qualoom

Monitorización de la estabilidad de tu eCommerce ante escenarios de pedidos punta o compras masivas

Bienvenidos a la segunda parte de nuestra entrega sobre Pedidos punta que aborda todos aquellos aspectos técnicos y organizativos a tener en cuenta a la hora de preparar nuestro e-commerce para afrontar momentos críticos de tráfico de usuarios, accesos masivos y operaciones de compra claves en determinados momentos del año.

En esta segunda entrega profundizaremos en todas aquellas buenas prácticas relativas a la monitorización de sistemas de comercio electrónico que permiten dotar a los administradores de TI de las herramientas necesarias para la depuración y remediación de manera eficaz y rápidamente de potenciales incidencias y anomalías, algo crítico durante campañas o promociones con alto tráfico de usuarios y pedidos.

1/ Métricas de rendimiento

Aunque hay algunas excepciones, la gran mayoría de sistemas de comercio electrónico comparten una arquitectura básica de tres capas:

1. Servidores web
2. Servidores de aplicación
3. Servidores de base de datos

Los sitios de mayor popularidad y tráfico suelen contar, adicionalmente, con sistemas de balance de carga, caché en memoria almacenamiento compartido y CDN para mejorar la experiencia de usuario y reducir la carga de los servidores, redundando habitualmente en una reducción en sus costes operativos.

En todos estos servicios y piezas de infraestructura, existen 5 KPIs que, de forma transversal, conviene tener debidamente monitorizados y alarmados:

- Uso de CPU
  - → Estadístico de mayor utilidad: PROMEDIO
  - → Aunque depende en gran medida de la implementación de cada aplicación, como punto de partida es razonable asumir que un sistema con un uso sostenido de CPU por encima del 75% es susceptible de estar entrando en un régimen de sobrecarga y penalizando los tiempos de respuesta que experimenta el usuario final.
  - → Cabe destacar que algunas implementaciones no están preparadas para aprovechar múltiples procesadores; por ejemplo, Redis es una de las bases de datos en memoria más populares y su implementación es estrictamente mono-proceso. En estos casos es importante ponderar los umbrales de alarmado de acuerdo al uso real que hacen las aplicaciones de los recursos hardware.
  - → Por continuar con el ejemplo de Redis, una alarma por consumo del 75% de CPU no tiene sentido en un sistema con más de una CPU y debería dividirse por el número de procesadores lógicos.
  - → Es importante destacar también que la mayoría de los proveedores cloud no se refieren en sus tarifas a núcleos físicos sino a núcleos de hyperthreading. Así, un servidor de 4 vCPUs realmente cuenta con solo 2 núcleos físicos y un total de 4 núcleos de hyperthreading. En ciertas implementaciones esto puede suponer una utilización subóptima de los recursos dependiendo del soporte que tengan las aplicaciones subyacentes para ejecución multi-hilo o multi-proceso, y las alarmas deben ajustarse según cada escenario.

- Uso de memoria

  → Estadístico de mayor utilidad: MÁXIMO

  → El principal problema con la memoria de un sistema es que, una vez agotada, el rendimiento se desploma de forma instantánea, incluso incurriendo en una caída completa del sistema. Es por esto que se debe tener un control estricto sobre el uso de este recurso y alarmar al equipo correspondiente cuando se experimente una utilización anormalmente elevada.

  → Los servidores web no suelen consumir grandes cantidades de memoria y, habitualmente, puede confiarse en que un uso por encima del 75% se trate de una anomalía que requiera ser investigada. Por supuesto, esto depende de la dotación de recursos del mismo.

  → Los servidores de aplicación, por su parte, sí suelen tener una tendencia de consumo elevado de memoria. Por suerte, la mayoría de los servidores de aplicaciones comerciales permiten ajustar el número de procesos concurrentes y la asignación máxima de memoria por proceso, facilitando a los administradores garantizar que un servidor sobrecargado no incurre en un consumo excesivo de memoria. Por desgracia, la práctica habitual es la de hacer un ligero sobredimensionamiento de estas características para permitir una mayor concurrencia de procesos ligeros y asumir que los procesos pesados serán puntuales y de poco riesgo. En estos casos, un punto de partida razonable es alarmar el sistema cuando el uso sobrepase el 85-90% de la memoria del sistema.

  → Los servidores de bases de datos implementan, en su inmensa mayoría, mecanismos de control de recursos que, debidamente optimizados, permiten mantener el uso de memoria estable en torno a un margen reducido. Es por esto que suele ser habitual que las alarmas en estos sistemas se disparan a partir del 90-95%.

- Espacio libre en disco

  → Estadístico de mayor utilidad: MÍNIMO

  → Todos los sistemas requieren de cierto margen para guardar datos de forma persistente o temporal, como Logs, ficheros temporales, tokens de sesión, servicios de datos, etc.

  → El agotar este espacio de intercambio/almacenamiento supone, normalmente, detener el procesamiento de nuevas cargas de trabajo. Un punto de partida razonable suele ser elevar una notificación cuando el espacio libre baja del 10% del total, siempre asumiendo que cada pieza de la infraestructura tiene un almacenamiento debidamente dimensionado para su función.

- Load Average

  → Estadístico de mayor utilidad: PROMEDIO

  → Esta métrica solo está disponible en sistemas Linux, y permite identificar el número de procesos en espera de recursos de CPU. Suele ser determinante a la hora de identificar cuellos de botella provocados por sistemas externos o de entrada/salida, ya que estos pueden provocar que los procesos productivos se queden bloqueados a la espera de contestación sin consumir recursos del sistema en que ejecutan.

  → También suele evaluarse en función del número de núcleos de procesador disponibles, y alarmando cuando el número de procesos bloqueados supera el número de procesadores.

- IOPS

  → Estadístico de mayor utilidad: PROMEDIO

  → Este estadístico suele ser de vital importancia en servicios de almacenamiento, principalmente en productos de base de datos relacional. Sobrecargar el subsistema de almacenamiento solicitando un número de operaciones mayor del que este es capaz de ejecutar redundará irremediablemente en esperas por entrada/salida, y con ello penalizando la experiencia de usuario.

  → En este caso, alarmar en cuanto se detecte un pico de IOPS suele resultar en falsos positivos, ya que los servicios de almacenamiento suelen tener tendencia a leer datos de disco en bloques lo más grandes posibles para reducir el número de veces que se requiere bajar a disco. Por tanto, lo habitual en este caso es enviar una notificación cuando el volumen de IOPS supere el 90% de la capacidad del sistema de forma sostenida durante un tiempo razonable dependiendo de cada aplicación.

A continuación, detallamos otros KPIs más específicos, pero no por ello menos importantes.

- Balanceadores de carga

  - Tiempo de respuesta promedio.
    - Estadístico de mayor utilidad: PROMEDIO
    - Prácticamente la totalidad de servicios y productos de balanceo de carga proporcionan esta métrica. Algunos incluso permiten discernir entre el tiempo que tarda el propio balanceador en procesar la petición y lo que tarda el backend en devolver el contenido.
    - En sitios de comercio electrónico es habitual que los tiempos oscilen en la horquilla de los 200-300ms, en promedio, con lo que superar un tiempo de respuesta promedio de 500ms-1000ms suele ser indicativo claro de que hay un problema que requiere de investigación inmediata.
  - Volumen de peticiones y conexiones activas
    - Estadístico de mayor utilidad: SUM
    - En este caso, el alarmado se hace complicado por ser unos KPIs que fluctúan recurrentemente en función del tráfico. Es por ello que lo recomendable es utilizar sistemas de detección de anomalías, si la solución de monitorización lo permite. Este tipo de alarmado utiliza algoritmos de Machine Learning para identificar picos inesperados en la volumetría de peticiones.
  - Tasa de errores HTTP 4xx y 5xx
    - Estadístico de mayor utilidad: SUM
    - Si analizamos el tráfico de prácticamente cualquier e-commerce, observaremos que la inmensa mayoría de peticiones se responden con códigos de éxito o HTTP 2xx. Normalmente las tasas de errores de cliente (4xx) y errores de servidor (5xx) suelen estar en rangos bastante contenidos y estables.
    - El alarmado en este caso depende en gran medida de cada producto de comercio electrónico y la naturaleza del tráfico. Si no existe ninguna información previa, lo razonable es notificar a los administradores en cuanto se produce alguno de estos eventos e ir ajustando el umbral en función de la actividad diaria.

- Servidores de backend disponibles
  - Estadístico de mayor utilidad: SUM
  - Los servicios y productos de balance de carga habitualmente permiten implementar mecanismos de comprobación ligera de estado para asegurarse de que sus servidores de backend están sanos, y reenviar tráfico solo a aquellos nodos que respondan correctamente a dicha comprobación. Asimismo, suelen ofrecer métricas con el número de servidores de backend tanto caídos como vivos. Es de vital importancia notificar a los administradores cuando se produce un fallo de comprobación de estado, ya que el sistema puede entrar en sobrecarga con facilidad al haber menor capacidad disponible mientras dure la incidencia.

- Servidores de bases de datos
  - Número de conexiones activas.
    - Estadístico de mayor utilidad: PROMEDIO
    - Lo normal en cualquier aplicación de comercio electrónico es que el número de conexiones activas en base de datos sea bastante estable. Una acumulación de conexiones en la base de datos suele ser un indicador de congestión por bloqueo de tablas, IO a disco o interbloqueo de queries mal optimizadas. El umbral de alarmado dependerá del comportamiento de cada aplicación, pero es recomendable que se notifique en cualquier caso que se supere el 50% del pool de conexiones global del servidor.

- Servidores de cache compartida en memoria
  - Tasa de acierto/fallo
    - Estadístico de mayor utilidad: PROMEDIO
    - Los productos y servicios de esta naturaleza ofrecen, casi en su totalidad, estadísticas de acierto y fallo de las operaciones en cache. Es normal que se produzcan eventos de fallo de forma recurrente, pero es raro que la tasa de fallo sea inferior al 50%, ya que esto indica que se están realizando una cantidad anormalmente elevada de viajes a la cache de forma innecesaria.
  - Expulsiones de la cache
    - Estadístico de mayor utilidad: SUM
    - Cuando la memoria asignada a la cache se agota, se ejecuta un algoritmo de selección de candidatos para su expulsión de la caché y con ello poder insertar el último objeto solicitado. Cuando se llega a este punto se producen dos problemas que penalizan a la

operación: Por un lado, los servicios de caché están invirtiendo tiempo innecesariamente en la ejecución del algoritmo de mantenimiento de cache, y por otro las aplicaciones no pueden guardar suficientes datos como realmente necesitan, requiriéndote de mayor cantidad de operaciones pesadas.

- Este KPI suele ser consecuencia directa del agotamiento de memoria del servidor, pero no siempre es así ya que las expulsiones se realizan en base a la memoria lógica asignada al servicio de cache y no a la memoria del sistema. Por eso, es recomendable monitorizar esta métrica de forma adicional y alarmar tan pronto se produzcan eventos de este tipo.

Por último, cabe destacar que la mayoría de los proveedores de CDN y WAF como Akamai, Cloudflare, Amazon Cloudfront o Azure CDN proporcionan un gran número de métricas de diversa utilidad, que pueden ayudar significativamente en la identificación de problemas. A continuación, recopilamos algunas de las más comunes:

- Volumen de errores HTTP 4xx/5xx
- Porcentaje de tráfico de bots
- Volumetría de tráfico por origen geográfico
- Tasas de baneo por regla de WAF
- Tiempos de respuesta del origen de datos
- Tasas de acierto en la cache de borde

2/ Analítica de logs

A continuación, haremos un repaso de los principales logs emitidos por la gran mayoría de infraestructuras de comercio electrónico, y comentaremos la utilidad de tener un sistema de recolección y analítica de logs centralizado que permita explotar y correlar esta información:

2.1 / Logs de acceso

Tanto los servidores web como la mayoría de los productos y servicios de balanceo de carga y CDN permiten generar logs de acceso. Esta información es de vital importancia para la depuración de errores y detección de patrones en la actividad de los usuarios, habilitando a los administradores de la plataforma para realizar consultas y correlaciones complejas en función de numerosas métricas de capa de aplicación como, por ejemplo:

- IPs de cliente
- Cabecera HTTP Host
- URL
- Parámetros de URL
- User-Agent
- Códigos y tiempos de respuesta por petición
- Tamaño de las peticiones y respuestas
- Protocolo de conexión
- Etc

Disponer de un buen sistema de analítica de logs de acceso permite depurar de forma sencilla el comportamiento de las aplicaciones cliente, identificar posibles ataques, generar informes para equipos de soporte de terceros, preparar la infraestructura para cambios futuros y auditar el cumplimiento de estándares de seguridad.

Algunos sistemas de analítica de logs, como ELK y Amazon Cloudwatch, permiten emitir notificaciones por volumetría de eventos de respuesta en una consulta compleja. Por ejemplo, se podría emitir una alarma cuando la latencia de la URL de checkout en concreto sobrepase cierto umbral. Estas URLs suelen tener un tiempo de respuesta promedio muy superior del resto del sitio web, a la vez que suelen ser las más propensas a tener errores y retrasos de ejecución por ser las más pesadas y que más dependencias tienen (por ejemplo, pasarelas de pago de terceros o sistemas de control de stocks). Este tipo de alarmado permite tener una mayor observabilidad del rendimiento de estas peticiones en concreto, sin tener que restringirse a la granularidad de las métricas generales anteriormente mencionadas.

2.2 / Logs de errores: Servidores web y servidores de aplicaciones

Recopilar los eventos de error de los productos de TI permitirá a los administradores trazar problemas normalmente relacionados con permisos, dependencias de software y utilización de recursos. Algunos de los eventos que suelen observarse en estos logs son:

- Problemas de permisos en el sistema de ficheros
- Inconsistencias en versiones de dependencias (por ejemplo, módulos PHP o librerías de NodeJS)
- Sobreutilización de memoria o buffers.
- Errores de configuración.
- Etc

Habitualmente estos errores se traducen en eventos 4xx y 5xx en el log de acceso, lo que permite cierto grado de correlación rápida de eventos en el caso de tener ambos logs disponibles.

2.3 / Logs de aplicación

En los logs emitidos por las aplicaciones es donde suelen encontrarse detalles de los problemas de tiempo de ejecución y donde suele requerirse del equipo de desarrollo para su análisis óptimo. En este tipo de logs podemos encontrar eventos muy variados a la vez que más cercanos al negocio.

- Errores de autenticación de usuarios
- Errores de conexión con servicios de datos
- Errores de interoperabilidad con pasarelas de pago y otros servicios de terceros
- Trazas personalizadas de los desarrolladores
- Eventos de depuración y error en tareas programadas
- Etc

De nuevo, es habitual que eventos de error en esta capa resulten en respuestas con código de respuesta HTTP 5xx, habilitando de nuevo posibilidad de correlar eventos entre capas de la aplicación.

3/ Auditoría de seguridad

Independientemente de las necesidades de cumplimiento de normativas de seguridad, es altamente recomendable emitir y recolectar logs de auditoría de todas las capas de la infraestructura para poder garantizar la seguridad del sistema. Se recomienda disponer, al menos, de la siguiente información:

- Auditoría de sistema operativo (Logins SSH y acciones administrativas).
- Auditoría de base de datos.
- Eventos del servicio de WAF, si existe.

Adicionalmente, es recomendable que las aplicaciones emitan logs de auditoría de acciones sobre las interfaces administrativas de la aplicación. Sin embargo, no todos los productos comerciales ofrecen esta posibilidad.

Disponer de estos datos permitirá a los administradores identificar vulnerabilidades y posibles agujeros de seguridad aprovechados por atacantes o usuarios maliciosos, y facilitando la implementación de medidas correctivas.

3.1 / Comprobaciones de estado

Hasta ahora hemos comentado las distintas herramientas de explotación de datos relevantes para informar al equipo de administración de posibles problemas. Sin embargo, se recomienda implementar, además de todas estas notificaciones, la ejecución de procesos de comprobación activa del estado de salud de cada uno de los componentes, y notificar tan pronto se detecte una anomalía en la respuesta. Por ejemplo:

- Pruebas de conectividad con ICMP
- Consultar periódicamente la home del sitio y verificar que se responde con un código de respuesta HTTP 200 y que la respuesta contiene un contenido concreto.
- Realizar conexiones autenticadas y consultas simples a la base de datos relacional
- Realizar conexiones que validen la salud de las interfaces administrativas (SSH o RDP) de los distintos servidores de la plataforma.
- Etc

Este tipo de comprobaciones de estado permite hacer una validación funcional end-to-end del estado de la plataforma, permitiendo identificar objetiva y rápidamente problemas operacionales de alta prioridad que, por ejemplo, en el caso de las alarmas por métricas de rendimiento, son en cierta medida susceptibles a la interpretación del equipo de administración.

3.2 / Instrumentación de aplicaciones

Este tipo de soluciones proporciona una gran cantidad de información de baja granularidad respecto al funcionamiento interno de las aplicaciones que pocas otras alternativas pueden proporcionar; el introducir marcas de tiempo dentro de la actividad de las aplicaciones permite identificar cuellos de botella en el código fuente de la aplicación y habilita la identificación rápida de consultas ineficientes que, de otra manera, podría resultar ciertamente complicado depurar.

Junto con los datos de trazabilidad y marcas de tiempo habituales, estas herramientas suelen recopilar de forma consolidada eventos de log, pilas de excepciones, métricas de rendimiento y multitud de metadatos que facilitan enormemente la depuración de problemas al equipo de desarrollo de forma relativamente rápida de implementar, motivo por el que estas soluciones suelen implementarse en empresas pequeñas que no cuentan con un departamento de TI consolidado.

Por contra, este tipo de herramientas suele requerir la contratación de servicios de terceros, instalación de agentes y librerías e incluso la modificación del código fuente, un peaje que muchas organizaciones no están dispuestas a asumir salvo en entornos no productivos o situaciones de emergencia. Sin embargo, a medida que la industria evolucione hacia infraestructuras de microservicios, veremos cómo este tipo de soluciones toma mayor relevancia y protagonismo ya que son las únicas que permiten dotar de una trazabilidad completa a las interacciones de un sistema de TI.