Documento técnico de AWS

Conceptos básicos sobre varias regiones de AWS

Conceptos básicos sobre varias regiones de AWS: Documento técnico de AWS

Table of Contents

Conceptos básicos sobre varias regiones de AWS

Fecha de publicación: 20 de diciembre de 2022 () Revisiones del documento

Resumen

Este documento avanzado de 300 niveles está dirigido a arquitectos de nube y líderes sénior que crean cargas de trabajo y AWS que estén interesados en utilizar una arquitectura multirregional para mejorar la resiliencia de sus cargas de trabajo. Este paper asume un conocimiento básico de la AWS infraestructura y los servicios. Describe los casos de uso comunes en varias regiones, comparte los conceptos y las implicaciones fundamentales de varias regiones en torno al diseño, el desarrollo y la implementación, y proporciona orientación prescriptiva para ayudarlo a determinar mejor si una arquitectura multirregional es adecuada para sus cargas de trabajo.

¿Usa Well-Architected?

El marco de AWS Well-Architected le ayuda a entender las ventajas y desventajas de las decisiones que toma al crear sistemas en la nube. Los seis pilares del marco le permitirán aprender las prácticas recomendadas de arquitectura para diseñar y utilizar sistemas fiables, seguros, eficientes, rentables y sostenibles. Mediante AWS Well-Architected Tool, disponible sin costo alguno en la AWS Management Console, puede comparar las cargas de trabajo con estas prácticas recomendadas respondiendo a una serie de preguntas para cada pilar.

Para obtener más orientación experta y prácticas recomendadas para la arquitectura de la nube (implementaciones de arquitectura de referencia, diagramas y documentos técnicos), consulte el Centro de arquitectura de AWS.

Introducción

Cada una Región de AWSconsta de varias zonas de disponibilidad independientes y físicamente separadas dentro de un área geográfica. Se mantiene una separación lógica estricta entre los servicios de software de cada región. Este diseño específico garantiza que una falla en la infraestructura o los servicios en una región no provoque una falla correlacionada en otra región.

La mayoría de AWS los clientes pueden alcanzar sus objetivos de resiliencia para una carga de trabajo en una sola región mediante múltiples zonas de disponibilidad (AZ) o servicios regionales.

AWS Sin embargo, un subconjunto de clientes opta por arquitecturas multirregionales por tres motivos.

- Tienen requisitos de alta disponibilidad y continuidad de las operaciones para sus cargas de trabajo de nivel más alto que, en su opinión, no se pueden cumplir en una sola región.
- Deben cumplir con los requisitos de soberanía de los datos (como el cumplimiento de las leyes y reglamentos locales), que exigen que las cargas de trabajo operen dentro de una jurisdicción determinada.
- Necesitan mejorar el rendimiento y la experiencia del cliente en relación con la carga de trabajo mediante la ejecución de las cargas de trabajo en las ubicaciones más cercanas a los usuarios finales.

Este paper se centra en los requisitos de alta disponibilidad y continuidad de las operaciones, y le ayuda a analizar las consideraciones necesarias para adoptar una arquitectura multirregional para una carga de trabajo. Describimos los conceptos fundamentales que se aplican al diseño, el desarrollo y el despliegue de una carga de trabajo multirregional, junto con un marco prescriptivo que le ayudará a determinar si una arquitectura multirregional es la opción correcta para una carga de trabajo concreta. Debe asegurarse de que una arquitectura multirregional sea la opción correcta para su carga de trabajo, ya que estas arquitecturas suponen un desafío y es posible que, si no se hace correctamente, la disponibilidad general de la carga de trabajo disminuya.

Diseñar y operar para la resiliencia en una sola región

Antes de profundizar en los conceptos multirregionales, comience por confirmar que su carga de trabajo ya es lo más resiliente posible en una sola región. Para lograrlo, evalúe su carga de trabajo comparándola con el pilar de confiabilidad y el pilar de excelencia operativa del AWS Well-Architected Framework y realice los cambios necesarios para adoptar las mejores prácticas recomendadas. Los siguientes conceptos se tratan en el AWS Well-Architected Framework:

- Segmentación de la carga de trabajo en función de los límites del dominio
- Contratos de servicio bien definidos
- Gestión de la dependencia y acoplamiento
- Gestionar los errores, los reintentos y las estrategias de espera
- Operaciones idempotentes y transacciones con estado frente a transacciones sin estado
- Preparación operativa y gestión de cambios
- Comprensión del estado de las cargas
- Responder a los eventos

Para mejorar la resiliencia de una sola región, revise y aplique los conceptos descritos en los patrones avanzados de resiliencia multizona de disponibilidad para gestionar los fallos grises. Este paper proporciona las mejores prácticas sobre el uso de réplicas en cada zona de disponibilidad para contener los errores y amplía los conceptos de zonas de disponibilidad múltiples introducidos en AWS Well Architected. Una vez que haya aplicado completamente los conceptos y las mejores prácticas recomendados para lograr la máxima resiliencia en una sola región, se puede evaluar una carga de trabajo específica comparándola con los fundamentos de las arquitecturas multirregionales para determinar si la resiliencia de la carga de trabajo se puede aumentar mediante un enfoque multirregional.

Fundamento multirregional 1: comprender los requisitos

Como se mencionó anteriormente, la alta disponibilidad y la continuidad de las operaciones son motivos habituales para optar por arquitecturas multirregionales. Las métricas de disponibilidad miden el porcentaje de tiempo que una carga de trabajo está disponible para su uso durante un período definido, mientras que las métricas de continuidad de las operaciones miden la recuperación en el caso de eventos a gran escala y, por lo general, de mayor duración.

Medir la disponibilidad es un proceso casi continuo. Las mediciones o métricas específicas pueden variar, pero normalmente se agrupan en torno a un objetivo de disponibilidad, que en la mayoría de los casos se denomina nueves (por ejemplo, una disponibilidad del 99,99%). Con los objetivos de disponibilidad, no hay una solución única para todos. Los objetivos de disponibilidad deben establecerse a nivel de carga de trabajo, en lugar de aplicar un objetivo único para todas las cargas de trabajo, separando los componentes no críticos de los críticos.

Para la continuidad de las operaciones, se suelen utilizar las siguientes point-in-time medidas:

- Objetivo de tiempo de recuperación (RTO): el RTO es el retraso máximo aceptable entre la interrupción del servicio y el restablecimiento del servicio. Este valor determina un período aceptable durante el cual el servicio estará interrumpido.
- Objetivo de punto de recuperación (RPO): el RPO es el tiempo máximo aceptable desde el último punto de recuperación de datos. Esto determina qué se considera una pérdida de datos aceptable entre el último punto de recuperación y una interrupción del servicio.

Al igual que cuando se establecen los objetivos de disponibilidad, el RTO y el RPO también deben definirse a nivel de carga de trabajo. Para lograr una continuidad de las operaciones más rigurosa o unos requisitos de alta disponibilidad, se requiere una mayor inversión. Dicho esto, no todas las aplicaciones pueden exigir o requerir el mismo nivel de resiliencia. La creación de un mecanismo de organización por niveles puede ayudar a establecer el marco que permita alinear a los propietarios de TI y las empresas a la hora de identificar las aplicaciones más exigentes en función del impacto en la empresa y organizarlas en niveles en consecuencia. En las siguientes tablas se pueden encontrar ejemplos de organización en niveles.

Tabla 1: Ejemplo de estratificación de resiliencia para el SLA

Acuerdo de nivel de servicio (SLA) de disponibilidad	Nivel de resiliencia	Tiempo de inactividad aceptable por año
99,99%	Platino	52,60 minutos
99,90%	Oro	8,77 horas
99,5%	Plata	1,83 días

Tabla 2: Ejemplo de estratificación de resiliencia para RTO y RPO

Nivel	RTO máximo	RPO máximo	Criterios	Coste
Platino	15 minutos	cinco minutos	Cargas de trabajo de misión crítica	$$$
Oro	15 minutos — seis horas	dos horas	Cargas de trabajo importantes, pero no esenciales	$$
Plata	seis horas, unos días	24 horas	Cargas de trabajo no críticas	$

Al diseñar cargas de trabajo para garantizar la resiliencia, es necesario comprender la relación entre la alta disponibilidad y la continuidad de las operaciones. Por ejemplo, si una carga de trabajo requiere una disponibilidad del 99,99%, no se toleran más de 53 minutos de inactividad al año. La detección de un fallo puede tardar al menos cinco minutos y un operador puede tardar otros diez minutos en intervenir, tomar decisiones sobre los pasos de recuperación y llevarlos a cabo. No es inusual que la recuperación de un solo problema tarde entre 30 y 45 minutos en recuperarse. En este caso, disponer de una estrategia multirregional destinada a proporcionar una instancia aislada que elimine el impacto correlacionado puede permitir la continuación de las operaciones, ya que las operaciones se cancelan en un plazo determinado y, al mismo tiempo, se evalúa el deterioro inicial de forma independiente. Aquí es donde es necesario definir el RTO y el RPO adecuados.

En el caso de las cargas de trabajo esenciales que tienen necesidades de disponibilidad extremas (por ejemplo, una disponibilidad del 99,99% o superior) o de los estrictos requisitos de continuidad de las operaciones que solo se pueden cumplir si se trasladan por error a otra región, podría ser adecuado adoptar un enfoque multirregional. Sin embargo, estos requisitos normalmente solo se aplican a un subconjunto pequeño de la cartera de cargas de trabajo de una empresa que tiene un tiempo de recuperación limitado medido en minutos u horas. A menos que una aplicación necesite un tiempo de recuperación de unos minutos o unas horas, esperar a que se solucione una interrupción regional de la aplicación en la región afectada podría ser una mejor opción y, por lo general, se adapta a las cargas de trabajo de nivel inferior.

Antes de implementar una arquitectura multirregional, los responsables de la toma de decisiones empresariales y los equipos técnicos deben tener en cuenta las implicaciones en materia de costos, incluidos los factores que influyen en los costos operativos y de infraestructura. Una arquitectura multirregional típica puede suponer un aumento de costes dos veces superior al de un enfoque de una sola región. Si bien existen varios patrones multirregionales para la continuidad del negocio, como funcionar con una estación de espera activa, una estación de espera caliente y una luz piloto, el patrón con el menor riesgo de cumplir los objetivos de recuperación implicará utilizar una estación de espera activa y duplicará el costo de la carga de trabajo.

Guía clave

- Los objetivos de disponibilidad y continuidad de las operaciones, como el RTO y el RPO, deben establecerse por carga de trabajo y alinearse con las partes interesadas de la empresa y de TI.
- La mayoría de los objetivos de disponibilidad y continuidad de las operaciones se pueden cumplir en una sola región. En el caso de los objetivos que no se puedan cumplir con una sola región, se debe considerar la opción de multirregión, con una visión clara de las compensaciones entre costo, complejidad y beneficios.

Fundamento multirregional 2: entender los datos

La administración de los datos no es un problema trivial en las arquitecturas multirregionales. La distancia geográfica entre las regiones impone una latencia inevitable, que se manifiesta en el tiempo que se tarda en replicar los datos en todas las regiones. Será necesario hacer concesiones entre la disponibilidad, la coherencia de los datos y la introducción de niveles de latencia superiores en una carga de trabajo que utilice una arquitectura multirregional. Ya sea que utilice la replicación asíncrona o sincrónica, tendrá que modificar su aplicación para gestionar los cambios de comportamiento que impone la tecnología de replicación. Es muy difícil convertir una aplicación existente diseñada para una sola región en varias regiones debido a los desafíos relacionados con la consistencia y la latencia de los datos. Comprender los requisitos de coherencia de los datos y los patrones de acceso a los datos para determinadas cargas de trabajo es fundamental para sopesar las desventajas.

2a: Comprender los requisitos de coherencia de los datos

El teorema CAP proporciona una referencia para razonar sobre las ventajas y desventajas entre la consistencia de los datos, la disponibilidad y las particiones de la red, de las cuales solo dos pueden satisfacerse al mismo tiempo para una carga de trabajo. La multirregión, por definición, incluye particiones de red entre regiones, por lo que debe elegir entre disponibilidad y coherencia.

Si selecciona la disponibilidad de los datos en todas las regiones, no incurrirá en una latencia significativa durante la escritura transaccional, ya que se depende de la replicación asíncrona de los datos comprometidos entre regiones, lo que reduce la coherencia entre las regiones hasta que se complete la replicación. Con la replicación asíncrona, cuando se produce un error en la región principal, existe una alta probabilidad de que haya escrituras pendientes de replicación desde la región principal. Esto lleva a una situación en la que los datos más recientes no están disponibles hasta que se reanude la replicación y se necesita un proceso de reconciliación para gestionar las transacciones en curso que no se replicaron desde la región en la que se produjo la interrupción.

Para las cargas de trabajo en las que se prefiere la replicación asíncrona, puede utilizar servicios como Amazon Aurora y Amazon DynamoDB, que proporcionan una replicación asíncrona entre regiones. Tanto las tablas globales de Amazon Aurora Global Database como las de Amazon DynamoDB tienen métricas de CloudWatchAmazon predeterminadas que ayudan a monitorizar el retraso en la replicación.

Diseñar la carga de trabajo para aprovechar las arquitecturas basadas en eventos es una ventaja para una estrategia multirregional, ya que permite que la carga de trabajo abarque la replicación

asincrónica de los datos y permite reconstruir el estado mediante la reproducción de los eventos. Dado que los servicios de streaming y mensajería almacenan los datos de carga útil de los mensajes en una sola región, un proceso de conmutación por error o devolución regional debe incluir un mecanismo para redirigir los flujos de datos de entrada del cliente, así como conciliar las cargas útiles en vuelo o no entregadas almacenadas en la región que sufrió la interrupción.

Si se selecciona la coherencia, se incurrirá en una latencia significativa, ya que los datos se replicarán de forma sincrónica durante las escrituras transaccionales. Al escribir en varias regiones de forma sincrónica, si la escritura no se realiza correctamente en todas, es posible que la disponibilidad disminuya porque la transacción no se confirmará y será necesario volver a intentarlo. Los intentos de volver a escribir los datos en todas las regiones de forma sincrónica se realizan a costa de la latencia en cada intento. En algún momento, cuando se hayan agotado los reintentos, habrá que tomar la decisión de anular por completo la transacción, lo que reducirá la disponibilidad, o enviar la transacción únicamente a las regiones disponibles, lo que generará incoherencias. Existen tecnologías de generación de quórum, como Paxos, que pueden ayudar a replicar y archivar datos de forma sincrónica, pero que requieren una importante inversión por parte de los desarrolladores.

Cuando las escrituras implican una replicación sincrónica en varias regiones para cumplir con estrictos requisitos de coherencia, la latencia de escritura aumenta en un orden de magnitud. Una latencia de escritura más alta no es algo que normalmente se pueda adaptar a una aplicación sin cambios significativos. Lo ideal es que se tenga en cuenta al diseñar la aplicación por primera vez. Para las cargas de trabajo multirregionales en las que la replicación sincrónica es una prioridad, las soluciones de los AWS socios pueden ayudar.

2b: Comprender los patrones de acceso a los datos

Los patrones de acceso a los datos de la carga de trabajo se clasifican en uno de los siguientes tipos: de lectura intensiva o de escritura intensiva. La comprensión de esta característica para una carga de trabajo concreta servirá de guía para seleccionar una arquitectura multirregional adecuada.

Para cargas de trabajo de lectura intensiva, como el contenido estático que es completamente de solo lectura, se puede lograr una arquitectura multirregional activa/activa sin una complejidad significativa. Servir contenido estático en la periferia mediante una red de distribución de contenido (CDN) garantiza la disponibilidad al almacenar en caché el contenido lo más cerca posible del usuario final; el uso de conjuntos de funciones como la conmutación por error de Origin en Amazon CloudFront puede ayudar a lograrlo. Otra opción es implementar la informática sin estado en varias regiones y utilizar el DNS para dirigir a los usuarios a la región más cercana para leer el contenido. Para ello, se puede utilizar Route 53 con una política de enrutamiento por geolocalización.

Para las cargas de trabajo de lectura intensiva que tienen un porcentaje mayor de lecturas que de escrituras, se puede utilizar una estrategia de lectura local y escritura global. Esto implica que todas las escrituras van a una base de datos de una región específica, con una replicación asíncrona de los datos en todas las demás regiones, y las lecturas se pueden realizar en cualquier región para lograrlo. Este enfoque requiere una carga de trabajo para lograr una coherencia final, ya que las lecturas locales pueden quedar obsoletas debido al aumento de la latencia necesaria para la replicación de escrituras entre regiones.

Aurora Global Database puede ayudar a aprovisionar réplicas de lectura en una región de espera que solo puede gestionar todo el tráfico de lectura de forma local y un único almacén de datos principal en una región específica para gestionar las escrituras. Los datos se replican de forma asíncrona desde la base de datos principal a la base de datos en espera (réplicas de lectura) y las bases de datos en espera se pueden pasar a la principal si es necesario realizar una conmutación por error de las operaciones a la región en espera. Si una carga de trabajo se adapta mejor a los modelos de datos no relacionales, DynamoDB también se puede utilizar en este enfoque. Una vez más, la carga de trabajo debe adoptar una coherencia eventual, lo que puede requerir que se vuelva a escribir si no se diseñó para ello desde el principio.

Para las cargas de trabajo con un uso intensivo de escritura, se debe seleccionar una región principal y se debe incorporar a la carga de trabajo la capacidad de realizar la conmutación por error a una región en espera. En comparación con un enfoque activo/activo, un enfoque primario/de reserva es menos complicado. Esto se debe a que, en el caso de una arquitectura activa/activa, será necesario reescribir la carga de trabajo para gestionar el enrutamiento inteligente a Regions, establecer la afinidad de las sesiones, garantizar la idempotencia de las transacciones y gestionar los posibles conflictos.

La mayoría de las cargas de trabajo que buscan resiliencia en varias regiones no requerirán un enfoque activo/activo. Se puede utilizar una estrategia de fragmentación para aumentar la resiliencia al limitar el radio de impacto de una avería en toda la base de clientes. Si puedes fragmentar una base de clientes de forma eficaz, puedes seleccionar diferentes regiones principales para cada fragmento. Por ejemplo, si puede dividir los clientes de manera que la mitad de ellos estén alineados con la región uno y la otra mitad con la región dos, tratando las regiones como celdas, se puede crear un enfoque de celdas multirregional, lo que reduce el radio de impacto de su carga de trabajo.

El enfoque de fragmentación se puede combinar con un enfoque primario/en espera para proporcionar capacidades de conmutación por error a los fragmentos. Será necesario incorporar a la carga de trabajo un proceso de conmutación por error probado y un proceso de reconciliación de

datos que garantice la coherencia transaccional de los almacenes de datos tras la conmutación por error. Estos se tratan con mayor detalle más adelante en este paper.

Guía clave

- Existe una alta probabilidad de que las escrituras pendientes de replicación no se envíen a la región en espera si se produce un error. Los datos no estarán disponibles hasta que se reanude la replicación (suponiendo que la replicación sea asíncrona).
- Como parte de la conmutación por error, se necesitará un proceso de reconciliación de datos para garantizar que los almacenes de datos que utilizan la replicación asincrónica mantengan un estado coherente desde el punto de vista de las transacciones.
- Cuando se requiera una coherencia sólida, las cargas de trabajo deberán modificarse para tolerar la latencia requerida del almacén de datos que se replica de forma sincrónica.

Fundamento multirregional 3: comprender las dependencias de su carga de trabajo

Una carga de trabajo específica puede tener varias dependencias en una región, como los AWS servicios utilizados, las dependencias internas, las dependencias de terceros, las dependencias de red, los certificados, las claves, los secretos y los parámetros. Para garantizar el funcionamiento de la carga de trabajo en caso de fallo, no debe haber dependencias entre la región principal y la región en espera; cada una debe poder funcionar de forma independiente. Para lograrlo, se deben analizar todas las dependencias de la carga de trabajo para garantizar que estén disponibles en cada región. Esto es necesario porque un error en la región principal no debería afectar a la región en espera. Además, es imprescindible conocer cómo funciona la carga de trabajo cuando una dependencia se encuentra en un estado degradado o no está disponible por completo, a fin de poder diseñar soluciones que lo gestionen de forma adecuada.

3a: servicios AWS

Al diseñar una arquitectura multirregional, es necesario comprender los AWS servicios específicos que se utilizarán. El primer aspecto es entender qué características tiene el servicio para permitir la multirregión y si se debe diseñar una solución para cumplir los objetivos multirregionales. Por ejemplo, con Amazon Aurora y Amazon DynamoDB, hay una función para replicar datos de forma asíncrona en una región en espera. Todas las dependencias AWS de servicio deberán estar disponibles en todas las regiones desde las que se vaya a ejecutar la carga de trabajo. Para asegurarse de que los servicios que se utilizarán estén disponibles en las regiones deseadas, consulte la lista de Región de AWS todos los servicios.

3b: Dependencias internas y de terceros

En el caso de cualquier dependencia interna que tenga una carga de trabajo, asegúrate de que esté disponible en las regiones desde las que operará la carga de trabajo. Por ejemplo, si la carga de trabajo está compuesta por muchos microservicios, conozca todos los microservicios que componen una capacidad empresarial. A partir de ahí, asegúrese de que todos esos microservicios se desplieguen en cada región desde la que vaya a operar la carga de trabajo.

No se recomiendan las llamadas interregionales entre microservicios dentro de una carga de trabajo, y se debe mantener el aislamiento regional. Esto se debe a que la creación de dependencias entre

regiones aumenta el riesgo de que se produzcan errores correlacionados, lo que anula los beneficios que se están intentando conseguir con las implementaciones regionales aisladas de la carga de trabajo. Las dependencias locales también pueden ser parte de la carga de trabajo, por lo que es imprescindible comprender cómo podrían cambiar las características de estas integraciones si cambiara la región principal. Por ejemplo, si la región en espera se encuentra más alejada del entorno local, el aumento de la latencia tendrá un impacto negativo.

Comprender las soluciones de software como servicio (SaaS), los kits de desarrollo de software (SDK) y otras dependencias de productos de terceros, y poder analizar situaciones en las que estas dependencias están degradadas o no están disponibles, proporcionará más información sobre cómo funciona y se comporta la cadena de sistemas en diferentes modos de falla. Estas dependencias pueden estar dentro del código de una aplicación, desde la forma en que se administran los secretos externamente mediante AWS Secrets Manager o una solución de almacén de terceros (como Hashicorp), hasta los sistemas de autenticación que dependen del IAM Identity Center para los inicios de sesión federados.

Disponer de redundancia en lo que respecta a las dependencias puede contribuir a aumentar la resiliencia. También existe la posibilidad de que una solución SaaS o una dependencia de terceros utilice la misma carga principal que la carga de Región de AWS trabajo. Si este es el caso, debe trabajar con el proveedor para determinar si su postura de resiliencia se ajusta a los requisitos de la carga de trabajo.

Además, tenga en cuenta el destino compartido entre la carga de trabajo y sus dependencias, como las aplicaciones de terceros. Si las dependencias no están disponibles en (o desde) una región secundaria tras una conmutación por error, es posible que la carga de trabajo no se recupere por completo.

3c: Mecanismo de conmutación por error

El sistema de nombres de dominio (DNS) se utiliza habitualmente como mecanismo de conmutación por error para desplazar el tráfico de la región principal a una región de reserva. Revise y analice de forma crítica todas las dependencias que implica el mecanismo de conmutación por error. Por ejemplo, si su carga de trabajo utiliza Amazon Route 53, entender que el plano de control está alojado en US-East-1 significa que está pasando a depender del plano de control de esa región específica. Esto no se recomienda como parte de un mecanismo de conmutación por error si la región principal también es US-East-1. Si se utiliza otro mecanismo de conmutación por error, es necesario conocer a fondo cualquier situación en la que no funcione como se espera. Una vez establecido este entendimiento, planifique una contingencia o desarrolle un nuevo mecanismo, si es

necesario. Consulte Creación de mecanismos de recuperación ante desastres con Amazon Route 53 para obtener información sobre los enfoques que puede utilizar para realizar correctamente la conmutación por error.

Como se explica en la sección sobre dependencias internas, todos los microservicios que forman parte de una capacidad empresarial deben estar disponibles en cada región en la que se despliegue la carga de trabajo. Como parte de la estrategia de conmutación por error, la capacidad empresarial debe combinarse para eliminar la posibilidad de que se produzcan llamadas entre regiones. Como alternativa, si los microservicios se conmutan por error de forma independiente, existe la posibilidad de que se produzcan comportamientos no deseados, ya que los microservicios podrían realizar llamadas entre regiones, lo que generaría latencia y podría provocar que la carga de trabajo no estuviera disponible en caso de que se agotara el tiempo de espera de los clientes.

3d: Dependencias de configuración

Los certificados, las claves, los secretos y los parámetros forman parte del análisis de dependencias necesario al diseñar para varias regiones. Siempre que sea posible, es mejor localizar estos componentes dentro de cada región para que estas dependencias no tengan un destino compartido entre las regiones. En el caso de los certificados, la caducidad debe variar de una región a otra y, si es posible, de una región a otra, para evitar que un certificado que caduque (con las alarmas configuradas para avisar con antelación) afecte a varias regiones.

Las claves y los secretos de cifrado también deben ser específicos de cada región. De esta forma, si se produce un error al rotar una clave o un secreto, el impacto se limita a una región específica.

Por último, todos los parámetros de la carga de trabajo deben almacenarse localmente para que la carga de trabajo se recupere en la región específica.

Guía clave

- Una arquitectura multirregional se beneficia de la separación física y lógica entre las regiones. La introducción de dependencias entre regiones en la capa de aplicación reduce este beneficio. Evite este tipo de dependencias.
- Los controles de conmutación por error deberían funcionar sin depender de la región principal.
- Es necesario coordinar la conmutación por error desde el punto de vista empresarial para eliminar la posibilidad de que aumenten la latencia y la dependencia de las llamadas entre regiones.

Fundamento multirregional 4: Preparación operativa

Gestionar una carga de trabajo multirregional es una tarea compleja que conlleva desafíos operativos específicos de cada región. Estos incluyen la Cuenta de AWS administración, la reorganización de los procesos de implementación, la creación de una estrategia de observabilidad multirregional, la creación y prueba de un manual de conmutación por error y recuperación y, por último, la administración de los costos. Una revisión de la preparación operativa (ORR) puede ayudar a los equipos a preparar una carga de trabajo para la producción, ya sea que se ejecute en una sola región o en varias regiones.

4a: Cuenta de AWS gestión

Para implementar una carga de trabajo en todas las regionesRegiones de AWS, asegúrate de que todas las cuotas de AWS servicio de una cuenta sean iguales en todas las regiones. En primer lugar, conozca todos los AWS servicios que forman parte de la arquitectura, analice el uso planificado en las regiones en espera y compárelos con el uso actual. En algunos casos, si la región en espera no se ha utilizado antes, puede consultar las cuotas de servicio predeterminadas para comprender el punto de partida. A continuación, solicite un aumento de cuota en todos los servicios que vaya a utilizar mediante la consola Service Quotas (es necesario iniciar sesión) o las API.

AWSLas funciones de Identity and Access Management (IAM) deben configurarse en cada región para garantizar que los operadores, las herramientas de automatización y los AWS servicios tengan los permisos adecuados para acceder a los recursos de la región en espera. Al aislar las funciones a nivel regional, se logra el aislamiento regional que buscamos para las arquitecturas multirregionales. Asegúrese de que estos permisos estén en vigor antes de empezar a funcionar con una región en espera.

4b: Prácticas de despliegue

Con las capacidades multirregionales, el despliegue de la carga de trabajo en varias regiones puede resultar complejo. AWS CloudFormationayuda a implementar la infraestructura en una o varias regiones y se puede personalizar en función de sus necesidades. AWS CodePipelineayuda a proporcionar una cartera de integración/entrega continua (CI/CD) casi continua, que incluye acciones interregionales que permiten el despliegue en regiones distintas de la región en la que se encuentra la cartera. Esto, combinado con estrategias de despliegue sólidas, como la azul/verde, permite un despliegue con un tiempo de inactividad mínimo o nulo.

Sin embargo, el despliegue de capacidades con estado puede resultar más complejo cuando el estado de la aplicación o los datos no se externalizan a un almacén persistente. En estas situaciones, adapte cuidadosamente el proceso de implementación para que se adapte a sus necesidades. Diseñe el proceso y el proceso de despliegue para que se desplieguen en una región a la vez, en lugar de hacerlo en varias regiones simultáneamente. Esto reduce la posibilidad de que se produzcan errores correlacionados entre las regiones. Para obtener más información sobre las técnicas que Amazon utiliza para automatizar las implementaciones de software, lee el artículo de Builder Library sobre cómo automatizar las implementaciones seguras y sin intervención.

4c: Observabilidad

Al diseñar para múltiples regiones, considere cómo se monitoreará la salud de todos los componentes de cada región para obtener una visión holística de la salud regional. Esto podría incluir el monitoreo de las métricas para detectar el retraso en la replicación, lo que no es una consideración para la carga de trabajo de una sola región.

Al crear una arquitectura multirregional, considere la posibilidad de observar también el rendimiento de la carga de trabajo desde las regiones en espera. Esto incluye realizar controles de salud y canarios (pruebas sintéticas) desde la región de reserva, lo que proporciona una visión externa del estado de salud de la primaria. Además, puede utilizar Amazon CloudWatch Internet Monitor para comprender el estado de la red externa y el rendimiento de sus cargas de trabajo desde la perspectiva del usuario final. Del mismo modo, la región principal debe tener la misma observabilidad para monitorear la región en espera. Estos canarios deberían monitorear las métricas de experiencia del cliente para obtener una visión general de la carga de trabajo. Esto es necesario porque si hubiera un problema en la región principal, la observabilidad en la principal podría verse afectada y afectar a la capacidad de evaluar el estado de la carga de trabajo.

En ese caso, observar fuera de esa región puede proporcionar información. Estas métricas deberían agruparse en los paneles de control disponibles en cada región y crearse alarmas en cada región. Dado que Amazon CloudWatch es un servicio regional, es obligatorio disponer de estos servicios en ambas regiones. Estos datos de monitoreo se utilizarán para realizar la llamada a la conmutación por error desde una región principal a una en espera.

4d: Procesos, procedimientos y pruebas

El mejor momento para responder a la pregunta «¿Cuándo debo realizar la conmutación por error?» es mucho antes de que lo necesites. Todos los planes de continuidad empresarial, que incluyan a

las personas, los procesos y la tecnología, deben definirse con bastante antelación a la aparición de un problema y probarse periódicamente. Decida un marco de decisión de recuperación. Si existe un proceso de recuperación bien practicado y se conoce bien el momento de la recuperación, se puede elegir el momento en que se inicie el proceso de recuperación de forma que se cumpla el objetivo de RTO mediante una conmutación por error. Este momento podría ser inmediatamente después de que se detecte un problema con la aplicación en la región principal, o bien podría ocurrir más adelante, cuando se hayan agotado las opciones de recuperación de la aplicación de la región y, en ese momento, se deba iniciar una conmutación por error para cumplir con la RTO.

Si bien la acción de conmutación por error en sí misma debe automatizarse al 100%, la decisión de activar la conmutación por error debe tomarla una persona (normalmente un número reducido de personas predeterminadas de la organización). Además, los criterios para decidir sobre una conmutación por error deben estar claramente definidos y ser entendidos a nivel global por la organización. Estos procesos se pueden definir y completar mediante los manuales de ejecución de AWS System Manager, lo que permite una end-to-end automatización completa y garantiza la coherencia de la ejecución del proceso durante las pruebas y la conmutación por error.

Estos manuales de ejecución deben estar disponibles en la región principal y en espera para iniciar los procesos de conmutación por error o conmutación por recuperación. Una vez implementada esta automatización, se debe definir y seguir una cadencia de pruebas regular. Esto garantiza que, cuando se produzca un evento real, la respuesta se base en un proceso bien definido y practicado en el que la organización confíe. También es importante tener en cuenta las tolerancias establecidas para los procesos de reconciliación de datos. Confirme que los requisitos de RPO/RTO establecidos se cumplen con el proceso propuesto.

4e: Costo y complejidad

Las implicaciones económicas de una arquitectura multirregional se deben al aumento del uso de la infraestructura, la sobrecarga operativa y el tiempo dedicado a los recursos. Como se mencionó anteriormente, el costo de infraestructura en una región en espera es similar al costo de infraestructura en una región principal cuando se realiza el aprovisionamiento previo, lo que se traduce en el doble del costo. Aprovisione la capacidad de forma que sea suficiente para las operaciones diarias, pero reserve suficiente capacidad de búfer para tolerar los picos de demanda, y configure los mismos límites en cada región.

Además, es posible que sea necesario realizar cambios en el nivel de las aplicaciones para que se ejecute correctamente en una arquitectura multirregional si va a adoptar una arquitectura activa-activa, ya que su diseño y funcionamiento pueden requerir mucho tiempo y recursos. Como

mínimo, las organizaciones deberían dedicar tiempo a comprender las dependencias técnicas y empresariales de cada región y a diseñar procesos de conmutación por error y recuperación.

Los equipos también deberían realizar los ejercicios habituales de conmutación por error y conmutación por recuperación para sentirse cómodos con los manuales de instrucciones que se utilizarán durante un evento. Si bien son increíblemente importantes y cruciales para obtener los resultados esperados de una inversión en varias regiones, estos ejercicios representan un costo de oportunidad y consumen tiempo y recursos que se destinan a otras actividades.

Guía clave

- AWSLas cuotas de servicio deben revisarse y ser paritarias en todas las regiones en las que operará la carga de trabajo.
- El proceso de despliegue debe centrarse en una región a la vez, en lugar de en varias regiones simultáneamente.
- Es necesario monitorear métricas adicionales, como el retraso en la replicación, y son específicas de los escenarios multirregionales.
- Amplíe la supervisión de la carga de trabajo más allá de la región principal. Las métricas de la experiencia del cliente deben supervisarse por región y medirse desde fuera de cada región en la que se ejecute una carga de trabajo.
- La conmutación por error y la conmutación por recuperación deben probarse periódicamente. Asegúrese de implementar un único manual para los procesos de conmutación por error y conmutación por recuperación, que se utilice tanto durante las pruebas como durante un evento en directo. Los manuales de ejecución para las pruebas y los eventos en directo no pueden ser diferentes.

Conclusión

Este documento técnico analiza los casos de uso comunes para múltiples regiones, los fundamentos sobre cómo implementar una arquitectura multirregional y las implicaciones de este enfoque. Estos fundamentos se pueden aplicar a cualquier carga de trabajo y se pueden utilizar como marco para ayudar a la toma de decisiones sobre si una arquitectura multirregional es o no el enfoque adecuado para una empresa en particular.

Colaboradores

Los colaboradores de este documento son:

Colaborador técnico:

- John Formento, Jr., arquitecto principal de soluciones, equipo AWS multirregional

Colaborador editorial:

- Lisi Lewis, gerente sénior de marketing de productos

Documentación adicional

Para obtener información adicional, consulte los siguientes recursos:

- Patrones avanzados de resiliencia Multi-AZ (documento AWS técnico)
- El pilar de la confiabilidad: AWS un marco de buena arquitectura
- La disponibilidad y más allá: comprender y mejorar la resiliencia de los sistemas distribuidos AWS (documento técnico) AWS
- AWSLímites del aislamiento de fallas (documento técnico) AWS

Revisiones del documento

Para recibir notificaciones sobre las actualizaciones de este documento técnico, suscríbase a la fuente RSS.

Cambio	Descripción	Fecha
Documento publicado	Primera publicación.	20 de diciembre de 2022

Avisos

Es responsabilidad de los clientes realizar su propia evaluación independiente de la información que contiene este documento. El presente documento: (a) tiene solo fines informativos, (b) representa las ofertas y prácticas actuales de los productos de AWS, que están sujetas a cambios sin previo aviso, y (c) no supone ningún compromiso ni garantía por parte de AWS y sus filiales, proveedores o licenciantes. Los productos o servicios de AWS se proporcionan “tal cual”, sin garantías, afirmaciones ni condiciones de ningún tipo, ya sean expresas o implícitas. Las responsabilidades y obligaciones de AWS con respecto a sus clientes se controlan mediante los acuerdos de AWS y este documento no forma parte ni modifica ningún acuerdo entre AWS y sus clientes.

Glosario de AWS

Para ver la terminología más reciente de AWS, consulte el Glosario de AWS en la Referencia de Glosario de AWS.

Las traducciones son generadas a través de traducción automática. En caso de conflicto entre la traducción y la version original de inglés, prevalecerá la version en inglés.