

Laboratorio 10 Balanceador de Carga -ARSW

Cesar Eduardo Lanos Camacho

Parte 0

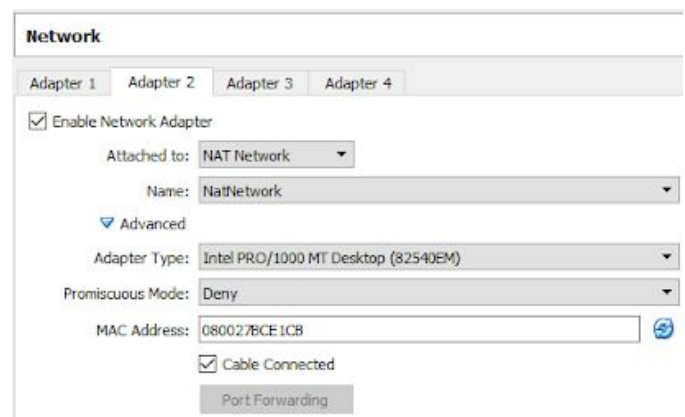
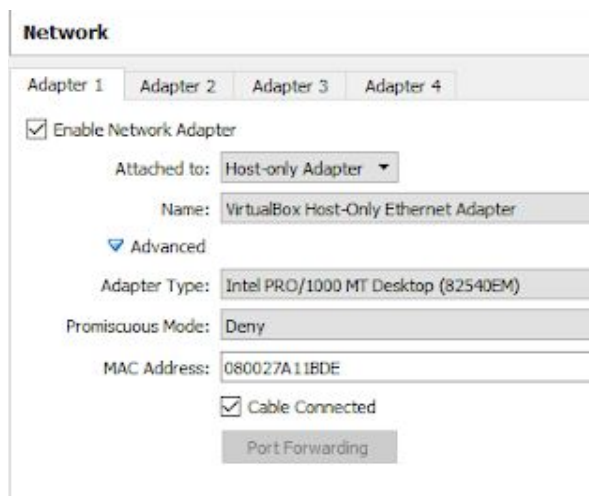
1. Importe la máquina virtual suministrada (extensión .ova).



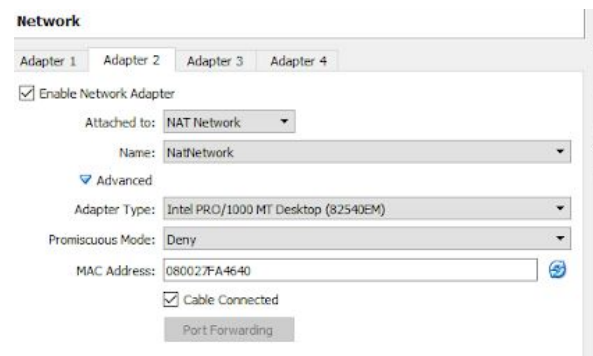
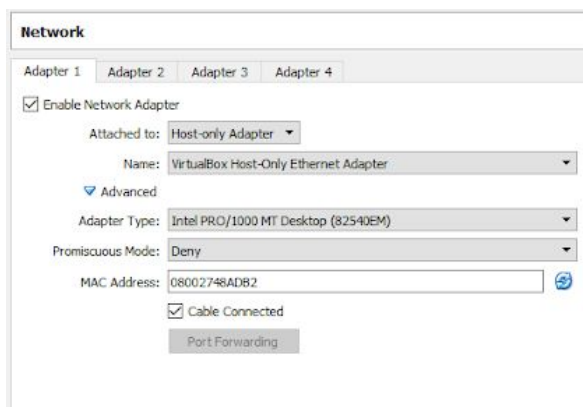
2. Antes de iniciar la máquina virtual, configure las redes de VirtualBox
Ambos puntos se realizan a continuacion.
3. Configure la máquina virtual (Settings/Network) y configure dos adaptadores de red. El primero de tipo 'Host-only' (asociado a la red vboxnet0), y el segundo de tipo NAT-Network (asociado a la red (NatNetwork)).

VARIOS PUNTOS RESUMIDOS

maquina original

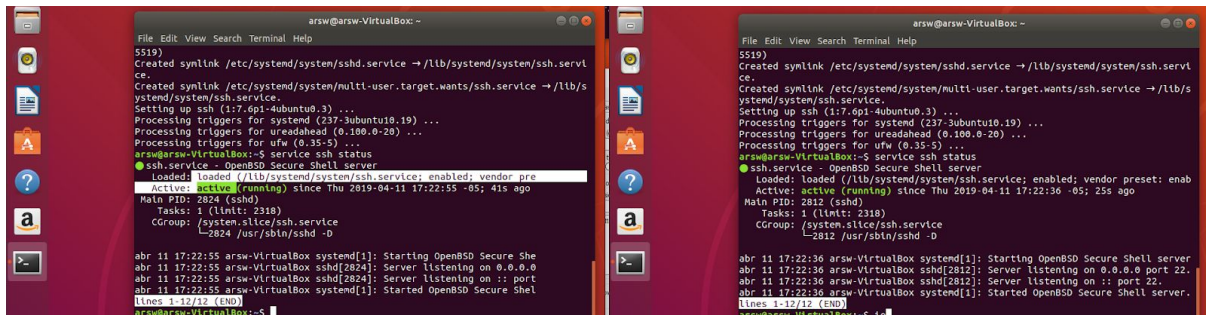


Maquina Clone



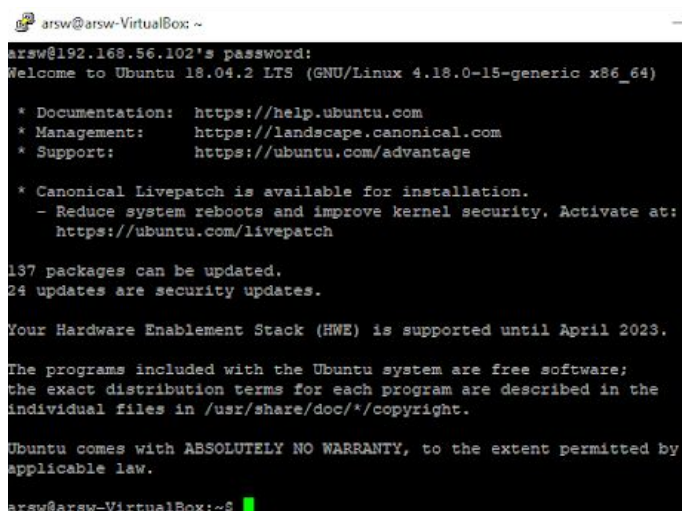
4. Inicie ambas máquinas y verifique que queden con sus respectivas direcciones, y que éstas sean accesibles. Una vez verificado esto, puede conectarse a las máquinas virtuales a través de ssh (para no tener que usar la terminal de la máquina virtual):

ssh funcionando



```
arsw@arsw-VirtualBox: ~  
File Edit View Search Terminal Help  
5519)  
Created symlink /etc/systemd/system/ssh.service → /lib/systemd/system/ssh.serv  
ce.  
Created symlink /etc/systemd/system/multi-user.target.wants/ssh.service → /lib/s  
ystemd/system/ssh.service.  
Setting up ssh (1:7.6p1-4ubuntu0.3) ...  
Processing triggers for systemd (237-3ubuntu10.19) ...  
Processing triggers for ureadahead (0.100.0-20) ...  
Processing triggers for ufw (0.35-5) ...  
arsw@arsw-VirtualBox:~$ service ssh status  
● ssh.service - OpenBSD Secure Shell server  
   Loaded: loaded (/lib/systemd/system/ssh.service; enabled; vendor preset: enab  
   Active: active (running) since Thu 2019-04-11 17:22:55 -05; 41s ago  
     Main PID: 2824 (sshd)  
       Tasks: 1 (limit: 2310)  
      CGroup: /system.slice/ssh.service  
              └─2824 /usr/sbin/sshd -D  
  
abr 11 17:22:55 arsw-VirtualBox systemd[1]: Starting OpenBSD Secure She  
abr 11 17:22:55 arsw-VirtualBox sshd[2824]: Server listening on 0.0.0.0  
abr 11 17:22:55 arsw-VirtualBox sshd[2824]: Server listening on :: port  
abr 11 17:22:55 arsw-VirtualBox systemd[1]: Started OpenBSD Secure Shel  
Lines 1-12/25 (END)  
arsw@arsw-VirtualBox:~$
```

Entrando por ssh

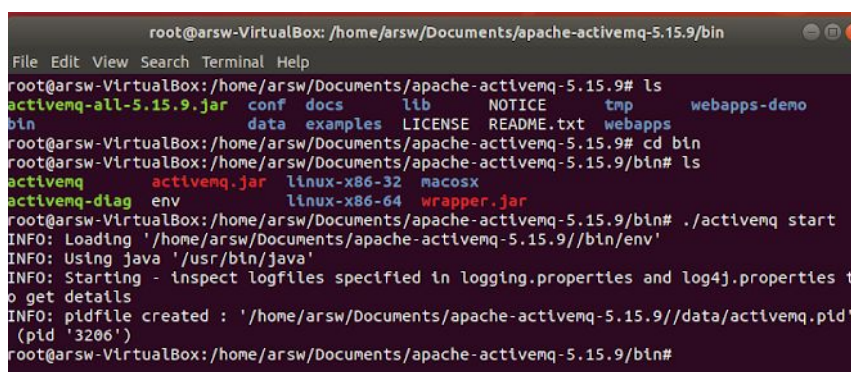


```
arsw@arsw-VirtualBox: ~  
arsw@192.168.56.102's password:  
Welcome to Ubuntu 18.04.2 LTS (GNU/Linux 4.18.0-15-generic x86_64)  
  
 * Documentation:  https://help.ubuntu.com  
 * Management:    https://landscape.canonical.com  
 * Support:        https://ubuntu.com/advantage  
  
 * Canonical Livepatch is available for installation.  
   - Reduce system reboots and improve kernel security. Activate at:  
     https://ubuntu.com/livepatch  
  
137 packages can be updated.  
24 updates are security updates.  
  
Your Hardware Enablement Stack (HWE) is supported until April 2023.  
  
The programs included with the Ubuntu system are free software;  
the exact distribution terms for each program are described in the  
individual files in /usr/share/doc/*/copyright.  
  
Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by  
applicable law.  
arsw@arsw-VirtualBox:~$
```

PARTE 1

1. En uno de los dos servidores virtuales, inicie el servidor ActiveMQ. Para esto, ubíquese en el directorio apache-activemq-5.14.1/bin (en el directorio raíz del usuario 'ubuntu'), y ejecute ./activemq start .

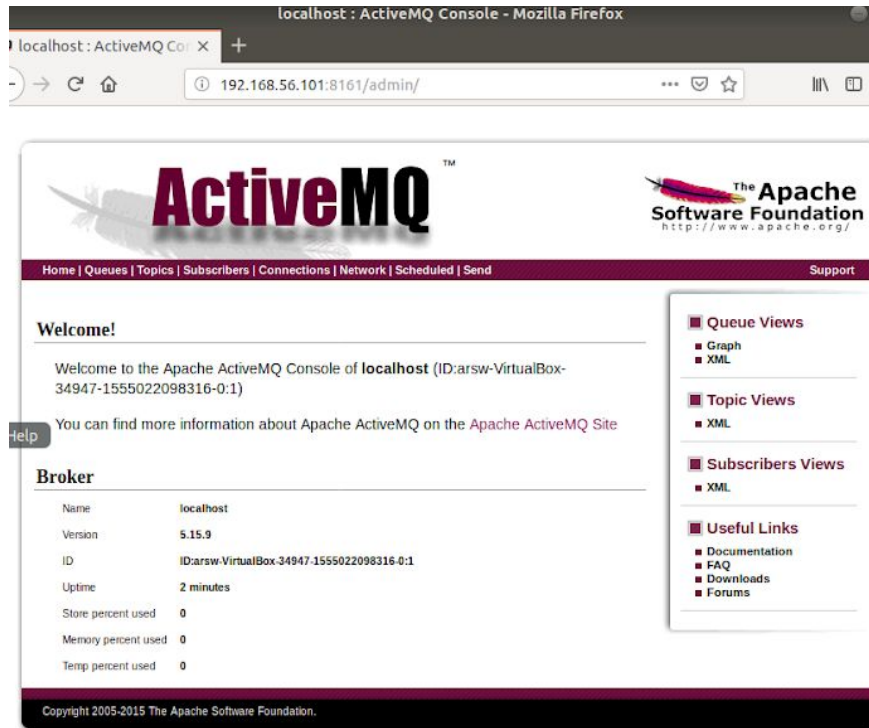
Activación en la máquina clon de de active qm



```
root@arsw-VirtualBox: /home/arsw/Documents/apache-activemq-5.15.9/bin  
File Edit View Search Terminal Help  
root@arsw-VirtualBox: /home/arsw/Documents/apache-activemq-5.15.9# ls  
activemq-all-5.15.9.jar  conf  docs  lib  NOTICE  tmp  webapps-demo  
bin  data  examples  LICENSE  README.txt  webapps  
root@arsw-VirtualBox: /home/arsw/Documents/apache-activemq-5.15.9# cd bin  
root@arsw-VirtualBox: /home/arsw/Documents/apache-activemq-5.15.9/bin# ls  
activemq  activemq.jar  linux-x86-32  macosx  
activemq-diag  env  linux-x86-64  wrapper.jar  
root@arsw-VirtualBox: /home/arsw/Documents/apache-activemq-5.15.9/bin# ./activemq start  
INFO: Loading '/home/arsw/Documents/apache-activemq-5.15.9/bin/env'  
INFO: Using java '/usr/bin/java'  
INFO: Starting - inspect logfiles specified in logging.properties and log4j.properties t  
o get details  
INFO: pidfile created : '/home/arsw/Documents/apache-activemq-5.15.9/data/activemq.pid'  
(pid '3206')  
root@arsw-VirtualBox: /home/arsw/Documents/apache-activemq-5.15.9/bin#
```

2. Para verificar que el servidor de mensajes esté arriba, abra la consola de administración de ActiveMQ: `http://IP_SERVIDOR:8161/admin/` . Consulte qué tópicos han sido creados en el momento.

Ingreso al active mq



3. Recupere la última versión del ejercicio realizado de WebSockets (creación colaborativa de polígonos). Modifíquese para que en lugar de usar el 'simple Broker' (un broker de mensajes embebido en la aplicación), delegue el manejo de los eventos a un servidor de mensajería dedicado (en este caso, ActiveMQ).

```
@Override
public void configureMessageBroker(MessageBrokerRegistry config) {
    //config.enableSimpleBroker("/topic");
    config.enableStompBrokerRelay("/topic").setRelayHost("192.168.56.101").setRelayPort(61613);

    config.setApplicationDestinationPrefixes("/app");
}
```

4. DModifique, también en la configuración, el registro del 'endpoint', para que permita mensajes de otros servidores (por defecto sólo acepta de sí mismo). Eso es requerido para permitir el manejo del balanceador de carga:

```
@Override
public void registerStompEndpoints(StompEndpointRegistry registry) {
    registry.addEndpoint("/stompendpoint").setAllowedOrigins("*").withSockJS();
}
```

5. Agregue las siguientes dependencias al proyecto:

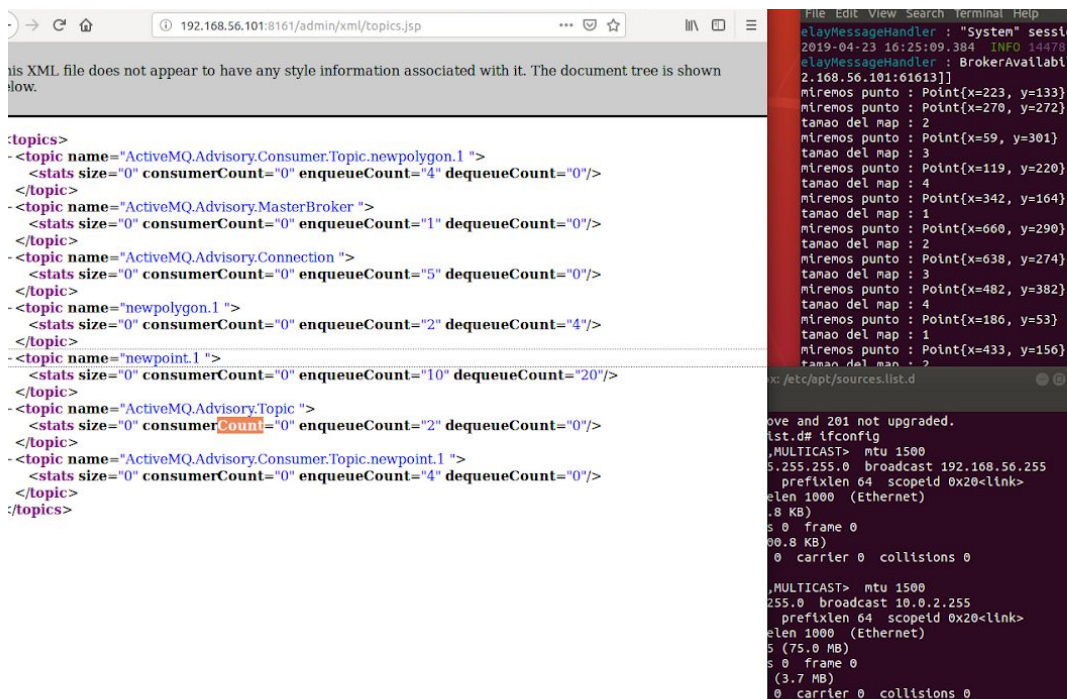
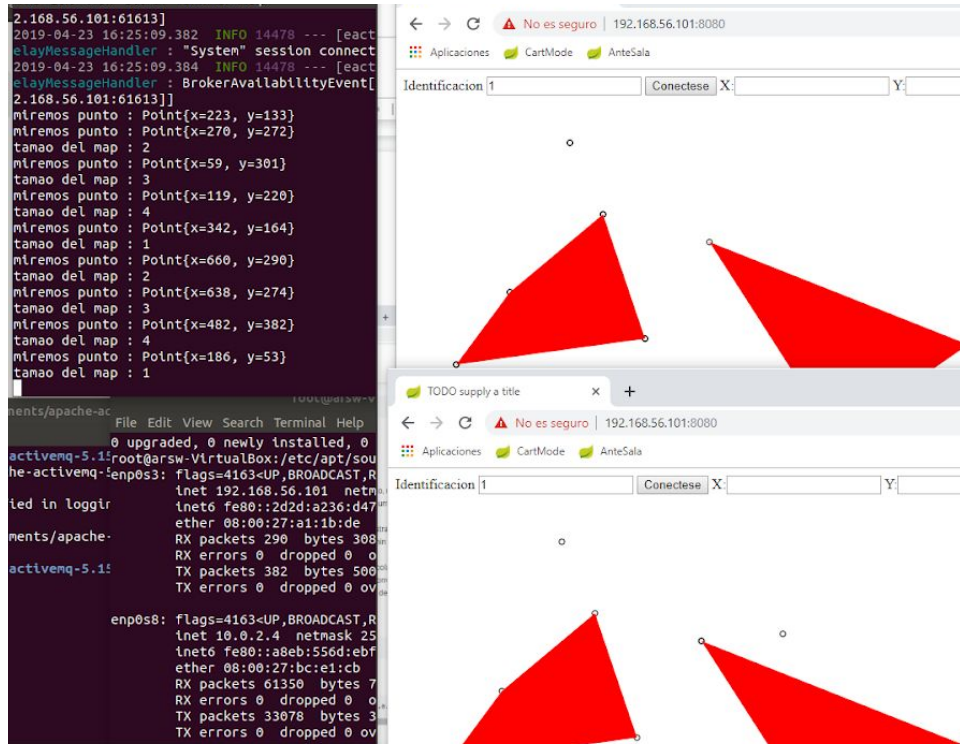
```
<dependencies>
    <dependency>
        <groupId>org.springframework.integration</groupId>
        <artifactId>spring-integration-amqp</artifactId>
    </dependency>
    <dependency>
        <groupId>io.projectreactor</groupId>
        <artifactId>reactor-core</artifactId>
        <version>2.0.8.RELEASE</version>
    </dependency>

    <dependency>
        <groupId>io.projectreactor</groupId>
        <artifactId>reactor-net</artifactId>
        <version>2.0.8.RELEASE</version>
    </dependency>
    <dependency>
        <groupId>io.netty</groupId>
        <artifactId>netty-transport</artifactId>
        <version>4.0.42.Final</version>
    </dependency>
    <dependency>
        <groupId>io.netty</groupId>
        <artifactId>netty-transport-native-epoll</artifactId>
        <version>4.0.42.Final</version>
    </dependency>
</dependencies>
```

6. En cada máquina ejecuta la aplicación, y desde el navegador verifique que las dos aplicaciones funcionen correctamente y al haber usado la aplicación, consulte nuevamente la consola Web de ActiveMQ, y revise qué información de tópicos se ha mostrado

Funcionamiento en ambas máquinas de la aplicación

máquina 1



máquina 2

The terminal window displays the following logs and network configuration:

```
File Edit View Search Terminal Help
eMessageHandler : BrokerAvailabilityEvent[available
.168.56.101:61613]]
2019-04-23 16:32:42.357 INFO 8406 --- [nio-8080-exe
dServletContainer : Tomcat started on port(s): 8080
2019-04-23 16:32:42.371 INFO 8406 --- [nio-8080-exe
lication : Started CollabPaintApplication t
for 15.838)
2019-04-23 16:32:52.597 INFO 8406 --- [nio-8080-exe
alhost].[/] : Initializing Spring FrameworkSer
2019-04-23 16:32:52.598 INFO 8406 --- [nio-8080-exe
herServlet : FrameworkServlet 'dispatcherServ
2019-04-23 16:32:52.636 INFO 8406 --- [nio-8080-exe
herServlet : FrameworkServlet 'dispatcherServ
ed in 38 ms
miremos punto : Point(x=105, y=158)
miremos punto : Point(x=506, y=213)
tamao del map : 2
miremos punto : Point(x=418, y=360)
tamao del map : 3
miremos punto : Point(x=249, y=349)
tamao del map : 4
miremos punto : Point(x=616, y=208)
tamao del map : 1

root@arsw-VirtualBox: /home/arsw# ifconfig
mp0s3: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> p
inet 192.168.56.102 netmask 255.255.255.0 b
inet6 fe80::d46a:1a:1663:e2a8 prefixlen 64
ether 08:00:27:10:9a:df txqueuelen 1000 (Et
RX packets 20370 bytes 30549952 (30.5 MB)
RX errors 0 dropped 0 overruns 0 frame 0
TX packets 624 bytes 295643 (295.6 KB)
TX errors 0 dropped 0 overruns 0 carrier 0

mp0s8: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> p
inet 10.0.2.6 netmask 255.255.255.0 broadca
inet6 fe80::305b:7e1f:e660:193f prefixlen 64
ether 08:00:27:2b:a9:d1 txqueuelen 1000 (Et
RX packets 53112 bytes 74567709 (74.5 MB)
RX errors 0 dropped 0 overruns 0 frame 0
TX packets 23469 bytes 2017778 (2.0 MB)
TX errors 0 dropped 0 overruns 0 carrier 0

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
inet 127.0.0.1 netmask 255.0.0.0

```

The browser displays the message: "This XML file does not appear to have any style information associated with it. The document tree is shown below."

```
<?xml version="1.0" encoding="UTF-8" standalone="no" ?>
<topics>
  <topic name="ActiveMQ.Advisory.Consumer.Topic.newpolygon.1">
    <stats size="0" consumerCount="0" enqueueCount="4" dequeueCount="0"/>
  </topic>
  <topic name="ActiveMQ.Advisory.Consumer.Topic.newpolygon.3">
    <stats size="0" consumerCount="0" enqueueCount="2" dequeueCount="0"/>
  </topic>
  <topic name="ActiveMQ.Advisory.MasterBroker">
    <stats size="0" consumerCount="0" enqueueCount="1" dequeueCount="0"/>
  </topic>
  <topic name="ActiveMQ.Advisory.Connection">
    <stats size="0" consumerCount="0" enqueueCount="8" dequeueCount="0"/>
  </topic>
  <topic name="newpolygon.1">
    <stats size="0" consumerCount="0" enqueueCount="2" dequeueCount="4"/>
  </topic>
  <topic name="newpolygon.3">
    <stats size="0" consumerCount="2" enqueueCount="1" dequeueCount="2"/>
  </topic>
  <topic name="ActiveMQ.Advisory.Consumer.Topic.newpoint.3">
    <stats size="0" consumerCount="0" enqueueCount="2" dequeueCount="0"/>
  </topic>
  <topic name="newpoint.1">
    <stats size="0" consumerCount="0" enqueueCount="10" dequeueCount="20"/>
  </topic>
  <topic name="newpoint.3">
    <stats size="0" consumerCount="2" enqueueCount="6" dequeueCount="12"/>
  </topic>
  <topic name="ActiveMQ.Advisory.Topic">
    <stats size="0" consumerCount="0" enqueueCount="4" dequeueCount="0"/>
  </topic>
  <topic name="ActiveMQ.Advisory.Consumer.Topic.newpoint.1">
    <stats size="0" consumerCount="0" enqueueCount="4" dequeueCount="0"/>
  </topic>
</topics>

```

The terminal window displays the following logs and network configuration:

```
File Edit View Search Terminal Help
eMessageHandler : "System" sessio
2019-04-23 16:25:09.384 INFO 14478
eMessageHandler : BrokerAvailabl
2.168.56.101:61613]]
miremos punto : Point(x=223, y=133)
miremos punto : Point(x=270, y=272)
tamao del map : 2
miremos punto : Point(x=59, y=301)
tamao del map : 3
miremos punto : Point(x=119, y=220)
tamao del map : 4
miremos punto : Point(x=342, y=164)
tamao del map : 1
miremos punto : Point(x=660, y=290)
tamao del map : 2
miremos punto : Point(x=638, y=274)
tamao del map : 3
miremos punto : Point(x=482, y=382)
tamao del map : 4
miremos punto : Point(x=186, y=53)
tamao del map : 1
miremos punto : Point(x=433, y=156)
tamao del map : 2

x:/etc/apt/sources.list.d

ove and 201 not upgraded.
lst.d# ifconfig
,MULTICAST> mtu 1500
5.255.255.0 broadcast 192.168.56.255
prefixlen 64 scopeid 0x20<link>
elen 1000 (Ethernet)
.8 KB)
s 0 frame 0
00.8 KB)
0 carrier 0 collisions 0

,MULTICAST> mtu 1500
255.0 broadcast 10.0.2.255
prefixlen 64 scopeid 0x20<link>
elen 1000 (Ethernet)
5 (75.0 MB)
s 0 frame 0
(3.7 MB)
0 carrier 0 collisions 0


```

Parte 2

Configuración NGINX

Se escogió la máquina Ubuntu (original)

1. Cree un archivo de configuración NGINX, compatible con WebSockets, a partir de la siguiente plantilla. Ajuste la configuración de 'upstream' para que use el host y el puerto de los dos servidores virtuales, y el parámetro 'listen' para que escuche en el puerto 8090 .



```
events {
    worker_connections 768;
    # multi_accept on;
}

http {
    log_format formatWithUpstreamLogging '[${time_local}] $remote_addr - $remote_user - $server_name to: $upstream_addr: $request';
    access_log access.log formatWithUpstreamLogging;
    error_log error.log;

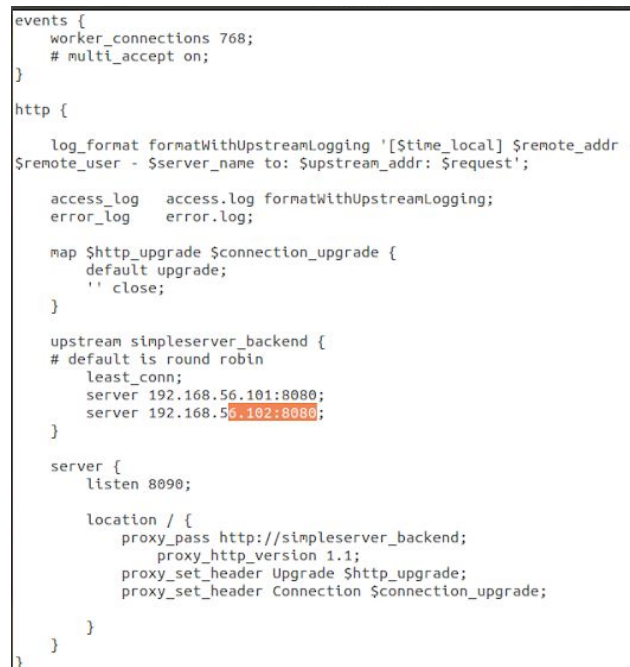
    map $http_upgrade $connection_upgrade {
        default upgrade;
        '' close;
    }

    upstream simpleserver_backend {
        # default is round robin
        server 192.168.56.101:8080;
        server 192.168.56.102:8080;
    }

    server {
        listen 8090;

        location / {
            proxy_pass http://simpleserver_backend;
            proxy_http_version 1.1;
            proxy_set_header Upgrade $http_upgrade;
            proxy_set_header Connection $connection_upgrade;
        }
    }
}
```

2. Revise en la documentación de NGINX, cómo cambiar la estrategia por defecto del balanceador por la estrategia 'least_conn'.



```
events {
    worker_connections 768;
    # multi_accept on;
}

http {
    log_format formatWithUpstreamLogging '[${time_local}] $remote_addr - $remote_user - $server_name to: $upstream_addr: $request';
    access_log access.log formatWithUpstreamLogging;
    error_log error.log;

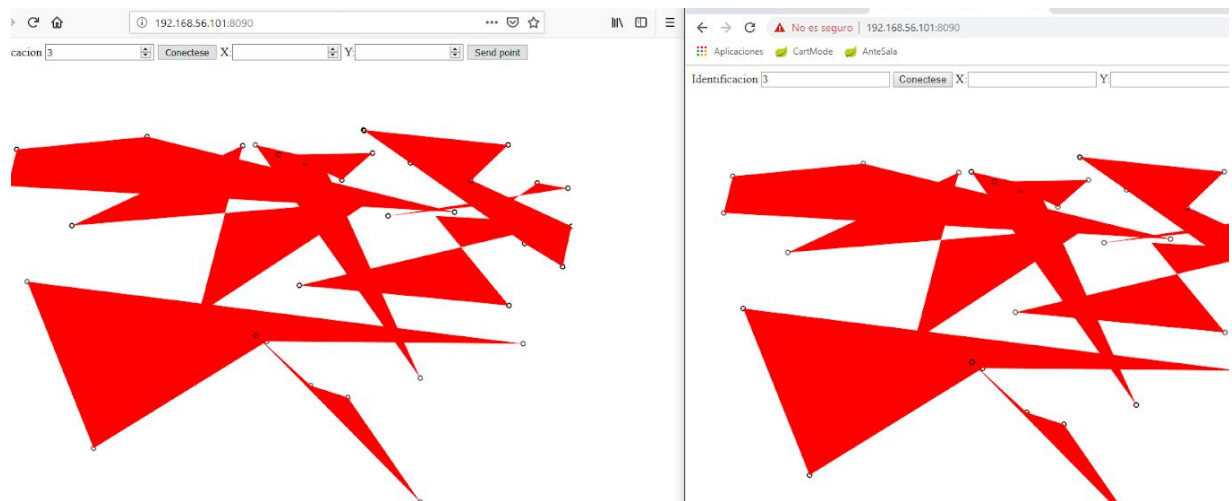
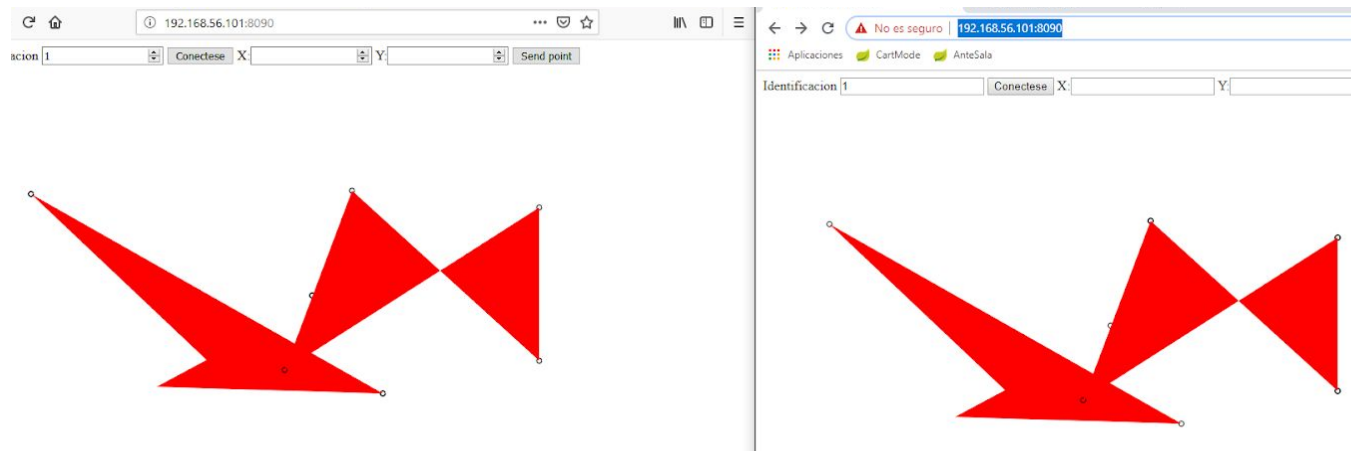
    map $http_upgrade $connection_upgrade {
        default upgrade;
        '' close;
    }

    upstream simpleserver_backend {
        # default is round robin
        least_conn;
        server 192.168.56.101:8080;
        server 192.168.56.102:8080;
    }

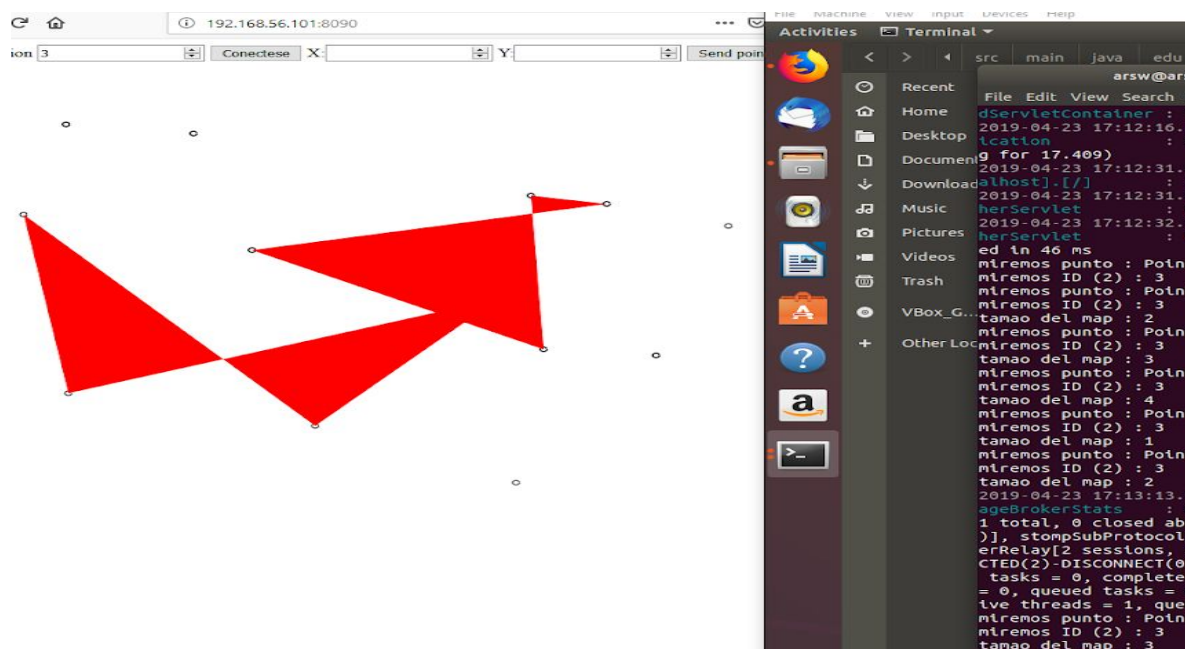
    server {
        listen 8090;

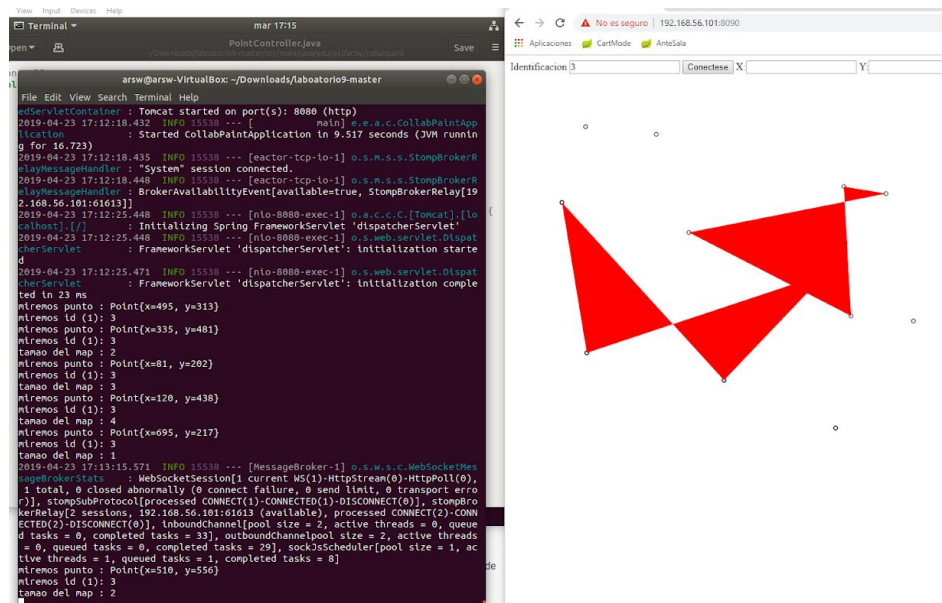
        location / {
            proxy_pass http://simpleserver_backend;
            proxy_http_version 1.1;
            proxy_set_header Upgrade $http_upgrade;
            proxy_set_header Connection $connection_upgrade;
        }
    }
}
```


3. Ejecute de nuevo la aplicación, pero esta vez abriendo la aplicación desde navegadores diferentes (p.e. Chrome y Firefox), y haciendo uso de la misma.

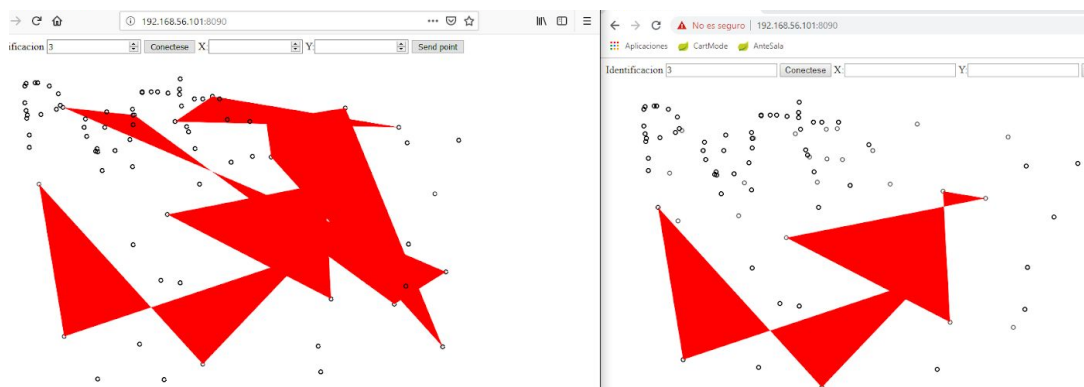


4. Revise, a través de los LOGs de cada servidor, si se están distribuyendo las peticiones. Revise qué instancia de la aplicación se le está asignando a cada cliente.





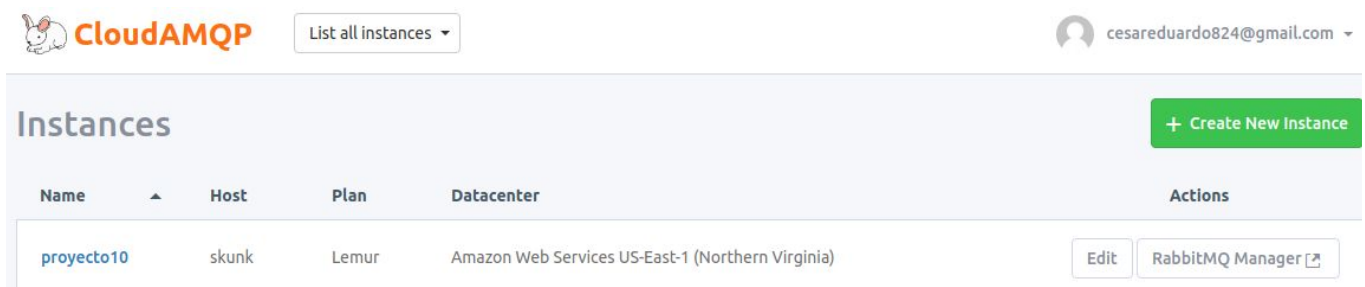
5. Apague una de las dos aplicaciones (Ctrl+C), y verifique qué pasa con el cliente que estaba trabajando con el servidor recién apagado.



Aunque se apago la maquina <http://192.168.56.101:8090/>, aun al poner puntos en ambas pantallas se dibujaban en el otro navegador, mas lento claramente pero se podía seguir utilizando la aplicación como si nada.

Parte 3

1. Cuenta creada e instancia lemur.



Details

Host(s)


skunk.rmq.cloudamqp.com (Load balanced)
skunk-01.rmq.cloudamqp.com

User & Vhost

rlysppti


Password


oaRMCy129N7HVMeuaW2HFrnao4VrnAZm

 Rotate password

AMQP URL

amqp://rlysppti:oaRMCy129N7HVMeuaW2HFrnao4VrnAZm@skunk.rmq.cloudamqp.com/rlysppti


3.7.4 Erlang 20.2


3.7.4 Erlang 20.2

Overview
Connections
Channels
Exchanges
Queues
Admin

Connections


▼ All connections (3)

Pagination

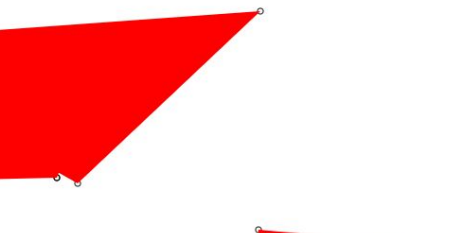
Page 1 of 1 - Filter: ☐ Regex ?

Overview			Details			Network	
Name	User name	State	SSL / TLS	Protocol	Channels	From client	To client
161.18.157.238:58300	rlsyppti	■ running	○	STOMP 1.2	1	0B/s	0B/s
161.18.157.238:58348	rlsyppti	■ running	○	STOMP 1.1	1	0B/s	0B/s
161.18.157.238:58520	rlsyppti	■ running	○	STOMP 1.1	1	0B/s	0B/s

← → ↺ ⌂ ⓘ localhost:8080 ☆ 🔔 🗑️ ⋮


Segurid PDF a Word PROYECTO

Identificacion 3
Conectese X: Y:
Send point



Queues


▼ All queues (4)

Pagination

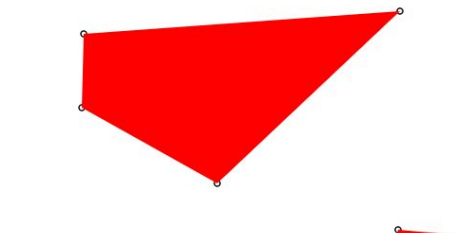
Page 1 of 1 - Filter: ☐ Regex ?

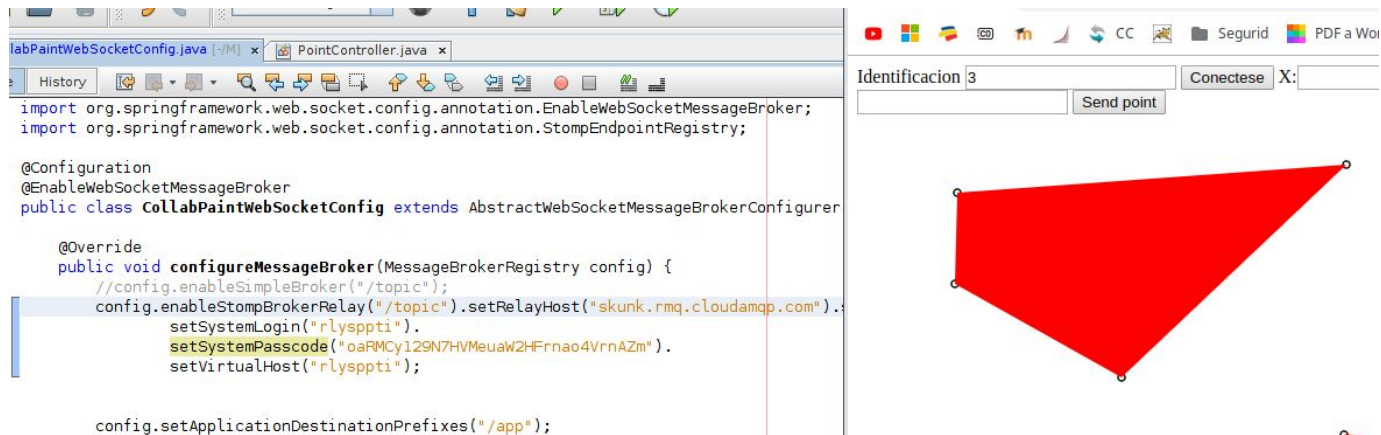
Overview			Messages			Message rates		
Name	Features	State	Ready	Unacked	Total	incoming	deliver / get	ack
stomp-subscription-7E0RJN9u8C_xiKSN8TX5dA	AD HA	■ running	0	0	0	0.00/s	0.00/s	0.00/s
stomp-subscription-VYJ_JJoEFrNDs8x19LDTQ	AD HA	■ idle	0	0	0	0.00/s	0.00/s	0.00/s
stomp-subscription-J0SZXd9YKoXsUTmDaKW5A	AD HA	■ running	0	0	0	0.00/s	0.00/s	0.00/s
stomp-subscription-rADT8YevdEAHgqL0AueKsw	AD HA	■ idle	0	0	0	0.00/s	0.00/s	0.00/s

← → ↺ ⌂ ⓘ localhost:8080 ☆ 🔔 🗑️ ⋮


Segurid PDF a Word

Identificacion 3
Conectese X: Y:
Send point





5. Consulte 'benchmarks' comparativos entre RabbitMQ y ActiveMQ, y analice cual sería más conveniente.

ActiveMQ está en el punto medio. Se puede implementar tanto con el intermediario como con las topologías P2P. Al igual que RabbitMQ, es más fácil implementar escenarios avanzados, pero generalmente a costa del rendimiento en bruto. Es la navaja suiza de mensajería .

RabbitMQ es una de las principales implementaciones del protocolo AMQP (junto con Apache Qpid). Por lo tanto, implementa una arquitectura de intermediario, lo que significa que los mensajes se ponen en cola en un nodo central antes de enviarlos a los clientes. Este enfoque hace que RabbitMQ sea muy fácil de usar e implementar, ya que solo en unas pocas líneas de código se admiten escenarios avanzados como enrutamiento, balanceo de carga o colas de mensajes persistentes. Sin embargo, también lo hace menos escalable y "más lento" porque el nodo central agrega latencia y los sobres de mensajes son bastante grandes.

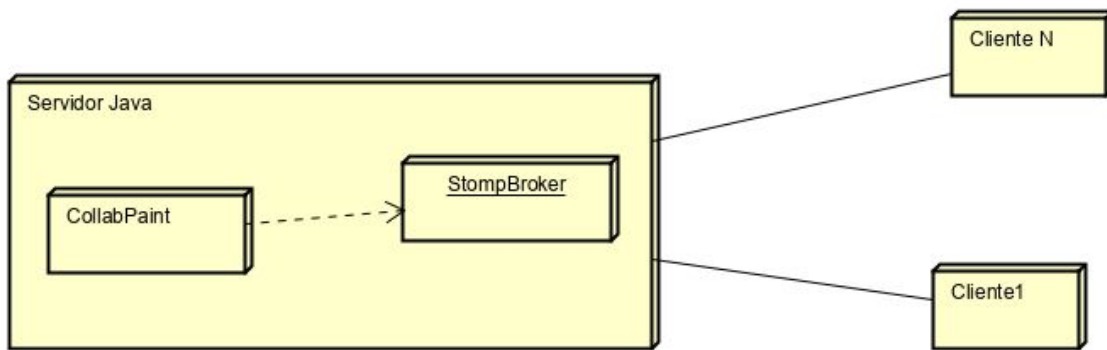
Finalmente, los 2 productos:

- Tienen API de cliente para los lenguajes más comunes (C ++, Java, .Net, Python, Php, Ruby, ...)
- Tienen documentación sólida
- Se admiten activamente

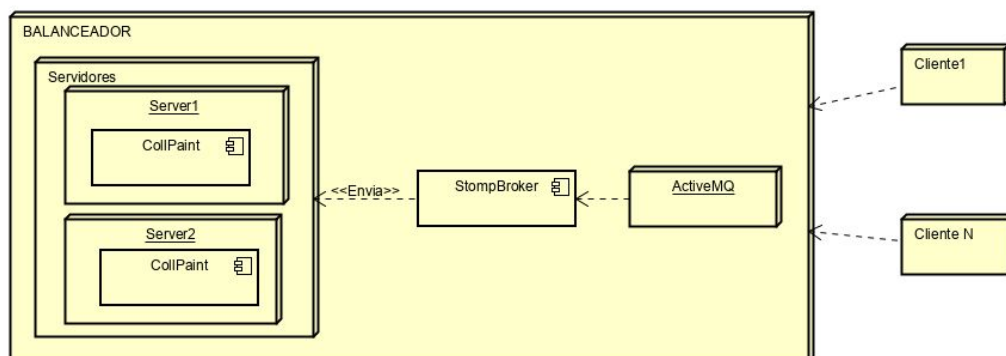
Después de un análisis de los benchmarks de cada aplicación, yo decido que lo mejor es utilizar ActiveMQ, dado que es sencillo y ligero y funciona excelente para este tipo de aplicación. Descarto a RabbitMQ sea fácil de usar e implementar y se puedan desarrollar escenarios muy cheveres, esto nos costaría en tiempo de respuesta y escalabilidad de de la aplicación. Además al ser una aplicación tan sencilla, de solo puntos, lo único que necesitamos es que sea algo rapido y sencillo.

Parte 4

1. Haga el diagrama de despliegue para la versión original del laboratorio.



2. Haga el diagrama de despliegue (incluyendo el detalle de componentes) para la nueva versión del laboratorio. En este caso suponga que los servidores no están en máquinas virtuales sino en máquinas reales.



3. Con la nueva arquitectura planteada las inconsistencias que se podrían presentar son, que un cliente al mandar un mensaje(en este caso puntos) apareciese en el de otro jugador con el mismo id en el de la otra aplicación.