Desarrollo de Aplicaciones Web

Nerea Nuevo Pascual

1. Arquitectura básica de un servidor web.

Un servidor Web: distribuye la información a los clientes que la necesitan a través de una conexión a la red.
Una conexión a la red: es Internet, es decir, lo que permite ese intercambio de información.
Uno o varios usuarios: son los clientes que acceden a la aplicación con el objetivo de buscar la información necesaria.

2. URL: definición, sintaxis (estructura) y diferentes ejemplos.

Una URL es un localizador único de recursos que permite encontrar e identificar una página web o cualquier recurso llevándote directamente hacia él.

La estructura de una URL es la siguiente:

1 Protocolo: formato en el que se envían los datos de la página.
2 Sub-dominio: extensión que se añade al dominio.
3 Dominio: suele ser el nombre del servidor.
4 TLD: extensión de la página web.
5 Sub-carpetas: directorio al que accedemos.
6 Página: nombre del archivo que mostramos.
7 Etiqueta: lugar de la página donde nos posicionamos.

3. HTTP/HTTPS.

Características de HTTP / HTTPS:

Toda la comunicación entre los clientes y servidores se realiza a partir de caracteres US-ASCII de 7 bits.
Permite transferencia de objetos multimedia.
Existen una serie de métodos para que el cliente pueda dialogar con el servidor.
No mantiene estado. El servidor trata cada operación de manera independiente al resto.
Cada objeto al que se aplican los métodos está identificado con un URL único.

Métodos de HTTP / HTTPS:

GET: solicita una representación de un recurso específico.
HEAD: pide una respuesta idéntica a la de una petición GET, pero sin el cuerpo de la respuesta.
POST: envía una entidad a un recurso en específico y puede cambiar el estado del servidor
PUT: reemplaza todas las representaciones actuales del recurso de destino con la carga de la petición.
DELETE: borra un recurso en específico.
CONNECT: establece una conexión hacia el servidor identificado por el recurso.
OPTIONS: opciones de comunicación para el recurso de destino.

TRACE: realiza una prueba de bucle de retorno de mensaje a lo largo de la ruta al recurso de destino.
PATCH: es utilizado para aplicar modificaciones parciales a un recurso.

El HTTP y el HTTPS tienen las mismas características y métodos, la única diferencia es que en el HTTPS la información transmitida va cifrada.

4. Comparativa de diferentes navegadores.

	Versión instalada	Versión actual	Motor de renderización	Interprete de JavaScript	Empresa	Acceso a los elementos
FireFox	69.0	69.0	Gecko	SpiderMonkey	Mozilla	F12
IE	11.0	11.95	Trident	Chakra	Microsoft	F12
Opera	63.0	63.0	Blink	V8	Opera Sofware	Ctrl+Alt
Safari		12.2	WebKit	JS Core	Apple	Opc. desarrollo
Chrome	76	77	Blink	V8	Google	F12

5. Comparativa de servidores web más utilizados actualmente.

Apache

Es el web server de referencia para Internet. Nació en 1996 y hasta el día de hoy sigue vigente.
Entre sus ventajas encontramos que es código abierto, es además software gratuito, y multiplataforma (Windows, Linux y Unix).
Entre sus desventajas está su bajo rendimiento cuando se reciben miles de requests simultáneos en procesamiento de pedidos de contenido dinámico o archivos estáticos.

Nginx

Es un servidor web de código abierto y gratuito (aunque también existe una versión comercial) que se destaca por su alto rendimiento. Incluye además funciones como servidor proxy reverso HTTP, balanceador de carga, así como POP3 y IMAP. Está disponible para Windows, Linux y Unix.
Está diseñado para ofrecer un bajo uso de memoria y alta concurrencia. En lugar de crear nuevos procesos para cada solicitud web, Nginx usa un enfoque asincrónico basado en eventos donde las solicitudes se manejan en un solo hilo.
Con Nginx, un proceso maestro puede controlar múltiples procesos de trabajo. El proceso maestro mantiene los procesos de trabajo, y son estos lo que hacen el procesamiento real.

GWS

Google Web Server, se trata de un servidor web privado escrito en C++, que es utilizado por Google para la mayoría de su infraestructura web. Está basado en Linux y no disponible para el público.
Alberga aproximadamente un 10 % de todas las páginas web activas del mundo.

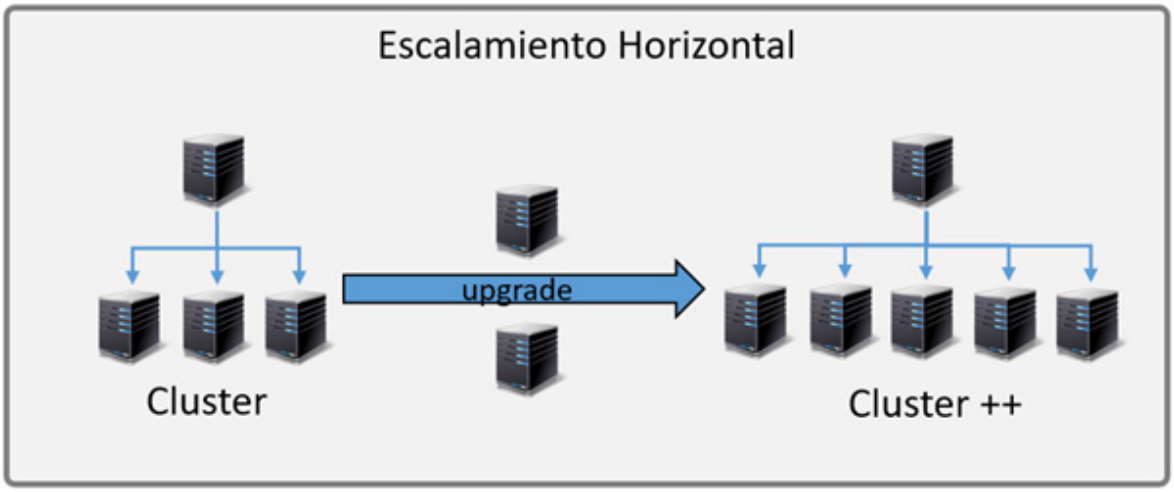
Microsoft IIS

Internet Information Services, también conocido como IIS, es un tipo de servidor web creado por Microsoft específicamente para su plataforma de sistemas operativos Windows. Tuvo su origen en el viejo «Option Pack» que corría en Windows NT, pero luego dada su creciente popularidad se integraría con Windows Server 2003, Windows Server 2008 y en posteriores ediciones.
Permite el procesamiento y despacho de páginas desarrolladas en tecnología ASP / ASP.NET, aunque también vale aclarar que sirve para interpretar páginas programadas en Perl o PHP.

No es sólo un servidor web, sino también una suite de servicios para la web, ya que ofrece también servicios de SMTP y FTP por ejemplo. Hoy se integra naturalmente con Microsoft Azure.
Tiene como gran desventaja que es un servidor web propietario exclusivo de Windows, y por lógica carece de integración para tantas tecnologías y lenguajes como otros servidores. Salvo uses ASP o ASP.NET con MSQL, siempre será mejor ir por Linux + Nginx o LiteSpeed.

6. Escalabilidad horizontal. Ventajas e inconvenientes.

El escalamiento horizontal es sin duda el más potente, pero también el más complicado. Este modelo implica tener varios servidores (conocidos como Nodos) trabajando como un todo. Se crea una red de servidores conocida como Cluster, con la finalidad de repartirse el trabajo entre todos nodos del cluster, cuando la performance del cluster se ve afectada con el incremento de usuarios, se añaden nuevos nodos al cluster, de esta forma a medida que son requeridos, más y más nodos son agregados al cluster.



Para que el escalamiento horizontal funcione deberá existir un servidor primario desde el cual se administra el cluster. Cada servidor del cluster deberá tener un software que permite integrarse al cluster, por ejemplo, para las aplicaciones Java, tenemos los servidores de aplicaciones como Weblogic, Widfly, Websphere, etc. y sobre estos se montan las aplicaciones que queremos escalar.

Ventajas:

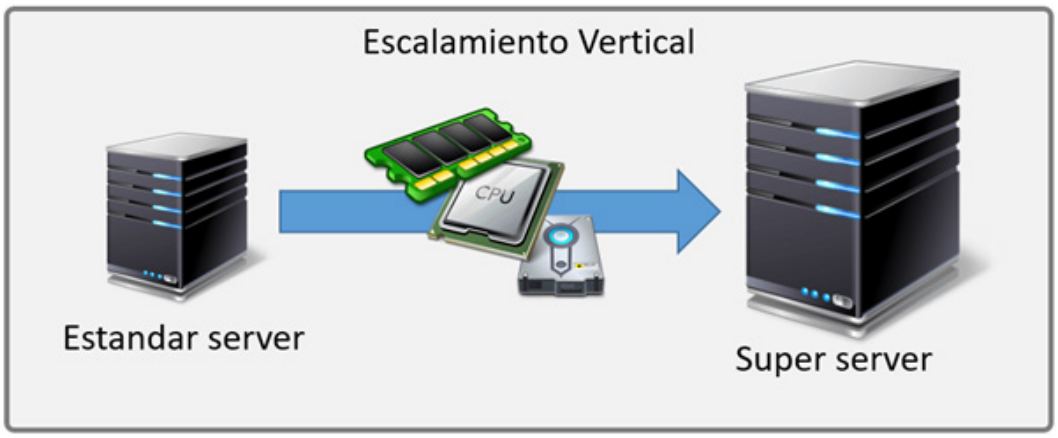
El crecimiento es prácticamente infinito, podríamos agregar cuantos servidores sean necesarios
Es posible combinarse con el escalamiento vertical.
Soporta la alta disponibilidad
Si un nodo falla, los demás sigue trabajando.
Soporta el balanceo de cargas.

Desventajas:

Requiere de mucho mantenimiento
Es difícil de configurar
Requiere de grandes cambios en las aplicaciones (si no fueron diseñadas para trabajar en cluster)
Requiere de una infraestructura más grande.

7. Escalabilidad vertical. Ventajas e inconvenientes.

La escalabilidad vertical o hacia arriba, este es el más simple, pues significa crecer el hardware de uno de los nodos, es decir aumentar el hardware por uno más potente, como disco duro, memoria, procesador, etc. pero también puede ser la migración completa del hardware por uno más potente. El esfuerzo de este crecimiento es mínimo, pues no tiene repercusiones en el software, ya que solo será respaldar y migrar los sistemas al nuevo hardware.



¿Bastante fácil no?, la realidad es que este tipo de escalamiento tiene algunos aspectos negativos, ya que nuestro crecimiento está ligado al hardware, y este; tarde o temprano tendrá un límite, llegara el momento que tengamos el mejor procesador, el mejor disco duro, la mejor memoria y no podamos crecer más o podríamos a lo mejor comprar el siguiente modelo de servidores que nos costara un ojo de la cara y el rendimiento solo mejorar un poco, lo que nos traerá el mismo problema el próximo año.

Ahora bien, no significa que este modelo de escalamiento sea malo, ya que lo podemos combinar con el escalamiento horizontal para obtener mejores resultados.

Ventajas:

No implica un gran problema para las aplicaciones, pues todo el cambio es sobre el hardware
Es mucho más fácil de implementar que el escalamiento horizontal.
Puede ser una solución rápida y económica (compara con modificar el software)

Desventajas:

El crecimiento está limitado por el hardware.
Una falla en el servidor implica que la aplicación se detenga.
No soporta la Alta disponibilidad.
Hacer un upgrade del hardware al máximo pues llegar a ser muy caro, ya que las partes más nuevas suelen ser caras con respecto al rendimiento de un modelo anterior.

8. Razona/relaciona entre la demanda de tu aplicación (número de usuarios que demanda tu aplicación) y el escalamiento horizontal y vertical.

La relación entre la demanda y ambos escalamientos es prácticamente similar porque cuantos más usuarios demanden y usen la aplicación más trabajo van a tener que hacer los/el nodo/s de la aplicación de manera que para evitar el exceso de trabajo en el escalamiento vertical se puede añadir más RAM, otro procesador, otra CPU… es decir, hacer que el nodo tenga más potencia, mientras que el escalamiento horizontal lo único que hay que hacer es añadir un o varios nodos a ese Clúster y con eso lograrán repartirse el trabajo.

9. ¿Qué es un cuello de botella en un sistema informático? Consecuencias que produce el cuello de botella o bottleneck.

Cualquier parte o componente que impida el paso fluido de la información requerida.
El cuello de botella puede darse en varias partes del PC, como pueden ser la RAM, el disco duro, en el procesador o en la tarjeta gráfica.

RAM: lo más habitual, un bajo rendimiento en general y los tirones y parones bruscos en juegos.

Procesador: es común sufrir pérdidas de rendimiento que pueden ir desde tirones hasta bloqueos o parones temporales.

Disco duro: tiempos de encendido y apagado del sistema muy altos, tiempos de carga demasiado lentos y problemas en la carga de texturas en juegos.

Tarjeta gráfica: Un rendimiento insuficiente en juegos incluso en calidades bajas es la señal más clara de que tenemos una tarjeta gráfica que no cubre con nuestras necesidades.

10. Comparativa uso de virtualización frente a contenedores.

Aunque ambas tecnologías tienen el mismo objetivo, aislar una aplicación de otros procesos y aplicaciones en el sistema host, ambas tienen enfoques bastante diferentes.

Máquinas virtuales: Como su nombre indica, este enfoque está mucho más involucrado en el alcance. Se basa en un hipervisor (por ejemplo, KVM, XEN) que emula una máquina física completa, asigna una cantidad deseada de memoria del sistema, núcleos de procesador y otros recursos como almacenamiento en disco, redes, complementos PCI, etc.

Contenedores: Los contenedores aíslan esencialmente una aplicación del host a través de varias técnicas, pero utilizan el mismo núcleo de sistemas host, procesos (por ejemplo, pila de red) para ejecutar la aplicación o VNFs.

La virtualización viene con muchas herramientas probadas a lo largo del tiempo, plataformas de gestión y orquestación, sondas virtuales, soluciones de infraestructura virtual hiperconvertidas y mucho más. La portabilidad y la interoperabilidad son las características que destacan frente a los contenedores.

Los contenedores ofrecen una mayor eficiencia de recursos y agilidad de servicio. Aunque no parezca mucho, abre la puerta a un modelo de microservicios que puede escalar más rápido y de manera más eficiente. Los contenedores de papel se ajustan más a las iniciativas de NFV/SDN y la industria se ha dado cuenta de que Kubernetes es uno de los proyectos de código abierto de más rápido crecimiento hasta la fecha.