August 27, 2018

USING BLOCKCHAIN TO SIMPLIFY SESSION INITIATION PROTOCOL OVERLOAD CONTROL

Kaustubh Inamdar

Ram Mohan R

Gonzalo Salgueiro

USING BLOCKCHAIN TO SIMPLIFY SESSION INITIATION PROTOCOL OVERLOAD CONTROL

AUTHORS:
Kaustubh Inamdar
Ram Mohan R
Gonzalo Salgueiro

ABSTRACT

Techniques are described herein by which the Session Initiation Protocol (SIP) server overload problem may be significantly simplified by using a distributed system where transactions can be authorized and stored. For example, a technology like blockchain may enable a centralized, shared, and secure transaction database to be used to communicate SIP server overload information. SIP server overload information may be shared between servers that are part of a trust domain. The trust domain may be confined within a network or span across network boundaries (e.g., between enterprise edges / SIP trunk providers / SIP calling cloud providers).

DETAILED DESCRIPTION

Blockchain has recently risen to prominence across the technology industry. Originally designed and used to support bitcoin, blockchain has started seeing applicability in diverse verticals such as Internet of Things (IoT) and telecommunications. For example, a blockchain based approach can be used to enable E.164 to Uniform Resource Identifier (URI) translation in a fast, secure, and reliable way. In general, a blockchain based approach is extremely useful in scenarios where the same data needs to be shared among several entities in a secure, reliable, and scalable capacity. Described herein are techniques for using blockchain technology to provide a more flexible alternative to handling Session Initiation Protocol (SIP) overload control.

SIP server overload is a condition wherein a SIP server is unable to process incoming requests due to resource exhaustion. Resources can include memory, Central Processing Unit (CPU), input/output disk, etc. Over the years, there have been many approaches designed to solve the SIP server overload problem. These approaches include using the 503 Service Unavailable with a Retry-After header and using the constructs of

the SIP overload control Internet Engineering Task Force (IETF) Request for Comments (RFCs).

Using a 503 Service Unavailable is of limited use and can contribute to congestion collapse as the SIP server would spend its CPU cycles sending 503 responses when it is already overloaded. The perils of using a SIP 503 response have already been well documented in numerous research publications.

The constructs of IETF RFCs, though useful, are quite complex. At a minimum, several items are required to ensure overload control is handled effectively using IETF RFCs. These items include logical components such as the processor, monitor, control function, and actuator; an explicit SUBSCRIBE/NOTIFY framework that communicates overload information over SIP or the modification of the Via: header field of SIP request and responses; and a certain degree of co-operation between servers along the call path (for end-to-end overload control).

The overload control framework may be significantly simplified by using a distributed system where transactions can be authorized and stored. For example, a technology like blockchain, wherein a centralized, shared and secure transaction database may be used to communicate SIP server overload information, may be useful. SIP server overload information is shared between servers that are part of a trust domain. Any party that is part of the blockchain network may be part of the trust domain. All interested parties (e.g., enterprise edge / SIP trunk provider, SIP cloud calling provider, SIP trunk aggregators, public switched telephone network backend providers, etc.) may be part of the blockchain network and may form the trust domain. The trust domain may be confined within a network or span across network boundaries (e.g., between enterprise edge / SIP trunk provider / SIP calling cloud provider).

Using blockchain technology may be preferable to using a secure, distributed database. First, with secure, distributed databases, to ensure that data is consistent across all nodes, it is required to either duplicate or replicate data. Data duplication across several SIP servers could be a long-drawn process and could lead to a condition wherein the nodes are informed of an overloaded condition much later than its onset. Data replication involves choosing one SIP server as a “master” and pushing any updates from the master’s database to the other SIP servers. This is not an ideal situation as this architecture introduces a single

point of failure. Even in cases where any node is allowed sufficient privileges to make changes and those changes are replicated to other participating nodes, the issue of "trust," especially across nodes that belong to different administrative domains, is not easily solvable using traditional databases.

Moreover, with databases, any malicious actor with "administrator" privileges could potentially alter information and present an incorrect picture of server overload to the entire community of participating servers. For example, a malicious actor could alter the state report of a legitimately overloaded server, thus falsely allowing downstream servers to send the same traffic loads to the affected server. In addition, given the large number of SIP servers that might want to federate and communicate overload information and the frequency with which servers toggle between normal and overloaded conditions, it might be a prohibitively expensive task to use databases.

Accordingly, described herein is the applicability and advantages of blockchain when used to solve the SIP server overload problem. A blockchain based approach to solving the SIP server overload problem may be advantageous for several reasons. First, it completely avoids the need to define and implement extensions to a baseline SIP. The constructs of the IETF RFCs workgroup require extensions to baseline SIP. Second, it provides each federated SIP server visibility about the state of any other server. With traditional mechanisms, a server's visibility is restricted to its immediate neighbor and such a restricted span of visibility is insufficient in addressing overload control in real-world SIP networks. Third, it avoids some of the problems commonly seen with information sharing via databases, specifically issues related to trustworthiness of server state information, ease of disseminating information across federating servers, and scalability of operations.

Figure 1 below illustrates a topology of a sample SIP network.

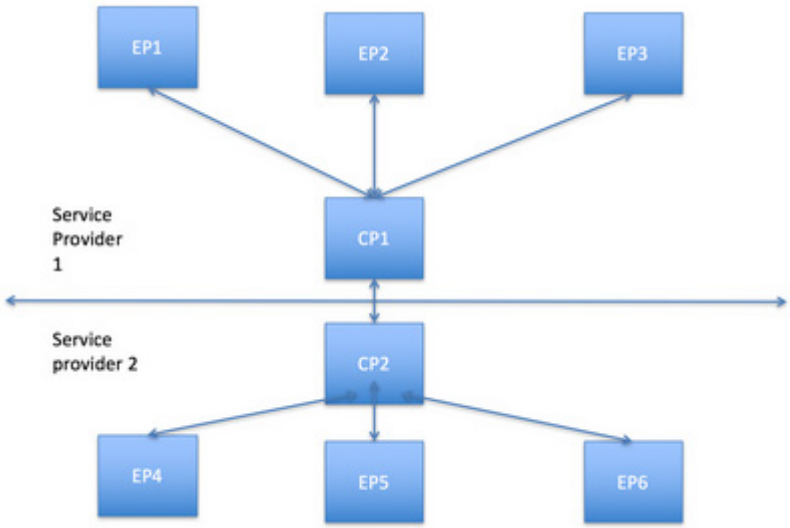

Figure 1

Service provider network 1 has three edge proxies (EP1, EP2, EP3) and one core proxy server (CPI). The core proxy server in service provider network 1 peers with the core proxy server in service provider network 2 (CP2). Service provider network 2 has three edge proxy servers (EP4, EP5 and EP6).

Each proxy server joins as a member of a blockchain network. Entities (the proxy servers) within the trust domain may agree to exchange the following data among one another: SIP server overload information (or lack thereof); the format in which this information is exchanged; and/or the rules using which information is interpreted.

Each proxy server functions as a blockchain node and generates transaction data. This transaction data is encrypted and may only be read by other blockchain nodes within the trust domain. Transaction data may include the following: the server state of each proxy server and their ability (or lack thereof) to process new requests; the number of requests that can be processed within a second or a given time frame by a node; the algorithm to be used while implementing load control or rate control of SIP messages; the domain(s) for which overload control has to be applied; and/or any alternate destinations to which requests may be routed.

Using the blockchain approach, a particular overload control scenario is described in accordance with Figure 1 above. EP1 may be assumed to service all numbers to a disaster relief service and numbers for several other services. During a natural disaster, the call volume to the disaster relief service is expected to increase dramatically. In anticipation of

increased volume, EP1 may update its transaction data to place emphasis on calls coming into the disaster relief hotlines as opposed to those that are for other services. Additionally, to ensure call coverage, it may redirect requests beyond a certain threshold to EP2 or EP3.

Figure 2 below illustrates example transaction data written by EP1.

Timestamp	18:01:53 November 10, 2000
Numbers of Priority	1-408-434-3470 To 1-408-434-3480
Time Validity	7 AM, 11th November 2000 to 7 PM, 12th November 2000.
SIP Server State	CPU: 20% Memory: 10%
Total Requests Per Second	1000
Total Load Per Minute	20,000 SIP Transactions
Request Redirect Destination	EP2, EP3
Load Control Algorithm	ABC
Rate Control Algorithm	XYZ
% Of Calls To Other Numbers	4%
Overload Action	Reject/Drop.

Figure 2

Once added to the blockchain, CP1 reads this transaction data and handles requests northbound to EP1 in accordance with the finer data points. For example, from 7AM on the 11th of November to 7 PM on the 12th of November, numbers that service the disaster relief hotline are given highest preference. Requests targeting other numbers serviced by EP1 are sent only 4 out of a 100 times. The total number of requests that can be sent to EP1 per second is limited to 1,000. The total number of active SIP transactions per minute on EP1 cannot exceed 20,000.

If at any point in time CP1 needs to send more than 1,000 requests per second, it may begin redirecting a percentage of requests to EP2 and EP3. Each transaction has a timestamp. While implementing rate or load control, peer SIP servers within the trust domain must act on transaction data with the latest timestamp.

Consider a second scenario where CP1 is approaching an overloaded state. Here, it updates its transaction data to provide the latest snapshot of the SIP server state and

metadata, as illustrated in Figure 3 below. CP2 and other edge servers may read this transaction data to understand how traffic to CP1 can be engineered. For example, EP4, EP5, and EP6 may fail all requests destined for CP1, thus ensuring that excessive traffic does not move into the core SIP network. The edge proxies may also propagate this information to downstream SIP servers and proxies (the edge proxies may be a part of yet another blockchain network with the downstream servers and proxies), so that requests are failed as close to the source as possible, thereby ensuring that traffic loads do not reach the edge and core SIP networks.

Timestamp	16:01:53 November 10, 2000
Numbers of Priority	None
Time Validity	N/A
SIP Server State	CPU: 70% Memory: 60%
Total Requests Per Second	10000
Total Load Per Minute	25,000 SIP Transactions
Request Redirect Destination	None
Load Control Algorithm	ABC
Rate Control Algorithm	XYZ
Overload Action	Reject/Drop

Figure 3

In summary, techniques are described herein by which the SIP server overload problem may be significantly simplified by using a distributed system where transactions can be authorized and stored. For example, a technology like blockchain may enable a centralized, shared, and secure transaction database to be used to communicate SIP server overload information. SIP server overload information may be shared between servers that are part of a trust domain. The trust domain may be confined within a network or span across network boundaries (e.g., between enterprise edges / SIP trunk providers / SIP calling cloud providers).