POC HTTP band limitation

A high level description

|  |  |  |  |
| --- | --- | --- | --- |
| Date | Version | Author | Comments |
| March, 21th 2018 | 1.0 | Zardoni, Matteo[[1]](#footnote-1) | First draft |
| September, 11st 2018 | 1.1 | Zardoni, Matteo | Fixed unit measure |

Since *nginx* permits only to limit the http band per request or per number of requests/ip, the aim of this POC is to enable nginx to shape the band by considering two parameters, which are the ip address and the (k)B/s value. Our solution relies on a personalization of a nginx module[[2]](#footnote-2)1, which permits to limit the rate of download requests.

In order to configure the server to throttle the band by considering the two parameters cited previously, the behaviour shown in the Fig. 1 has been implemented.

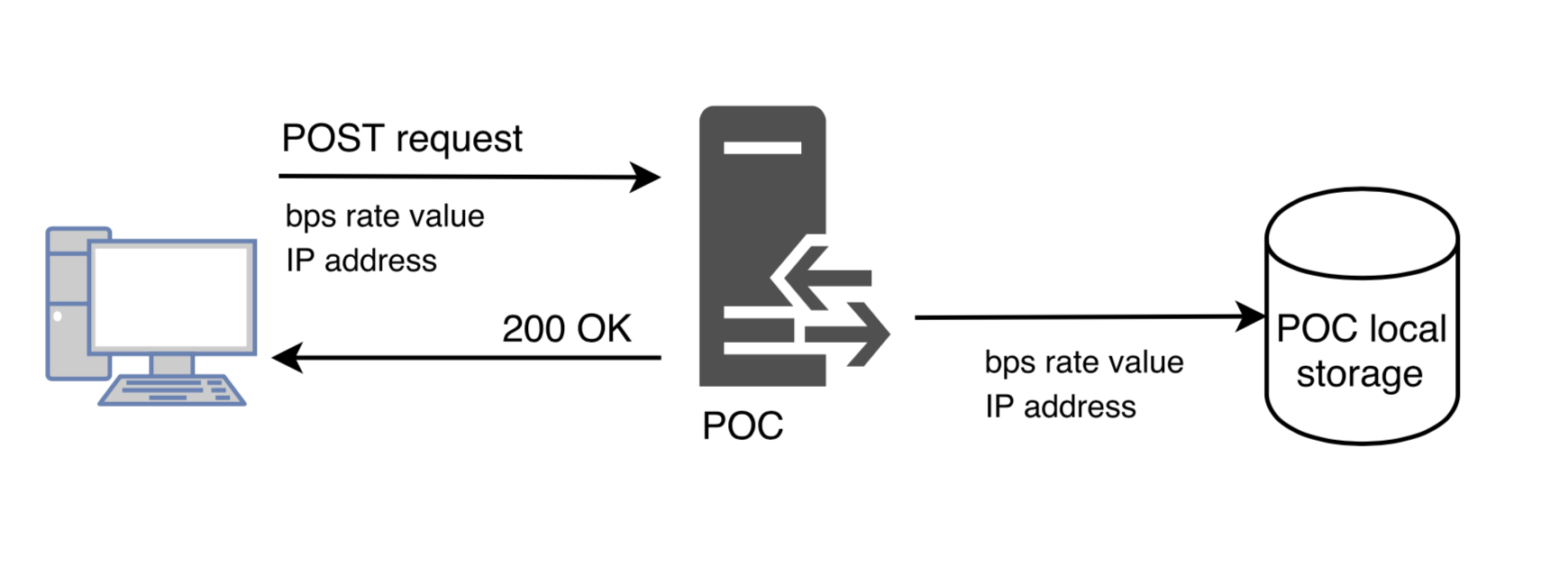


Fig. 1 – “bps” is treated as (k)B/s value

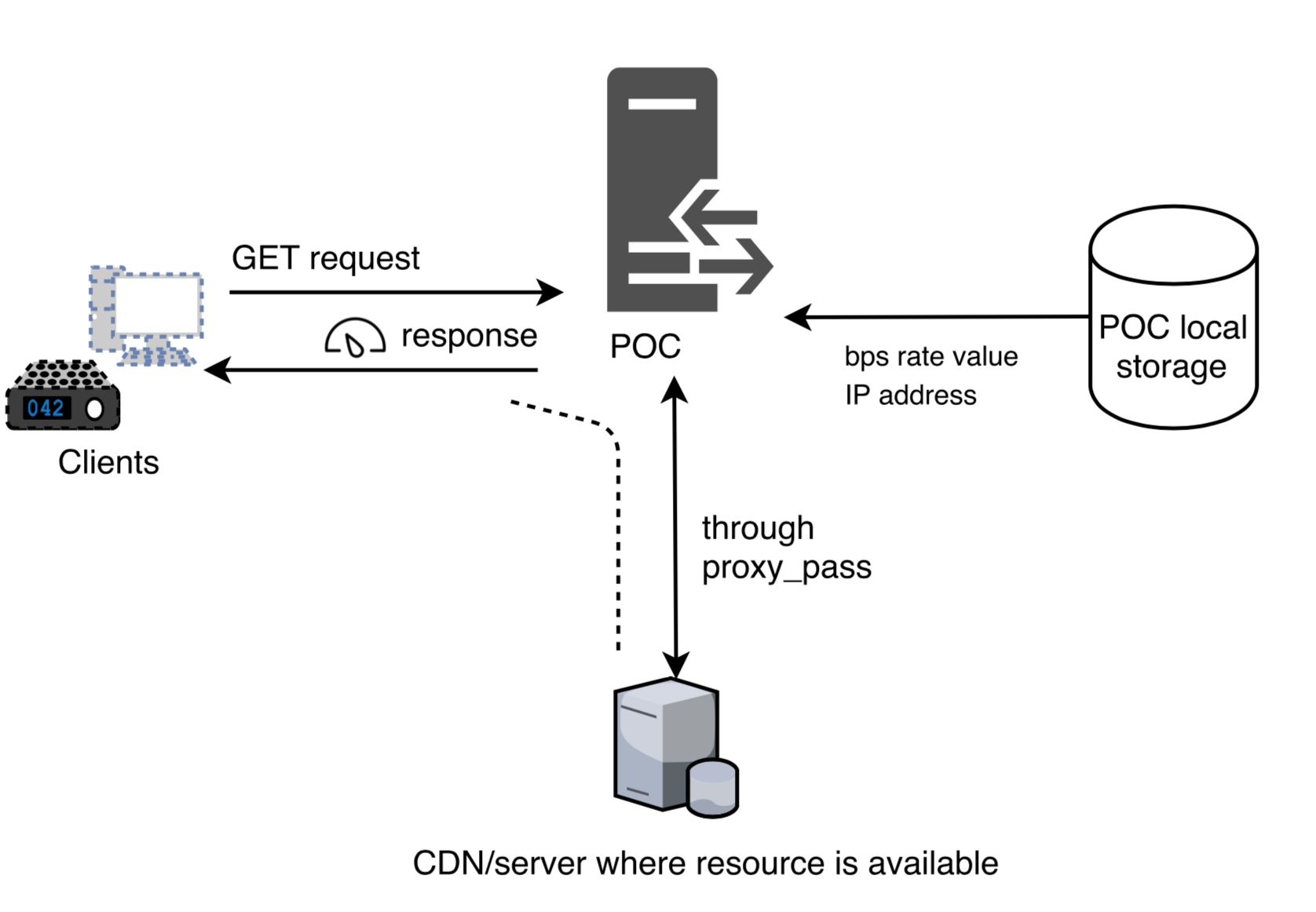
More precisely, our POC receives POST requests, and it stores in the local storage the (k)B/s value and the related ip address that are available within the requests. In order to store this information, the POC creates a new text file for each POST request received. Since every POST request has details about the ip address and the (k)B/s value, the POC defines a text file having the extension “.dat”, which is named through the first parameter. The second one is applied as content of the file, therefore each “.dat” file consists of a unique row composed by the (k)B/s value. If for a particular ip address a value is already specified, this rate value is overwritten by the new one just specified. This storage mechanism permits to obtain very quickly the information about an exisiting (k)B/s rate value, since the time taken to access and consequently read the file is measured in constant time. These “.dat” files are then applied in order to throttle the band of incoming GET requests.

Fig. 2 – “bps” is treated as (k)B/s value

Indeed, as the Fig. 2 shows, once a client forwards a GET request, the POC checks its local storage, in order to know if for a particular IP address, from which the GET requests arrives, a (k)B/s value exists. If it exists, the POC throttles the band, in such a way that the speed does not exceed the value specified. Thanks to the personalization of the module that we import, the checking activity is performed also when a GET request is served, not only when it arrives. These facts enable the POC to condition the speed of (1) new incoming requests and (2) requests that have been already opened. In addition, this POC exploits the proxy\_pass feature of nginx. Hence, as the Fig. 2 shows, the POC receives the GET request forwarded by the client, it sends to the original server where the resource requested is available and receives the response. Then the POC takes the reponse, and forwards it back to the client. In this manner, it is possible to condition the speed also for all the resources available on the domain specified in proxy\_pass.

1. [ma.zardoni@reply.it](mailto:ma.zardoni@reply.it) [↑](#footnote-ref-1)
2. 1 <https://github.com/bigplum/Nginx-limit-traffic-rate-module> [↑](#footnote-ref-2)