. In our approach, the AcMe consistently probes available resources on boundary routers according to ARMP messages. In particular, the AcMe sends ARMP messages in order to probe the egress network interfaces of its domain (the nearest to the user) to understand the real traffic users' profile. In case this profile is significantly different from that previously configured, the AcMe performs a new network configuration on the ingress domain interface (the one nearest to the service provider). This new network configuration is enforced using ARMP messages related to the SLS negotiation phase. Obviously, this distribution of network resources inside the domain will be managed by the AcMe on the basis of the customer's contract. For instance, customers who have subscribed for higher quality SLAs (e.g., Gold) are preferred. Hence the AcMe performs network probing and commands re-configuration of specific network entities. This activity is carried out according to theoretical results showed in [4] using Diffserv [5] architecture in the core of the network. End users are interested in end-to-end service. Under the assumption that each network domain supports traffic accepted by its ingress routers [6], the AcMe limits admission control activities on ingress routers. Once the flows enter the network, they must be propagated inside the rest of the network according to the quality of service requested. AcMes managing network domains crossed by flows take into the account the number and the class of services of these flows and consistently configure the domain without any knowledge of what happens in near domains. Interaction between AcMes of different domains is eventually needed for inter-domain communications.
The need for an entity which manages SLAs is also motivated by scalability reasons, especially inside the core of the network. It is self-evident that when the number of users increases, many different requests (SLAs) come into play. By simply considering a separate SLA for each different request coming from each user there would be a big SLS jam to be accommodated in each network domain. The presence of a management entity helps in such a kind of aggregation process.
As represented in Figure 1, the AcMe pertains to a network domain and accomplishes the task of local domain devices re-configuration in order to manage internal domain resources. It directly interacts with network devices sending messages to boundary nodes, which are based on an experimental architecture that is detailed in next section.
The AcMe works in two times: in a first moment it sends probing requests (SLS_Network_Condition_request messages) to its network resources and waits for collected information concerning current traffic load (SLS_Network_Condition_response messages). In a second time, a *closed loop control* is activated. The AcMe performs evaluation on received data and, if needed, it operates a new configuration (SLS_configuration_enforce): this new configuration is sent to right network devices. AcMe configures the boundary network nodes by directly sending commands to the boundary nodes. We can summarize the AcMe activities in the following steps:

1. After the initial-static network configuration, the network provider, by means of ARMP messages, consistently probes router network interfaces. In the showed example, the network provider probes the Egress network interfaces of its domain (the nearest to the user) to understand the real traffic profile of the user.
2. In case this profile is significantly different from that previously configured, the provider, still using ARMP messages, performs a new network configuration, on the Ingress router interfaces (the one nearest to the service provider). Obviously, this distribution of network resources inside the domain will be managed by the AcMe, based on the user rights according to their requested service (e.g., Gold better than Silver).

Probed information are then collected by the AcMe, which compares the updated information about traffic profile of a single user or a bundle of profiles related to a users' organization with the previously requested network configuration. When needed, the network configuration is updated to meet the changed requirements.

IV. OPEN CONFIGURABLE ROUTER

In order to emulate the scenario we depicted above, we have designed and implemented an Open Configurable Router to act as ingress/egress router of a network domain. Figure 2 depicts the scheme of our Open Programmable Router prototype. The yellow part represents the egress network interface; the green part is the ingress network interface. The open router definition is based on Programmable Network principles [7]. During its design, we tried to getting over the limitations of the traditional models (*Open Control Networks* and *Active Networks*) by proposing an OCN platform which has the flexibility proper of the AN architecture, in particular it has to provide support for the code installation. However, it is not supposed to perform computations on data path, thus, the whole node performances are not compromised.

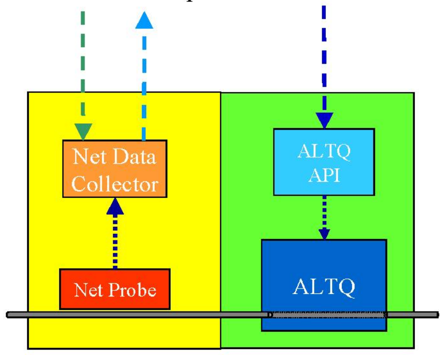

Figure 2 : Open Programmable Router Architecture

Operations on network interfaces like probing and Traffic Control configuration can be performed remotely. The platform runs on a FreeBSD environment with ALTQ [9] Traffic Control Module enabled.

The remote network configuration is based on standard ALTQ APIs. A remote network probing module is also implemented in the router. The implementation allows retrieval of information based on users' traffic profile, for example, number of HTTP requests coming from the users' organization. An open interface allows the interaction with the upper management entity (AcMe). In the next Section we present an analysis of results we achieved from some experiments carried out on a laboratory testbed based on the presented architecture.

V. EXPERIMENTAL TESTBED

To demonstrate AcMe functionalities, we made use of an experimental testbed exploiting functionality of an prototype implementation of the open router. The scenario emulates a single network domain where users of a corporate network exploit a service offered by a single service provider. We implemented a distributed prototype of AcMe and Open Configurable Nodes that acts as enhanced gateways (ingress and egress routers) of a network domain aiming at distributed network management activities [8]..

```
# a config for hierachical sharing

interface fxp0 bandwidth 10M hfsc
#
# 100Kb/s of the bandwidth to the default class
#
class hfsc fxp0 def_class root grate 100K default
#
#    bandwidth share    guaranteed rate
#    tcp_class:1Mbps
#    udp_class:6Mbps
#
class hfsc fxp0 tcp_class root grate 1M
class hfsc fxp0 udp_class root grate 6M

#   bandwidth share    guaranteed rate
#    tcp_other:0,5Mbps
#    http_class0  Mbps
#

#TCP
class hfsc fxp0 tcp_other tcp_class grate 0.5M
filter fxp0 tcp_other0 0 0 0 6
# HTTP CLASS
class hfsc fxp0 HTTP_class tcp_class grate 0M
filter fxp0 HTTP_class0 0 0 80 6

#UDP
class hfsc fxp0 VoIP_class udp_class grate 0M
filter fxp0 VoIP_class 0 10000 0 0 17
filter fxp0 VoIP_class 0 10001 0 0 17
filter fxp0 VoIP_class 0 10002 0 0 17
filter fxp0 VoIP_class 0 10003 0 0 17
filter fxp0 VoIP_class 0 10004 0 0 17
filter fxp0 VoIP_class 0 10005 0 0 17
filter fxp0 VoIP_class 0 10006 0 0 17
filter fxp0 VoIP_class 0 10007 0 0 17
filter fxp0 VoIP_class 0 10008 0 0 17
filter fxp0 VoIP_class 0 10009 0 0 17
filter fxp0 VoIP_class 0 10010 0 0 17
filter fxp0 VoIP_class 0 10011 0 0 17
filter fxp0 VoIP_class 0 10012 0 0 17
class hfsc fxp0 RTP_class udp_class grate 0M
filter fxp0 RTP_class0 1234 0 0 17
```

Figure 3 : ALTQ Configuration

The experimental testbed consists of an ingress/egress routers with two network interfaces, interposed between several client/server VoIP applications running on opposite sides of router. By using the testbed, we performed a comprehensive analysis that demonstrate how our system transparently understands users' requirements (VoIP calls) in real-time, and configures the network dynamically to meet any new (updated) requirements of the user (a new VoIP call). The VoIP traffic, as well as cross traffic, have been generated with D-ITG (*Distributed Internet Traffic Generator* [10]).

The scheduler installed on ALTQ is HFSC. The initial configuration gives no bandwidth to VoIP class, just 100Kb/s to default traffic, 500Kb/s to TCP class. We make the assumption that VoIP flows have destination port numbers starting from 10000 up to 10012.

The flows generation at server side is described in figure 3. A 4 Mb/s UDP cross-traffic is persistently active from the beginning until t=300s, while 12 VoIP flows, each one of 70.4 Kb/s, start in progression, one every 30 seconds. Thus, at time t=150s, 5 VoIP flows are active, beyond the UDP cross traffic. At time t=300s, all 10 VoIP flows stopped, as well as UDP cross traffic. Finally, at time t=350s and t=380s, two new VoIP flows begin, and terminate at t=430s.

From an analysis of throughput graphic at receiver side, as reported in figure 4, we can see how cross traffic is strongly penalized because it just enters the default class. Although at time t=0s there are no resources for VoIP flows, the router adapts its configuration when it becomes aware of each active VoIP flow. Finally, the figure 5 presents a packet loss report. Of course, while UDP cross traffic experiences an increasing number of packet loss, after a re-configuration and adaptation time of the router,losses related to VoIP flows asymptotically stabilize to almost zero. The anomaly in flows 5 and 9 are probably caused by a not well triggered choice of configuration parameters. We will repeat experimentations in order to reach the correct router configuration.

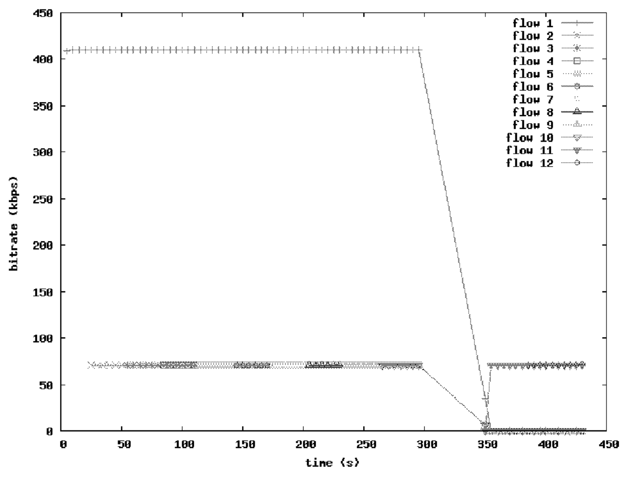

Figure 4 : Throughput at Sender Side

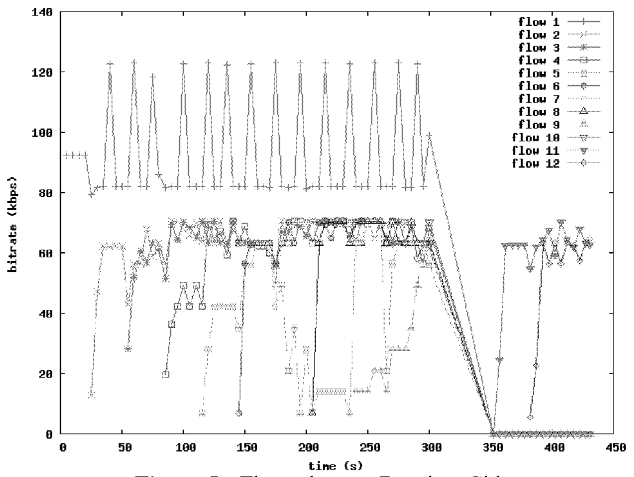

Figure 5 : Throughput at Receiver Side

VI. CONCLUSIONS AND ISSUES FOR FUTURE RESEARCH

This paper presented an architecture for the dynamic and seamless configuration of the network nodes based on the on-line measuring of the "*user traffic profile*". Activities of this architecture has been described and a preliminary experimental analysis has been used for the architecture validation. In order to show the powerful of the proposed system we used VoIP traffic. A more precise analysis of such a system is presented in [11].

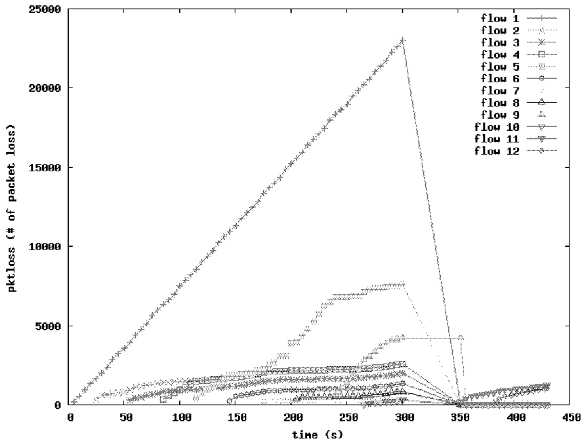

Figure 6 : Packet Loss at Receiver Side

Our project entails interesting research for future investigation. For example, how do we "systematically" configure Dynamic SLAs/SLSs for networks based on different traffic types?

We will like to extend this work to include real-time, and other types of flows, based on the diverse requirements of the residential user (i.e. network games). Presently, our testbed allows experiments on a small-scale. We will like to test the scalability of the system on a real network and on a much wider-scale. Mobile Internet access using WLANs and GPRS/3G has gained good popularity. We are investigating an extension of Dynamic-SLAs/SLSs even in the future heterogeneous wireless environments [12].

ACKNOWLEDGMENT

This work has been partially supported by the Italian Ministry for Education, University and Research (MIUR) in the framework of the RECIPE Prin Project and by EU in the field of OneLab and Netqos Projects and Content NoE.

REFERENCES

[1] G. Cortese, R. Fiutem, P. Cremonese, S. D'Antonio, M. Esposito, S.P. Romano, A Dioconescu, "CADENUS: Creation and Deployment of End-User Services in Premium IP Networks", IEEE Communications Magazine, pages 54-69, Vol.41, No. 1, January 2003

[2] M. D. Arienzo, M. Esposito, S. P. Romano, G. Ventre. "Dynamic SLA-based Management of VPNs". In Proceedings of IWDC 2001 (LNCS–2170), Italy.

[3] D. Tennenhouse, "Embedding the Internet: Proactive Computing". Communications of the ACM, Vol. 43, Issue 5, p. 43-50, May 2000.

[4] M. Mellia, C. Casetti, G. Mardente, M. Ajmone Marsan, "An Analytical Framework for SLA Admission Control in a DiffServ Domain", INFOCOM 2003

[5] S. Blake, D. Black, M. Carlson, E. Davies, Z. Wang, and W. Weiss, "An Architecture for Differentiated Services", RFC 2475, Dec. 1998.

[6] C. Cetinkaya,V. Kanodia, and Edward W. Knightly. "Scalable Services via Egress Admission Control", IEEE Transactions on Multimedia, Vol. 3, No. 1, March 2001 p. 69

[7] A.T. Campbell, H.G. DeMeer, M.E. Kouvanis, K. Miki, J.B. Vicente, and D. Villela. "A Survey of Programmable Networks". Computer communication Review, 29(2):7-23, April 1999.

[8] R. Boutaba, A. Polyrakis "Projecting Advance Enterprise Network and Service Management to Active Networks". IEEE Network, Janauary/February 2002.

[9] Kenjiro Cho, "Managing Traffic with ALTQ", In proceedings of USENIX 1999, Annual Technical Conference: FREENIX Track, Monterey CA

[10] http://www.grid.unina.it/software/ITG

[11] M. D'Arienzo, A. Pescape', G. Ventre "Dynamic Service Management in Heterogeneous Networks", International Journal on Network and System Management, Vol. 12, No. 3, pp. 349-370, Sep. 2004, ISSN 1064-7570

[12] M. D. Arienzo, R. Chakravorty, I. Pratt, J. Crowcroft, "A Framework for Dynamic SLA-based QoS Control for UMTS", in IEEE Wireless Communications Magazine, October 2003.

This page intentionally left blank.