Protocolos de Comunicación 2010

CSSC HTTP PROXY

Integrantes

- Colotto, Matías Ezequiel
- Cura, María Eugenia
- Samra, Santiago José
- Scaruli, Jorge Ezequiel

Introducción

Se implementó un proxy HTTP 1.1 que cumple con los requisitos de la cátedra, y sigue los estándares expuestos en la RFC 2616 (http://tools.ietf.org/html/rfc2616)

Approaches Iniciales

- Se evaluaron diferentes αpproaches para decidir de qué manera atender los pedidos de un cliente
- Para ello se implementaron servicios simples, como echo y se evaluó su performance y dificultad de implementación
- Se pensaron teniendo en cuenta que luego serían implementadas para el servidor proxy

Threaded Echo Server

- Consiste básicamente en un servidor echo que lanza un nuevo thread al recibir un pedido de un cliente
- Ventajas:
 - Simplicidad en su implementación
 - Posee manejo de concurrencia
- Desventajas:
 - Consume muchos recursos
 - Existe un overhead al iniciar un thread

Select Echo Server

- Esta alternativa utiliza la técnica "Select" que proveen los SocketChannels de Java para lograr E/S asincrónica
- Ventajas:
 - Muy eficiente en recursos
 - Se evitan problemas de concurrencia (1 thread)
- Desventajas:
 - No tiene buena performance en ámbitos de alta concurrencia

Threaded Select Echo Server

- Es una implementación que utiliza ambas alternativas anteriores y que funciona mediante tres threads: uno para lectura, uno para escritura y uno para aceptar conexiones
- Ventajas:
 - Puede escribir y leer al mismo tiempo
 - No tiene overhead al recibir una nueva conexión
- Desventajas:
 - La implementación es extremadamente compleja

Decisión final

- Se optó por implementar una variante de la primera opción
- Consiste en tener un pool de threads inicial, con una cantidad fija de threads, que atienden a las conexiones entrantes. Si no hay threads disponibles, se encolan las conexiones
- Se logró eliminar así el overhead de lanzar un nuevo thread ante cada nuevo pedido de un cliente, y tener un funcionamiento concurrente

Configuración

- Se hizo mediante un archivo XML
- Permite fijar algunos parámetros configurables de proxy que afectan a la performance del mismo, entre otras cosas, la cantidad de threads que hay en el pool
- Especifica los filtros y ACLs que fueron pedidos por la cátedra

Configuración: Formato

Ver archivo config.xml

Filtros y ACLs

- Se aplican en el orden especificados en el archivo de configuración
- Se pueden modificar en runtime sin problemas
- Se debe tener en cuenta que sólo se aplica el primer filtro que cumple su condición para ser aplicado

Configuración Remota

- Se implementó un simple HTTP server iterativo con el objetivo de servir una interfaz para configurar y monitorear el proxy
- Permite modificar la configuración actual subiendo un nuevo archivo de configuración
- Hay que autenticarse para acceder a este servicio, se utilizó HTTP Basic Authentication
- Se implementó una versión muy reducida de la API de Servlets

Configuración Remota

CSSC Proxy Server Configuration

Welcome, ssamra

Filter condition:

You may monitor the Proxy server status here

Configuration parameters

_			
Parameter	Current value		
Thread pool size	10		
Chaining proxy address	No chaining proxy		
Chaining proxy port	No chaining proxy		
Maximum server persistent connections	10		
Maximum server persistent connections per server	2		
Client persistent connection timeout	30000		
Server persistent connection timeout	30000		
Logging file name	log.txt		
New configuration:	Browse		
	Upload configuration		
Current filters:			
Filter 1:			

· OS: Mac OS

Monitoreo Remoto

CSSC Proxy Server Monitoring

Back

Monitoring parameters

Transferred bytes

Parameter	Value
Total transferred bytes	1605307
Bytes sent from client to proxy	121580
Bytes sent from proxy to origin server	111420
Bytes sent from origin server to proxy	689690
Bytes sent from proxy to client	682617

Blocks

Parameter	Value
Whole blocks	3
IP blocks	12
URI blocks	5
Media type blocks	19
Size blocks	1

Transformations

Parameter	Value
L33t transformations	2
Image rotations	43

Conexiones Persistentes

- Se separó el manejo de conexiones persistentes entre los clientes y el proxy de aquellas entre el proxy y los origin servers
- Fue la principal causa de problemas en el trabajo práctico

ServerManager

- Es una interfaz que caracteriza a clases que manejen conexiones a origin servers
- El método principal es getSocket(InetAddress addr, int port), que permite obtener un Socket para hablar con un determinado origin server
- Se realizaron varias implementaciones de la misma

SimpleServerManager

- Es la implementación más simple, que no soporta conexiones persistentes
- Devuelve un nuevo Socket cada vez que se le pide el Socket correspondiente
- Solamente hecha para fines de testeo

PersistentServerManager

- Es una implementación basada en un mapa de IP a Socket y en el uso de monitores
- Permite solamente un Socket por servidor, por lo cual si se deseaba hablar con un servidor mientras dicho Socket estaba en uso, el thread quedaba bloqueado
- Produjo gran cantidad de bugs relacionados con condiciones de carrera

PersistentSemaphorizedServer Manager

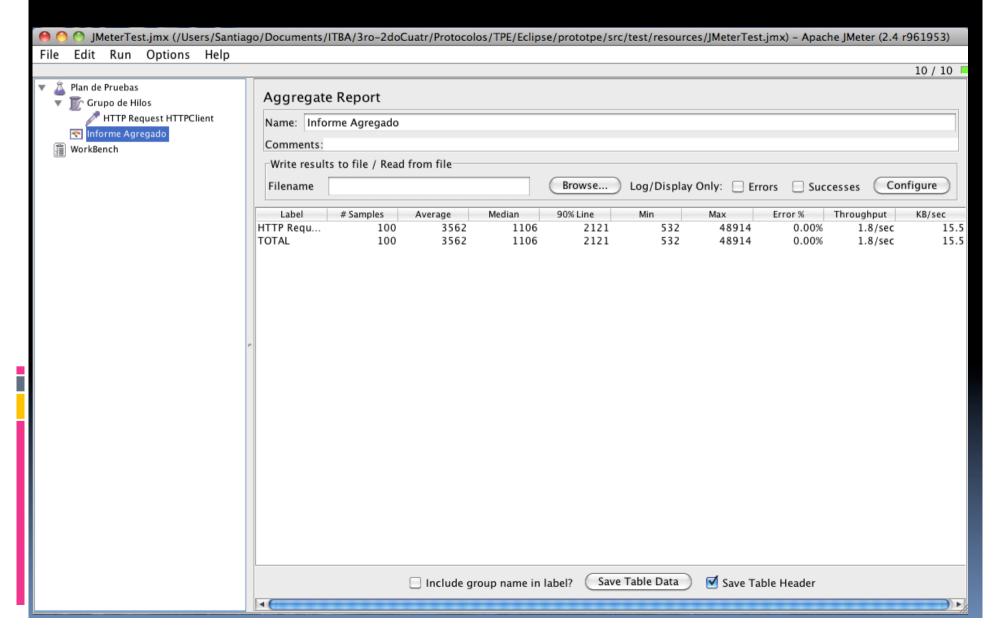
- Es la implementación definitiva, basada en dos mapas paralelos: de IP a cola de Socket, y de IP a semáforo
- Permite hasta N sockets diferentes por servidor y hasta M sockets abiertos en total al mismo tiempo
- Es rápida y no se han visto problemas en su funcionamiento

Logging

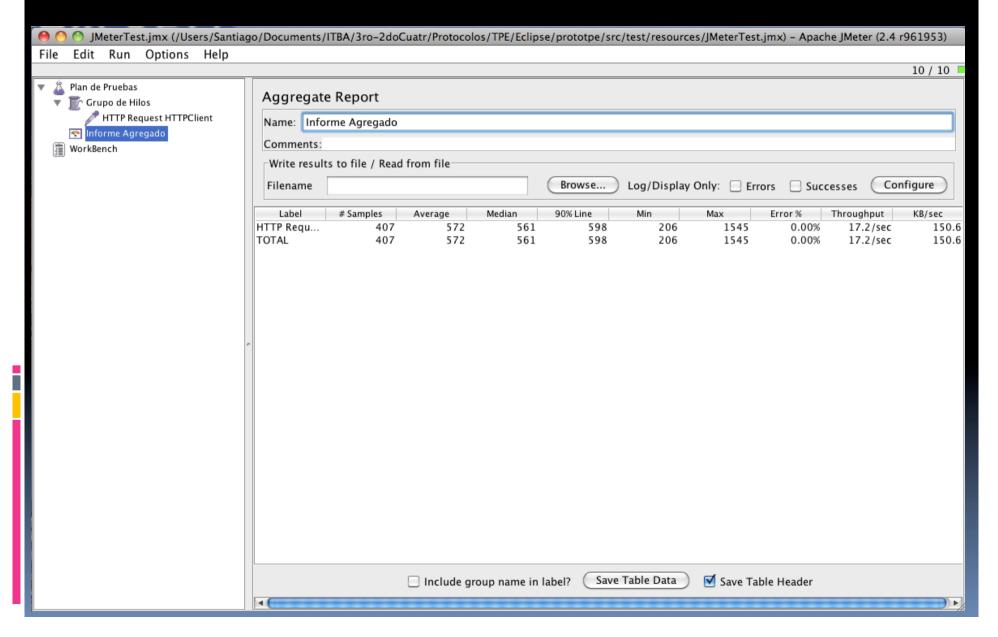
- Se guardan en un archivo log.txt
- Almacena requests y responses HTTP
- Contenido de ejemplo:

```
/10.6.0.82 - GET to www.lanacion.com.ar, path: http://www.lanacion.com.ar/
www.lanacion.com.ar - Replied 200 OK to /
10.6.0.82, requested path: http://
www.lanacion.com.ar/
```

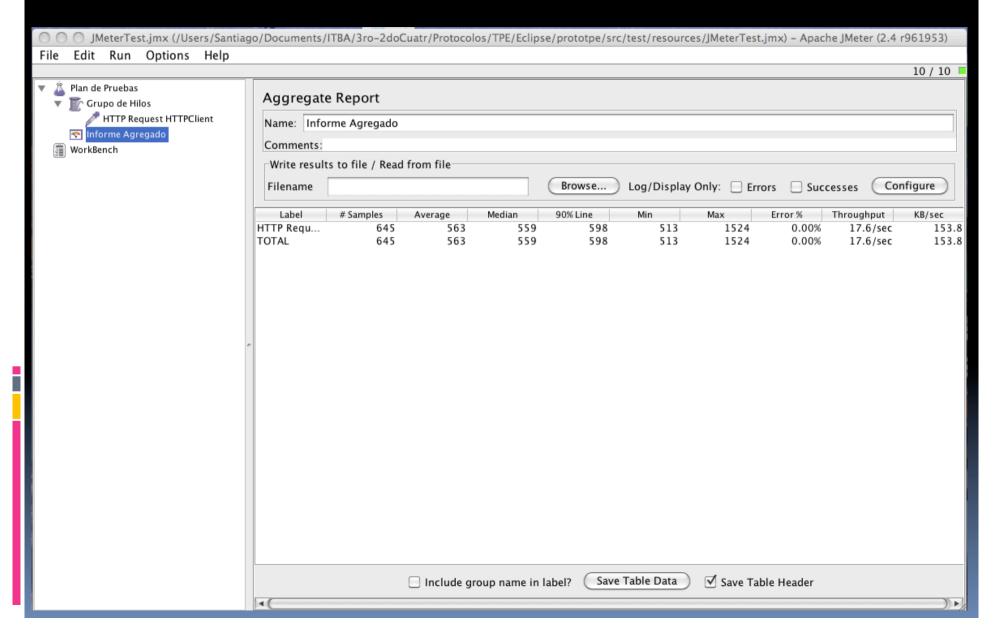
Benchmarking - PSM



Benchmarking - SSM



Benchmarking - PSSM



Problemas

- Conexiones persistentes
- Servidor SQUID del ITBA (utiliza HTTP 1.0)
- Headers fuera del estándar (Proxy-Connection)
- Manejo de conexiones de los Browser (depende si es explícito o transparente)

Limitaciones

- No se soporta pipelining de requests al servidor
- El número de conexiones hacia un origin server está limitado
- Hay configuraciones que no se pueden modificar en runtime
- No soporta todos los métodos HTTP

Conclusión

- Si se respetaran más los estándares, el trabajo de realizar un proxy HTTP sería relativamente más simple
- Es interesante ver cómo un proxy HTTP podría modificar todo el contenido por el cual navega un cliente sin que él se entere