Consideraciones para la verificación de ambientes de producción

Mtro. Alberto González Guízar

Dirección General de Cómputo y de Tecnologías de Información y Comunicación, UNAM

albertgg@unam.mx

Hoy en día los sistemas de información se han convertido en un actor clave dentro de la sociedad actual y una herramienta para cualquier organización que requiere procesar información para la toma de decisiones o brindar servicios. Es por ello que su disponibilidad toma relevancia y cualquier afectación impacta en los usuarios de los sistemas en la imagen, confianza, costos y cumplimiento normativo de las organizaciones.

Una buena práctica para la operación de un sistema de información es mantener separados los ambientes de trabajo: desarrollo, pruebas y producción, ya que cuando empieza a dar un servicio cualquier cambio en el código, configuración o ejecución de pruebas puede comprometer la disponibilidad e integridad de la información.

Es por ello que se deben revisar los ambientes donde operan los sistemas de información para evaluar la capacidad y seguridad que tienen para tomar las acciones necesarias que permitan garantizar la disponibilidad del servicio que ofrecerán. Todo ello debe realizarse antes de liberar el servicio y periódicamente verificar durante su operación.

Verificación de la capacidad

Respecto a la capacidad prevista para la operación deben considerarse la cantidad de usuarios que estarán trabajando simultáneamente, la cantidad de información que se estará incorporando o modificando y los requerimientos de las plataformas (sistema operativo, servidor web/aplicaciones, sistema manejador de bases de datos) que requerirá el sistema para funcionar.

Omitir los puntos anteriores puede ocasionar que el servicio sea deficiente para los usuarios debido a un tiempo de respuesta alto (el usuario lo percibe como lento) o no puede ejecutar algunos módulos por falta de recursos.

Se debe estimar por ello, la capacidad del hardware que soportará el servicio y definir la configuración más adecuada a la demanda que se tendrá de los recursos. Así por ejemplo, el utilizar determinado sistema operativo, de tipo Linux o Windows Server, requiere cierto tipo de arquitectura de procesador (procesador de 64 bits a 1.4 GHz para Linux CentOS 7 o Windows 2019 Server), memoria (2 GB RAM recomendado para ambos) y espacio en disco (32 GB para Windows 2019 Server o 40 GB para Linux CentOS 7). Lo anterior se puede revisar en la documentación técnica para la instalación del sistema operativo.

Cuando el servidor no cuenta con suficiente memoria para el sistema operativo y los procesos en ejecución, empieza a ocupar memoria virtual en disco (swap, espacio de intercambio), lo cual hace lenta la respuesta del equipo y puede llegar a darse de baja los procesos no esenciales, afectando la operación del sistema de información y su disponibilidad.

A lo anterior, se suma el requerimiento de memoria para el servidor web o de aplicaciones, por cada conexión realizada consumirá cierta cantidad. En Apache por ejemplo, el número de conexiones que estará atendiendo se define en el parámetro: *límite de conexiones simultáneas*. Si el número definido es menor a la cantidad de usuarios que se están intentando acceder o conectarse al mismo tiempo al servicio ofrecido por el sistema de información, empezará a rechazar nuevas conexiones indicando que se agotó el tiempo de espera con el servidor.

Una forma para calcular la cantidad de memoria que consume cada conexión al servidor Web, en este caso del Apache y sobre Linux, se puede utilizar el siguiente comando:

```
ps -ylC apache2 --sort:rss | awk '{SUM += $8; I += 1} END {print SUM/I/1024}'
```

Este comando dará como resultado la cantidad de memoria en MB acorde a las conexiones activas y el tipo de páginas a las que se accede. Se recomienda realizar la prueba con diferentes pestañas abiertas en el navegador y mostrando distintos contenidos a ser posible del sistema.

Otro elemento que influye en el dimensionamiento de la capacidad que debe tener el servidor físico o virtual es el sistema manejador de bases de datos, por lo regular suele ser un RDBMS (Relational Database Management System en inglés) como PostgreSQL, MySQL, Oracle, MS SQL Server, entre otros. Para lo cual se deben considerar de manera inicial, las características del procesador, la memoria necesaria para su funcionamiento y el espacio en disco necesario para la creación de la estructura de las bases de datos.

Posteriormente, es necesario considerar respecto al almacenamiento, el tamaño de la base de datos que ocupará el sistema de información (tamaño de la estructura y número de registros de datos iniciales) y el crecimiento de registros anual que tendrá.

A nivel de la memoria, cada conexión y consulta a la base de datos consume recursos. Por ejemplo, en PostgreSQL se generan aproximadamente 16 KB por conexión, por lo que es importante configurar adecuadamente el parámetro ***max_connections*** con relación a ***work_mem ,*** si se tienen demasiadas conexiones inactivas puede afectar el rendimiento del servidor.

Cada consulta que se ejecuta en el sistema manejador de bases de datos hace uso de memoria para construir los datos del resultado, por lo que una consulta mal construida puede hacer uso inadecuado de los recursos y afectar el rendimiento.

Para corroborar el estado y cómo son consumidos los recursos dentro del sistema manejador de bases de datos se pueden hacer uso de las herramientas que ofrezca. Por ejemplo, en MS SQL Server consulta las vistas del sistema: **sys.dm_os_sys_info** (proporciona información de memoria a nivel del servidor físico), **sys.dm_os_sys_memory** (proporciona información de la memoria actual del sistema) y **sys.dm_os_process_memory** (proporciona información para determinar si la memoria de proceso física o virtual de SQL Server es baja) o en su caso utilizar SQL Server Management Studio para monitorear la memoria consumida por objetos en la base u optimizar consultas SQL.

Uso de máquinas virtuales

Hacer uso de la virtualización permite a los administradores aprovechar mejor la capacidad de los servidores, ahorra energía y reduce el tiempo de inactividad, de recuperación o de instalación al hacer respaldos de las máquinas virtuales o clonar ambientes.

Para ello se pueden virtualizar los servidores físicos con el uso de herramientas (fig. 1) , algunas de las cuales son de licenciamiento comercial (por ejemplo: VMware o Virtualbox) y otras de licencia pública (por ejemplo: KVM-Qemu, Xen o Hyper-V).

Para poder utilizarla se requiere que el equipo soporte la tecnología de virtualización. En Intel recibe la denominación VT-x y en AMD conocida como AMD-V, además se aconseja disponer al menos de un servidor de mediana capacidad (8 núcleos, 64 GB RAM) para implementar mínimo 2 o 3 máquinas virtuales.

Fig. 1 Herramientas de virtualización para Linux

Verificación inicial del ambiente de producción

Otro punto importante de verificación en un ambiente de producción es la revisión de los puertos (fig.2) abiertos y los procesos en ejecución en el servidor, además de ser un riesgo para la seguridad. En el caso de los procesos pueden estar consumiendo recursos de forma innecesaria al no ser requeridos para la operación del sistema de información.

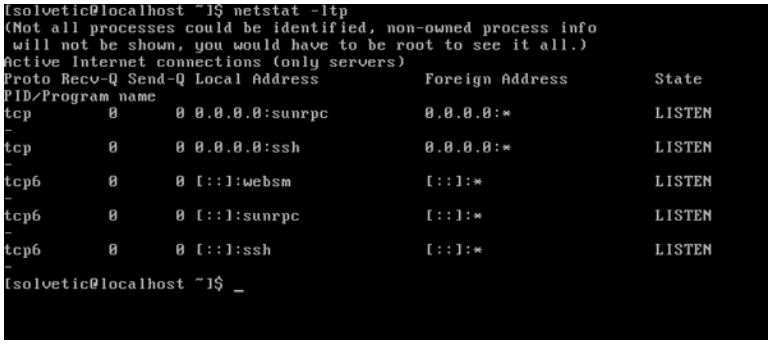


Fig. 2 Ejemplo de revisión de puertos activos TCP

Se recomienda eliminar los servicios en el sistema operativo que no son necesarios, cerrar los puertos sin ocupar, eliminar cuentas de usuarios de invitado, de prueba o que sean innecesarias, proteger las bitácoras (logs) contra su alteración y acceso no autorizado, cambiar los puertos y las contraseñas de las cuentas por defecto en los sistemas manejadores de bases de datos, así como eliminar las bases de datos de ejemplo.

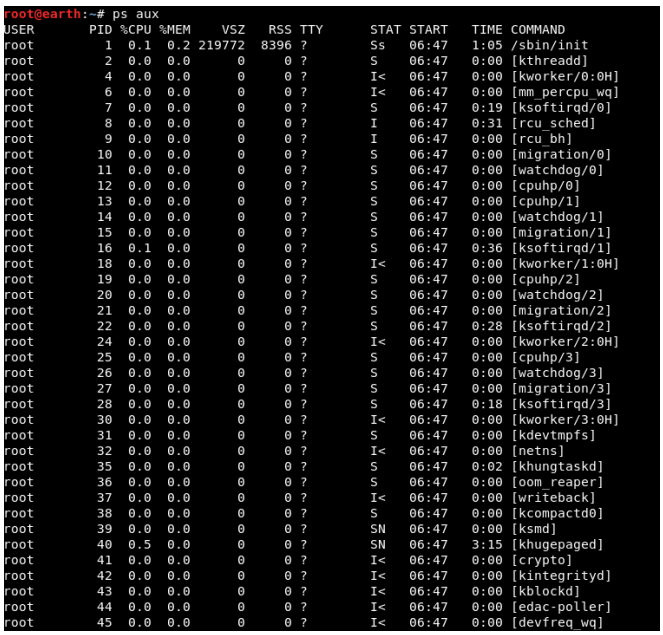
De igual manera es recomendable verificar que no se tengan vulnerabilidades críticas o de impacto alto en las versiones del sistema operativo, servidor web/aplicaciones, sistema manejador de bases de datos y plataforma de desarrollo utilizada (PHP Cake, Laravel, JDK, Spring, entre otras), para lo cual se pueden consultar base de datos de vulnerabilidades como CVE (Common Vulnerabilities and Exposures, https://cve.mitre.org/), Google Hacking Database (https://www.exploit-db.com/google-hacking-database) y National Vulnerability Database del NIST (https://nvd.nist.gov/), por citar algunas.

Como otro punto relevante a considerar, se recomienda utilizar las últimas versiones estables del software utilizado para el funcionamiento del aplicativo a desplegarse en el ambiente de producción.

Seguimiento a la operación del ambiente de producción

Es recomendable revisar regularmente los registros de eventos de las actividades de los usuarios, así como las excepciones, fallas y eventos de seguridad de la información para detectar algún problema con el aplicativo o con algún elemento de la plataforma que lo soporta.

De igual forma revisar de manera constante la utilización de los recursos (CPU, memoria, disco y/o red) en el servidor (máquina virtual) para detectar problemas en el rendimiento o capacidad. Se pueden utilizar comandos o herramientas que proporcionan los sistemas operativos, por ejemplo en Linux, se puede utilizar: ***ps aux*** o ***top -C***, en el caso de Windows Server se pueden usar las herramientas de monitoreo proporcionadas por Microsoft como son el administrador de aplicaciones o el monitor de rendimiento.



```
root@earth:~# ps aux
USER         PID %CPU %MEM    VSZ   RSS TTY      STAT START   TIME COMMAND
root           1  0.1  0.2 219772  8396 ?        Ss   06:47   1:05 /sbin/init
root           2  0.0  0.0      0     0 ?        S    06:47   0:00 [kthreadd]
root           4  0.0  0.0      0     0 ?        I<   06:47   0:00 [kworker/0:0H]
root           6  0.0  0.0      0     0 ?        I<   06:47   0:00 [mm_percpu_wq]
root           7  0.0  0.0      0     0 ?        S    06:47   0:19 [ksoftirqd/0]
root           8  0.0  0.0      0     0 ?        I    06:47   0:31 [rcu_sched]
root           9  0.0  0.0      0     0 ?        I    06:47   0:00 [rcu_bh]
root          10  0.0  0.0      0     0 ?        S    06:47   0:00 [migration/0]
root          11  0.0  0.0      0     0 ?        S    06:47   0:00 [watchdog/0]
root          12  0.0  0.0      0     0 ?        S    06:47   0:00 [cpuhp/0]
root          13  0.0  0.0      0     0 ?        S    06:47   0:00 [cpuhp/1]
root          14  0.0  0.0      0     0 ?        S    06:47   0:00 [watchdog/1]
root          15  0.0  0.0      0     0 ?        S    06:47   0:00 [migration/1]
root          16  0.1  0.0      0     0 ?        S    06:47   0:36 [ksoftirqd/1]
root          18  0.0  0.0      0     0 ?        I<   06:47   0:00 [kworker/1:0H]
root          19  0.0  0.0      0     0 ?        S    06:47   0:00 [cpuhp/2]
root          20  0.0  0.0      0     0 ?        S    06:47   0:00 [watchdog/2]
root          21  0.0  0.0      0     0 ?        S    06:47   0:00 [migration/2]
root          22  0.0  0.0      0     0 ?        S    06:47   0:28 [ksoftirqd/2]
root          24  0.0  0.0      0     0 ?        I<   06:47   0:00 [kworker/2:0H]
root          25  0.0  0.0      0     0 ?        S    06:47   0:00 [cpuhp/3]
root          26  0.0  0.0      0     0 ?        S    06:47   0:00 [watchdog/3]
root          27  0.0  0.0      0     0 ?        S    06:47   0:00 [migration/3]
root          28  0.0  0.0      0     0 ?        S    06:47   0:18 [ksoftirqd/3]
root          30  0.0  0.0      0     0 ?        I<   06:47   0:00 [kworker/3:0H]
root          31  0.0  0.0      0     0 ?        S    06:47   0:00 [kdevtmpfs]
root          32  0.0  0.0      0     0 ?        I<   06:47   0:00 [netns]
root          35  0.0  0.0      0     0 ?        S    06:47   0:02 [khungtaskd]
root          36  0.0  0.0      0     0 ?        S    06:47   0:00 [oom_reaper]
root          37  0.0  0.0      0     0 ?        I<   06:47   0:00 [writeback]
root          38  0.0  0.0      0     0 ?        S    06:47   0:00 [kcompactd0]
root          39  0.0  0.0      0     0 ?        SN   06:47   0:00 [ksmd]
root          40  0.5  0.0      0     0 ?        SN   06:47   3:15 [khugepaged]
root          41  0.0  0.0      0     0 ?        I<   06:47   0:00 [crypto]
root          42  0.0  0.0      0     0 ?        I<   06:47   0:00 [kintegrityd]
root          43  0.0  0.0      0     0 ?        I<   06:47   0:00 [kblockd]
root          44  0.0  0.0      0     0 ?        I<   06:47   0:00 [edac-poller]
root          45  0.0  0.0      0     0 ?        I<   06:47   0:00 [devfreq_wq]
```

Ejemplo de salida del comando ps aux

Es aconsejable definir valores en el consumo de los recursos que se consideran sanos para la operación del aplicativo y cuáles deben disparar una alerta que permita llevar acciones para detectar cualquier posible problema, efectuar su análisis y llevar a cabo su solución, antes de que empeore y represente una amenaza para la continuidad del servicio y este se vea comprometido. Por ejemplo, si el procesador o la memoria sobrepasa el 60% de utilización se deberá mandar una alerta a través de un mensaje vía SMS (Short Message Service) o correo electrónico.

De igual forma, es aconsejable revisar el uso de la memoria swap, si aumenta el consumo de esta nos puede indicar que existe algún problema, donde se ve afectado el rendimiento del servidor y por tanto del aplicativo. Cabe recordar que la memoria swap está en disco y por lo tanto es más lenta su lectura/escritura que en la memoria RAM.

A través de estas acciones se puede actuar de forma pro-activa en la solución de cualquier problema que comience y mantener la disponibilidad de nuestro ambiente de producción de acuerdo con los niveles de servicio comprometidos a los usuarios. Siempre será mejor tener una actitud pro-activa que reactiva, esta última podría llevarnos a interrumpir los servicios proporcionados por el ambiente y a la insatisfacción de nuestros usuarios, ocasionando una mala imagen de la organización en la que trabajamos.

Referencias

Alvarado Guillermo. (2018). Virtualización a fondo ¿Cuáles son las diferencias entre KVM y QEMU? ¿Son sinónimos?. Recuperado: 02 de diciembre de 2021, de Guillermo Alvarado Blog. URL: https://galvarado.com.mx/post/kvmqemu/

Martínez David. (2020). **Trucos para Mejorar el Rendimiento de Apache**. Recuperado: 02 de diciembre de 2021, de Puerto53.com. URL: https://puerto53.com/internet-y-aplicaciones/trucos-para-mejorar-el-rendimiento-de-apache/

MySQL Calculator. (2021) MySQL Memory Calculator. Recuperado: 02 de diciembre de 2021, de Precedence. URL: http://www.mysqlcalculator.com/

Portillo Sergio. (2020). Como detectar problemas de rendimiento causados por falta de memoria. 02 de diciembre de 2021, de Tutoriales IT Sitio web: https://tutorialesit.com/sql-server-como-detectar-problemas-de-rendimiento-causados-por-falta-de-memoria/

CRÉDITOS

DGTIC-UNAM

Elaborado por: Mtro. Alberto González Guízar

Revisión técnica:

Ing. José Othoniel Chamú Arias

Mtra. Susana Laura Corona Correa

L.I. Leticia Martínez Calixto

Mtro. Hugo Alonso Reyes Herrera

Revisión de estilo: Pamela Valdés Reséndiz

Autorización de publicación: Dra. Marcela J. Peñaloza Báez

Dirección de Colaboración y Vinculación, 2021