End-to-end Session Initiation Protocol (SIP) overload control
draft-wang-appsawg-end2end-overload-control-00

Abstract

This draft proposes end-to-end Session Initiation Protocol (SIP) overload control, in which the edge servers of the SIP network throttle the arriving calls in order to control overload for the SIP network. Compared to the local and hop-by-hop SIP overload control, the end-to-end SIP overload control can achieve best performance.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at http://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

Copyright Notice

Copyright (c) 2013 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust’s Legal Provisions Relating to IETF Documents (http://trustee.ietf.org/license-info) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.
1. Introduction

Session Initiation Protocol (SIP) serves as a foundation for many of today’s session-oriented applications, such as Voice over IP (VoIP), multimedia distributions, video conferencing, instant messaging and presence service. The widespread popularity and rapidly growing deployments of SIP require that SIP servers provide adequate control mechanisms to handle overload. Overload of a SIP server occurs if the message arrival rate to the server exceeds its message processing capacity. Under overload, the throughput of a SIP server can drop significantly and can even reach zero. Besides, the call setup delay becomes unacceptable for a real-time media call. In this case, the server enters into a congestion collapse.

[RFC6357] has classified the SIP overload control approaches into local, hop-by-hop and end-to-end overload control. In local overload control, the SIP server monitors its load and starts to reject requests locally by using 503 (Service Unavailable) responses when it detects overload. In hop-by-hop overload control, the overloaded SIP server can provide feedback to its direct upstream neighbors, which then adjust the amount of traffic forwarded to this SIP server to eliminate overload. In end-to-end overload control, the edge servers, which are considered as the closest servers to the sources of traffic in a SIP network, are responsible for adjusting the amount of traffic forwarded to the overloaded server to eliminate overload.

In the deployment scenarios (such as IMS) where the SIP call traverses through multiple SIP servers in a SIP network, local and hop-by-hop overload control are inefficient since overload is resolved near the overloaded sever. In this case, the SIP servers located between the edge server and the overloaded server waste their processing resources on processing a request that will finally be
rejected. On the other hand, in end-to-end overload control minimum resources of SIP networks are wasted on processing a request that will finally be rejected since the edge servers are responsible for rejecting requests.

The research in [Hilt] indicates that the end-to-end overload control achieves the best performance (highest throughput) although it is the most complex among all types of overload control approaches. Based on them, this document proposes an end-to-end overload control mechanism for networks of SIP servers.

2. End-to-end overload control scheme

2.1. Overview

The SIP network consists of edge servers and core servers. Each UA is connected to the network via an edge server located closest to it. When a SIP call between two UAs goes through the network, the first server the call arrives at is denoted as the ingress server, and the last server the call arrives at is denoted as the target server. It is clear that both ingress server and target server are edge servers.

The design of end-to-end overload control should follow the principles as below:

- Overload MUST be controlled at ingress servers. That is, arriving calls from UAs are throttled at ingress servers. Overload control works best if applied at the servers closest to the source of traffic because in this way minimum resources of SIP networks are wasted on processing a request that will finally be rejected.
- Overload SHOULD be controlled on a per-target basis. That is, each ingress server throttles the arriving call from UAs based on its target server. Without the per-target basis, an ingress server should identify which server is overloaded and throttle arriving calls that will be routed through the overloaded server. On the other hand, with the per-target basis, an ingress server only needs to identify which target server the call passing through the overloaded server is related to, and throttle arriving calls that will be forwarded to this target server. Thus per-target basis makes it much easier for ingress servers to control overload for the SIP network.

2.2. End-to-end overload control design

In end-to-end overload control, the core servers SHOULD only implement local overload control that rejects requests by using 503 responses. When receiving a 503 response from a downstream neighbor, the server SHOULD forward this response to the upstream neighbor,
from which the INVITE request related to this response has been received. In this way, the 503 response will finally be forwarded to the edge server. The edge servers SHOULD calculate and follow the restrictions on the traffic admitted to the SIP network based on the received 503 responses.

In end-to-end overload control, a set of control-units is deployed at each ingress server to control overload for the SIP network. At an ingress server, each control-unit is related to a specific target server and controls the arriving calls from UAs that take this ingress server as first-hop and the target server as last-hop in the network. Thus, the load of the network is controlled by control-units at all ingress servers. Note that this approach is completely distributed: there is no centralized entity to control control-units and each control-unit is functionally identical and operates independently. Besides, there is no communication between control-units. Finally, this approach can be deployed incrementally, by installing control-units on ingress servers, with no need to alter other servers in the SIP network. Therefore, this approach is easy to implement.

2.3. End-to-end overload control algorithm

The main function of the control-unit is to decide the call admission rate, which is calculated by using the end-to-end overload control algorithm.

2.3.1. End-to-end overload control algorithm metrics

Aggressiveness: The network is underutilized when the call admission rate is below the capacity of the network. In this case, control-unit needs to increase the call admission rate as fast as possible in order to make full use of network resources and avoid unnecessary call rejections. Aggressiveness measures how fast a control-unit makes use of network resources as they are available. The aggressiveness is defined as the inverse of the time needed for the control-unit to achieve the increment of a certain amount of call admission rate, in response to: (1) a step increase of available network resources or (2) a step increase of call arrival rate when there are available resources in the network. Obviously, high aggressiveness, implying potentially high utilization, is desirable.

Responsiveness: The network is overloaded when the call admission rate exceeds the capacity of the network. In this case, control-unit needs to decrease the call admission rate as fast as possible in order to eliminate overload. Responsiveness measures how fast a control-unit decreases the call admission rate in response to overload. We define responsiveness as the inverse of the time needed
for the control-unit to achieve the decrement of a certain amount of call admission rate, in response to a step increase of network overload. Obviously, high responsiveness, which allows control-unit to decrease the call admission rate quickly when overload occurs, is desirable.

Throughput: The network is fully utilized when the call admission rate is close to the capacity of the network. In this case, the throughput is determined by the overload control algorithm. Obviously, high throughput is desirable.

2.3.2. Default End-to-end overload control algorithm

The default end-to-end overload control algorithm presented here is only an example. Other algorithms that can achieve high aggressiveness, high responsiveness and high throughput may be used.

The default end-to-end overload control algorithm consists of an increasing rule and a decreasing rule. When there is no overload feedback, the algorithm increases call admission rate according to the increasing rule. When receiving the overload feedback, the algorithm decreases call admission rate according to the decreasing rule. The control-unit periodically executes the overload control algorithm (with interval T) and takes the number of received 503 responses during each T as the overload feedback to the algorithm.

The increasing rule and decreasing rule are shown as follows:

- increasing: \( r(t+1) = r(t) + a \), \( a > 0 \)
- decreasing: \( r(t+1) = r(t) - b * r(t) \), \( 1 > b > 0 \)

where \( r(t) \) is the call admission rate at time \( t \). \( a \) and \( b \) are constant factors. That is, if no call rejection is received, the call admission rate is increased additively. Otherwise, it is decreased multiplicatively.

3. Security Considerations

TBA

4. IANA Considerations

None.

5. References

5.1. Normative References
5.2. Informative References


Authors’ Addresses

Jin Zhu Wang
China Mobile
Email: wangjinzhuzinamobile.com

Qing Yu
China Mobile
Email: yuqing@chinamobile.com

Lingli Deng
China Mobile
Email: denglingli@chinamobile.com