TokenGen - Developer Documentation

Design Overview

The distributed TokenGen is implemented as a FastCGI extension to the Apache web server. All requests at the Apache server is handled by the FastCGI module. Additional context about the request via environment variables is also passed on by Apache. The main thread in the FastCGI module then computes the wait time for the request, and returns the appropriate HTTP response to the client. The HTTP response contains the meta HTTP refresh header, that automatically redirects the client to the original web server after the prescribed wait time. The redirect URL contains the original URL that the client requested from the application server, along with the following information embedded in the URL string: the current time stamp, the wait time relative to the current time stamp, and a HMAC token that is used to check the authenticity of the reported wait time.

Each of the TokenGen replica reads from a configuration file the network (IP and port) details of other replicas. Each replica has as many threads as the number of peers replicas, each of which sends the wait time array to the corresponding peer periodically. Apart from the above the there are three running threads. One thread prints debug logs. Another thread gathers capacity updates from the capacity estimation module The main thread assigns wait times.

Every TokenGen instance has $n + 3$ running threads. Where n is the number of peer proxies. The main thread is responsible for assigning wait times to user requests. Second thread is a listening thread which listens to all the inbound communication towards the TokenGen replica. The third thread is used to print debug logs periodically. Inbound communication contains of data from other peer TokenGen replicas and capacity updates from the capacity estimation module. The remaining n threads communicate the current wait time data and current incoming load with all other peer TokenGen replicas periodically. Finally, one of the replicas runs the capacity estimation module that provides the total server capacity estimates to all other replicas.

Source Code Overview - TokenGen

The core functions and operations performed in the source files are listed below

functions.h contains helper function for various tasks in proxy1.c parser.h - Reads the config file and populates various variables with the correct values.

proxy1.c

- readFromClient - This function reads incoming communications from TokenGen and CapacityEstimator. It is spawned in a different thread for each incoming connection and the data is accepted.Once the data transfer is over the thread is closed. This functions handles all the incoming requests weather it is from TokenGen or CapacityEstimator. It distinguishes the source based on the sentinel character sent by the client.

- `create_server_socket`- creates a server socket where incoming requests from TokenGen and CapacityEstimator arrives. Once the connection is accepted, this function calls the readFromClient function to get the data.

- writeToServer - n instances of this function run is parellel, where n is the number of peers. Once connected to a server(peer TokenGen) it will keep on sending the incoming rate and `visitor_count`at specified time intervals in an infinite loop.
- main - core logic of TokenGen, we find the share of capacity based on the data gathered by the above functions and redirect the requests to TokenCheck with the calculated wait time.

queue.h - Contains helper function written to maintain a queue. not used currently.
timer.h - contains functions to write logs.