INSTITUTO TECNOLÓGICO BUENOS AIRES

Proyecto Final

Proyecto TiX

Desarrollo e implementación de una arquitectura horizontalmente escalable

Autores
Matías Gabriel
Domingues
Facundo Nahuel
Martinez Correa
Javier Pérez Cuñarro

Tutor
Dr. Ing. José Ignacio
Alvarez-Hamelin

18 de diciembre de 2017

Índice

4. Ensayos

4.1. Prueba de Sistema

4.1.1. Objetivos

La Prueba de Sistema está apuntada a corroborar el correcto funcionamiento del Servicio de TiX en su totalidad en condiciones normales. Esto significa, que se debe probar de punta a punta el Sistema, con una carga normal, para una persona determinada. La idea de esto es asegurar una calidad mínima de servicio en base a parámetros preestablecidos.

4.1.2. Indicadores propuestos

Las métricas sobre las cuales se va a determinar que el sistema funciona correctamente son:

1. Tiempo de funcionamiento, entendiéndose como la cantidad de tiempo que estuvo vivo el sistema durante la prueba, para lo cual pudo recibir mediciones del Subsistema Cliente y mostrar el progreso de la prueba a través del Subsistema de Presentación.
2. Tiempo medio de demora en obtenerse una métrica, entendiéndose como el promedio de los tiempos que tarda el Subsistema de Procesamiento en entregar el resultado de un conjunto de mediciones más el Subsistema de Presentación en mostrarlo.
3. La media del valor absoluto del error de la métrica de Uso de Ancho de Banda, entendiéndose como el promedio de los valores absolutos de las diferencias entre el valor mostrado y el valor real para cada punto de la métrica calculado durante la prueba.

4.1.3. Consideraciones

Para esta prueba se tienen lo siguientes considerandos y supuestos:

- Se considera que el sistema es robusto y que el Tiempo de funcionamiento a lo largo de la prueba es una muestra lo suficientemente representativa del mismo en todo momento.
- Se considera que la prueba es lo suficientemente compleja para que su automatización total no sea relevante ni un punto a considerar para la presente disertación.
- Se considera una métrica aquella que puede ser visualizada a través del servicio tix-web del Subsistema de Presentación y Administración.

- Se considera negligible el efecto que tiene en el Uso de Ancho de Banda la carga de páginas web, la descarga esporádica de archivos con tamaño inferior a 1 MB y el uso en segundo plano de la conexión por parte de servicios como los son las redes sociales (e.g.: Facebook, Instagram, Twitter), medios de comunicación instantánea (e.g.: Slack, Whatsapp, Discord), portales web y agentes de correo electróncio (e.g.: GMail, Outlook, Outlook Express, Thunderbird).
- No se considera negligible el efecto que tiene en el Uso de Ancho de Banda la descarga constante de archivos o de archivos con tamaño superior a 1 MB, el uso de servicios o portales web de flujo de datos, sea de video (e.g.: YouTube, Netflix), audio (e.g.: iTunes, Spotify), u otro tipos de aplicaciones en tiempo real (i.e.: video juegos on-line, proyectos de cómputo distribuido, etc).
- Si bien en general los Proveedores de Servicio de Internet no entregan velocidades simétricas, lo que podría suponer una diferencia en la calidad de las conexiones de subida y bajada, se considera que:
  - Como método para medir el Uso de Ancho de Banda es el mismo tanto para subida como para la bajada.
  - Y que el método es robusto a esto ante la diferencia de calidad.
- Por el considerando anterior, sólo se medirá el Uso de Ancho de Banda de bajada.
- No se medirá la Calidad de la conexión debido a los muchos factores externos que juegan con ella.
- Se considera que una vez insertada la métrica en la base de datos del Subsistema de Presentación, está lista para ser mostrada. Con lo que si no puede ser mostrada es que está caído el Subsistema de Presentación en sí.

4.1.4. Metodología

Se crea un usuario en el Sistema TiX a través del Subsistema de Presentación, por medio de la página web. Luego se descarga y se instala el cliente.

Se instala un cliente de bittorrent en la computadora que va a realizar la prueba. La idea es saturar el enlace de forma programada para así poder tener valores de medición contra los cuales comparar. A su vez se debe asegurar que durante el periodo de la prueba ningún otro servicio salvo el cliente de bittorrent, el cliente de TiX y aquellos considerados negligibles para la saturación del ancho de banda, mantengan conexiones con internet. Durante las pruebas se utilizó el cliente Vuze [61]. Este permite programar de forma muy granular las velocidades máximas de transferencia de carga y descarga. Para la gran mayoría de las pruebas se utilizó la configuración del programador de velocidades de Vuze que se muestran en el Cuadro 1.

```
daily pause_all from 00:00 to 01:00 # No downloads
daily unlimited from 01:00 to 02:00 # Download at 100%
daily 80_pct_speed from 02:00 to 03:00 # Download at 80%
daily 75_pct_speed from 03:00 to 04:00 # Download at 75%
daily half_speed from 04:00 to 05:00 Download at 50%
daily 25_pct_speed from 05:00 to 06:00 # Download at 25%
daily pause_all from 06:00 to 23:59 # No downloads
```

Cuadro 1: Programación de las velocidades de descarga de Vuze

Para saturar la conexión se usaron los torrents de varias distribuciones de linux, ya que esto asegura descargas de gran tamaño por el tiempo prolongado que dura la prueba. Una vez iniciado el cliente de bittorrent, se procede a iniciar el cliente de TiX y se los deja corriendo durante todo el tiempo de la prueba.

Tras la finalización de la mismas se ejecutan las consultas que se encuentran en el repositorio de la prueba de sistema sobre la base de datos del Subsistema de Presentación. Estas arrojan los resultados de las métricas que nos interesan tener para verificar el éxito de la prueba.

4.1.5. Resultados

Tras correr algunas pruebas los resultados obtenidos fueron.

- Tiempo de funcionamiento: 95.0 %
- Tiempo medio de demora en obtenerse una métrica: $2' 30''$
- Media del error total de la métrica de Uso de Ancho de Banda: 0.12

Observaciones

- Aunque el tiempo de funcionamiento es menor al esperado, hay que tener en cuenta que se usa una única locación al momento de hacer las métricas, lo cual puede confundir el problema de un funcionamiento subpar con una caída del servicio desde el lado del cliente. Ejemplo de esto son las imágenes 16 y 18, donde para experimentos similares existen muchos menos puntos de ciertos periodos de la prueba.
- El error medio relativo puede dar más que el esperado debido a que el método utilizado, saturación mediante la descarga de torrents. La saturación del canal en este caso, no depende de la máquina desde donde se están haciendo las pruebas, sino de los pares y las semillas del torrent en sí.
- Como se puede apreciar desde las imágenes 16, 18, el error no es homogéneo en todas las mediciones. Hay mayor error cuanto menor es el uso

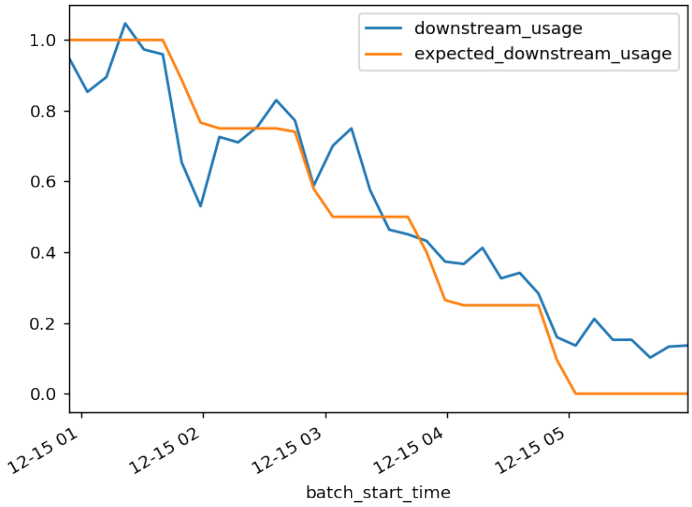

Figura 16: Medición contra valor esperado

de la conexión. Esto se hace evidente también en la figura 17, que además muestra que los puntos de mayor error son aquellos donde existen saltos bruscos de uso de datos.

4.2. Prueba de Carga

4.2.1. Objetivos

La Prueba de Carga está apuntada a medir la carga máxima del Sistema TiX y encontrar los cuellos de botella del mismo. Además, sirve como experimento para entender cómo es que se comporta un Sistema Complejo como esto y cuales son sus posibles mejoras.

Debido a la dinámica operativa de TiX, el servicio genera un gran nivel de carga, la cual, si bien es constante, es mucho más alta de lo que suele recibir un servidor web con la misma cantidad de usuarios. En un escenario ideal, el servicio tendría que ejecutarse en un espacio el cual permita el libre escalamiento de recursos, para cubrir eventuales picos de tráfico e infinitos usuarios. Lamentablemente esto es financieramente inviable, por lo que es necesario conocer con gran precisión cuál es el máximo número de usuarios que la plataforma puede soportar bajo la infraestructura en la cual se encuentra corriendo el servicio.

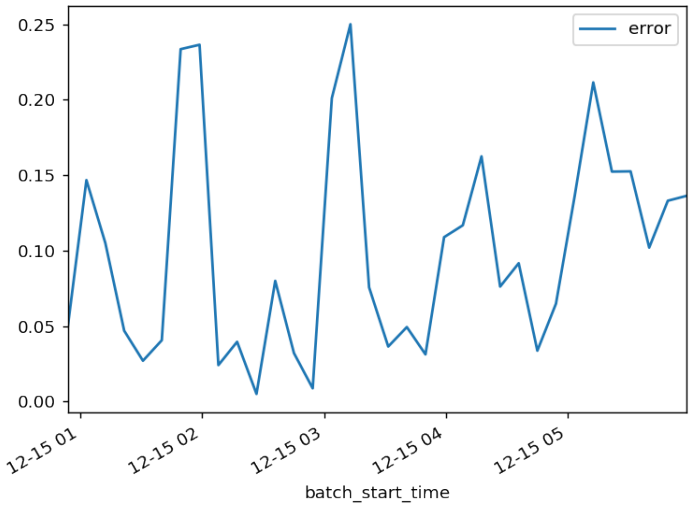

Figura 17: Error de la medición de la Figura 16

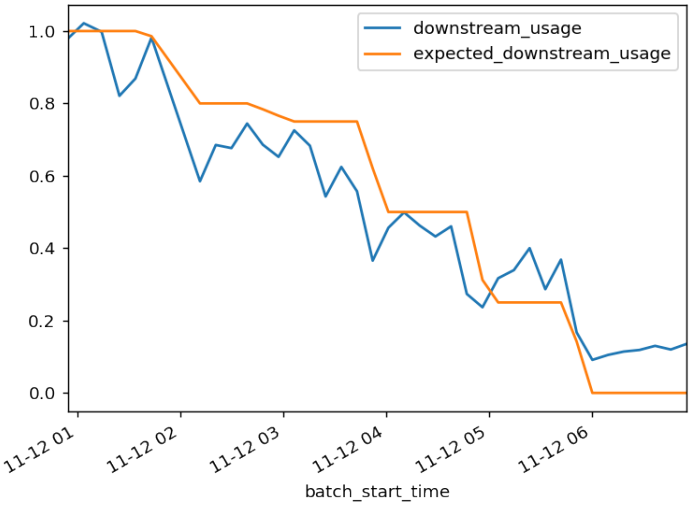

Figura 18: Otra medición contra su valor esperado

4.2.2. Indicadores propuestos

Para esta Prueba se considera una única

1. Cantidad máxima de usuarios con el sistema saturado.

Si bien existen muchas otras métricas que podrían resultar interesantes, como tiempo de funcionamiento para el sistema saturado, o degradación de la calidad del servicio con el sistema saturado, se entiende que el objetivo del presente Proyecto no es hacer una auditoría exhaustiva del Sistema, sino asegurar métricas mínimas de funcionamiento.

4.2.3. Consideraciones

Para esta prueba se tienen los siguientes considerandos y supuestos:

- La infraestructura sobre la que corre el sistema está compuesta por un servidor con las especificaciones descritas en el Cuadro 2.
- Dado el funcionamiento del Subsistema de Ingesta y Procesamiento, se considera que para probar la carga del sistema no es necesario usar múltiples usuarios, pero sí múltiples instalaciones.
- Se considera que para probar la carga del sistema, los resultados de las métricas no influyen. Es decir, el procesamiento de las mediciones y los resultados que estos arrojen no cambian el resultado de la Prueba de Carga. Por lo que los clientes pueden ser desplegados en cualquier infraestructura, desde cualquier lugar del mundo.

4.2.4. Metodología

La metodología para realizar la prueba de carga consta de dos partes: por un lado la generación de clientes para realizar la prueba, y por el otro el análisis de los resultados para saber si la prueba fue exitosa.

Generación de Clientes

Para la Generación de clientes, se utiliza una versión modificada del cliente actual, la cual cumple el mismo rol que el cliente estándar (autenticarse al servicio, crear una instalación y generar reportes). Esto permite correr múltiples instancias del cliente en el mismo dispositivo, con un consumo de recursos menor.

Por otro lado se utiliza JMeter [63], una herramienta especializada en generación de pruebas de carga. Esta permite crear múltiples instancias del cliente, las cuales son ejecutadas en simultáneo y genera un reporte basado en el resultado de su ejecución.

```
System:
    Host: tix.innova-red.net
    Kernel: 4.4.0-83-generic x86_64 (64 bit gcc: 5.4.0)
    Console: tty 1
    Distro: Ubuntu 16.04 xenial
Machine:
    System: Dell product: PowerEdge R210 II serial: BQK2TW1
    Mobo: Dell model: 03X6X0 v: A02 serial: ..CN708212BR2HYQ.
    Bios: Dell v: 2.2.3 date: 10/25/2012
CPU:
    Quad core Intel Xeon E31220 (-HT-MCP-) cache: 8192 KB flags:
(lm nx sse sse2 sse3 sse4_1 sse4_2 ssse3 vmx) bmips: 24744
    clock speeds: max: 3400 MHz 1: 1652 MHz 2: 1600 MHz 3: 1686
MHz 4: 1600 MHz
Graphics:
    Card: Matrox Systems MGA G200eW WPCM450 bus-ID: 03:03.0
    Display Server: N/A driver: N/A tty size: 237x56
    Advanced Data: N/A for root out of X
Network:
    Card-1: Broadcom NetXtreme II BCM5716 Gigabit Ethernet
driver: bnx2 v: 2.2.6 bus-ID: 02:00.0
        IF: eno1 state: up speed: 1000 Mbps duplex: full mac:
d4:ae:52:c7:83:53
    Card-2: Broadcom NetXtreme II BCM5716 Gigabit Ethernet
driver: bnx2 v: 2.2.6 bus-ID: 02:00.1
        IF: eno2 state: down mac: d4:ae:52:c7:83:54
Drives:
    HDD Total Size: 499.6GB (9.8% used) ID-1: /dev/sda model:
    Virtual_Disk size: 499.6GB temp: 0C
Partition:
    ID-1: / size: 19G used: 13G (75%) fs: ext4 dev: /dev/sda7
    ID-2: swap-1 size: 4.09GB used: 0.87GB (21%) fs: swap dev:
/dev/sda6
RAID: No RAID devices: /proc/mdstat, md_mod kernel module
present
Sensors:
    System Temperatures: cpu: 29.8C mobo: N/A
    Fan Speeds (in rpm): cpu: N/A
Info:
    Processes: 203
    Uptime: 144 days
    Memory: 1166.0/3921.2MB
    Init: systemd runlevel: 5 Gcc sys: 5.4.0
    Client: Shell (bash 4.3.481) inxi: 2.2.35
```

Cuadro 2: Especificaciones del Servidor de acuerdo a la salida del script inxi [62]

Finalmente, se necesita un servidor para poder correr la prueba de stress. Dada la carga generada y la necesidad de mantener el servicio en funcionamiento por al menos 24 horas, es recomendable que se realice en un servidor donde se pueda monitorear su funcionamiento. Aunque también puede ser realizado en cualquier PC.

Encontramos dos indicadores clave para conocer si la prueba podía ser corrida de manera exitosa:

1. Cantidad de operaciones de E/S en el cliente.
2. Estado del consumo de CPU en el servidor de los clientes.

El primero es necesario para asegurarse que la prueba está efectivamente corriendo al ratio que le fue solicitado. Por ejemplo, si se encuentra corriendo aproximadamente 1000 instancias, entonces tendrían que haber aproximadamente 1000 escrituras en archivo por segundo.

En segundo lugar, el estado de consumo de CPU es un factor clave para conocer si el servidor donde están corriendo los clientes se encuentra al límite de procesamiento. Al momento en que este se quede sin recursos, los clientes se volverán más lentos.

Si alguno de estos indicadores muestra algún problema, entonces prueba no está corriendo correctamente y sus resultados no pueden ser fiables.

Analisis de Resultados

El análisis de resultados comprende todas aquellas herramientas que pueden ser utilizadas para conocer cómo está funcionando internamente el Servicio. Esto incluye:

- Análisis de logs.
- Análisis de carga del sistema.
- Análisis del estado de las colas de procesamiento.

Siendo este último en particularmente crítico para conocer si el sistema está funcionando correctamente o se encuentra saturado.

El Análisis de logs se hace utilizando Docker Compose y su comando para visualizar logs. La principal idea de esto es detectar problemas en la y excepciones arrojadas por los servicios. También es entender el funcionamiento de cada uno de los servicios bajo estrés de manera más específica.

El Análisis de carga del sistema incluye medir y monitorear el uso de recursos, como CPU, memoria y operaciones de entrada y salida. La finalidad de esto es ver cuál es el cuello de botella general del Sistema. Si es intensivo en capacidad

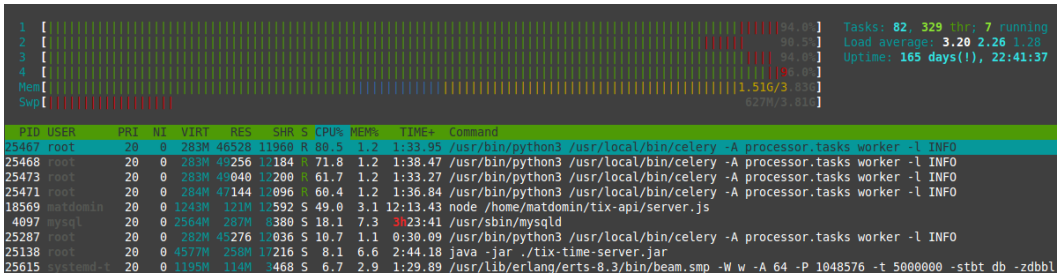

Figura 19: Captura de Pantalla mostrando la salida de HTOP al momento en que hay un pico de consumo de recursos en el Servidor del Sistema

de cómputo, en memoria o tráfico de red y operaciones de lecto-escritura. Para este análisis se usaron herramientas como htop, atop e iotop.

Finalmente, Analizar el estado de la cola de procesamiento es el factor determinante para entender el resultado de la prueba. En términos generales, dada la característica de TiX en donde el tráfico es constante, si la cantidad de paquetes que se están encolando para procesamiento es mayor que la capacidad de procesarlos, el sistema se va a ir acercando lentamente al colapso. Esto se puede visualizar analizando el gráfico que provee RabbitMQ sobre el estado de la cola.

4.2.5. Resultados

Luego de correr la prueba de carga múltiples veces, la conclusión es que el sistema, en las condiciones actuales, es capaz de soportar 2200 usuarios concurrentes.

Durante las distintas pruebas se pudo ver si bien la carga del servidor aumenta conforme aumenta la cantidad de clientes, el servidor no está todo el tiempo consumiendo al máximo sus recursos. Sino que por el contrario, que durante el tiempo de máxima carga hay momentos de actividad baja y momentos de saturación.

El momento de mayor consumo de recursos o de saturación es cuando se están procesando las métricas, como se puede ver en la figura 19. Esto se debe a que además del procesamiento, la API está recibiendo las métricas generadas e insertándolas en la base de datos.

Luego, el momento de baja actividad corresponde con el régimen regular del Sistema de Ingesta, como se muestra en la figura 20. Esto es, la toma de paquetes

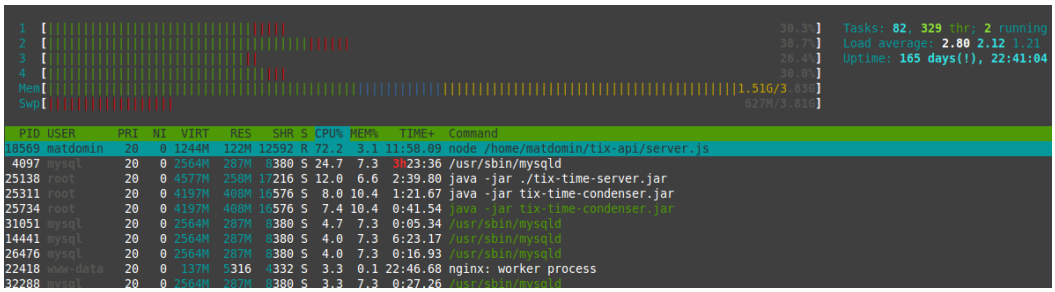

Figura 20: Captura de Pantalla mostrando la salida de HTOP al momento en que hay un consumo moderado a bajo de recursos en el Servidor del Sistema

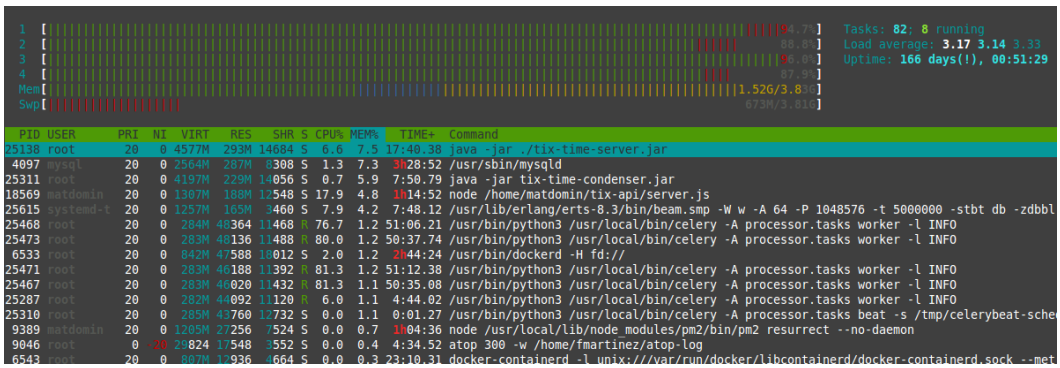

Figura 21: Captura de Pantalla mostrando la salida de HTOP al momento en que hay un pico de consumo de recursos en el Servidor del Sistema con los procesor ordenados por consumo de memoria

desde el tix-time-server, el procesamiento de los datos en el tix-time-condenser y los consultados generadas contra la tix-api.

Lo lleva a la conclusión que existen tres servicios bien diferenciados con distintos niveles de requerimientos. Por un lado la tix-api, debido a su conexión con la base de datos. Por otro el tix-time-server y el tix-time-condenser, que realizan un trabajo más rutinario y monótono con una carga constante del sistema. Y finalmente el tix-time-processor, que realiza un trabajo de procesamiento por lotes con un consumo fuerte en cortos periodos, y un consumo nulo el tiempo restante.

También, a partir de estas imágenes podemos ver que el Sistema es intensivo en Procesamiento, pero no en memoria. Ocupando en momentos de máxima saturación el 34 % de la memoria disponible con una configuración de una instancia de cada servicio, como se ve en la figura 21. Para esta configuración es que el sistema logró procesar 1900 usuarios concurrentes.

El sistema entero no satura los recursos del servidor, sino que el tiempo de

Figura 22: Panel de Control de RabbitMQ mostrando la cola que conecta el tix-time-server con el tix-time-condenser saturada

respuesta para atender los mensajes que llegan en la cola no es el suficiente.

La primera cola en saturarse, como se puede ver en las imágenes 22 e 23 es la que conecta el tix-time-server con el tix-time-condenser. Agregar más instancias de este último servicio permite mejorar el rendimiento general del sistema. Pero cada instancia nueva lo mejora marginalmente. Las razones de esto no fueron profundizadas, pero las hipótesis que surgieron son:

- Un problema de configuración del mismo servicio tix-time-condenser, que no le permite tomar una mayor cantidad de mensajes de la cola.
- Un problema de configuración de alguno de los otros servicios con los que interactúa (RabbitMQ y tix-api), que no le permite escalar linealmente.
- Un problema de infraestructura, consecuencia directa de estar corriendo todo en un mismo servidor. Es decir, la misma infraestructura es el cuello de botella.

Para poder hacer esto se debe hacer un perfilado más específico de estos servicios con herramientas como NewRelic, Pingdom Server Monitor o DataDog.

Como se menciona, se probaron varias configuraciones con distintos resultados, mostradas en la tabla 3.

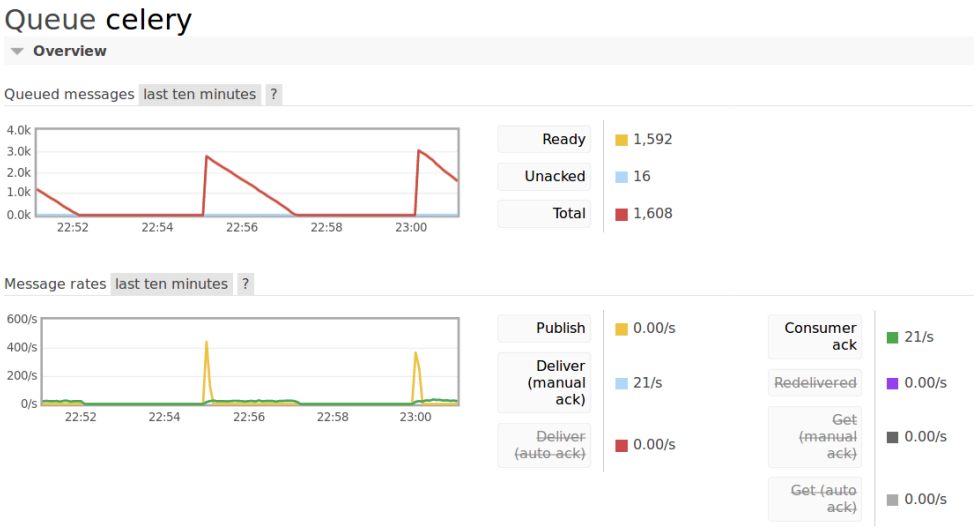

Figura 23: Panel de Control de RabbitMQ mostrando la cola que usa Celery al momento de saturación del sistema

server	condenser	processor (BEAT)	processor (WORKER)[1]	API	Usuarios
1	1	1	1 (1 min)	1	1800
1	1	1	2 (1 min)	1	1800
1	1	1	1 (5 min)	1	1800
1	2	1	1 (5 min)	1	1900
1	3	1	1 (5 min)	1	2200

Cuadro 3: Configuraciones usadas para la prueba de carga

Referencias

[1] J. Postel, "User datagram protocol," Internet Engineering Task Force, Tech. Rep., aug 1980. [Online]. Available: https://www.ietf.org/rfc/rfc768.txt

[2] D. Mills, "Network time protocol (NTP)," Internet Engineering Task Force, Tech. Rep., sep 1985. [Online]. Available: https://tools.ietf.org/html/rfc958

[3] M. E. Crovella and A. Bestavros, "Self-similarity in world wide web traffic: evidence and possible causes," *IEEE/ACM Transactions on networking*, vol. 5, no. 6, pp. 835–846, 1997.

[4] W. E. Leland, M. S. Taqqu, W. Willinger, and D. V. Wilson, "On the self-similar nature of ethernet traffic (extended version)," *IEEE/ACM Transactions on networking*, vol. 2, no. 1, pp. 1–15, 1994.

[5] Oracle, "Java Language Documentation." [Online]. Available: https://docs.oracle.com/en/java/

[6] R. Johnson, J. Hoeller, K. Donald, C. Sampaleanu, R. Harrop, A. Arendsen, T. Risberg, D. Davison, D. Kopylenko, M. Pollack, T. Templier, E. Vervaet, P. Tung, B. Hale, A. Colyer, J. Lewis, C. Leau, M. Fisher, S. Brannen, R. Laddad, A. Poutsma, C. Beams, T. Abedrabbo, A. Clement, D. Syer, O. Gierke, and R. Stoyanchev, "Spring Framework Documentation." [Online]. Available: https://docs.spring.io/spring/docs/3.1.1.RELEASE/spring-framework-reference/html/

[7] Red Hat JBoss Middleware, "Hibernate ORM Documentation." [Online]. Available: http://hibernate.org/orm/documentation/

[8] Apache Software Foundation, "Apache Wicket Framework Documentation." [Online]. Available: https://ci.apache.org/projects/wicket/apidocs/1.5.x/index.html

[9] R. Mordani, Oracle, N. Abramson, Apache Software Foundation Art Technology Group Inc.(ATG), K. Avedal, BEA Systems, H. Bergsten, Boeing, Borland Software Corporation, Developmentor, J. Hunter, IBM, InterX PLC, R. Johnson, Lutris Technologies, New Atlanta Communications, LLC, Novell, Inc., Persistence Software, Inc., Pramati Technologies Progress Software, SAS Institute, Inc., Sun Microsystems, Inc., Sybase, and G. Wilkins, "JavaTM Servlet 2.4 Specification," Java Community Process, techreport 154, Nov. 2003. [Online]. Available: https://jcp.org/en/jsr/detail?id=154

[10] Python Software Foundation, "Python 2 Language Documentation." [Online]. Available: https://docs.python.org/2/

[11] R Development Core Team, "The R Language Manuals." [Online]. Available: https://cran.r-project.org/manuals.html

[12] J. Buck and L. Hambley, “Capistrano Documentation.” [Online]. Available: http://capistranorb.com/

[13] Y. Matsumoto, “Ruby Language Documentation.” [Online]. Available: https://www.ruby-lang.org/en/documentation/

[14] The Open Group and IEEE Std, “crontab,” The Open Group, techreport, 2001. [Online]. Available: http://pubs.opengroup.org/onlinepubs/9699919799/utilities/crontab.html

[15] R. T. Fielding, “Architectural styles and the design of network-based software architectures,” phdthesis, University of California, 2000. [Online]. Available: http://www.ics.uci.edu/~fielding/pubs/dissertation/top.htm

[16] Travis-CI Community, “Travis-CI.” [Online]. Available: http://travis-ci.org/

[17] Docker, Inc., “Docker.” [Online]. Available: https://www.docker.com/

[18] ——, “Dockerhub.” [Online]. Available: https://hub.docker.com/

[19] ——, “Docker Compose Documentation.” [Online]. Available: https://docs.docker.com/compose/

[20] Apache Software Foundation, “Apache Maven.” [Online]. Available: https://maven.apache.org/

[21] T. Lindholm, F. Yellin, G. Bracha, and A. Buckley, *The Java Virtual Machine Specification, Java SE 8 Edition*, A.-W. Professional, Ed. Addison-Wesley Professional, 2014. [Online]. Available: https://docs.oracle.com/javase/specs/jvms/se8/html/index.html

[22] Facebook, Instagram, and Community, “React framework (reactjs).” [Online]. Available: https://reactjs.org/

[23] T. Koppers, S. Larkin, J. Ewald, J. Vepsäläinen, K. Kluskens, and W. contributors, “Webpack.” [Online]. Available: https://webpack.js.org/

[24] M. Jones, J. Bradley, and N. Sakimura, “JSON web token (JWT),” IETF, Tech. Rep., may 2015. [Online]. Available: https://tools.ietf.org/html/rfc7519

[25] E. R. Fielding and E. J. Reschke, “Hypertext transfer protocol (HTTP/1.1): Authentication,” IETF, Tech. Rep., jun 2014. [Online]. Available: https://tools.ietf.org/html/rfc7235

[26] Netty Project Community, “Netty.” [Online]. Available: https://netty.io/

[27] P. Webb, D. Syer, J. Long, S. Nicoll, R. Winch, A. Wilkinson, M. Overdijk, C. Dupuis, S. Deleuze, and M. Simons, “Spring Boot Documentation,” 2017. [Online]. Available: https://docs.spring.io/spring-boot/docs/current/reference/htmlsingle/

[28] The Scipy Community, "NumPy Reference Guide." [Online]. Available: https://docs.scipy.org/doc/numpy-1.13.0/reference/

[29] ——, "Scipy reference guide." [Online]. Available: https://docs.scipy.org/doc/scipy-1.0.0/reference/

[30] J. D. Hunter, M. Droettboom, T. A. Caswell, and The Matplotlib Community, "Matplotlib user's guide." [Online]. Available: http://matplotlib.org/users/index.html

[31] The scikit-learn Community, "scikit-learn documentation." [Online]. Available: http://scikit-learn.org/stable/documentation.html

[32] Jupyter Team, "Jupyter documentation." [Online]. Available: https://jupyter.readthedocs.io/en/latest/contents.html

[33] A. Solem, "Celery user manual." [Online]. Available: http://docs.celeryproject.org/en/latest/

[34] Pivotal, "Rabbitmq documentation." [Online]. Available: https://www.rabbitmq.com/documentation.html

[35] J. Postel, "Transmission control protocol," DARPA Internet Program, Tech. Rep., sep 1981. [Online]. Available: https://tools.ietf.org/html/rfc793

[36] J. Nagle, "Congestion control in IP/TCP internetworks," Internet Engineering Task Force, Tech. Rep., jan 1984. [Online]. Available: https://tools.ietf.org/html/rfc896

[37] F. Yergeau, "UTF-8, a transformation format of ISO 10646," Internet Engineering Task Force, Tech. Rep., nov 2003. [Online]. Available: https://tools.ietf.org/html/rfc3629

[38] R. L. Rivest, A. Shamir, and L. M. Adleman, "Cryptographic communications system and method," United States Patent Patent 4,405,829, 1983. [Online]. Available: http://patft.uspto.gov/netacgi/nph-Parser?Sect1=PTO1&Sect2=HITOFF&d=PALL&p=1&u=%2Fnetahtml%2FPTO%2Fsrchnum.htm&r=1&f=G&l=50&s1=4405829.PN.&OS=PN/4405829&RS=PN/4405829

[39] M. Cooper, Y. Dzambasow, P. Hesse, S. Joseph, and R. Nicholas, "Internet x.509 public key infrastructure: Certification path building," Internet Engineering Task Force, Tech. Rep., sep 2005. [Online]. Available: https://tools.ietf.org/html/rfc4158

[40] Q. H. Dang, "Secure hash standard," National Institute Of Standards and Technology, Tech. Rep., jul 2015. [Online]. Available: http://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.180-4.pdf

[41] B. Kaliski, "PKCS #1: RSA encryption version 1.5," Internet Engineering Task Force, Tech. Rep., mar 1998. [Online]. Available: https://tools.ietf.org/html/rfc2437

[42] S. Aiyagari, A. Richardson, M. Arrott, M. Ritchie, M. Atwell, S. Sadjadi, J. Brome, R. Schloming, A. Conway, S. Shaw, R. Godfrey, M. Sustrik, R. Greig, C. Trieloff, P. Hintjens, K. van der Riet, J. O'Hara, S. Vinoski, and M. Radestock, "Advanced message queuing protocol," , techreport, 2008. [Online]. Available: https://www.rabbitmq.com/resources/specs/amqp0-9-1.pdf

[43] T. Bray, "The JavaScript object notation (JSON) data interchange format," Internet Engineering Task Force, Tech. Rep., mar 2014. [Online]. Available: https://tools.ietf.org/html/rfc7159

[44] F. Wasilewski, "Pywavelets documentation." [Online]. Available: https://pywavelets.readthedocs.io/en/latest/#license

[45] Oracle, "Javafx." [Online]. Available: http://www.oracle.com/technetwork/java/javase/overview/javafx-overview-2158620.html

[46] ——, "Swing." [Online]. Available: https://docs.oracle.com/javase/7/docs/api/javax/swing/package-summary.html

[47] ——, "Fxml." [Online]. Available: https://docs.oracle.com/javafx/2/api/javafx/fxml/doc-files/introduction_to_fxml.html

[48] ——, "Java preferences api." [Online]. Available: https://docs.oracle.com/javase/8/docs/technotes/guides/preferences/index.html

[49] E. Syse and F. contributors, "Fxlauncher." [Online]. Available: https://github.com/edvin/fxlauncher

[50] Call-Em-All, "Material-ui." [Online]. Available: http://www.material-ui.com

[51] E. Rasmussen, "Redux." [Online]. Available: https://redux.js.org/

[52] ——, "Redux-form." [Online]. Available: https://redux-form.com

[53] J. y. c. Linux Foundation, "Node.js." [Online]. Available: https://nodejs.org

[54] I. y. c. StrongLoop, "Expressjs." [Online]. Available: https://expressjs.com/

[55] A. y colaboradores, "Passportjs." [Online]. Available: http://www.passportjs.org/

[56] T. G. y colaboradores, "Knexjs." [Online]. Available: http://knexjs.org/

[57] B. y colaboradores, "Bookshelfjs." [Online]. Available: http://bookshelfjs.org/

[58] R. y colaboradores, “Ramdajs.” [Online]. Available: http://ramdajs.com/

[59] Google, “Chrome v8.” [Online]. Available: https://developers.google.com/v8/

[60] I. Z. S. y colaboradores, “Npm.” [Online]. Available: https://www.npmjs.com/

[61] Azureus Software, Inc., “Vuze bittorrent client web page.” [Online]. Available: https://www.vuze.com/

[62] H. Hope, “inxi documentation.” [Online]. Available: https://smxi.org/docs/inxi-man.htm

[63] Apache Software Foundation, “Apache JMeter Documentation.” [Online]. Available: http://jmeter.apache.org/usermanual/index.html