Aislamiento de las cargas de trabajo a través de la fragmentación aleatoria

Colm MacCárthaigh

Aislamiento de las cargas de trabajo a través de la fragmentación aleatoria

En la actualidad, Amazon Route 53 aloja muchas de las empresas más grandes del mundo y de los sitios web más populares, pero sus comienzos fueron mucho más humildes.

Asumir el desafío de alojar DNS

No mucho después de que AWS comenzara a ofrecer sus servicios, los clientes dejaron en claro que querían contar con la posibilidad de utilizar los servicios Amazon Simple Storage Service (S3), Amazon CloudFront y Elastic Load Balancing desde la "raíz" del dominio, es decir, para nombres como "amazon.com", no solo para nombres como "www.amazon.com".

Esto puede parecer muy sencillo. Sin embargo, a causa de una decisión de diseño en el protocolo DNS que se tomó en la década de 1980, es más difícil de lo que parece. DNS cuenta con una característica llamada CNAME, que permite al propietario de un dominio delegar una parte de este a otro proveedor para su alojamiento, pero esto no funciona en el nivel superior o de raíz de un dominio. Para satisfacer las necesidades de nuestros clientes, tendríamos que alojar sus dominios. Cuando alojamos el dominio de un cliente, podemos proporcionar el conjunto actual de direcciones IP que correspondan a Amazon S3, Amazon CloudFront o Elastic Load Balancing. Estos servicios se expanden todo el tiempo y agregan direcciones IP constantemente, por lo que tampoco se trata de parámetros que el cliente pueda ingresar de manera simple como código rígido en la configuración del dominio.

Alojar DNS no es una tarea insignificante. Si hay algún problema con el DNS, toda una empresa puede quedar sin conexión. Sin embargo, una vez que identificamos esa necesidad, nos decidimos a satisfacerla a la manera típica de Amazon: con urgencia. Reunimos un pequeño equipo de ingenieros y nos pusimos manos a la obra.

Lidiar con los ataques DDOS

Consulte a cualquier proveedor de DNS cuál es su mayor desafío, y le dirá que es lidiar con los ataques de denegación de servicio distribuido (DDOS). El DNS se basa en el protocolo UDP, lo que implica que sus solicitudes se pueden falsificar en gran parte del indomable lejano oeste que es Internet. Como el DNS es también infraestructura fundamental, la combinación de estos factores lo convierte en un blanco atractivo para aquellos inescrupulosos que buscan extorsionar empresas, para quienes desean, por numerosas razones, causar interrupciones en el servicio utilizando "booters" y para los ocasionales molestos que no parecen darse cuenta de que están cometiendo un crimen grave con consecuencias reales para las personas. Sin importar cuál sea la razón, todos los días se producen miles de ataques DDOS contra dominios.

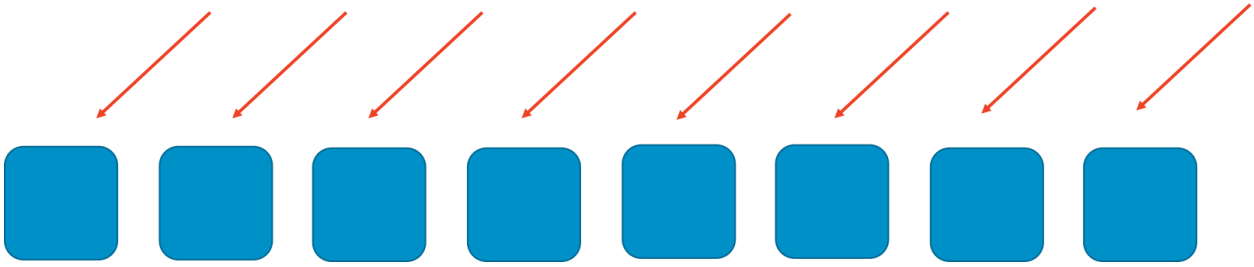
Una alternativa para mitigar estos ataques es utilizar enormes volúmenes de capacidad en el servidor. Aunque contar con una buena base de capacidad es importante, esta estrategia no se ajusta realmente al problema. Cada servidor que un proveedor agrega cuesta miles de dólares, mientras que, para agregar clientes falsos, los atacantes solo deben gastar centavos si utilizan una botnet controlada de manera remota. Para los proveedores, aumentar la capacidad del servidor con volúmenes enormes es una estrategia condenada al fracaso.

Cuando desarrollamos Amazon Route 53, lo más nuevo en mecanismos de defensa para DNS era utilizar dispositivos de red especializados que contaban con una variedad de trucos para filtrar el tráfico a gran velocidad. Teníamos varios de estos dispositivos para los servicios de DNS internos de Amazon y consultamos a proveedores de hardware sobre qué otras soluciones estaban disponibles. Nos dimos cuenta de que nos costaría decenas de millones de dólares comprar dispositivos suficientes para proteger cada uno de los dominios de Route 53 y que, además, nos atrasaríamos meses en nuestro cronograma esperando que nos los entregaran, los instaláramos y quedaran en funcionamiento. Esa idea no encajaba con nuestros planes de resolver el problema de forma urgente y ahorrativa, así que nunca la consideramos seriamente. Necesitábamos encontrar la forma de gastar recursos solamente en defender aquellos dominios que estuvieran bajo ataque. Así es que recurrimos al viejo dicho que enuncia que la necesidad agudiza el ingenio. La necesidad era desarrollar un servicio de DNS de primera categoría con un tiempo de actividad del 100 %, para el que solo se emplearían unos pocos recursos. Nuestro ingenio dio lugar a la fragmentación aleatoria.

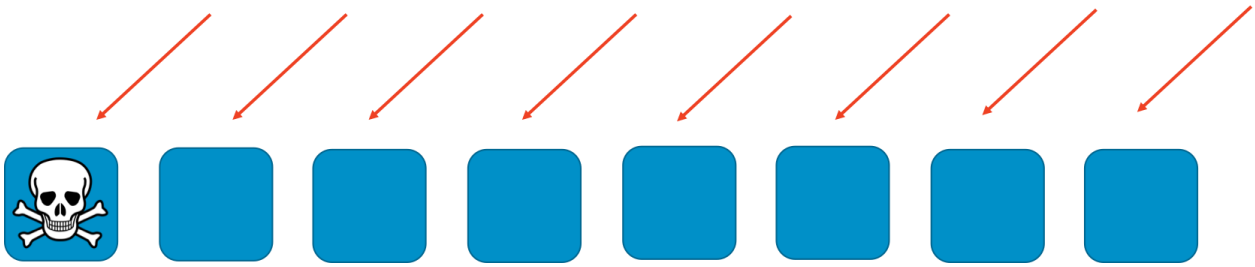
¿Qué es la fragmentación aleatoria?

Se trata de un procedimiento simple, pero potente. Es más potente todavía de lo que habíamos creído en un principio. Lo hemos utilizado una y otra vez, y se ha convertido en el patrón central que permite a AWS prestar al cliente servicios rentables diseñados para varios usuarios, pero que brindan la experiencia de exclusividad de los servicios diseñados para un único usuario.

Para entender cómo funciona la fragmentación aleatoria, piense primero en cómo la fragmentación común puede mejorar la escalabilidad y la resiliencia de un sistema. Imagine un sistema o un servicio capaz de escalado horizontal que está compuesto por ocho trabajadores. La imagen que sigue ilustra a los trabajadores y sus solicitudes. Los trabajadores podrían ser servidores, colas, bases de datos o lo que sea que componga su sistema.



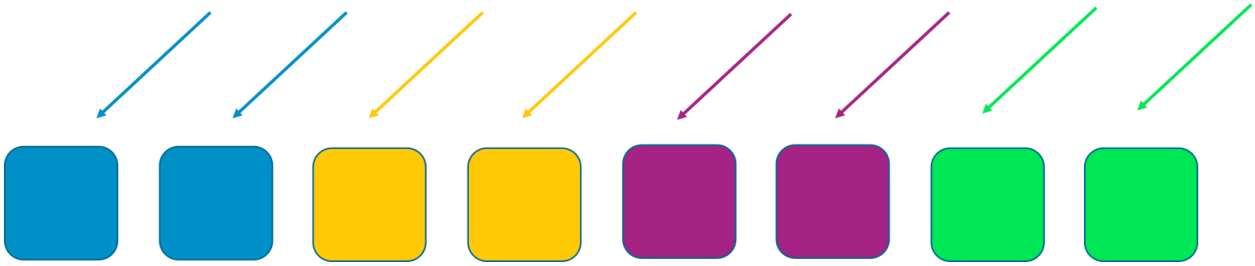
Sin ningún tipo de fragmentación, el grupo de trabajadores debe encargarse de todo el trabajo. Todos los trabajadores deben ser capaces de responder a cualquier tipo de solicitud. Esto es muy bueno en términos de eficiencia y redundancia. Si un trabajador falla, los otros siete pueden absorber el trabajo pendiente. De esta manera, el sistema necesita relativamente pocos recursos ociosos. Sin embargo, si se pueden activar errores con un tipo particular de solicitud o con una inundación de solicitudes, como en un ataque DDOS, surge un gran problema. Las siguientes dos imágenes ilustran un ataque de este tipo en curso.



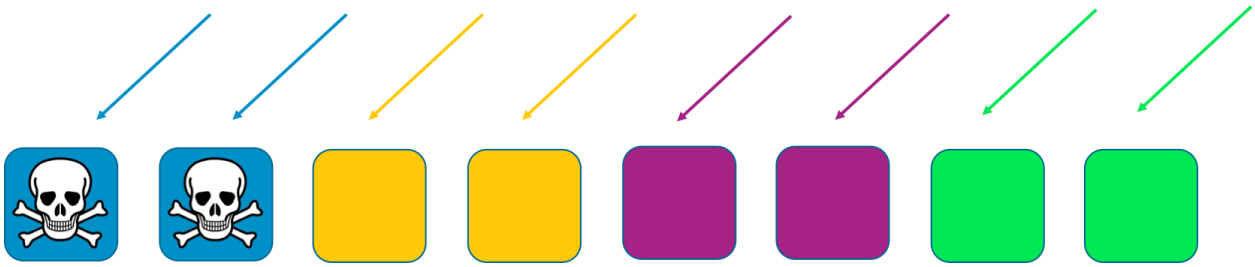
El problema eliminará al primer trabajador al que afecte y, cuando los trabajadores restantes retomen el trabajo pendiente, se producirá una reacción en cadena. Así, el problema puede eliminar muy rápidamente a todos los trabajadores y al servicio en su totalidad.

El alcance del impacto en estos casos es “a todo y a todos”. Todo el servicio se cae. Y se afecta a cada uno de los clientes. Como decimos los ingenieros en sistemas cuando hablamos de disponibilidad: “Esto no es óptimo”.

Podemos mejorar la situación con la fragmentación común. Si dividimos el grupo en 4 particiones de trabajadores, perdemos eficiencia, pero el alcance del impacto es menor. Las siguientes dos imágenes ilustran cómo la fragmentación puede limitar el impacto de un ataque DDOS.



En este ejemplo, cada partición contiene dos trabajadores. Repartimos los recursos, como los dominios de los clientes, entre las particiones. Todavía tenemos redundancia, pero como solo hay dos trabajadores por partición, necesitamos más capacidad ociosa para que se ocupe de los errores que pudieran ocurrir en el sistema. A cambio, obtenemos una reducción considerable en el alcance del impacto.

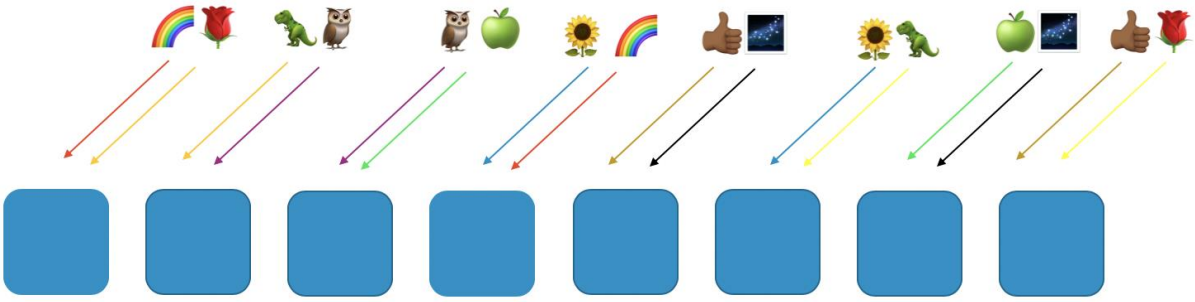


En este mundo fragmentado, el alcance del impacto se reduce mediante el aumento de la cantidad de particiones. Con cuatro particiones, si un cliente experimenta un problema, la partición que lo aloja se puede ver afectada, así como todos los demás clientes alojados en esa misma partición. Sin embargo, esa partición representa solo un cuarto de la totalidad del servicio. Un porcentaje de impacto al 25 % es mucho mejor que uno al 100 %. Con la fragmentación aleatoria, podemos mejorar todo de forma exponencial otra vez.

La fragmentación aleatoria permite crear particiones virtuales con dos trabajadores cada una, y podemos asignar a una de estas particiones virtuales nuestros clientes, nuestros recursos o cualquier componente que deseemos aislar.

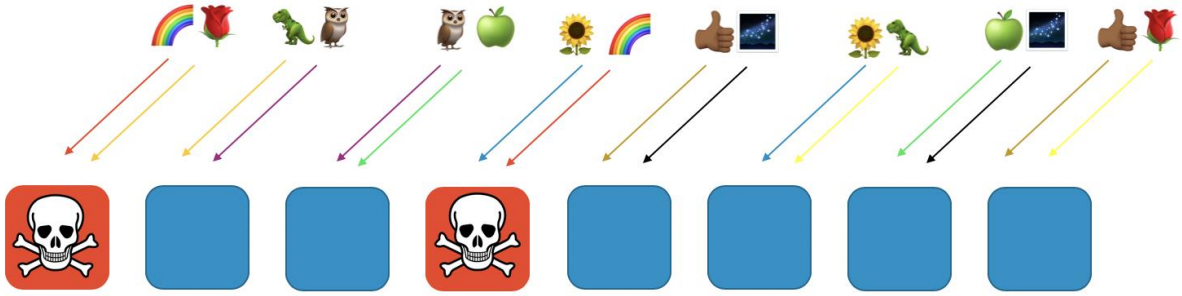
La siguiente imagen muestra un ejemplo de fragmentación aleatoria con ocho trabajadores y ocho clientes, cada uno asignado a dos trabajadores. Normalmente, tendríamos muchos más clientes que trabajadores, pero el ejemplo es más fácil de entender si la escala es más pequeña. Vamos a enfocarnos en dos clientes: el cliente ilustrado con un arcoíris y el cliente ilustrado con una rosa.

En el ejemplo, asignamos el cliente del arcoíris al primer trabajador y al cuarto. La combinación de ambos trabajadores conforma la partición aleatoria de ese cliente. Los clientes restantes se asignarán a distintas particiones virtuales, con su propia combinación de dos trabajadores. Por ejemplo, el cliente de la rosa también se asigna al primer trabajador, pero su segundo trabajador será el octavo.



Si el cliente del arcoíris, asignado a los trabajadores uno y cuatro, tiene un problema (como envenenamiento o inundación de solicitudes), este afectará su partición virtual, pero no

tendrá efecto total sobre ninguna otra. De hecho, como máximo, se verá afectado uno de los trabajadores de otra partición aleatoria. Si los solicitantes son tolerantes a errores y pueden solucionar el problema (con reintentos, por ejemplo), el servicio puede continuar funcionando de manera ininterrumpida para los clientes o los recursos en las otras particiones, como se ilustra en la siguiente imagen.



En otras palabras, mientras todos los trabajadores que prestan servicio al cliente del arcoíris pueden experimentar un problema o un ataque, los demás trabajadores no se verán afectados en absoluto. Para los clientes, esto significa que aunque el cliente de la rosa y el cliente del girasol compartan un trabajador con el cliente del arcoíris cada uno, ellos no se verán afectados. Como se ve en la siguiente imagen, el octavo trabajador puede prestar servicio al cliente de la rosa, y el sexto, al del girasol.

Cuando hay un problema, todavía podemos perder un cuarto del servicio total, pero con esta distribución de los clientes o los recursos a través de la fragmentación aleatoria, el alcance del impacto es mucho menor. Con ocho trabajadores, existen 28 combinaciones únicas de dos trabajadores, es decir, 28 posibles particiones aleatorias. Si tenemos cien clientes o más, y asignamos una partición a cada cliente, entonces el alcance del impacto correspondiente a un problema alcanzará solo la fracción 1/28. Esto es 7 veces mejor que con la fragmentación común.

Es muy emocionante ver cómo los números mejoran de manera exponencial mientras más clientes y trabajadores tenga. La mayoría de los desafíos de escalado se complican cada vez más con esas dimensiones, pero la fragmentación aleatoria mejora en términos de efectividad. De hecho, si se cuenta con los trabajadores suficientes, puede haber más particiones aleatorias que clientes, por lo que se puede aislar a cada cliente.

Amazon Route 53 y la fragmentación aleatoria

¿Cómo contribuye todo esto a Amazon Route 53? Para Route 53, decidimos distribuir nuestra capacidad en un total de 2048 servidores de nombres virtuales. Estos son virtuales porque no se corresponden con los servidores físicos que alojan Route 53. Podemos ir cambiándolos de lugar para administrar mejor la capacidad. Luego, asignamos cada dominio de cliente a una partición aleatoria de cuatro servidores de nombres virtuales. Con estas cifras, obtenemos la asombrosa cantidad de 730 mil millones de posibles particiones aleatorias. Tenemos tantas posibles particiones aleatorias que podemos asignar cada dominio a una partición en particular. De hecho, podemos dar un paso más y asegurarnos de que un dominio de cliente nunca comparta más de dos servidores de nombres virtuales con otro dominio de cliente.

Los resultados son impresionantes. Si el dominio de un cliente es blanco de un ataque DDOS, los cuatro servidores de nombres virtuales asignados a ese dominio experimentarán un aumento en el tráfico, pero el resto de los dominios de clientes permanecerá sin cambios. De todas maneras, no nos resignamos a que el cliente que ha sido blanco del ataque tenga un mal día. La fragmentación aleatoria implica que podemos identificar y aislar el cliente atacado para dedicar capacidad especial a la tarea de detener el ataque. Además, desarrollamos nuestra propia capa patentada de filtros de tráfico, AWS Shield. Sin embargo, la fragmentación aleatoria logra marcar una gran diferencia a la hora de garantizar una experiencia de cliente sin contratiempos con Route 53, aun cuando surgen problemas.

Conclusión

Hemos decidido integrar la fragmentación aleatoria en muchos de nuestros otros sistemas. Además, hemos introducido mejoras, tales como la fragmentación aleatoria recursiva, con la que podemos fragmentar elementos en múltiples capas y así aislar al cliente de un cliente. La fragmentación aleatoria es extremadamente adaptable. Es una manera inteligente de organizar los recursos existentes. Además, por lo general, no implica costos adicionales, por lo que es una gran mejora que se puede implementar de manera económica.

Si está interesado en utilizar la fragmentación aleatoria, consulte nuestra biblioteca de código abierto, Route 53 Infima. Esta biblioteca incluye varias implementaciones diferentes de fragmentación aleatoria que pueden usarse para asignar u organizar recursos.