HAL
open science

Design and Evaluation of a Session-Aware Admission Control Framework for Improving Service Providers' Profitability

Narjess Ayari, Laurent Lefèvre, Denis Barbaron

To cite this version:

Narjess Ayari, Laurent Lefèvre, Denis Barbaron. Design and Evaluation of a Session-Aware Admission Control Framework for Improving Service Providers' Profitability. Journal of Internet Engineering, 2010, 4 (1), pp.260-268. hal-00793633

HAL Id: hal-00793633

https://inria.hal.science/hal-00793633v1

Submitted on 26 Feb 2013

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Design and Evaluation of a Session-Aware Admission Control Framework for Improving Service Providers' Profitability

Narjess Ayari*, Laurent Lefèvre† and Denis Barbaron*

***Abstract*— Service-aware network management is a key issue for the current and next generation service-oriented networks. In this work, we focus on improving the QoS user experience (QoE) of Internet subscribers connecting to heavily loaded servers. Firstly, we advocate an innovative architecture for fine-grained admission-control of Internet servers subjected to multiple flow-based sessions. The proposed architecture is based on original concepts which meet the constraints of multiple-flow based sessions by explicitly identifying the flows pertaining to a single subscriber's session. Secondly, based on an economically-justified service model, we investigate novel session-aware admission-control for improving providers' profitability. Our fundamental observation is that it is sometimes desirable to reject some customers' new sessions so that others may be completed and thereby generate some revenue for the service provider. We evaluate the performance of the advocated model by conducting advanced simulations where we consider realistic CBR voice traffic. Results demonstrate that responsive session-aware admission-control improves service provider's benefits. For a satisfactory operator's revenue, session-aware admission-control improves the success rate by more than *69 %*, compared to standard request-aware admission-control.**

***Index Terms*— Admission-control, session awareness, next generation services.**

I. Introduction

When overloaded, an Internet server has not sufficient resources to provide service to all clients. It contributes to decreasing the Quality of Service (QoS) through sustaining heavy request queuing delays and prolonged processing time. In this work, QoS refers to the satisfaction of application key performance indicators (KPIs) such as delay, jitter and packet loss, while QoE refers to the high end user experience. Since a poor perceived performance (QoS) is a foremost impediment to the success of any service, an operator needs to provide an acceptable Qo(E/S) for the admitted network traffic.

Admission-control is a well-known means to prevent a server or a network resource from overloading. It consists of regulating the acceptance of the offered network traffic according to the usage of the controlled resources. Research on admission control can be split into server-side and network-side mechanisms. While network-side admission control mechanisms allocate bandwidth according to resources' usage of network paths, they do not consider routers' or servers' contribution to QoS and perceived users' QoE degradation. Instead, they assume routers have sufficient buffers and are rapid enough to accommodate large amounts of the already admitted network traffic. Conversely, network paths typically consist of a multitude of network devices that definitely, have different processing capacities, in terms of CPU and memory. Additionally, dimensioning edge processing servers according to the expected network traffic and access links' capacities does not guarantee zero congestion. Indeed, the network operator may choose to dimension the links according to the expected average rather than the peak traffic rates due to cost effectiveness. Such approach is preferable to over-sizing the network, particularly at the access. However, it leads to more traffic burstiness and congestion. Hence, in order to guarantee acceptable end users' QoE, the network traffic must be controlled, both according to the network and the servers' constraints.

In this work, we focus on server-side traffic control mechanisms. Indeed, edge servers may be subjected to non-controlled network traffic. As such, not only does a processing server require enough processing resources, its resources, also, need to be prevented from overloading.

Most state-of-the-art researches on server-side admission-control advocate session-oblivious mechanisms where dropping requests pertaining to an accepted session can occur at any time during the session's lifespan. From an operator's perspective, this means that the server, which seems to sustain a high throughput, is in reality wasting its resources on failed or on interrupted sessions. Well investigated admission-control mechanisms fall mostly into packet-level and flow-level admission-control strategies [1,2,3,4]. These approaches are very suitable for a single flow-based session, i.e. for a subscriber's session that spans over a single transport level flow. Conversely, the concept of customer-session is of particular interest for many applications. For commercial web sites, for instance, a customer sends a number of HTTP requests to the web site during a single session. For the web site to be successful, it is important that as many customers as possible complete their sessions, since only completed sessions may generate revenue. Voice-over-IP is another emerging application requiring improved QoS. Firstly, from an economically-motivated point of view, rejected or interrupted calls are not charged to customers and result in an increasing number of angry clients. Secondly, they waste the operator's processing resources (network and server sides). Since the overload of a call server or an intermediate gateway counts among the potential reasons for the interruption of ongoing calls, admission-control is very desirable to guarantee an acceptable QoS for the admitted multiple-flow-based VoIP calls. Intelligent admission control is expected to considerably reduce network resources' wasting across congestion.

Manuscript received on February 15th, 2010. This work was supported in part by Orange Labs (France Telecom R&D) and by the INRIA, France.

*Narjess AYARI was with Orange Labs, 2, Avenue Pierre Marzin, 22307 Lannion, France (phone: +33(0)296052738; fax: +33(0)296051111; e-mail: narjess.ayari@orange-ftgroup.com).

†Laurent Lefèvre is with INRIA RESO - LIP (UMR CNRS, ENS, INRIA, UCB), Université de Lyon, 46, allée d'Italie - 69364 LYON, France (phone: +33(0)472728228; fax: +33(0)472728080;e-mail: laurent.lefevre@inria.fr).

*Denis Barbaron is with Orange Labs, 2, Avenue Pierre Marzin, 22307 Lannion, France (phone: +33(0)296050357; fax: +33(0)296051111; e-mail: denis.barbaron@orange-ftgroup.com).

In this work, we advocate an admission-control architecture that uses deep packet inspection to provide session awareness to network nodes subjected to multiple-flow-based next generation services. Firstly, we describe the building blocks of the session-aware packet forwarding architecture. Secondly, we investigate a novel economically-justified session-aware admission-control paradigm for improving resources' usage and profitability of session-oriented servers. Our fundamental observation is that it is desirable to reject new sessions from some customers, so that others may be completed successfully, and thereby generate some revenue for the service provider.

Session aware admission control is expected to reduce network resources' usage as accepted sessions are guaranteed completion independently of their duration. Hence, only signalling requests, required for establishing sessions, are firstly subjected to rejection. When congestion persists, requests pertaining to the already accepted sessions may be dropped. However, we can imagine that for well behaved sessions, sources are intelligent enough to control their sending rates according to experienced packet loss. Unless some higher levels flow control is implemented, UDP-based sources, however, may continue sending traffic across congestion. This issue has been already addressed at the network level with reliability built-in evolved UDP protocols that support some kind of congestion control, such as reliable UDP [10].

Through simulations, we demonstrate that responsive session-aware admission-control improves the service provider's benefits and contributes to providing better QoE to subscribers. Particularly, we show that session-aware responsive admission control provides improved completion of sessions, independently of their durations.

The remainder of this paper is organized as follows. In section II, we review some research works on admission-control for Internet servers. In section III, we describe the architecture of the session-aware admission-control framework that we advocate. Section IV is devoted to presenting the session-aware mechanism that we advocate to control the offered network traffic to an Internet server. In section V, we evaluate the performance of our proposal and show that our approach provides better profitability to the service provider and contributes to the completion of the offered sessions independently of their duration. Finally, we conclude this paper and outline interesting future directions.

II. Background and Related Works

Admission-control mechanisms for Internet servers have almost the same objectives as admission-control mechanisms for the core network. The latter typically involves a gateway that estimates the offered network load by measuring the queue length of its backlogged packets. The admission-control policy uses this value to regulate the acceptance of the incoming packets. Drop Tail algorithm describes the simplest form of packet discarding. It suggests dropping the incoming packets when the router's buffer space becomes full. This idea is obviously applicable to the admission-control of requests made to Internet servers. However, its major drawback is that it rejects aggressively and arbitrarily the incoming traffic while similarly approving different classes of traffic. Both the Early Random Drop (ERD) and the Random Early Detection (RED) approaches [1,2] addressed Drop Tail's deficiencies. They suggested preventing the router's congestion rather than simply reacting to it. In essence, the idea is to prevent the backlog saturation by anticipating the rejection of the incoming packets as soon as the backlog queue's length goes beyond a predefined threshold. A packet will be dropped with a probability computed according to the router's backlog queue length. Randomization tends to ensure that all classes of traffic will suffer the same loss rate. Chen and Mohapatra [3] applied the ERD approach to regulate the acceptance of web server's requests. They applied a double-threshold-based admission-control on the application server's listen queue. Requests of lower priority are rejected with a higher probability as soon as the server's utilisation exceeds the first threshold. All requests are rejected when the second threshold is reached. This approach was shown to be effective for controlling differentiated services, mainly in terms of queuing delays between lower and higher traffic-priority classes. However, it has one major drawback: the application server's queue length does not necessarily represent a good indicator of the server's experienced load. Abdelzaher *et al.* [4] assumed a linear regression method which estimates the impact of the handled requests on the system's utilization. They used a linear feedback control-theory which admits an appropriate number of requests while keeping the system utilization bounded. However, they didn't consider any further constraint on the handled traffic. Lee *et al*. [5] focused on the web server and assumed the knowledge of the request arrival rate as well as the maximum waiting time for each incoming traffic class. They suggested two admission-control approaches. The first approach maximizes the potential profit of the service provider, while the second one admits as many clients as possible into the controlled web server. Carlstrom and Rom [6] presented algorithms and an architecture for optimizing the performance of web services. They advocated an analytical admission-control model, to maximize the service provider's objective. Their key idea was to break down a given web session into stages each having specific requirements and transition probabilities. The server must be aware of this structure so as to provide QoS-aware processing.

III. Architecture of the Session-Aware Admission-Control Framework

The system that we advocate achieves session awareness by explicitly identifying flows pertaining to a single user session. The rejection of an offered datagram is based on the server resource's usage as well as on the information conveyed by that datagram.

As illustrated by Fig. 1, traffic flowing from clients to the server is firstly processed by the session identifier module, which associates each incoming datagram either to a new session or to an already established one.

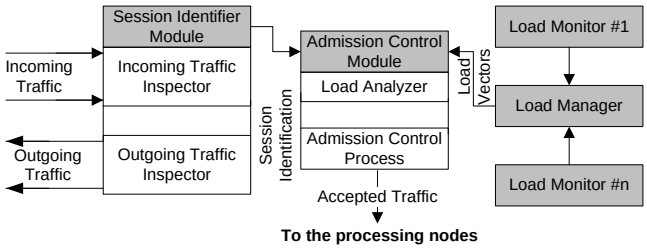

Fig. 1. The functional architecture of the session-aware admission-control framework

The incoming traffic is then delivered to the admission-control module which decides whether to accept it or reject it, according to the server's current load information. In essence, the admission-control module aims for maximizing the provider's profitability by giving, during congestion time, a higher priority to the already established sessions, compared to the newly offered ones. The server's load information is periodically collected by a load manager process. In case the server is based on a cluster of nodes, independent load monitor processes are deployed on every node to collect and periodically send load information updates to a single load manager process that is deployed at the cluster entry point.

A. *The session identifier module*

The session identifier module is a stateful engine which inspects the payload of both the incoming and the outgoing datagrams, searching for given *patterns*. Datagrams which are subjected to content inspection are those exchanged over the signalling flows. Indeed, a multiple-flow based session, such as a video streaming session or a voice over IP session, negotiates data flow's as well as control-data flow's identifiers using messages sent over the main signalling flow. This is typically the case of an RTSP/RTP/RTCP-based video streaming session where identifiers of data flows (RTP-based) and control-data flows (RTCP-based) are negotiated at session setup using RTSP signalling messages. Hence, assuming a typical multiple-flow-based session model [1], a searched pattern corresponds to a specific application-layer protocol header field that holds the information required to identify any incoming flow that expected to be associated to the already established session. Hence, the content inspection used by our architecture is application-layer-specific and is performed with respect to the syntax of the used signalling protocol.

The session identifier module maintains an in-memory session table which is updated with the receiving of an incoming traffic or with the inspection of the outgoing traffic. Table updates consist in adding new entries or updating already existing information.

A session is identified as a set of transport-level flows, necessary for signalling the session and transporting related data and control-data packets. Assuming an IPv4-based network protocol stack, a given flow is identified using a 13-bytes-long vector, which holds the source and destination IP addresses, the source and destination port numbers, as well as the used transport protocol. In our architecture, the flow state maintained within the session table includes moreover a set of variables necessary to track a given flow during its lifespan. These variables are a timeout, a timestamp, an identity flag and a status flag. The timeout and the timestamp are used to detect the flow inactivity. The status flag is used to differentiate between NEW, ESTABLISHED and INACTIVE flows.

A flow is in the NEW status during the receiving of the first datagram asking for its establishment. The activity of each established flow is tracked over time. When no data is exchanged over an ESTABLISHED flow for a certain time duration, it enters the INACTIVE status.

The identity flag says whether the corresponding flow is a SIGNALLING flow, an ANNOUNCED flow or a SECONDARY flow. We define a signalling flow as a flow that carries application-level signalling messages used to establish the End-to-End subscriber session. ANNOUNCED flows are either data flows (such as RTP flows) or control-data flows (such as RTCP flows) that are expected, but not yet established with the server, within the same subscriber session. The session identifier process reads the announced flows' identifiers by inspecting the content of either the incoming or outgoing signalling messages. An announced flow becomes SECONDARY as soon as it is successfully established with the server.

Fig. 2 below summarizes the processing of the session identifier engine.

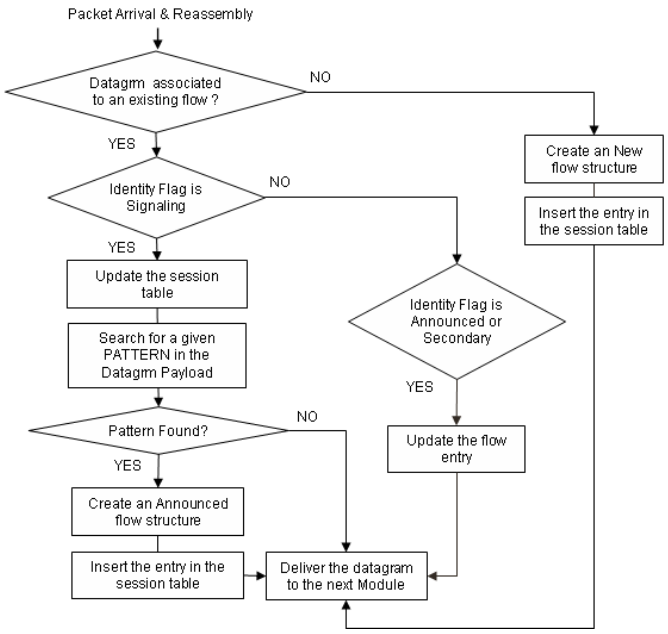

Fig. 2. Activity diagram of the session-identifier module.

The flow identifiers of an offered datagram, received by the session identifier module, are firstly determined. If the offered datagram does not pertain to any recorded flow within the session table, it is assumed to hold a request for the establishment of a new signalling flow. A new flow structure is therefore created and added to the session table. Particularly, the flow is marked as NEW, its identity flag is set to reference a SIGNALLING flow and its inactivity timeout is armed. The datagram is then delivered to the next engine, i.e. to the admission control engine. If on the other hand the offered datagram is associated to an already recorded flow, the identity flag of this flow is checked. The first alternative is that the offered datagram pertains to an already established signalling flow. The corresponding flow entry in the session table is then updated. Particularly, the flow is marked as ESTABLISHED, its inactivity timeout is restarted and its timestamp is updated. The datagram payload is then inspected searching for a specific application-level pattern. If the searched pattern is not found, the datagram is delivered to the next engine. Otherwise, the found pattern is used to build a

new entry in the session table that is referencing an ANNOUNCED flow. The datagram is then delivered to the next engine. In most cases, the outgoing traffic is also required to be inspected so as to complete the identification of the announced flows. Once the entire identity of the announced flow is acquired, the corresponding flow entry is added to the session table.

The second alternative is that the offered datagram pertains to an ANNOUNCED flow or to a SECONDARY flow. The corresponding entry is then updated. Particularly, an ANNOUNCED flow enters the SECONDARY status. In all cases, the inactivity timeout is restarted and the timestamp value is updated. The datagram is then delivered to the next engine.

In order to incur a reduced latency to the handled packets, our architecture requires a session table structure that provides efficient search and insertion performance. On the other hand, since the number of the handled flows can reach up to some thousands, each flow is abstracted using a hash transformation which quickly computes a flow digest while avoiding collisions. Hash transformations are indeed well known techniques for uniquely identifying flows while occupying relatively small memory space [8]. Also, the detection of inactive flows is particularly critical for the session identifier module because it affects the session table size and therefore the search and insertion performance. In practice, the timeout value ranges between 10 and 60 seconds. Finally, in order to take into account any possible packet delay inside the network, an inactive flow is not immediately flushed from the session table as soon as its timeout expires. Instead, a purging process periodically removes inactive flows' entries, when at least twice the corresponding timeout value elapses [9].

B. *The admission-control module*

We advocate different strategies for admitting or rejecting an offered packet to a server that runs close to congestion. Admission-control decision is based on the information that the offered packet holds as well on the instantaneous server's load. Section IV is devoted for describing the evaluated strategies.

C. *The load management module*

The load management engine involves two peer components aiming respectively, at collecting and monitoring the load of the server or the cluster nodes' load. Load vectors are periodically collected and sent to the monitor process over an UDP channel. A measure of the load includes significant indicators of the usage of the server's resources, such as CPU usage, memory usage, network buffers' usage, I/O and application server backlog queue length.

IV. Proposal of Session-Aware Admission-Control Strategies for Internet Servers

A. *Problem Statement and motivations*

Fundamental to our proposal is the characterization of the operator's reward as the mean monetary equivalent due to the blocking, the completion and the interruption of the offered sessions over a long time scale. On the basis of economical arguments, we define the operator's reward as the mean pay-off that encompasses the following three performance criteria.

Firstly, in an ideal situation when the Internet server does not suffer overload, an accepted session terminates normally. The consequence of that good functioning is user satisfaction, which translates into a positive image of the operator regarding competition. On the contrary, when the server experiences overload, any newly offered session will be blocked, resulting in bad user perception of the QoS provided by the system. Finally, when the server runs close to its capacity and is overloaded, the already active sessions are most likely to be interrupted.

Based on the operator's point of view, we assume that interrupting existing sessions is even worse than blocking new sessions, since interrupted calls are not charged to customers, while resulting in an increasing number of angry clients and causing the operator's processing resources to be wasted.

Let us associate to the successful completion, rejection, and interruption of a session an equivalent monetary value that describes a per-session contribution to the total operator's benefit, denoted by R_c, C_b, and C_i, respectively. If N_c, N_b, and N_i stand for the mean number of completed, rejected, and interrupted (respectively) sessions per unit of time, then the operator will try to maximize the following objective function (1),

$$R := R_c N_c - C_b N_b - C_i N_i , \quad (1)$$

The decision variables that the operator would play on to maximize this objective R are the acceptance probabilities of the newly offered sessions, denoted below by (p).

B. *Proposed session-aware admission-control*

From the above analysis, it turns out that the optimal network traffic acceptance policy would accept an offered session to the server if and only if the reward resulting from accepting the session is more important than the penalty resulting from rejecting it. This policy is expected to maximize the operator's profitability by maximizing the number of completed sessions under heavy load.

In the rest of this section, we describe the structural properties of the acceptance strategies that we advocate to optimize the operator's global reward (R). We believe that in a more realistic and precise model, the admission-control decision should depend on the load of the system [7, 8], which could, for simplicity reasons, be measured in terms of number of packets pending in the server's network buffers.

Firstly, when the server's load value is less than a threshold denoted by (T_1), the incoming traffic is accepted and forwarded to the next processing entity. As soon as the server's load exceeds the (T_1) threshold, any incoming traffic holding a request for the establishment of a new session is dropped with a probability (p) that is computed as a function of the server's instantaneous load. In the remaining of this paper, the probability (p) denotes the probability of rejection of a new session. When the server's load value exceeds a second threshold value, denoted by (T_2), only the traffic pertaining to the already established sessions is admitted. This rule is introduced mainly to avoid the interruption or the QoS degradation of the already established sessions during congestion. Particularly, we will show that this rule is necessary to reduce discrimination against long-term sessions. Indeed, short-term conversations typically have a higher chance to be completed successfully. When the server runs very close to its maximal capacity, denoted by (C), the offered traffic is dropped, until some processing resources are released. For improved responsiveness, an offered datagram is dropped with a probability (p) derived as a function of the

instantaneously measured server's load denoted by (l_s), as follows (2),

$$p = \begin{cases} 0, & if\left(l_s \le T_1\right) \\ f\left(l_s\right), & if\left(\left(NewSession\right) and \left(T_1 < l_s \le T_2\right)\right) \\ 1, & if\left(\left(NewSession\right) and \left(T_2 < l_s \le C\right)\right) \\ 1, & if\left(l_s > C\right) \end{cases}, \quad (2)$$

In the following subsection, we describe the conducted simulations to evaluate the performance of the advocated session-aware admission-control strategy.

V. Performance Evaluation

To evaluate the proposed admission-control strategies, we have implemented a generic simulator which consists of a processing server with a limited buffer-sized listen queue, a set of generators of clients, sessions, requests and feedbacks, as well as a set of estimators useful to maintain real-time information about the server's load as well as the rates it achieved at session- and request-level. We aim at showing that responsive session-aware admission-control improves the operator's benefit as well as the QoS provided to subscribers by improving its session-level rate. Session-level rate will be defined below as the number of successfully completed sessions per unit of time.

A. Simulation model overview

We use a discrete event-based simulation model to describe the interaction between the requests' arrivals to the system and their departures from it. In our model, we define a session as a sequence of individual requests that are generated for the duration of the session. A new session is produced by a session generator according to specified input parameters such as the client's identifier, the time at which the session was initiated, its original and current durations, its source throughput, and more. A session is normally completed when all related requests are completed successfully. For a given simulation run, the client generator generates *N* sources of traffic each corresponding to one client. Every client is able to generate sessions at a source rate δ. The request generator issues session's requests according to specific session's parameters such as the inter-request think-time and the number of retries in case of a request loss.

The processing server is characterised by its capacity and service-processing time. The service-processing time is assumed to be proportional to a request's size.

B. Performance Metrics

We aim to study the impact of different admission-control policies, session aware and session oblivious, on the achieved operator's profitability. Profitability has been typically estimated by measuring the servers' throughput. Actually, a server throughput is defined as the number of requests or packets that are processed per unit of time. Usually, a server running close to its edge capacity is considered satisfactory.

In this work, we define the "useful throughput" or the "success rate" as the number of sessions completed successfully per unit of time. We believe that this metric provides truthful operator's benefit estimation since a server that sustains a high request level rate may be actually wasting its resources on failed or on interrupted sessions.

In our simulations, we considered a realistic CBR voice traffic. Table 1 below summarizes the parameters of the model used to generate the offered sessions [see Table 1].

Simulation results presented below average the output of ten simulation runs, made up of new arrivals of client sessions.

C. Simulation Scenarios and Methodology

TABLE I
Traffic profile model for voice over IP conversations

Model	Distribution
Packet Inter Arrival Times	Constant
Session Inter Arrival Times	Exponential
Session Duration	Exponential
Packet Size	Constant

In this subsection, we detail our study of the effect of the different advocated measurement-based admission-control strategies on the server's achieved useful throughput. We simulate the behaviour of a processing server over a 100 seconds-long simulation time. A client initiates a conversation with an average talking time that obeys an exponential distribution. Session duration is varied over the simulation run, ranging from medium- to long-term sessions that have exponentially distributed durations. The traffic rate is set to 50 packets per unit of time.

Firstly, we consider a server deploying Request-Aware admission-control (RAC), where dropping requests pertaining to the same session occurs at any time during the subscriber's session lifespan. The request-aware strategy that we consider responsively drops the offered packets on the basis of a probabilistic model, as follows *(3)*,

$$p = \begin{cases} 0, & if\left(load \le T_1\right) \\ \left\| \dfrac{load - T_1}{capacity - T_1} \right\|, & if\left(T_{1<load\le C}\right) \\ 1, & otherwise. \end{cases}, \quad (3)$$

Secondly, we consider a server that deploys Session-Aware admission-control (SAC). When the server runs under a heavy load, the offered packets pertaining to the already established sessions are given a higher priority compared to those holding requests for establishing new sessions.

In the first variant of this model, which is depicted as the ON/OFF Session-Aware admission-control (ON/OFF SAC), the dropping probability *(p)* of the offered new sessions is derived as follows *(4)*,

$$p = \begin{cases} 0, & if\left(load \le T_1\right) \\ 1, & if\left(\left(NewSession\right) and \left(load > T_1\right)\right) \end{cases}, \quad (4)$$

The second variant of this strategy advocates a responsive Session-Aware admission-Control model (RESP SAC), where new sessions are dropped with a probability *(p)* computed as a function of the server's current load as follows *(5)*,

$$p = \begin{cases} 0, & if\left(load \le T_1\right) \\ \left\| \dfrac{load - T_1}{capacity - T_1} \right\|, & if\left(\left(NewSession\right) and \left(load > T_1\right)\right) \end{cases}, \quad (5)$$

In the following subsection, we will be studying the effect of session awareness on the achieved average useful throughput for different session durations. Next, we will turn our attention to studying the effectiveness of the model described by equation (2) in fairly competing sessions independently of their duration. For this purpose, we will consider two simulation scenarios:

1) Generating homogeneous sessions over the same simulation run: duration of a session ranges from medium- to long-term.

2) Generating mixed sessions over the same simulation run: In order to focus on the fairness property, we have chosen that medium- and long-term conversations be equitably generated over the simulation time.

D. Simulation Results and Discussion

Fig. 3 illustrates the effect of session unawareness on the achieved average throughput compared to the achieved average useful throughput for the simulation run. The generated sessions are medium-term. Session-unaware admission-control is deployed either by involving no admission-control (No AC) or by deploying request-aware admission-control (RAC).

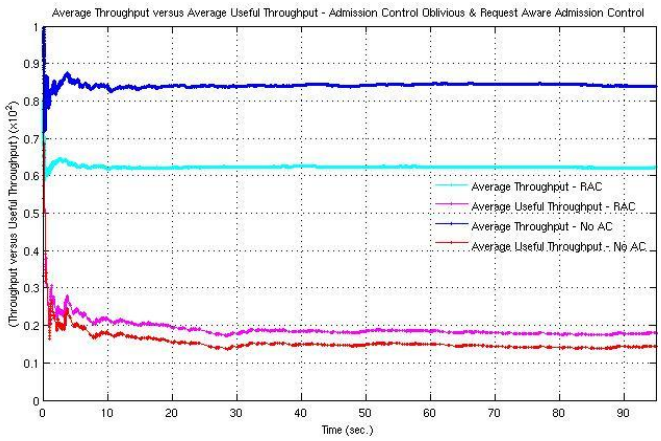

Fig. 3. Effect of Session-Unaware Admission-control on the Average Useful Throughput - Medium-Term Sessions & Single-Threshold-based RAC (75 % of the Server's Capacity).

As we could have expected, when the server is admission-control-oblivious or when it deploys request-aware admission-control, the achieved average useful throughput has lower values compared to the achieved average throughput. Particularly, when the server is oblivious to admission-control (No AC), while it seems to fully satisfy the clients' demands, only nearly 14 % of the offered sessions are successfully completed compared to the completion of nearly 17 % of the offered sessions on average, when responsive request-aware admission-control (RAC) is deployed on the server. Indeed, dropping the offered packets to the server at any time during the session's lifespan does not efficiently prevent session interruptions. This observation confirms our economically-justified point of view, advocating that request-aware admission-control does not deterministically contribute to improving the operator's profitability and the QoS perceived by clients.

In contrast, fig. 4 illustrates the average achieved useful throughput, when session-aware admission-control strategies are deployed on the server, compared to when responsive request-aware admission-control is used by the server.

Firstly, we can see that when responsive session-aware admission-control is used, nearly 76 % of the offered sessions are completed successfully, for a threshold value of 75 % of the server's capacity. The choice of the threshold value is justified by further simulation results where we observed that, under the above assumptions, the best useful throughput is provided when using a threshold of 75% of the server's capacity.

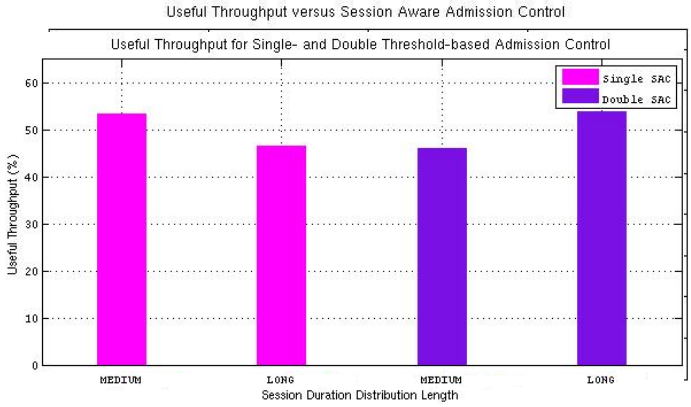

Fig. 8. Percentage Histogram of Completed Calls for Different Admission-Control Strategies - Mixed Generated Traffic.

Fig. 5 below describes the observed average useful throughput versus the threshold, assuming medium-term sessions and a responsive session-aware admission-control.

On the other hand, we can see that responsive session-aware admission-control outperforms the basic

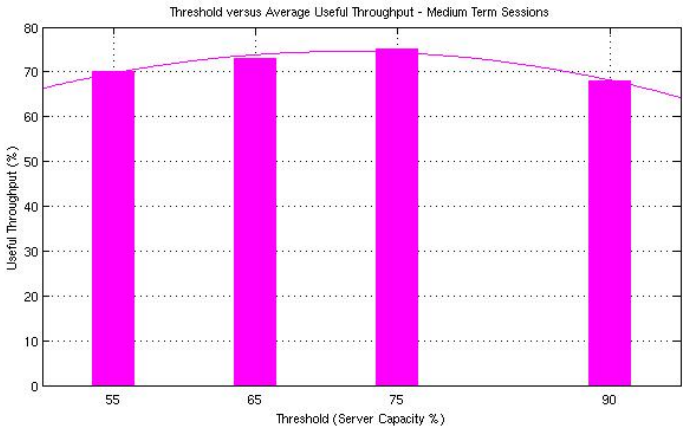

Fig. 5. Average Useful Throughput versus Threshold - Medium-Term Sessions & Responsive Session-Aware Admission-control.

session-aware admission-control model (ON/OFF SAC) by completing, on average, 8.83 % more offered calls.

It is also worth pointing out that the gain achieved when using responsive session-aware admission-control (RESP SAC), compared to when using responsive request-aware admission-control (RAC), is the completion of nearly 69 % more offered sessions, on average, under the same previously considered assumptions.

Next, we focus on comparing the effectiveness of single- and double-threshold based session-aware admission-control strategies. Fig. 6 focuses on revealing the gain achieved by a session-aware admission-control-enabled server.

It compares the achieved average useful throughput when the server uses either single-threshold-based responsive session-aware admission-control (Single SAC) or double-threshold-based session-aware admission-control (double SAC). The durations of the generated traffic are homogeneous over simulation runs, meaning that all generated sessions are either medium- or long-term over a given simulation run. The threshold value of the used Single SAC policy is set to 75 % of the server's capacity. The used double SAC involves a first threshold value set to 75 % of the server's capacity and a second threshold value set to 90 % of the server's capacity.

As we can see, when the offered sessions are medium-term, double-threshold-based admission-control performs slightly

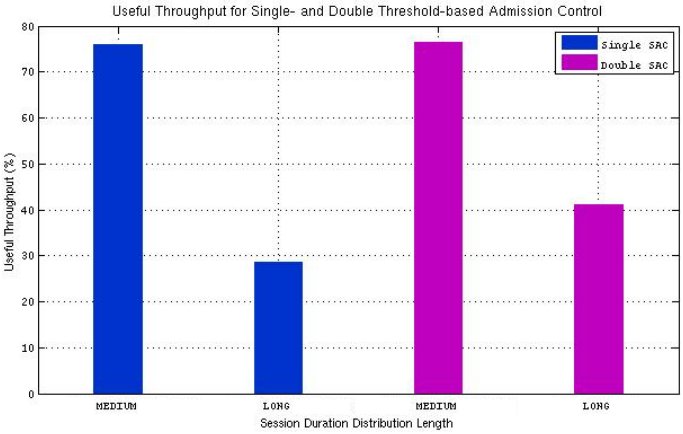

Fig. 6. Percentage Histogram of Completed Calls for Different-Admission-control Strategies – Homogeneous Generated Traffic.

better than single-threshold-based admission-control. Indeed, only 0.52 % of performance gain is obtained, on average. However, when the offered sessions are long-term, double-threshold-based session-aware admission-control offers a performance gain of nearly 44 % more completed sessions, compared to when single-threshold-based policy is used. This result confirms with intuition, since this model is actually designed to suit long-term sessions.

Let us now examine the average useful throughput when single- and double-threshold-based session-aware admission-control strategies are performed on mixed traffic. Medium-term and long-term sessions are equitably generated over the simulation runs. Fig. 7 plots the percentage histogram of the achieved average of completed calls in that case.

We can globally see that double-threshold-based session-aware admission-control (Double SAC) contributes

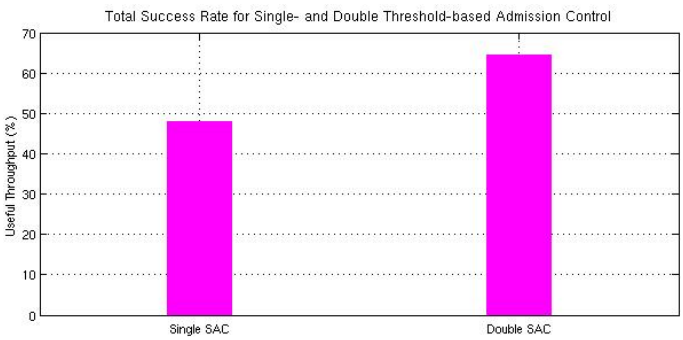

Fig. 7. Percentage Histogram of Completed Calls for Different Admission-Control Strategies - Mixed Traffic.

to improving the average success rate by completing nearly 25 % more offered calls, compared to when single-threshold-based admission-control (Single SAC) is used.

Fig. 8 highlights the contribution of long-term sessions' success rate in this achieved total average useful throughput. It plots the percentage histogram of the average completed calls as a function of session duration for both single- and double-threshold-based session-aware admission-control.

As we can see, double-threshold-based session-aware admission-control privileges the fair completion of the offered sessions independently of their duration. Indeed, the second threshold aims at a kind of protection of long-term sessions under overload. We observe that more than 13.69 % more long-term calls, established through the server, have satisfactory quality. In contrast, this percentage drops to 2 % completed calls, when responsive request-aware admission-control is used. As well, since the double-threshold-based admission-control is giving a higher priority to long-term calls, it completes less medium-term calls.

In realistic use-cases, calls range from short to long term. If we define QoS as the guarantee of completing more sessions independently of their duration, hence, based on the above simulation results, we can say that the session-aware admission-control that we have advocated contributes to having better QoS provided to subscribers.

VI. Discussion

In the previous section, we investigated the performance of session-aware admission-control strategies to efficiently prevent Internet servers' overload, while improving the QoS perceived by subscribers and maximizing the provider's profitability.

We compared the average achieved useful throughput for the different considered scenarios, which involve the generation of either mixed- or homogeneous-duration sessions. Fig. 9 summarizes the main obtained results. It shows that session-aware admission-control significantly surpasses standard strategies. For a satisfactory operator's revenue, session-aware admission-control improves the success rate by more than 69 % compared to standard request-aware admission-control when the server deals with medium-term offered sessions. On the other hand, we have shown that using responsive double-threshold-based admission-control is beneficial to increase the performance of servers processing long-term sessions. A performance gain of nearly 44 % more successfully completed long term calls is noticed when double-threshold-based session-aware admission-control is used, compared to when using single-threshold-based session-aware admission-control. Finally, we showed that in more realistic scenarios, where calls' durations range from short- to

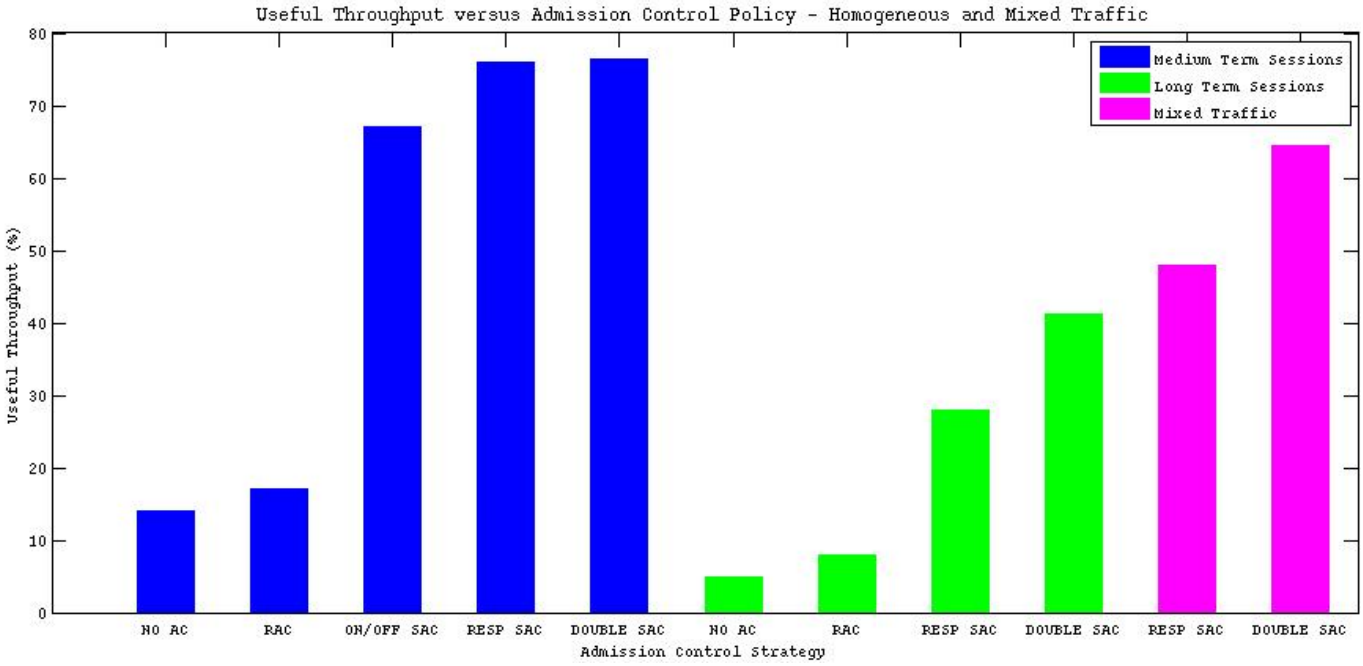

Fig. 9. Comparison Between Different Admission-control Strategies - Homogeneous and Mixed Traffic Profiles.

long-term, double-threshold-based session-aware admission-control provides improved QoS to subscribers by contributing to the completion of more sessions, independently of their duration.

VII. Conclusion

A. Lessons learned

Service-aware network management is a key issue for the current and future next generation service-oriented networks. Providing session-aware "intelligence" at the access, the core and the edge of the current Internet are hot topics for network operators and service providers. In this work, we investigate the contribution of session awareness to improving Internet servers' QoS.

Firstly, we questioned the appropriateness of state-of-the-art admission-control mechanisms for preventing Internet servers' resources from overloading. We showed that most state-of-the-art approaches, originally intended to ensure an acceptable QoS for the admitted traffic, operate either at the packet level or at the flow level. Applied to subscribers' sessions, they may drop requests pertaining to an accepted session at any time during the session lifespan, resulting in reduced QoS. Hence, these mechanisms poorly adapt to Internet servers subjected to multiple-flow-based sessions.

Based on an economically-justified service model, we derived a novel session-aware admission-control model for Internet servers subjected to session-oriented services. We described the building blocks of a session aware admission-control framework that aims for improving the operator's profitability, by maximizing the Internet server useful throughput, which is defined as the number of sessions completed successfully per unit of time.

The proposed architecture is based on original concepts which meet the constraints of multiple-flow based sessions by explicitly identifying the flows pertaining to a single subscriber session.

Secondly, we evaluated the performance of the advocated session-aware admission control model for improving service provider's profitability. We conducted advanced simulations considering realistic CBR voice traffic. We showed that session-aware admission-control improves the providers' benefit as well as the perceived QoS by subscribers, compared to session-oblivious strategies. We also showed that responsive double-threshold-based session-aware admission-control adapts to more realistic use-cases by guaranteeing the completion of sessions independently of their duration.

In this work, we defined QoS as the completion of the accepted sessions, independently of their duration. We demonstrated that session awareness is a very desirable property for edge network equipments. Indeed, sessions related to sale activities typically yield long conversations. Conversely, short sessions established with Internet servers have higher chance to complete normally, even when the Internet server is admission-control oblivious.

We believe that providing session-aware intelligence to edge network traffic processing equipments will play a strategic role in encouraging customers to migrate core business functions to the emerging technologies. Particularly, in an IMS context, session awareness will condition the network operator's profitability from an economic-justified point of view, as well as the service success regarding the competition.

B. Perspectives

Many significant perspectives are possible to this work. Firstly, we are interested in extending the advocated session-aware model to handle more QoS metrics, such as the client's category and other composite QoS metrics. Secondly, we need more proofs on the robustness of the advocated session-aware admission-control model on the QoS of highly variable Internet traffic. We believe that enhancing the advocated model with advanced forecasting-based techniques is an interesting perspective for improving the stability pattern of admission-control decisions, under overload.

ACKNOWLEDGMENT

We would like to address our sincere thanks to Mr. Frederic Dang Tran, the leader of the Magneto project at Orange Labs, for his valuable support to this research. Our sincere thanks go to Mr. Maillé, assistant professor from the ENST Bretagne for discussion and help on analytical work inherent to this work.

REFERENCES

[1] S. Floyd and V. Jacobson, "Random Early Detection for congestion avoidance", *Proceedings of the IEEE/ACM Transactions on Networking*, Aug. 1993, pp. 60-64.

[2] E. Hashem, "Analysis of random drop for gateway congestion control", *Technical report MIT-LCS-TR-467*, Laboratory of Computer Science, 1989, pp. 60.

[3] X. Chen and P. Mohapatra, "Providing differentiated services from an Internet server", *Proceedings of the 8th IEEE Transactions on Computer Communication and Networks*, 1999, pp. 214-216.

[4] T. Abdelzaher, K. Shin and N. Bhatti, "Performance guarantee for Web server end-systems: a control theoretical approach", *Proceedings of the IEEE Transactions on Parallel and Distributed Systems*, 2002, pp. 60-131.

[5] S. Lee, J. Lui, Y. Yau, "Admission-control and dynamic adaptation for a proportional delay diffserv-enabled web server", *Proceedings of the IEEE SIGMETRICS'02*, 2002, pp. 60.

[6] L. Cherkasova and P. Phaal, "A mechanism for peak load management of commercial web sites", *Proceedings of the IEEE Transactions on Computers*, 2002, pp.61.

[7] N. Ayari, D. Barbaron, L. Lefèvre and P. Primet, "Session awareness issues of the next generation cluster based network load balancing frameworks", *Proceedings of the* ACS/IEEE International Conference on Computer Systems and Applications, AICCSA07, May 2007.

[8] Z. Cao, Z. Wang, E. Zegura, "Performance of hashing based schemes for Internet load balancing", *Proceedings IEEE INFOCOM'00*, 2000, vol. 1, pp. 332-341.

[9] K.C. Claffy, H.-W. Braun and G.C. Polyzos, "A parameterizable methodology for Internet traffic flow profiling", *Proceeding of the IEEE Journal on Selected Areas in Communications*, vol. 13, Oct. 1995 pp. 1481–1494.

[10] Partridge, "Reliable UDP, RFC 1151", *BBN Systems and Technologies*, April 1990.

Narjess AYARI (narjess.ayari@orange-ftgroup.com) received her Engineering Diploma in Computer Science from the National School of Computer Science of Tunis (ENSI), in 2001 and her M. Sc. in Networking and Distributed Systems from the same School, in 2003. From 2005 to 2008, she joined France Telecom R&D Labs (Orange Labs) as a research engineer, where she conducted her PhD with a collaboration with the RESO team at the LIP laboratory (Laboratoire de l'Informatique du Parallélisme), in the Ecole Normale Supérieure of Lyon (ENS Lyon). She obtained her PhD. in Computer Science in 2008. She is the author and co-author of many research reports, IEEE papers and patents. Her current research interests include (and are not limited to) IMS and QoS in Broadband 3G/4G networks. Her web site is: http://perso.ens-lyon.fr/narjess.ayari.

Denis BARBARON (denis.barbaron@orange-ftgroup.com) graduated in Computer Science from the University of Science of Nantes, France, in 1994. He joined Orange Labs (formerly France Telecom R&D Labs) in 1997 as an R&D engineer. Since 2005, he is Senior Expert in cluster architectures at Orange Labs in Lannion. His current research interests include IMS and carrier-grade telecom service platforms.

Laurent LEFEVRE (laurent.lefevre@inria.fr). Laurent Lefèvre obtained his Ph.D. in Computer Science in January 1997 from the Ecole Normale Supérieure of Lyon, France. From 1997 to 2001, he was assistant professor in computer science in Lyon 1 University. Since 2001, he is a permanent researcher in computer science at INRIA (the French Institute for Research in Computer Science and Control). He is a member of the RESO team (High Performance Networks, Protocols and Services) from the LIP laboratory in Lyon, France. He has organized several conferences in high performance networking and computing and he is a member of several program committees. He has co-authored more than 50 papers published in refereed journals and conference proceedings. He is a member of IEEE and takes part in several research projects. His research interests include: grid and distributed computing and networking, power aware and energy efficient large scale distributed systems, autonomic networking, fault tolerance, high performance networks protocols and services, active networks and services, disruption tolerant networking, cluster computing, distributed shared memory systems and data consistency. His web site is: http://perso.ens-lyon.fr/laurent.lefevre