SLA related Control of Internet Metering: Overview

Marcelo Pias Steve Wilbur
University College London (UCL)
Department of Computer Science
Gower Street, WC1E 6BT, London, UK
{m.pias, s.wilbur}@cs.ucl.ac.uk

Abstract

In this short paper we give an overview of work in progress to propose a distributed metering model using Active Network concepts. The model is dynamically controlled by high level policies such as tariffs and service level agreements (SLAs). We briefly describe some of the issues being investigated such as Tariff/SLA translation and metering performance.

1 Introduction

Our intention is to provide a means for building a safe distributed meter system for QoS monitoring in IP networks. The model is dynamically controlled by high level policies such as charging tariffs and service level agreements (SLAs). The Meter system has the intelligence for deciding the data needed, when should be forwarded and for whom. In [1] the author describes another approach for IP accounting using Active Networks with accounting functions performed by the Provider's system only. Unlike in [1], our model not only allows accounting processes to be performed at provider's system but also at customer's premises. Furthermore, as stated before we are very concerned with the security and performance issues imposed by the creation and operation of the system.

2 Distributed Meter Model Proposal

Our model proposal is composed by two phases (states) and two events. The event of *Meter Creation* moves the system from initialization to operation. A *Meter system Update* event occurs when the system demands any software/hardware update. Basically, there is a process of cutting over to the new code associated to this latter event. We have modeled the system using some of Active Network (AN) concepts [2].

2.1 Definitions

- **Proxylet:** third parties' mobile code to be executed into a Generic Active Node (GAN) [2].
- **Generic Active Node (GAN):** it receives proxylets for execution on behalf of third parties.
- **Specialized Active Node (SAN):** in certain cases it might be useful to specialize a GAN through filters which accept specific group of proxylets to be executed.
- **Meter System:** a Meter system consists of a set of Meters plus a set of Dynamic Accounting Controllers (DAC).
- **Meter:** specialized active node created when a GAN receives a Meter_Code proxylet. A Meter captures packets from the network and apply policies in order to create a table of measurements.
- **Dynamic Accounting Controller (DAC):** another specialized active node created upon a GAN has received a Dynamic Accounting Controller (DAC) proxylet.

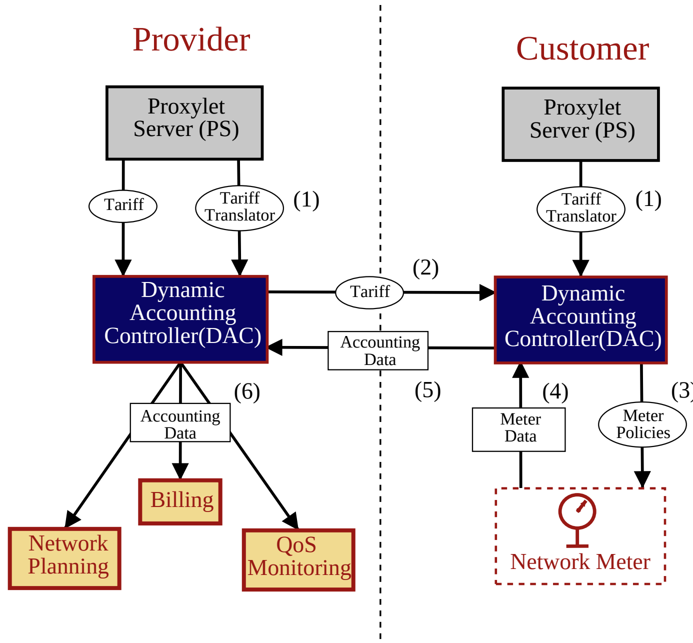

Figure 1: Phase 2 (Operation)

- **Tariff/SLA Translator (TT):** proxylet responsible for compiling the Tariff/SLA specification into Meter policies.

2.2 Model Phase 1: Initialization

In this phase, the main components of the Meter system (Meter + DAC) are sent as mobile code over the network. Initially, the DAC proxylet version 1 is delivered to the Provider premise. The Meter_Code version 1 is transmitted to the Provider's Generic Active Nodes. A copy is stored at local Proxylet Server at Provider's premises. In steps 3, 4 and 5 the Meter, DAC and Tariff/SLA Translator proxylets are shipped to Customer's premises. They carry version numbers and Meter Manufacturer (MM) identifiers.

2.3 Model Phase 2: Operation

In step (1), Tariff/SLA and its translator proxylets are loaded into Dynamic Accounting Controller (DAC). The Provider sends Tariff/SLA proxylets to his Customers, either through multicast or unicast communication (2). Once a Customer's DAC has received it, a Tariff/SLA Translator proxylet is loaded in order to perform the translation into appropriate Meter/Accounting policies (3). They specify explicitly requirements in a way that the Customer's Meter system can cope with. The DAC collects meter data either using push or pull modes(4). The Customer's DAC sends accounting data (detailed or summarized) to Provider's DAC. Finally, such collected data may be used for Billing, Network planning, QoS monitoring applications.

3 Tariff/SLA-related Control of the Model

In this section we look at a mechanism to perform a translation from a tariff (high-level) to appropriate Metering policies (low-level). A tariff/SLA expresses a set of charging and QoS monitoring policies in a declarative way. However, there exists a binding mechanism between tariff/SLA (high-level) and Meter configuration (low-level). Such relation may be achieved through metrics specification which includes: (a) type; (b) constraints; (c) implementation scheme; (d) associated event; (e) pattern matching filters. The language that we have designed is able to express metrics such as (a) traffic-related (e.g. session duration, volume, loss, throughput, delay, jitter); (b) environment-related (e.g. congestion indication); (c) classification-based such as DiffServ DSCodePoint, protocol-related, address and time attributes.

Following we present a simple arbitrary congestion-based tariff with event-condition-action (ECA) rules. It relies on the ECN (Explicit Congestion Notification) proposal [3] as a means for getting congestion feedback from the network.

Tariff/SLA description	**Translated meter policies**

<pre>price_byte_unit = 0.4;
ecn_marks: counter;
bytes_sent: counter in bytes;
on ecn_marks.congestion
   if (price < 1000)
     do {
       price = ecn_price +
              price_byte_unit*bytes_sent;
     }</pre>
	<pre>on initialization
	   do {
	     create counter name ecn_marks bound to
	       packet_arrival.pkt[NETWORK_ECN_BITS]=11;
	     create counter name bytes_sent bound to
	       packet_arrival.pkt.length;
	   }
	on packet_arrival
	   do {
	     update bytes_sent;
	     update ecn_marks;
	     prepare output;
	   }</pre>

Table 1: Tariff/SLA and its translation to meter policies

4 Meter System Performance

We have performed initial experiments to assess meter calibration metrics such as Meter Error Rate (MER) and traffic-related metrics such as Volume(V) and Packet Rate (PR). We also have used a tariff based on the IETF DiffServ architecture [4]. An arbitrary tariff was delineated with basis on six classes as described in [5]. We have done the mapping from that model to IETF DiffServ classes. It had considered the traffic characteristics of each class (interactive realtime, non-interactive realtime and non-realtime).

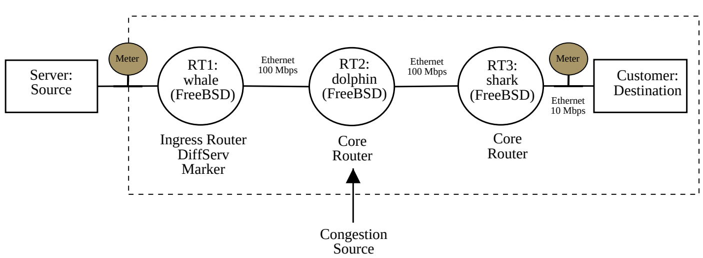

Figure 2: DiffServ Testbed

We have generated traffic for six distinct application categories: (1) (Gold, Interactive Realtime); (2) (Gold, Non-Interactive Realtime); (3) (Silver, Non-Interactive Realtime);(4) (Gold, Non-Realtime); (5) (Silver, Interactive Realtime); (6) (Silver, Non-Realtime). When alternating the meter position throughout the testbed we can assess the Volume (V) and Packet Rate (PR) metrics. The testbed used in the experiment is shown in Figure 2. The server (source) and the customer machine (destination) run Linux. Router RT1 marks ingress packets with DSCodePoint in respect of the six categories of applications. A multi field (MF) classifier operates at RT1 mapping applications into the desired DSCodePoint. Three experiment scenarios are used: (1) Meter at Customer and Network without Congestion; (2) Meter at Customer and Network Congested and (3) Meter at Source.

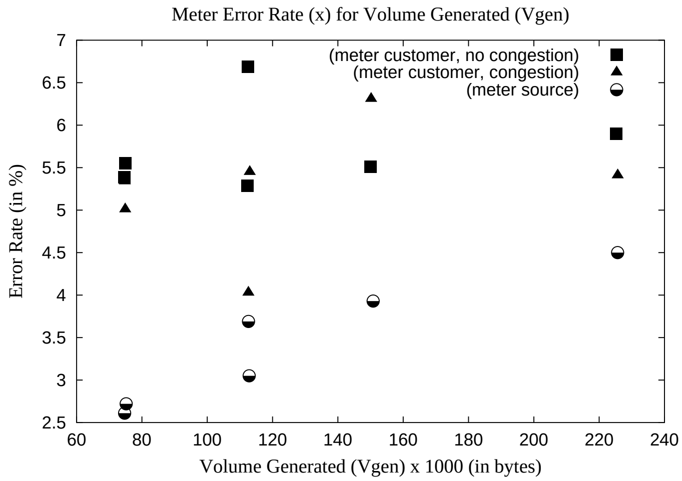

Figure 3: Meter Error Rate(x) over Volume Generated (Vgen)

5 Results

Some interesting results might be deduced from the data collected. There is relative discrepancy in the actual metered volume. Although is true to say that, we do not know whether this holds or not regardless of meter positions. Thus, more experiments switching meter points over the Provider network (IETF DiffServ domain in Figure 2) need to be be performed. We want to work out an association between the volume metered, generated and charged in congestion periods. At present, we define the volume relation being the following equation:

$$V_{met} = (1 - x).V_{gen} = (y).V_{chg} \tag{1}$$

where: V_{met}: Volume metered, V_{gen}: Volume generated, V_{chg}: Volume charged, x: Meter error rate, y: Volume charged coefficient

It should be pointed out that those volumes are semantically distinct. The parameter x in the equation 1 was determined by the experiment data (meter error rate). It has floated around 5% on average according to Fig. 3. The possible cause may be linked to the FreeBSD Kernel (meter machine) difficulties for handling all packets seen. The ambition of assessing DiffServ classification-based metrics has been reached successful. This may prove that we can meter and charge for such metrics.

References

[1] Franco Travostino, "Towards an Active IP Accounting Infrastructure," in *Third IEEE Conference on Open Architectures and Network Programming - OpenArch 2000*, Mar. 2000.

[2] Michael Fry and Atanu Ghosh, "Application Level Active Networking," *Computer Networks*, vol. 31, no. 7, pp. 655–667, 1999.

[3] K. Ramakrishnan and S. Floyd, "RFC 2481 - A Proposal to Add Explicit Congestion Notification (ECN) to IP," *IETF*, Jan. 1999.

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

[5] M. Alfano, M. Krampell, and M. Smirnov, "End-to-End Quality in IP networks: Can we offer and charge it?," in *XVII World Telecommunications Congress, Birmingham (UK)*, May 2000.