Détection et atténuation des défaillances grises

Modèles de résilience multi-AZ avancés

Modèles de résilience multi-AZ avancés: Détection et atténuation des défaillances grises

Table of Contents

Modèles de résilience multi-AZ avancés

Date de publication :11 juillet 2023(Révisions du document)

De nombreux clients exécutent leurs charges de travail dans des configurations de zones de disponibilité multiples (AZ) à haute disponibilité. Ces architectures fonctionnent bien en cas de défaillance binaire, mais rencontrent souvent des problèmes liés àgriséchecs. Les manifestations de ce type de défaillance peuvent être subtiles et ne peuvent être détectées rapidement et définitivement. Ce document fournit des conseils sur la manière d'instrumenter les charges de travail afin de détecter l'impact des défaillances grises isolées dans une seule zone de disponibilité, puis de prendre des mesures pour atténuer cet impact dans la zone de disponibilité.

Introduction

L'objectif de ce document est de vous aider à implémenter plus efficacement des architectures multi-AZ résilientes. L'une des meilleures pratiques pour créer des systèmes résilients dansCloud privé virtuel AmazonLes réseaux (VPC) doivent[déployer chaque charge de travail dans plusieurs zones de disponibilité](#).

Un[Zone de disponibilité](#)est un ou plusieurs centres de données discrets dotés d'une alimentation, d'une mise en réseau et d'une connectivité redondantes. L'utilisation de plusieurs zones de disponibilité vous permet de gérer des charges de travail qui sont plus hautement disponibles, tolérantes aux pannes et évolutives que ce qui serait possible dans un centre de données unique.

De nombreuxAWSdes services, tels que[Mise à l'échelle automatique d'Amazon Elastic Compute Cloud (EC2)](#)ou[Service de base de données relationnelle Amazon](#)(Amazon RDS), fournissez une configuration multi-AZ. Ces services ne vous obligent pas à créer d'outils d'observabilité ou de basculement supplémentaires. Ils rendent les charges de travail résilientes aux modes de défaillance binaires facilement détectables au sein d'un[Région AWS](#)qui concernent une seule zone de disponibilité. Il peut s'agir d'une panne matérielle complète, d'une coupure de courant ou d'un bogue logiciel latent affectant la majorité des ressources.

Mais il existe une autre catégorie de défaillances appeléepannes grises, dont les manifestations sont subtiles et défient toute détection rapide et définitive. Cela entraîne à son tour des délais plus longs pour atténuer l'impact causé par la panne. Cet article se concentre sur les impacts que les pannes grises peuvent avoir sur les architectures multi-AZ, sur la manière de les détecter et, enfin, sur la manière de les atténuer.

Les conseils fournis dans ce livre blanc s'appliquent principalement à des classes spécifiques de charges de travail qui :

- Utiliser principalement zonalAWSservices
- Nécessité d'améliorer la résilience d'une région
- sont prêts à faire un investissement important pour créer les modèles d'observabilité et de résilience requis

Dans le cadre de ces charges de travail, il se peut que vous ne soyez pas disposé à faire certains ou tous les compromis présentés dans???, ou ne pas avoir la possibilité d'utiliser plusieurs régions. Ces types de charges de travail sont susceptibles de représenter un petit sous-ensemble de votre portefeuille global. Par conséquent, ces conseils doivent être pris en compte au niveau de la charge de travail plutôt qu'au niveau de la plate-forme.

Défaillances grises

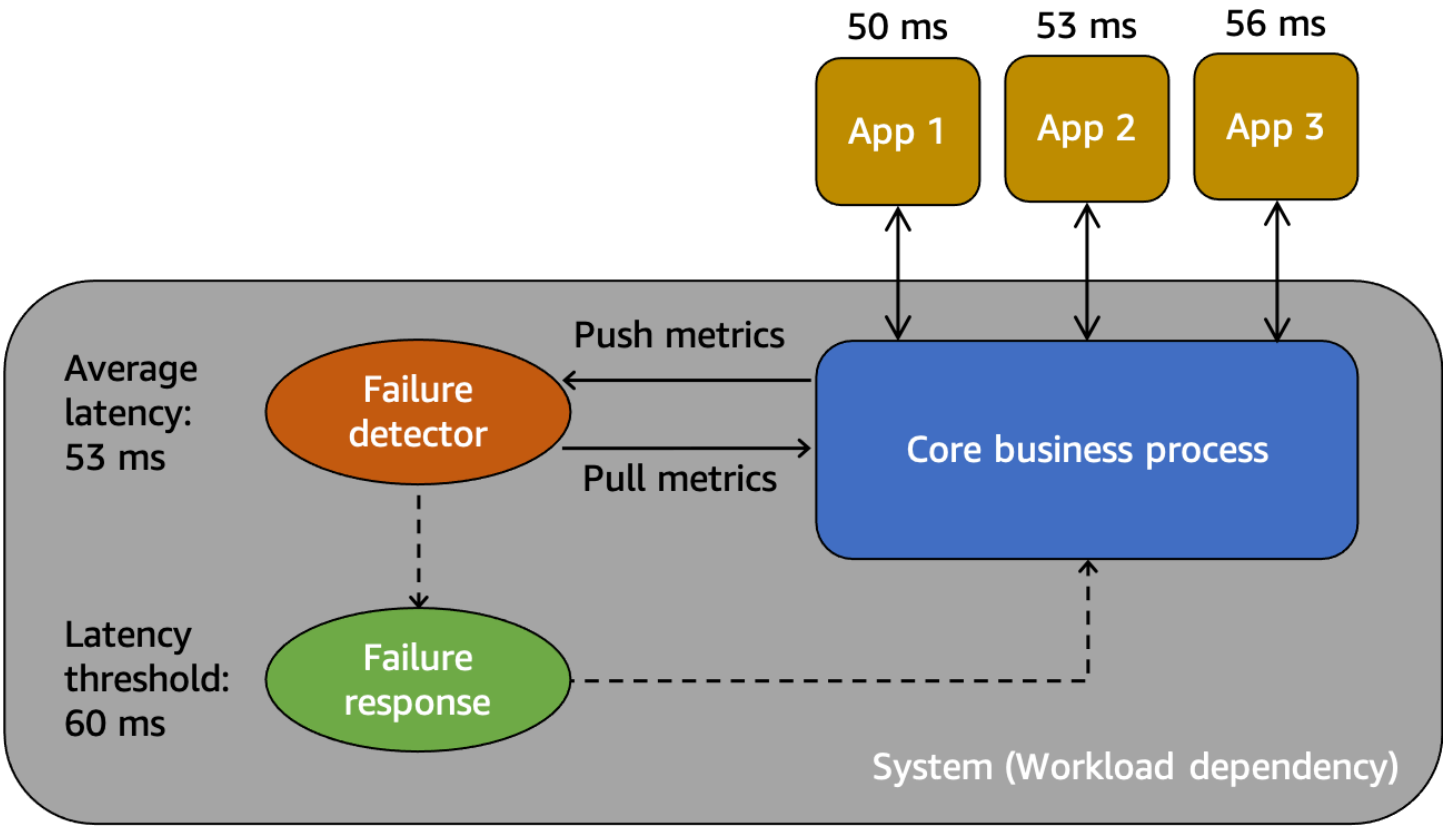
Les défaillances grises sont définies par la caractéristique deobservabilité différentielle, ce qui signifie que différentes entités observent la panne différemment. Définissons ce que cela signifie.

Observabilité différentielle

Les charges de travail que vous gérez sont généralement dépendantes. Par exemple, il peut s'agir desAWSles services cloud que vous utilisez pour créer votre charge de travail ou un fournisseur d'identité (IdP) tiers que vous utilisez pour la fédération. Ces dépendances implémentent presque toujours leur propre observabilité, en enregistrant des indicateurs concernant les erreurs, la disponibilité et la latence, entre autres éléments générés par l'utilisation par leurs clients. Lorsqu'un seuil est dépassé pour l'une de ces métriques, la dépendance prend généralement des mesures pour le corriger.

Ces dépendances ont généralement de multiples consommateurs de leurs services. Les consommateurs mettent également en œuvre leur propre observabilité et enregistrent des métriques et des journaux concernant leurs interactions avec leurs dépendances, en enregistrant des éléments tels que le niveau de latence des lectures sur disque, le nombre de demandes d'API ayant échoué ou la durée d'une requête de base de données.

Ces interactions et mesures sont décrites dans un modèle abstrait illustré dans la figure suivante.

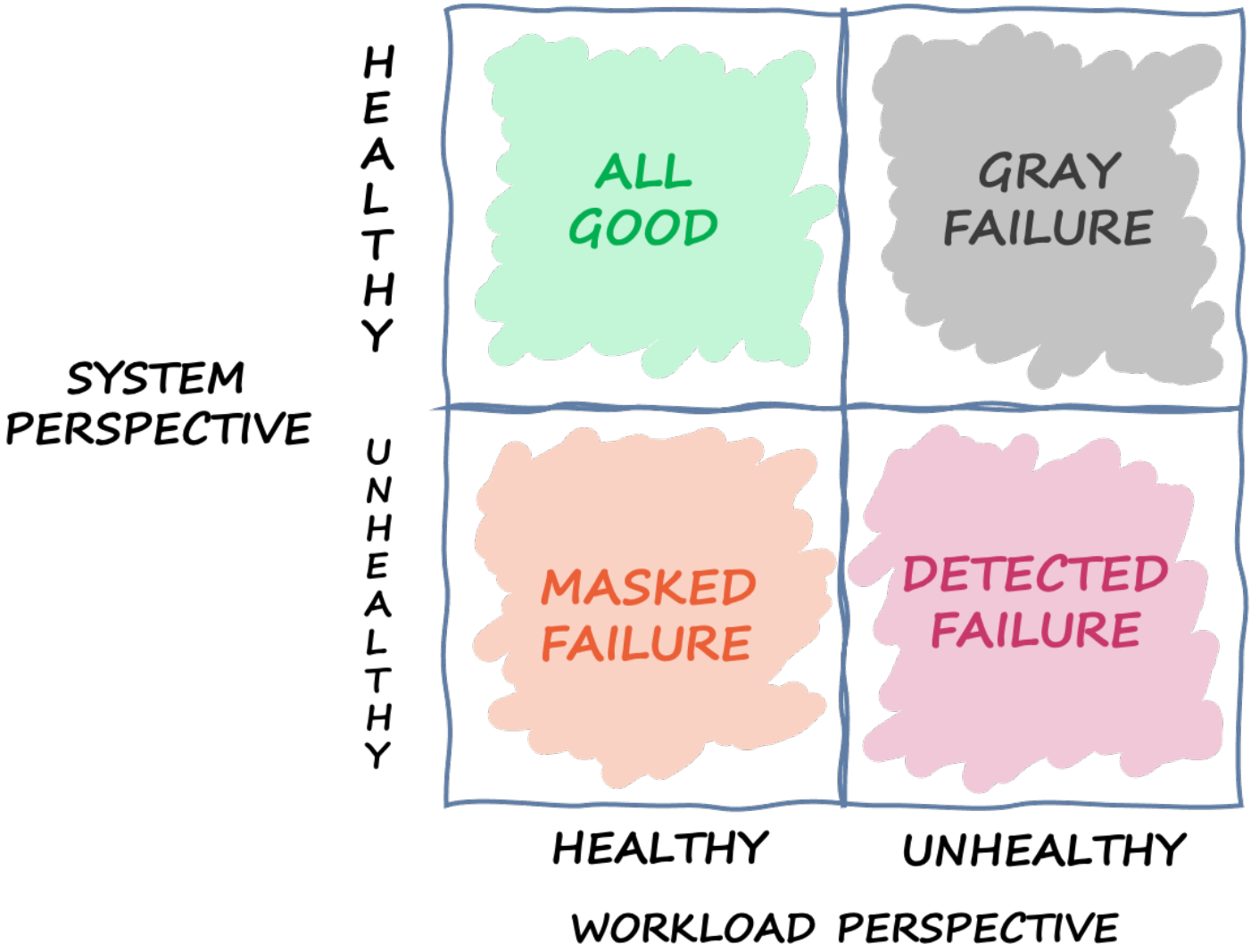


Un modèle abstrait pour comprendre les défaillances grises

Tout d'abord, nous avonssystème, qui dépend de l'App 1, de l'App 2 et de l'App 3 des consommateurs dans ce scénario. Le système dispose d'un détecteur de défaillance qui examine les métriques créées à partir du processus métier de base. Il dispose également d'un mécanisme de réponse aux défaillances pour atténuer ou corriger les problèmes observés par le détecteur de défaillance. Le système constate une latence moyenne globale de 53 ms et a défini un seuil pour invoquer le mécanisme de réponse aux pannes lorsque la latence moyenne dépasse 60 ms. Les applications 1, 2 et 3 font également leurs propres observations sur leur interaction avec le système, enregistrant une latence moyenne de 50 ms, 53 ms et 56 ms respectivement.

L'observabilité différentielle est la situation dans laquelle l'un des utilisateurs du système détecte que le système est défectueux, mais que le système de surveillance lui-même ne détecte pas le problème ou que l'impact ne dépasse pas le seuil d'alarme. Imaginons que l'App 1 commence à connaître une latence moyenne de 70 ms au lieu de 50 ms. Les applications 2 et 3 ne constatent aucun changement dans leurs latences moyennes. Cela augmente la latence moyenne du système sous-jacent à 59,66 ms, mais cela ne dépasse pas le seuil de latence nécessaire à l'activation du mécanisme de réponse aux pannes. Cependant, l'App 1 constate une augmentation de 40 % de la latence. Cela peut avoir un impact sur sa disponibilité en dépassant le délai d'expiration du client

configuré pour l'App 1, ou cela peut avoir des répercussions en cascade sur une chaîne d'interactions plus longue. Du point de vue de l'application 1, le système sous-jacent dont elle dépend n'est pas sain, mais du point de vue du système lui-même ainsi que des applications 2 et 3, le système est sain. La figure suivante résume ces différentes perspectives.



Un quadrant définissant les différents états dans lesquels un système peut se trouver en fonction de différentes perspectives

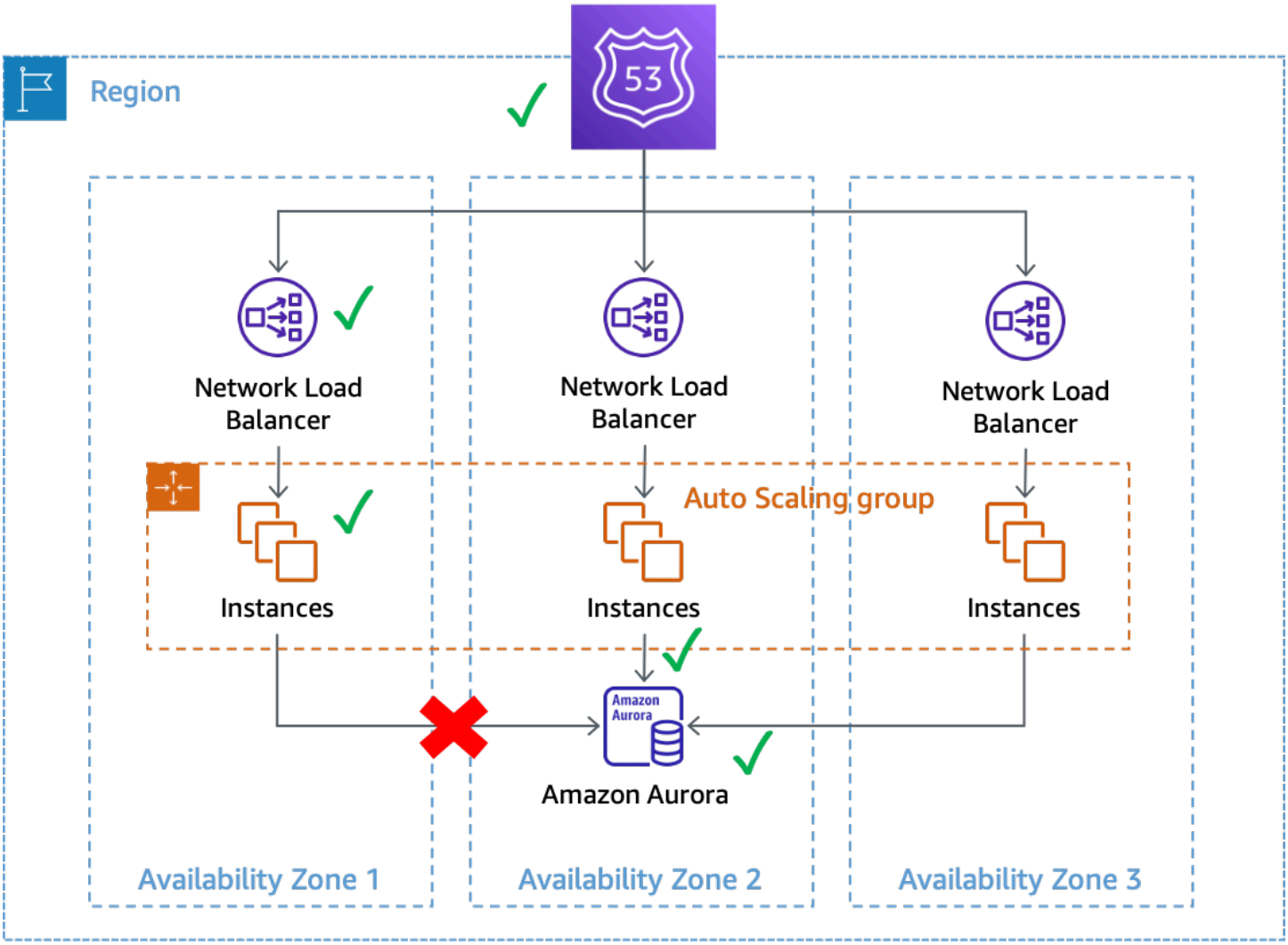
La défaillance peut également traverser ce quadrant. Un événement peut commencer sous la forme d'un échec gris, puis devenir un échec détecté, puis passer à un échec masqué, puis peut-être revenir à un échec gris. Il n'existe pas de cycle défini et il existe presque toujours un risque de récurrence d'une défaillance jusqu'à ce que sa cause première soit traitée.

La conclusion que nous en tirons est que les charges de travail ne peuvent pas toujours compter sur le système sous-jacent pour détecter et atténuer les défaillances. Quelle que soit la complexité et

la résilience du système sous-jacent, il est toujours possible qu'une défaillance passe inaperçue ou reste en dessous du seuil de réaction. Les utilisateurs de ce système, comme App 1, doivent être équipés pour détecter rapidement et atténuer l'impact d'une panne grise. Cela nécessite de mettre en place des mécanismes d'observabilité et de rétablissement pour ces situations.

Exemple d'échec de Gray

Les pannes grises peuvent avoir un impact sur les systèmes multi-AZ dansAWS. Prenons l'exemple d'une flotte deAmazon EC2instances d'un groupe Auto Scaling déployées dans trois zones de disponibilité. Ils se connectent tous à une base de données Amazon Aurora dans une seule zone de disponibilité. Ensuite, une panne grise se produit qui a un impact sur la mise en réseau entre la zone de disponibilité 1 et la zone de disponibilité 2. Cette déficience a pour conséquence qu'un pourcentage des connexions de base de données nouvelles et existantes provenant d'instances de la zone de disponibilité 1 échouent. Cette situation est illustrée dans la figure suivante.



Une panne grise qui a un impact sur les connexions aux bases de données depuis les instances de la zone de disponibilité 1

Dans cet exemple, Amazon EC2 considère que les instances de la zone de disponibilité 1 sont saines car elles continuent à être transmisesvérifications de l'état du système et de l'instance. Amazon EC2 Auto Scaling ne détecte pas non plus d'impact direct sur aucune zone de disponibilité et continue decapacité de lancement dans les zones de disponibilité configurées. Le Network Load Balancer (NLB) considère également que les instances qui le sous-tendent sont saines, tout comme les contrôles de santé de Route 53 effectués sur le point de terminaison NLB. De même, Amazon Relational Database Service (Amazon RDS) considère que le cluster de bases de données est sain et ne le considère pasdéclencher un basculement automatique. Nous avons de nombreux services différents qui considèrent tous que leurs services et leurs ressources sont sains, mais la charge de travail détecte une défaillance qui a une incidence sur leur disponibilité. Il s'agit d'un échec flagrant.

Réagir aux pannes grises

Lorsque vous rencontrez une défaillance grise dans votreAWSenvironnement, vous avez généralement trois options disponibles :

- Ne faites rien et attendez que la déficience prenne fin.
- Si la déficience est limitée à une seule zone de disponibilité, évacuez cette zone de disponibilité.
- Basculement vers un autreRégion AWSet profitez des avantages deAWSIsolement régional pour atténuer l'impact.

De nombreuxAWSles clients sont d'accord avec la première option pour la majorité de leurs charges de travail. Ils acceptent d'avoir une extension éventuelleObjectif de temps de restauration (RTO)avec le compromis qu'ils n'ont pas eu à créer de solutions d'observabilité ou de résilience supplémentaires. D'autres clients choisissent de mettre en œuvre la troisième option,Reprise après sinistre multirégionale(DR), en tant que plan d'atténuation pour un certain nombre de modes de défaillance. Les architectures multirégionales peuvent bien fonctionner dans ces scénarios. Cependant, l'utilisation de cette approche comporte quelques inconvénients (voirAWSPrincipes fondamentaux relatifs à plusieurs régionspour une discussion complète sur les considérations multirégionales).

Tout d'abord, la création et l'exploitation d'une architecture multirégionale peuvent s'avérer difficiles, complexes et potentiellement coûteuses. Les architectures multirégionales nécessitent un examen attentif des architecturesStratégie DRvous sélectionnez. Il n'est peut-être pas rentable de mettre en

œuvre une solution de reprise après sinistre active-active multirégionale uniquement pour gérer les déficiences zonales, tandis qu'une stratégie de sauvegarde et de restauration peut ne pas répondre à vos exigences de résilience. En outre, les basculements multirégionaux doivent être pratiqués en continu en production afin que vous soyez sûr qu'ils fonctionneront en cas de besoin. Tout cela nécessite beaucoup de temps et de ressources dédiés à la création, à l'exploitation et aux tests.

Ensuite, la réplication des données entreRégions AWSen utilisantAWSles services actuels sont tous effectués de manière asynchrone. La réplication asynchrone peut entraîner une perte de données. Cela signifie que lors d'un basculement régional, il existe un risque de perte de données et d'incohérence. Votre tolérance à la quantité de perte de données est définie comme votreObjectif de point de restauration (RPO). Les clients, pour lesquels une forte cohérence des données est requise, doivent créer des systèmes de réconciliation pour résoudre ces problèmes de cohérence lorsque la région principale est à nouveau disponible. Ils doivent également créer leurs propres systèmes de réplication synchrone ou à double écriture, ce qui peut avoir un impact significatif sur la latence des réponses, les coûts et la complexité. Ils font également de la région secondaire une dépendance absolue pour chaque transaction, ce qui peut potentiellement réduire la disponibilité de l'ensemble du système.

Enfin, pour de nombreuses charges de travail utilisant une approche active/en veille, le temps nécessaire pour effectuer le basculement vers une autre région est non nul. Il se peut que votre portefeuille de charges de travail doive être transféré dans la région principale dans un ordre spécifique, qu'il soit nécessaire de vider les connexions ou d'arrêter des processus spécifiques. Ensuite, il peut être nécessaire de rétablir les services dans un ordre spécifique. Les nouvelles ressources peuvent également devoir être provisionnées ou nécessiter du temps pour passer les contrôles de santé requis avant d'être mises en service. Ce processus de basculement peut être vécu comme une période d'indisponibilité totale. C'est ce qui préoccupe les RTO.

Au sein d'une région, de nombreuxAWSles services offrent une persistance des données très cohérente. Les déploiements multi-AZ d'Amazon RDS utilisentréplication synchrone. Service de stockage simple Amazon(Amazon S3) offresfortread-after-writeconsistance. Amazon Elastic Block StorageOffres (Amazon EBS)instantanés cohérents en cas de crash sur plusieurs volumes. Amazon DynamoDBpeuteffectuer des lectures très cohérentes. Ces fonctionnalités peuvent vous aider à obtenir un RPO inférieur (dans la plupart des cas, un RPO nul) dans une seule région par rapport aux architectures multirégions.

L'évacuation d'une zone de disponibilité peut avoir un RTO inférieur à celui d'une stratégie multirégionale, car votre infrastructure et vos ressources sont déjà provisionnées entre les zones de disponibilité. Au lieu de devoir ordonner avec soin la mise hors service et la restauration des

services, ou de vider les connexions, les architectures multi-AZ peuvent continuer à fonctionner de manière statique lorsqu'une zone de disponibilité est altérée. Au lieu d'une période d'indisponibilité complète qui peut survenir lors d'un basculement régional, lors d'une évacuation d'une zone de disponibilité, de nombreux systèmes peuvent ne subir qu'une légère dégradation, car le travail est transféré vers les zones de disponibilité restantes. Si le système a été conçu pour êtrestatiquement stableen cas de défaillance d'une zone de disponibilité (dans ce cas, cela signifierait que la capacité est préprovisionnée dans les autres zones de disponibilité pour absorber la charge), les clients concernés peuvent ne pas en ressentir le moindre impact.

> Il est possible que la dégradation d'une seule zone de disponibilité ait un impact sur une ou plusieursAWS Services régionauxen plus de votre charge de travail. Si vous constatez un impact régional, vous devez traiter l'événement comme une perturbation du service régional, bien que la source de cet impact provienne d'une seule zone de disponibilité. L'évacuation d'une zone de disponibilité ne résoudra pas ce type de problème. Utilisez les plans d'intervention que vous avez mis en place pour répondre à une panne du service régional lorsque cela se produit.

Le reste de ce document se concentre sur la deuxième option, à savoir l'évacuation de la zone de disponibilité, afin de réduire les RTO et RPO en cas de pannes grises d'une seule AZ. Ces modèles peuvent contribuer à améliorer la valeur et l'efficacité des architectures multi-AZ et, pour la plupart des classes de charges de travail, peuvent réduire la nécessité de créer des architectures multirégions pour gérer ce type d'événements.

Observabilité multi-AZ

Pour pouvoir évacuer une zone de disponibilité lors d'un événement isolé dans une seule zone de disponibilité, vous devez d'abord être en mesure de détecter que la panne est, en fait, isolée dans une seule zone de disponibilité. Cela nécessite une visibilité très fidèle du comportement du système dans chaque zone de disponibilité. De nombreux AWS services fournissent out-of-the-box des indicateurs qui fournissent des informations opérationnelles sur vos ressources. Par exemple, Amazon EC2 fournit de nombreux indicateurs tels que CPU l'utilisation, les lectures et écritures sur disque, ainsi que le trafic réseau entrant et sortant.

Toutefois, lorsque vous créez des charges de travail qui utilisent ces services, vous avez besoin de plus de visibilité que ces indicateurs standard. Vous souhaitez avoir une visibilité sur l'expérience client fournie par votre charge de travail. En outre, vos indicateurs doivent être alignés sur les zones de disponibilité dans lesquelles ils sont produits. C'est l'information dont vous avez besoin pour détecter les défaillances de gris observables de manière différentielle. Ce niveau de visibilité nécessite des instruments.

L'instrumentation nécessite l'écriture d'un code explicite. Ce code doit notamment enregistrer la durée des tâches, compter le nombre d'éléments réussis ou échoués, collecter des métadonnées sur les demandes, etc. Vous devez également définir des seuils à l'avance pour définir ce qui est considéré comme normal et ce qui ne l'est pas. Vous devez définir les objectifs et les différents seuils de gravité relatifs à la latence, à la disponibilité et au nombre d'erreurs dans votre charge de travail. L'article de la bibliothèque Amazon Builders' Instrumenting distributed systems for operational visibility fournit un certain nombre de bonnes pratiques.

Les métriques doivent être générées à la fois du côté serveur et du côté client. L'une des meilleures pratiques pour générer des indicateurs côté client et comprendre l'expérience client consiste à utiliser des canaries, un logiciel qui analyse régulièrement votre charge de travail et enregistre les indicateurs.

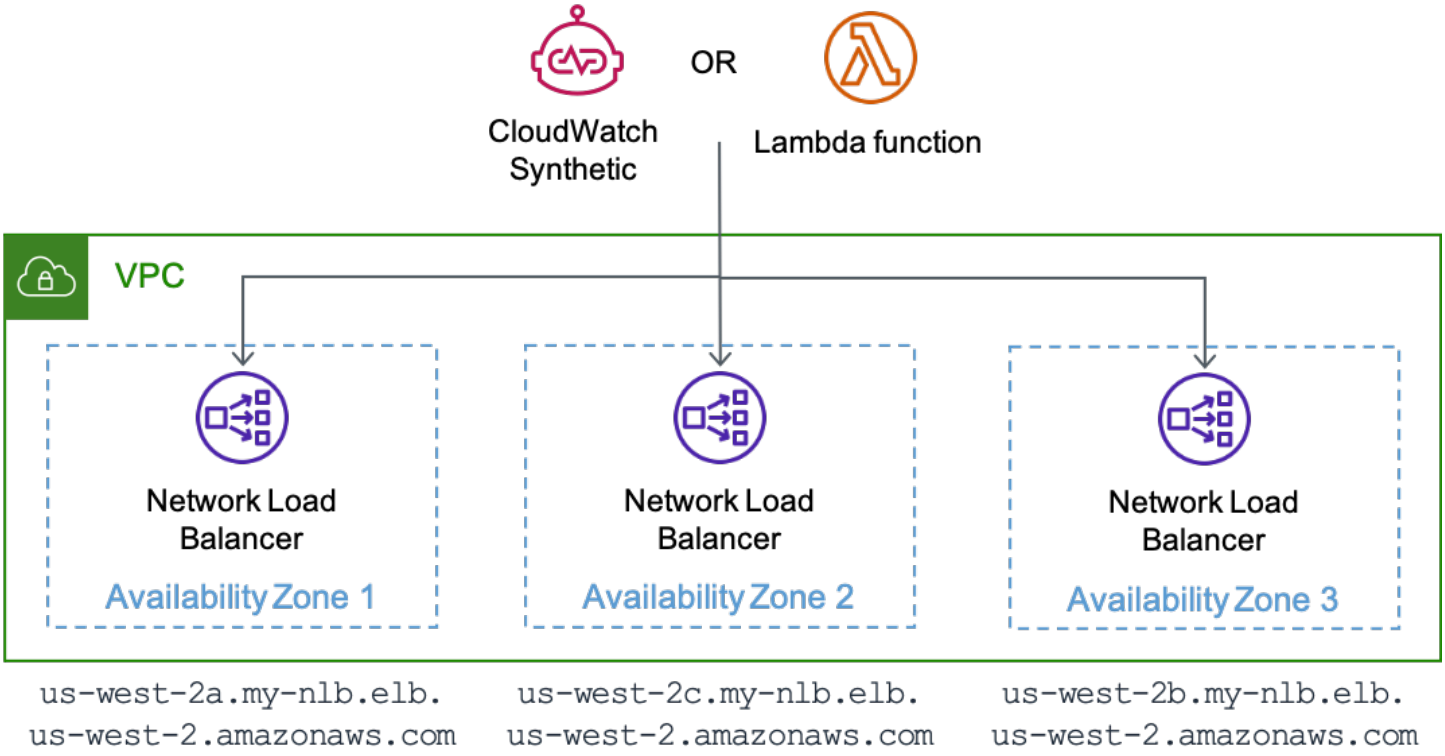
En plus de produire ces indicateurs, vous devez également comprendre leur contexte. Pour ce faire, vous pouvez utiliser des dimensions. Les dimensions confèrent à une métrique une identité unique et aident à expliquer ce que les métriques vous indiquent. Pour les métriques utilisées pour identifier les défaillances de votre charge de travail (par exemple, latence, disponibilité ou nombre d'erreurs), vous devez utiliser des dimensions alignées sur vos limites d'isolation des pannes.

Par exemple, si vous exécutez un service Web dans une région, dans plusieurs zones de disponibilité, à l'aide d'un framework Web M odel-view-controller (MVC), vous devez utiliser`Region`,,

`Availability Zone ID`, `Controller`, `Action`, et `InstanceId` comme dimensions pour vos ensembles de dimensions (si vous utilisiez des microservices, vous pouvez utiliser le nom du service et la HTTP méthode au lieu du contrôleur et des noms d'action). Cela est dû au fait que vous vous attendez à ce que les différents types de défaillances soient isolés par ces limites. Vous ne vous attendriez pas à ce qu'un bogue dans le code de votre service Web affectant sa capacité à répertorier des produits ait également un impact sur la page d'accueil. De même, vous ne vous attendez pas à ce qu'un EBS volume complet sur une seule EC2 instance empêche EC2 les autres instances de diffuser votre contenu Web. La dimension ID de zone de disponibilité vous permet d'identifier de manière cohérente les impacts liés à la zone de disponibilité. Comptes AWS Vous pouvez trouver l'ID de zone de disponibilité dans vos charges de travail de différentes manières. Reportez-vous à Annexe A — Obtenir l'ID de la zone de disponibilité pour quelques exemples.

Bien que ce document utilise principalement Amazon EC2 comme ressource de calcul dans les exemples, il `InstanceId` pourrait être remplacé par un ID de conteneur pour les ressources de calcul Amazon Elastic Container Service (AmazonECS) et Amazon Elastic Kubernetes Service (EKSAmazon) en tant que composants de vos dimensions.

Vos canaris peuvent également utiliser`Controller`, `Action`, `AZ-ID`, et `Region` comme dimensions dans leurs métriques si vous disposez de points de terminaison zonaux pour votre charge de travail. Dans ce cas, alignez vos canaris pour qu'ils s'exécutent dans la zone de disponibilité qu'ils testent. Cela garantit que si un événement de zone de disponibilité isolé a un impact sur la zone de disponibilité dans laquelle votre Canary fonctionne, il n'enregistre pas de statistiques indiquant qu'une autre zone de disponibilité testée ne semble pas saine. Par exemple, votre Canary peut tester chaque point de terminaison zonal pour un service situé derrière un Network Load Balancer NLB () ou un Application Load Balancer () en utilisant ALB ses noms de zone. DNS



Un canari utilisant des CloudWatch Synthetics ou AWS Lambda une fonction testant chaque extrémité zonale d'un NLB

En produisant des métriques avec ces dimensions, vous pouvez établir des CloudWatch alarmes Amazon qui vous avertissent lorsque des changements de disponibilité ou de latence surviennent dans ces limites. Vous pouvez également analyser rapidement ces données à l'aide de tableaux de bord. Pour utiliser efficacement les métriques et les journaux, Amazon CloudWatch propose le format de métrique intégré (EMF) qui vous permet d'intégrer des métriques personnalisées aux données des journaux. CloudWatchextrait automatiquement les métriques personnalisées afin que vous puissiez les visualiser et les analyser. AWS fournit plusieurs bibliothèques clientes pour différents langages de programmation, ce qui facilite la prise en mainEMF. Ils peuvent être utilisés avec AmazonEC2, Amazon ECS EKS AWS Lambda, Amazon et dans des environnements sur site. Grâce aux statistiques intégrées à vos journaux, vous pouvez également utiliser Amazon CloudWatch Contributor Insights pour créer des graphiques de séries chronologiques présentant les données des contributeurs. Dans ce scénario, nous pourrions afficher des données groupées par dimensions telles que `AZ-ID` `InstanceId`, ou `Controller` ainsi que tout autre champ du journal tel que `SuccessLatency` ou`HttpResponseCode`.

```
{
  "_aws": {
```

```
    "Timestamp": 1634319245221,
    "CloudWatchMetrics": [
      {
        "Namespace": "workloadname/frontend",
        "Metrics": [
          { "Name": "2xx", "Unit": "Count" },
          { "Name": "3xx", "Unit": "Count" },
          { "Name": "4xx", "Unit": "Count" },
          { "Name": "5xx", "Unit": "Count" },
          { "Name": "SuccessLatency", "Unit": "Milliseconds" }
        ],
        "Dimensions": [
          [ "Controller", "Action", "Region", "AZ-ID", "InstanceId"],
          [ "Controller", "Action", "Region", "AZ-ID"],
          [ "Controller", "Action", "Region"]
        ]
      }
    ],
    "LogGroupName": "/loggroupname"
  },
  "CacheRefresh": false,
  "Host": "use1-az2-name.example.com",
  "SourceIp": "34.230.82.196",
  "TraceId": "|e3628548-42e164ee4d1379bf.",
  "Path": "/home",
  "OneBox": false,
  "Controller": "Home",
  "Action": "Index",
  "Region": "us-east-1",
  "AZ-ID": "use1-az2",
  "InstanceId": "i-01ab0b7241214d494",
  "LogGroupName": "/loggroupname",
  "HttpResponseCode": 200,
  "2xx": 1,
  "3xx": 0,
  "4xx": 0,
  "5xx": 0,
  "SuccessLatency": 20
}
```


Ce journal comporte trois ensembles de dimensions. Ils progressent par ordre de granularité, de l'instance à la zone de disponibilité puis à la région (`Controller`et `Action` sont toujours inclus dans cet exemple). Ils permettent de créer des alarmes sur l'ensemble de votre charge de travail qui

indiquent quand une action spécifique du contrôleur a un impact sur une seule instance, dans une seule zone de disponibilité ou dans un ensemble Région AWS. Ces dimensions sont utilisées pour le nombre de mesures de HTTP réponse 2xx, 3xx, 4xx et 5xx, ainsi que pour la latence des mesures de demande réussies (si la demande échoue, elle enregistre également une métrique de latence de demande échouée). Le journal enregistre également d'autres informations telles que le HTTP chemin, l'adresse IP source du demandeur et indique si cette demande a nécessité l'actualisation du cache local. Ces points de données peuvent ensuite être utilisés pour calculer la disponibilité et la latence de chaque charge API de travail fournie.

> **Remarque sur l'utilisation des codes de HTTP réponse pour les mesures de disponibilité**
>
> En général, vous pouvez considérer les réponses 2xx et 3xx comme des réussites, et les réponses 5xx comme des échecs. Les codes de réponse 4xx se situent quelque part entre les deux. Ils sont généralement produits en raison d'une erreur du client. Peut-être qu'un paramètre est hors de portée, ce qui entraîne une réponse 400, ou qu'il demande quelque chose qui n'existe pas, ce qui entraîne une réponse 404. Vous ne compteriez pas ces réponses par rapport à la disponibilité de votre charge de travail. Cependant, cela peut également être le résultat d'un bogue dans le logiciel.
>
> Par exemple, si vous avez introduit une validation des entrées plus stricte qui rejette une demande qui aurait abouti auparavant, la réponse 400 peut être considérée comme une baisse de disponibilité. Ou peut-être que vous limitez le client et que vous renvoyez une réponse 429. Bien que la limitation d'un client protège votre service afin de maintenir sa disponibilité, du point de vue du client, le service n'est pas disponible pour traiter sa demande. Vous devrez décider si les codes de réponse 4xx font partie de votre calcul de disponibilité.

Bien que cette section ait décrit l'utilisation CloudWatch comme moyen de collecter et d'analyser des métriques, ce n'est pas la seule solution que vous pouvez utiliser. Vous pouvez également choisir d'envoyer des métriques vers Amazon Managed Service for Prometheus et Amazon Managed Grafana, une table Amazon DynamoDB, ou d'utiliser une solution de surveillance tierce. L'essentiel est que les métriques produites par votre charge de travail doivent contenir le contexte des limites d'isolation des pannes de votre charge de travail.

Avec des charges de travail qui produisent des métriques dont les dimensions sont alignées sur les limites d'isolation des pannes, vous pouvez créer une observabilité qui détecte les défaillances isolées dans les zones de disponibilité. Les sections suivantes décrivent trois approches

complémentaires permettant de détecter les défaillances résultant de la détérioration d'une seule zone de disponibilité.

Rubriques

Détection des défaillances à l'aide d'alarmes CloudWatch composites

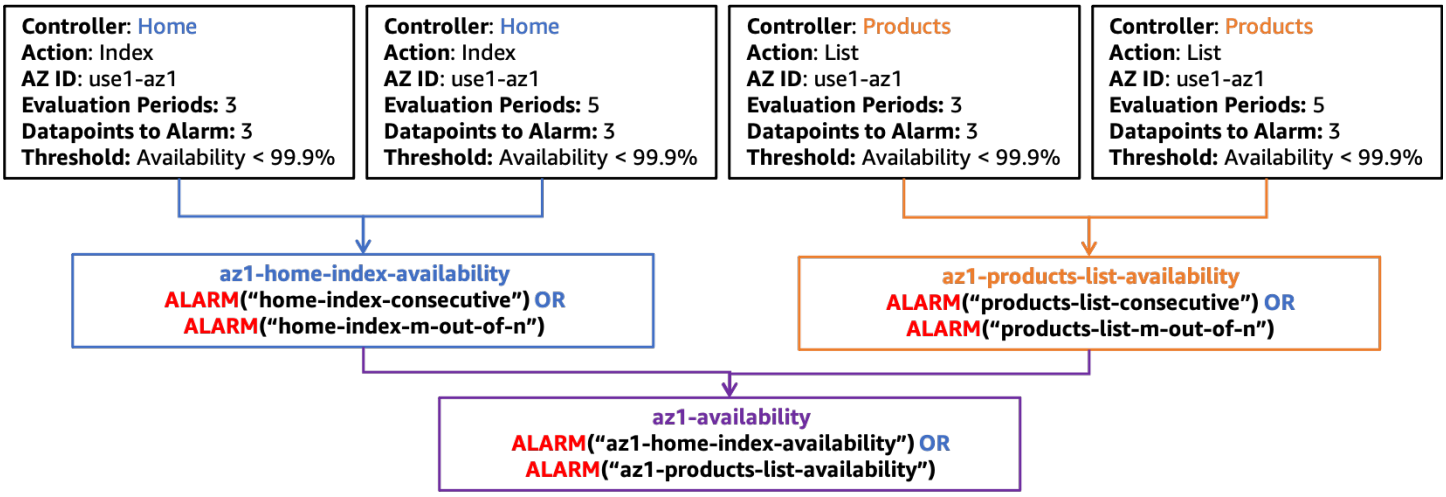
Dans CloudWatch les métriques, chaque ensemble de dimensions est une métrique unique, et vous pouvez créer une CloudWatch alarme pour chacune d'elles. Vous pouvez ensuite créer des alarmes CloudWatch composites Amazon pour agréger ces métriques.

Afin de détecter un impact avec précision, les exemples présentés dans ce paper utiliseront deux structures CloudWatch d'alarme différentes pour chaque dimension sur laquelle l'alarme est réglée. Chaque alarme utilisera une période d'une minute, ce qui signifie que la métrique est évaluée une fois par minute. La première approche consiste à utiliser trois points de données violés consécutifs en réglant les périodes d'évaluation et les points de données d'alarme à trois, ce qui signifie un impact pendant trois minutes au total. La deuxième approche consiste à utiliser un « M sur N » lorsque 3 points de données dans une fenêtre de cinq minutes sont violés en réglant les périodes d'évaluation à cinq et les points de données à alarme à trois. Cela permet de détecter un signal constant, ainsi qu'un signal qui fluctue sur une courte période. Les durées et le nombre de points de données contenus ici sont des suggestions. Utilisez des valeurs adaptées à vos charges de travail.

Détectez l'impact dans une seule zone de disponibilité

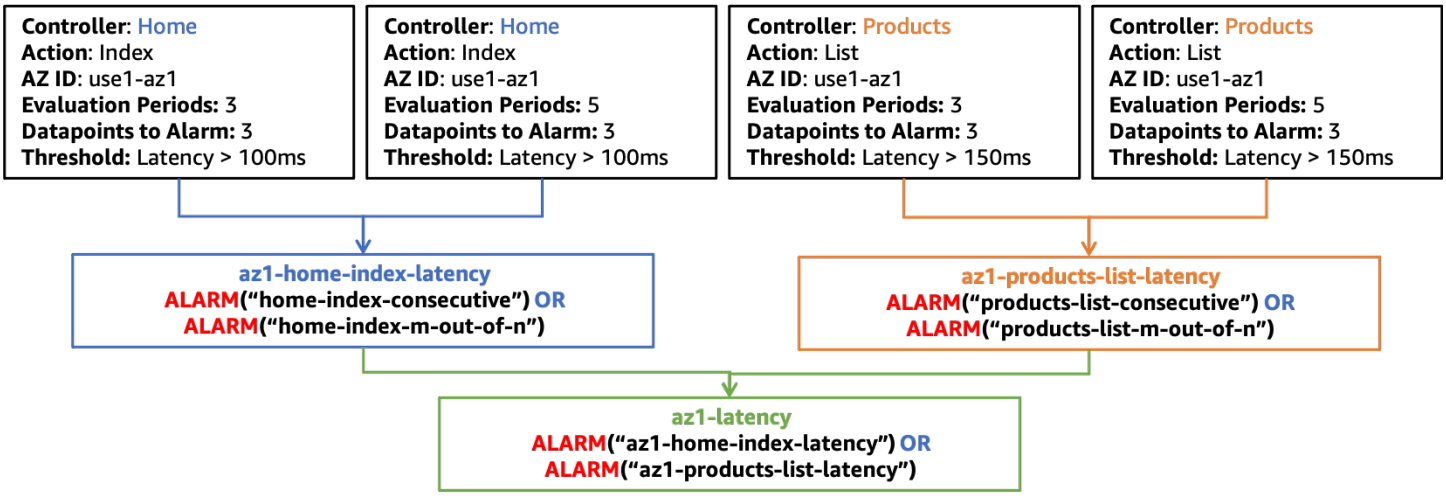
À l'aide de cette construction, considérez une charge de travail qui utilise `ControllerAction`,`InstanceId`,`AZ-ID`, et `Region` en tant que dimensions. La charge de travail comporte deux contrôleurs, Products et Home, et une action par contrôleur, List et Index respectivement. Il opère dans trois zones de disponibilité de la `us-east-1` région. Vous devez créer deux alarmes de disponibilité pour chacune `Controller` et `Action` une combinaison dans chaque zone de disponibilité, ainsi que deux alarmes de latence pour chacune d'entre elles. Ensuite, vous pouvez éventuellement choisir de créer une alarme composite pour vérifier la disponibilité de

chaque alarme `Controller` et d'`Action`une combinaison. Enfin, vous créez une alarme composite qui regroupe toutes les alarmes de disponibilité pour la zone de disponibilité. Cela est illustré dans la figure suivante pour une seule zone de disponibilité`use1-az1`, en utilisant l'alarme composite optionnelle pour chaque zone `Controller` et une `Action` combinaison (des alarmes similaires existeraient également pour les zones de `use1-az3` disponibilité `use1-az2` et, pour des raisons de simplicité, elles ne sont pas affichées).



Structure d'alarme composite pour une disponibilité dans *`use1-az1`*

Vous pouvez également créer une structure d'alarme similaire pour la latence, comme indiqué dans la figure suivante.



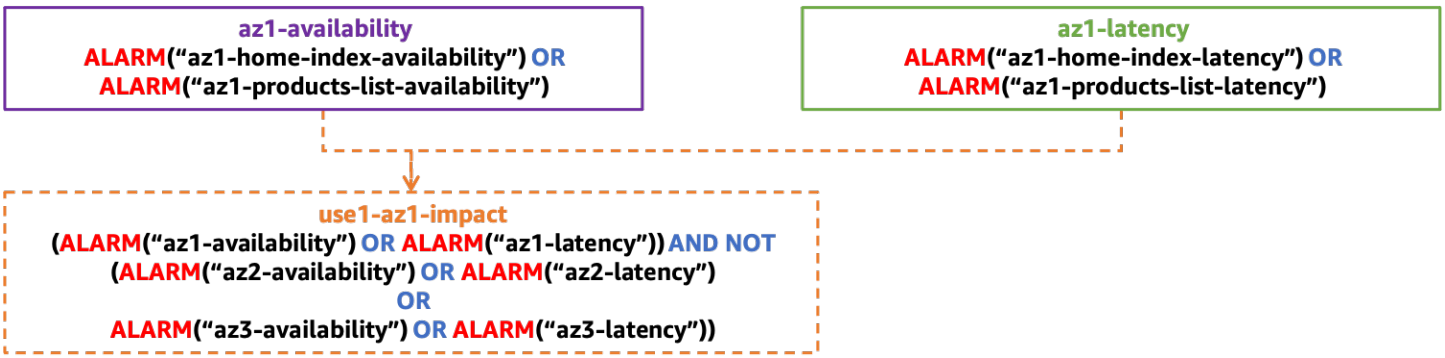
Structure d'alarme composite pour la latence dans *`use1-az1`*

Pour les autres figures de cette section, seules les alarmes `az1-latency` composites `az1-availability` et les alarmes seront affichées au niveau supérieur. Ces alarmes composites vous indiqueront si la disponibilité tombe en dessous ou si la latence augmente au-dessus des

seuils définis dans une zone de disponibilité donnée pour une partie quelconque de votre charge de travail. `az1-availability` `az1-latency` Vous pouvez également envisager de mesurer le débit pour détecter l'impact qui empêche votre charge de travail dans une seule zone de disponibilité de recevoir du travail. Vous pouvez également intégrer les alarmes produites à partir des métriques émises par vos canaris dans ces alarmes composites. Ainsi, si le côté serveur ou le côté client constatent des impacts sur la disponibilité ou la latence, l'alarme créera une alerte.

Assurez-vous que l'impact n'est pas régional

Un autre ensemble d'alarmes composites peut être utilisé pour garantir que seul un événement de zone de disponibilité isolé provoque l'activation de l'alarme. Pour ce faire, il faut s'assurer qu'une alarme composite de zone de disponibilité est dans l'`ALARM`état alors que les alarmes composites des autres zones de disponibilité sont dans `OK` cet état. Cela se traduira par une alarme composite par zone de disponibilité que vous utilisez. Un exemple est illustré dans la figure suivante (n'oubliez pas qu'il existe des alarmes de latence et de disponibilité dans `use1-az2` et`use1-az3`,`az2-latency`, et `az2-availability` `az3-latency`, `az3-availability`qui ne sont pas illustrées pour des raisons de simplicité).



Structure d'alarme composite pour détecter l'impact isolée à un seul AZ

Assurez-vous que l'impact n'est pas causé par une seule instance

Une seule instance (ou un faible pourcentage de votre parc global) peut avoir un impact disproportionné sur les indicateurs de disponibilité et de latence, ce qui pourrait donner l'impression que l'ensemble de la zone de disponibilité est affectée, alors qu'en fait ce n'est pas le cas. Il est plus rapide et tout aussi efficace de supprimer une seule instance problématique que d'évacuer une zone de disponibilité.

Les instances et les conteneurs sont généralement traités comme des ressources éphémères, fréquemment remplacés par des services tels que. AWS Auto Scaling Il est difficile de créer une

nouvelle CloudWatch alarme à chaque fois qu'une nouvelle instance est créée (mais cela est certainement possible en utilisant les hooks de cycle de vie d'Amazon EventBridge ou d'Amazon EC2 Auto Scaling). Vous pouvez plutôt utiliser CloudWatch Contributor Insights pour identifier le nombre de contributeurs aux métriques de disponibilité et de latence.

Par exemple, pour une application HTTP Web, vous pouvez créer une règle pour identifier les principaux contributeurs pour 5 x HTTP réponses dans chaque zone de disponibilité. Cela permettra d'identifier les instances qui contribuent à une baisse de disponibilité (notre indicateur de disponibilité défini ci-dessus est déterminé par la présence d'erreurs 5xx). À l'aide de l'exemple du EMF journal, créez une règle à l'aide de la clé de`InstanceId`. Filtrez ensuite le journal en fonction du `HttpResponseCode` champ. Cet exemple est une règle pour la zone de `use1-az1` disponibilité.

```
{
    "AggregateOn": "Count",
    "Contribution": {
        "Filters": [
            {
                "Match": "$.InstanceId",
                "IsPresent": true
            },
            {
                "Match": "$.HttpStatusCode",
                "IsPresent": true
            },
            {
                "Match": "$.HttpStatusCode",
                "GreaterThan": 499
            },
            {
                "Match": "$.HttpStatusCode",
                "LessThan": 600
            },
            {
                "Match": "$.AZ-ID",
                "In": ["use1-az1"]
            },
        ],
        "Keys": [
            "$.InstanceId"
        ]
    },
    "LogFormat": "JSON",
```

```
    "LogGroupNames": [
        "/loggroupname"
    ],
    "Schema": {
        "Name": "CloudWatchLogRule",
        "Version": 1
    }
}
```

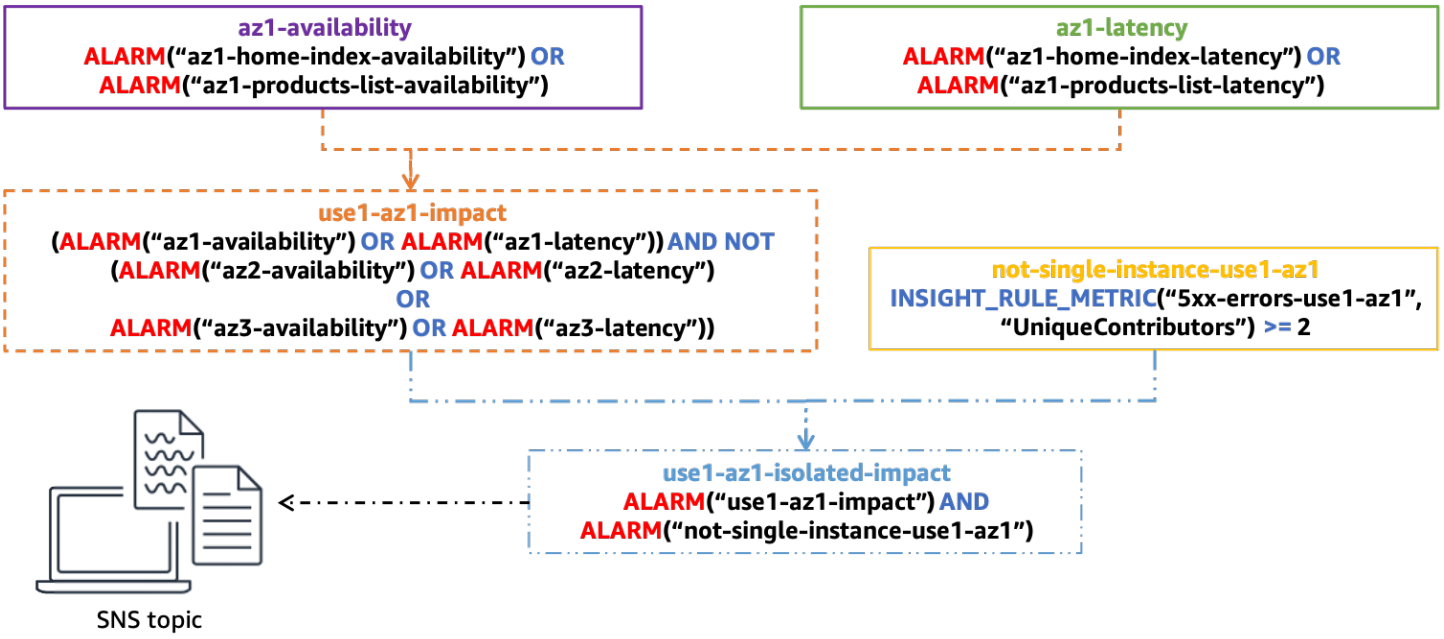
CloudWatch des alarmes peuvent également être créées en fonction de ces règles. Vous pouvez créer des alarmes en fonction des règles de Contributor Insights à l'aide des mathématiques métriques et de la `INSIGHT_RULE_METRIC` fonction associée à la `UniqueContributors` métrique. Vous pouvez également créer des règles supplémentaires pour Contributor Insights avec des CloudWatch alarmes pour des indicateurs tels que la latence ou le nombre d'erreurs, en plus de celles relatives à la disponibilité. Ces alarmes peuvent être utilisées avec les alarmes composites d'impact de zone de disponibilité isolées afin de garantir que des instances uniques n'activent pas l'alarme. La métrique de la règle d'analyse pour `use1-az1` peut ressembler à ce qui suit :

```
INSIGHT_RULE_METRIC("5xx-errors-use1-az1", "UniqueContributors")
```

Vous pouvez définir une alarme lorsque cette métrique est supérieure à un seuil ; dans cet exemple, deux. Il est activé lorsque le nombre de contributeurs uniques aux réponses 5xx dépasse ce seuil, ce qui indique que l'impact provient de plus de deux instances. La raison pour laquelle cette alarme utilise une comparaison supérieure à plutôt qu'inférieure à est de s'assurer qu'une valeur nulle pour les contributeurs uniques ne déclenche pas l'alarme. Cela indique que l'impact ne provient pas d'une seule instance. Ajustez ce seuil en fonction de votre charge de travail individuelle. Un guide général consiste à faire de ce chiffre 5 % ou plus du total des ressources de la zone de disponibilité. Plus de 5 % de vos ressources touchées présentent une signification statistique, compte tenu de la taille de l'échantillon suffisante.

Tout assembler

La figure suivante montre la structure d'alarme composite complète pour une seule zone de disponibilité :



Structure d'alarme composite complète pour déterminer l'impact mono-AZ

L'alarme composite finale est activée lorsque l'alarme composite indiquant un impact de zone de disponibilité isolé lié à la latence ou à `ALARM` la disponibilité est activée et lorsque l'alarme basée sur la règle Contributor Insights pour cette même zone de disponibilité est également active `ALARM` (ce qui signifie que l'impact est supérieur à une seule instance). `use1-az1-isolated-impact` `use1-az1-aggregate-alarm` `not-single-instance-use1-az1` Vous devez créer cette pile d'alarmes pour chaque zone de disponibilité utilisée par votre charge de travail.

Vous pouvez joindre une alerte Amazon Simple Notification Service (AmazonSNS) à cette dernière alarme. Toutes les alarmes précédentes sont configurées sans action. L'alerte pourrait informer un opérateur par e-mail pour qu'il lance une enquête manuelle. Cela pourrait également initier l'automatisation pour évacuer la zone de disponibilité. Cependant, attention à l'automatisation des bâtiments pour répondre à ces alertes. Après l'évacuation d'une zone de disponibilité, les taux d'erreur accrus devraient être atténués et l'alarme devrait revenir à son `OK` état normal. Si un impact se produit dans une autre zone de disponibilité, il est possible que l'automatisation évacue une deuxième ou une troisième zone de disponibilité, supprimant ainsi toute la capacité disponible de la charge de travail. L'automatisation doit vérifier si une évacuation a déjà été effectuée avant de prendre des mesures. Vous devrez peut-être également augmenter les ressources dans d'autres zones de disponibilité pour qu'une évacuation soit réussie.

Lorsque vous ajoutez de nouveaux contrôleurs ou actions à votre application MVC Web, à un nouveau microservice ou, en général, à toute fonctionnalité supplémentaire que vous souhaitez

surveiller séparément, il vous suffit de modifier quelques alarmes dans cette configuration. Vous allez créer de nouvelles alarmes de disponibilité et de latence pour cette nouvelle fonctionnalité, puis les ajouter aux alarmes composites de disponibilité et de latence alignées sur la zone de disponibilité appropriée, `az1-latency` comme `az1-availability` dans l'exemple que nous avons utilisé ici. Les autres alarmes composites restent statiques une fois configurées. Cela simplifie le processus d'intégration de nouvelles fonctionnalités avec cette approche.

Détection des défaillances grâce à la détection des valeurs aberrantes

Une lacune par rapport à l'approche précédente peut survenir lorsque vous constatez des taux d'erreur élevés dans plusieurs zones de disponibilité pour une raison non corrélée. Imaginez un scénario dans lequel des EC2 instances sont déployées dans trois zones de disponibilité et où votre seuil d'alarme de disponibilité est de 99 %. Ensuite, une défaillance d'une seule zone de disponibilité se produit, isolant de nombreuses instances et faisant chuter la disponibilité dans cette zone à 55 %. Dans le même temps, mais dans une autre zone de disponibilité, une EC2 instance unique épuise tout le stockage de son EBS volume et ne peut plus écrire de fichiers journaux. Cela l'amène à renvoyer des erreurs, mais il passe tout de même les tests de santé de l'équilibreur de charge, car ceux-ci ne déclenchent pas l'écriture d'un fichier journal. Cela se traduit par une baisse de la disponibilité à 98 % dans cette zone de disponibilité. Dans ce cas, votre seule alarme d'impact de zone de disponibilité ne s'activera pas car vous constatez un impact sur la disponibilité dans plusieurs zones de disponibilité. Cependant, vous pouvez tout de même atténuer la quasi-totalité de l'impact en évacuant la zone de disponibilité altérée.

Dans certains types de charges de travail, il est possible que vous rencontriez régulièrement des erreurs dans toutes les zones de disponibilité où l'indicateur de disponibilité précédent peut ne pas être utile. Prenons par AWS Lambda exemple. AWS permet aux clients de créer leur propre code à exécuter dans la fonction Lambda. Pour utiliser le service, vous devez télécharger votre code dans un ZIP fichier, y compris les dépendances, et définir le point d'entrée de la fonction. Mais parfois, les clients se trompent sur cette partie. Par exemple, ils peuvent oublier une dépendance critique dans le ZIP fichier ou mal saisir le nom de la méthode dans la définition de la fonction Lambda. Cela empêche l'invocation de la fonction et entraîne une erreur. AWS Lambda voit ce genre d'erreurs tout le temps, mais cela n'indique pas que quelque chose ne va pas nécessairement mal. Cependant, un problème tel qu'une altération de la zone de disponibilité peut également provoquer l'apparition de ces erreurs.

Pour détecter un signal dans ce type de bruit, vous pouvez utiliser la détection des valeurs aberrantes afin de déterminer s'il existe un écart statistiquement significatif dans le nombre d'erreurs entre les zones de disponibilité. Bien que nous constations des erreurs dans plusieurs zones de disponibilité, en cas de défaillance réelle de l'une d'entre elles, nous pouvons nous attendre à un taux d'erreur beaucoup plus élevé dans cette zone de disponibilité par rapport aux autres, voire à un taux potentiellement bien inférieur. Mais combien plus haut ou plus bas ?

L'une des méthodes de cette analyse consiste à utiliser un test du Khi carré (x^2) pour détecter les différences statistiquement significatives dans les taux d'erreur entre les zones de disponibilité (il existe de nombreux algorithmes différents pour effectuer la détection des valeurs aberrantes). Voyons comment fonctionne le test du chi carré.

Un test du Khi deux évalue la probabilité qu'une certaine distribution des résultats se produise. Dans ce cas, nous nous intéressons à la distribution des erreurs sur un ensemble défini deAZs. Pour cet exemple, pour faciliter les calculs, considérez quatre zones de disponibilité.

Établissez d'abord l'hypothèse nulle, qui définit ce que vous pensez être le résultat par défaut. Dans ce test, l'hypothèse nulle est que vous vous attendez à ce que les erreurs soient réparties uniformément dans chaque zone de disponibilité. Générez ensuite l'hypothèse alternative, selon laquelle les erreurs ne sont pas réparties uniformément, ce qui indique une altération de la zone de disponibilité. Vous pouvez désormais tester ces hypothèses à l'aide des données de vos indicateurs. À cette fin, vous allez échantillonner vos statistiques sur une fenêtre de cinq minutes. Supposons que vous obteniez 1 000 points de données publiés dans cette fenêtre, dans lesquels vous voyez 100 erreurs au total. Vous vous attendez à ce qu'avec une distribution uniforme, les erreurs se produisent 25 % du temps dans chacune des quatre zones de disponibilité. Supposons que le tableau suivant montre ce que vous attendiez par rapport à ce que vous avez réellement vu.

Tableau 1 : Erreurs attendues et réelles observées

AZ	Expected	Réel
`use1-az1`	25	20
`use1-az2`	25	20
`use1-az3`	25	25
`use1-az4`	25	35

Donc, vous voyez qu'en réalité, la distribution n'est pas uniforme. Cependant, vous pourriez penser que cela s'est produit en raison d'un certain degré de hasard dans les points de données que vous avez échantillonnés. Il existe un certain niveau de probabilité que ce type de distribution se produise dans l'échantillon tout en supposant que l'hypothèse nulle est vraie. Cela amène à la question suivante : Quelle est la probabilité d'obtenir un résultat au moins aussi extrême ? Si cette probabilité est inférieure à un seuil défini, vous rejetez l'hypothèse nulle. Pour être statistiquement significative, cette probabilité doit être de 5 % ou moins. [1]

[1] Craparo, Robert M. (2007). « Niveau de signification ». Dans Salkind, Neil J. Encyclopedia of Measurement and Statistics 3. Thousand Oaks, CA : SAGE Publications. pages 889 à 891. ISBN1-412-91611-9.

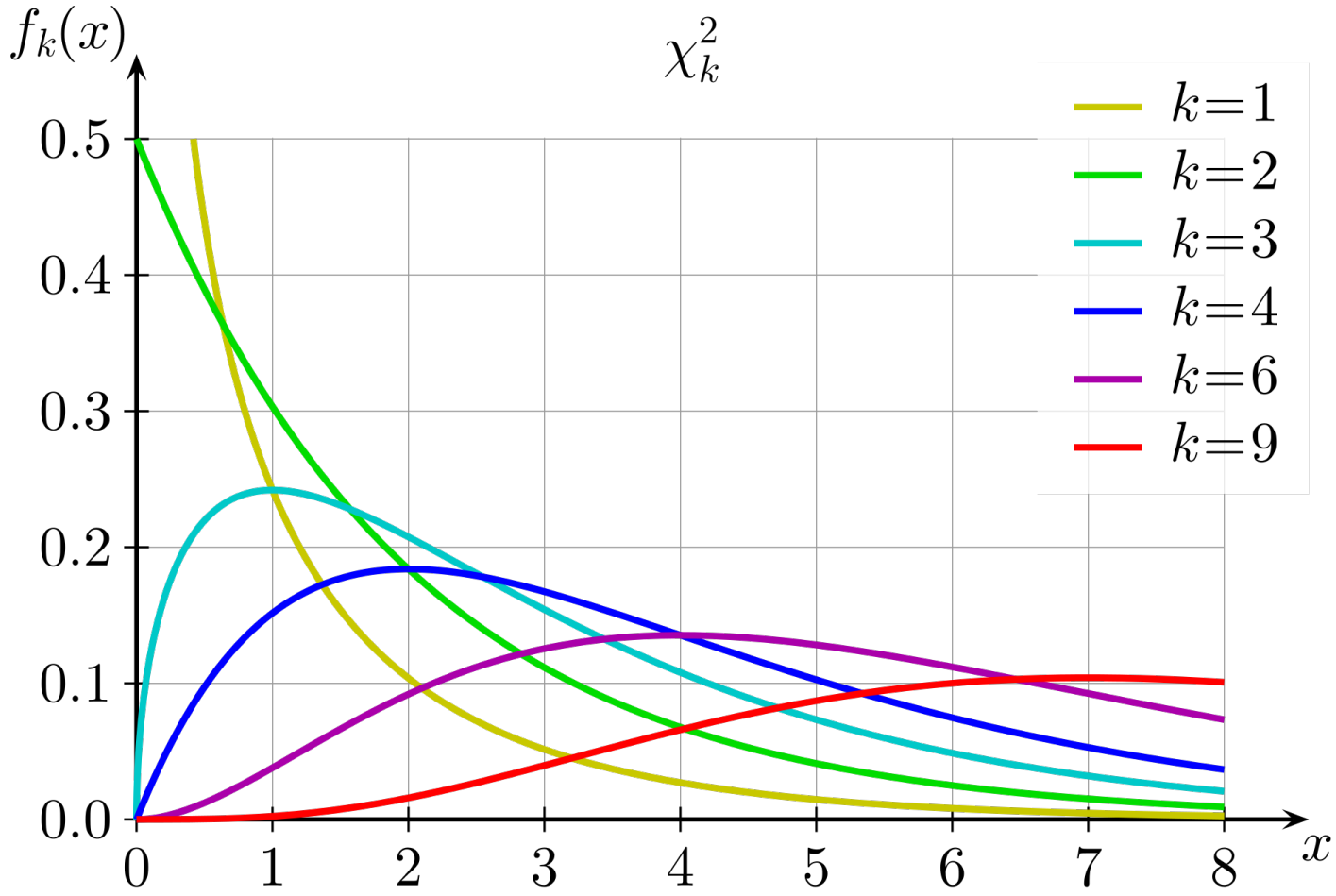
Comment calculez-vous la probabilité d'un tel résultat ? Vous utilisez la statistique x [2] qui fournit des distributions très bien étudiées et qui peut être utilisée pour déterminer la probabilité d'obtenir un résultat aussi extrême ou plus extrême à l'aide de cette formule.

$$E_i = expected\ observations\ of\ type\ i$$
$$O_i = actual\ observations\ of\ type\ i \qquad (1)$$
$$\chi^2 = \sum \frac{(O_i - E_i)^2}{E_i}$$

Dans notre exemple, cela se traduit par :

$$\chi^2 = \frac{(20-25)^2}{25} + \frac{(20-25)^2}{25} + \frac{(25-25)^2}{25} + \frac{(35-25)^2}{25}$$
$$\chi^2 = \frac{-5^2}{25} + \frac{-5^2}{25} + \frac{0^2}{25} + \frac{10^2}{25} \qquad (2)$$
$$\chi^2 = 1 + 1 + 0 + 4$$
$$\chi^2 = 6$$

Alors, qu'est-ce 6 que cela signifie en termes de probabilité ? Vous devez examiner une distribution du chi carré avec le degré de liberté approprié. La figure suivante montre plusieurs distributions du Khi carré pour différents degrés de liberté.



Distributions du Khi carré pour différents degrés de liberté

Le degré de liberté est calculé comme étant inférieur d'un au nombre de choix du test. Dans ce cas, étant donné qu'il existe quatre zones de disponibilité, le degré de liberté est de trois. Ensuite, vous voulez connaître l'aire sous la courbe (l'intégrale) pour x ≥ 6 sur le diagramme k = 3. Vous pouvez également utiliser un tableau précalculé avec les valeurs couramment utilisées pour obtenir une approximation de cette valeur.

Tableau 2 : Valeurs critiques du Khi au carré

Degrés de liberté	Probabilité inférieure à la valeur critique				
	0,75	0,90	0,95	0,99	0,999
1	1,323	2,706	3,841	6,635	10,828

Degrés de liberté	Probabilité inférieure à la valeur critique				
	0,75	0,90	0,95	0,99	0,999
2	2,773	4,605	5,991	9,210	13,816
3	4,108	6,251	7,815	11,345	16,266
4	5,385	7,779	9,488	13,277	18,467

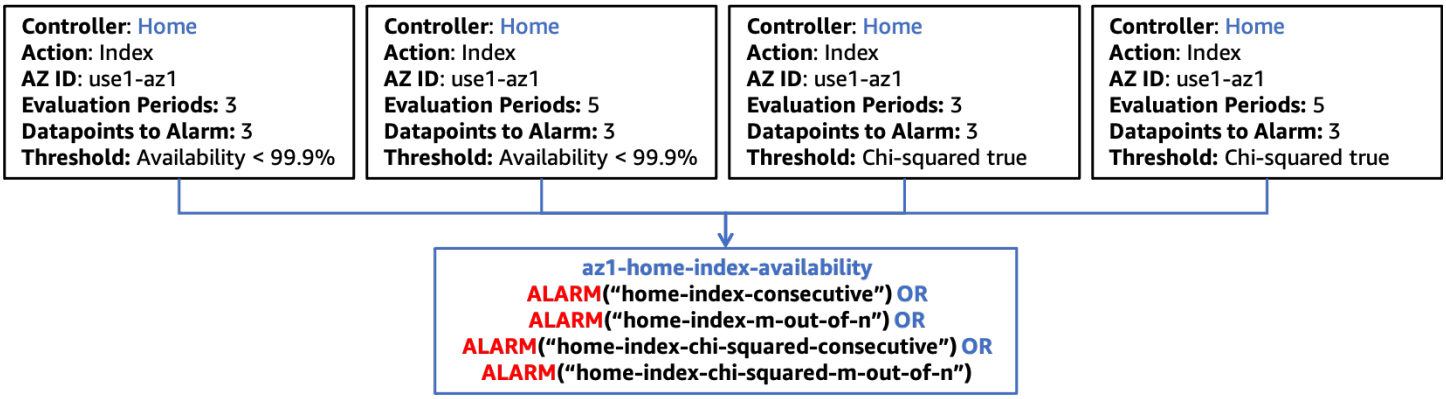
Pour trois degrés de liberté, la valeur du Khi carré de six se situe entre les colonnes de probabilité de 0,75 et 0,9. Cela signifie qu'il y a plus de 10 % de chances que cette distribution se produise, ce qui n'est pas inférieur au seuil de 5 %. Par conséquent, vous acceptez l'hypothèse nulle et déterminez qu'il n'y a pas de différence statistiquement significative dans les taux d'erreur entre les zones de disponibilité.

L'exécution d'un test de statistiques du Khi carré n'est pas prise en charge de manière native dans les mathématiques CloudWatch métriques. Vous devrez donc collecter les mesures d'erreur applicables CloudWatch et exécuter le test dans un environnement informatique tel que Lambda. Vous pouvez décider d'effectuer ce test au niveau d'un MVC contrôleur/action ou d'un microservice individuel, ou au niveau de la zone de disponibilité. Vous devez déterminer si une altération de la zone de disponibilité affecterait de la même manière chaque contrôleur/action ou microservice, ou si une DNS défaillance peut avoir un impact sur un service à faible débit et non sur un service à haut débit, ce qui pourrait masquer l'impact une fois agrégé. Dans les deux cas, sélectionnez les dimensions appropriées pour créer la requête. Le niveau de granularité aura également un impact sur les CloudWatch alarmes que vous créez.

Collectez la métrique du nombre d'erreurs pour chaque AZ et contrôleur/action dans une fenêtre temporelle spécifiée. Tout d'abord, calculez le résultat du test du Khi deux comme étant vrai (il y avait un biais statistiquement significatif) ou faux (il n'y en avait pas, c'est-à-dire que l'hypothèse nulle est valable). Si le résultat est faux, publiez un point de données égal à 0 dans votre flux de mesures pour obtenir des résultats au chi-carré pour chaque zone de disponibilité. Si le résultat est vrai, publiez un point de données de 1 pour la zone de disponibilité avec les erreurs les plus éloignées de la valeur attendue et un 0 pour les autres (voir un Annexe B — Exemple de calcul du Khi deux exemple de code pouvant être utilisé dans une fonction Lambda). Vous pouvez suivre la même approche que les alarmes de disponibilité précédentes en créant une alarme CloudWatch métrique de 3 lignes et une alarme métrique de 3 sur 5 CloudWatch en fonction des points de données produits par la fonction

Lambda. Comme dans les exemples précédents, cette approche peut être modifiée pour utiliser plus ou moins de points de données dans une fenêtre plus ou moins longue.

Ajoutez ensuite ces alarmes à votre alarme de disponibilité de zone de disponibilité existante pour la combinaison contrôleur/action, illustrée dans la figure suivante.



Intégration du test des statistiques du chi-carré aux alarmes composites

Comme indiqué précédemment, lorsque vous intégrez une nouvelle fonctionnalité à votre charge de travail, il vous suffit de créer les alarmes CloudWatch métriques appropriées spécifiques à cette nouvelle fonctionnalité et de mettre à jour le niveau suivant de la hiérarchie composite des alarmes pour inclure ces alarmes. Le reste de la structure de l'alarme reste statique.

Détection des défaillances des ressources zonales à instance unique

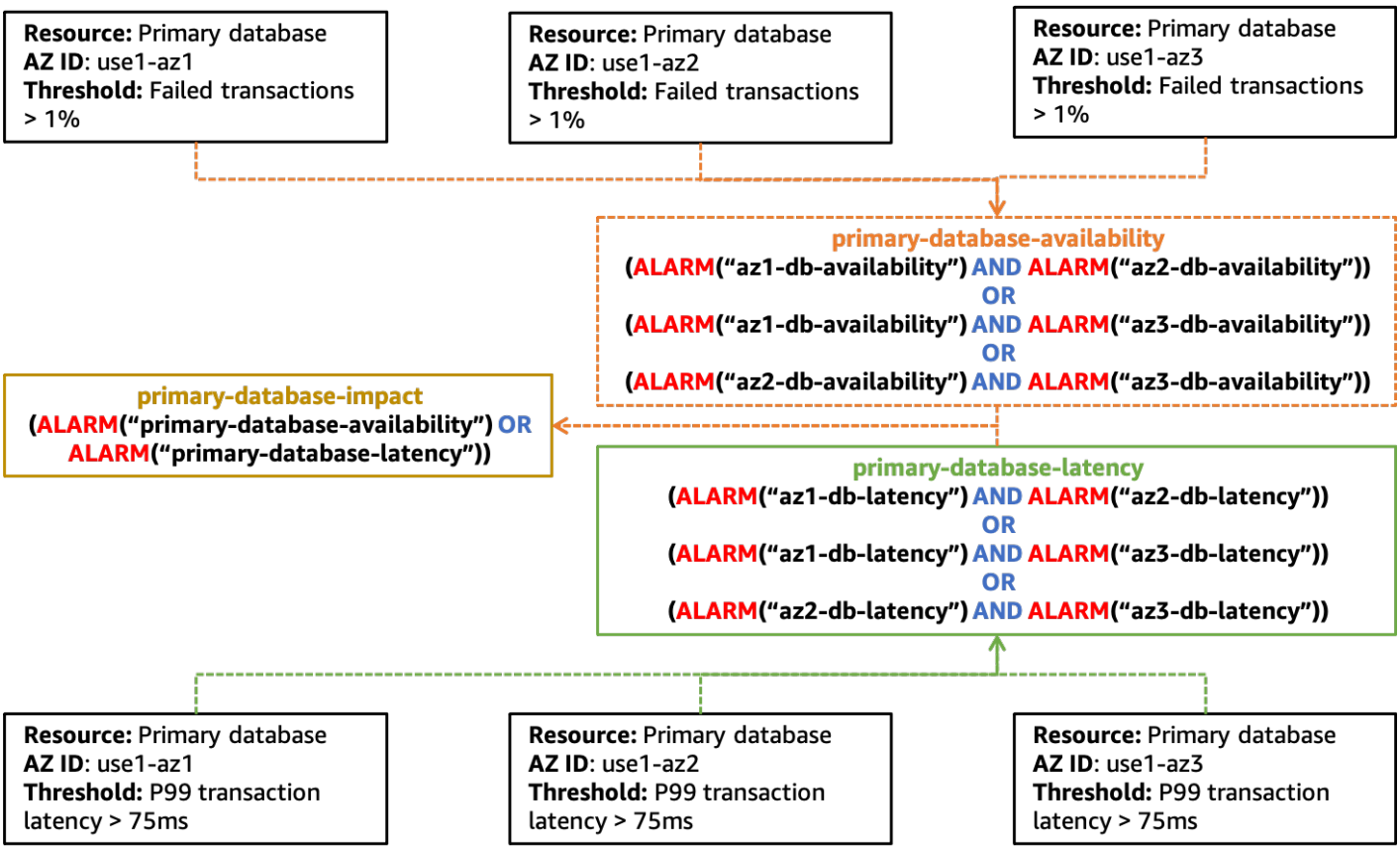
Dans certains cas, vous pouvez avoir une seule instance active d'une ressource zonale, le plus souvent des systèmes qui nécessitent un composant d'écriture unique tel qu'une base de données relationnelle (telle qu'AmazonRDS) ou un cache distribué (tel qu'Amazon ElastiCache (Redis OSS)). Si une seule défaillance de la zone de disponibilité affecte la zone de disponibilité dans laquelle se trouve la ressource principale, elle peut avoir un impact sur chaque zone de disponibilité qui accède à la ressource. Cela pourrait entraîner le franchissement des seuils de disponibilité dans chaque zone de disponibilité, ce qui signifie que la première approche ne permettrait pas d'identifier correctement la source d'impact de la zone de disponibilité unique. En outre, vous constaterez probablement des taux d'erreur similaires dans chaque zone de disponibilité, ce qui signifie que l'analyse des valeurs aberrantes ne détectera pas non plus le problème. Cela signifie que vous devez implémenter une observabilité supplémentaire pour détecter spécifiquement ce scénario.

Il est probable que la ressource qui vous préoccupe produise ses propres indicateurs concernant son état de santé, mais lors d'une détérioration de la zone de disponibilité, il se peut que cette ressource ne soit pas en mesure de fournir ces indicateurs. Dans ce scénario, vous devez créer ou mettre à jour des alarmes pour savoir si vous volez à l'aveugle. S'il existe des métriques importantes que vous surveillez déjà et que vous déclenchez une alarme, vous pouvez configurer l'alarme pour traiter les données manquantes comme des violations. Cela vous aidera à savoir si la ressource cesse de communiquer des données et si elle peut être incluse dans la même ressource d'affilée et si elle peut être utilisée dans les alarmes m sur n utilisées précédemment.

Il est également possible que, dans certains indicateurs indiquant l'état de santé de la ressource, celle-ci publie un point de données de valeur nulle en l'absence d'activité. Si la déficience empêche les interactions avec la ressource, vous ne pouvez pas utiliser l'approche des données manquantes pour ce type de mesures. Vous ne voulez probablement pas non plus vous alarmer si la valeur est nulle, car il peut y avoir des scénarios légitimes où elle se situe dans les limites des seuils normaux. La meilleure approche pour détecter ce type de problème consiste à utiliser des métriques produites par les ressources utilisant cette dépendance. Dans ce cas, nous voulons détecter l'impact dans plusieurs zones de disponibilité à l'aide d'alarmes composites. Ces alarmes doivent utiliser une poignée de catégories de métriques critiques liées à la ressource. Voici quelques exemples :

- Débit : taux d'unités de travail entrantes. Il peut s'agir de transactions, de lectures, d'écritures, etc.
- Disponibilité — Mesurez le nombre d'unités de travail réussies par rapport à celles qui ont échoué.
- Latence : mesurez plusieurs percentiles de latence pour garantir la réussite du travail effectué dans le cadre d'opérations critiques.

  Une fois encore, vous pouvez créer les alarmes métriques in a row et m sur n pour chaque métrique de chaque catégorie de métrique que vous souhaitez mesurer. Comme auparavant, elles peuvent être combinées dans une alarme composite afin de déterminer que cette ressource partagée est la source d'impact sur l'ensemble des zones de disponibilité. Vous souhaitez être en mesure d'identifier l'impact sur plusieurs zones de disponibilité grâce aux alarmes composites, mais il n'est pas nécessaire que l'impact concerne toutes les zones de disponibilité. La structure d'alarme composite de haut niveau pour ce type d'approche est illustrée dans la figure suivante.



Exemple de création d'alarmes pour détecter l'impact sur plusieurs zones de disponibilité causé par une seule ressource

Vous remarquerez que ce diagramme est moins prescriptif quant au type d'alarmes métriques à utiliser et à la hiérarchie des alarmes composites. En effet, la découverte de ce type de problème peut être difficile et nécessitera une attention particulière aux bons signaux pour la ressource partagée. Il peut également être nécessaire d'évaluer ces signaux de manière spécifique.

En outre, vous devez remarquer que l'`primary-database-impact`alarme n'est pas associée à une zone de disponibilité spécifique. Cela est dû au fait que l'instance de base de données principale peut être située dans n'importe quelle zone de disponibilité qu'elle est configurée pour utiliser, et aucune CloudWatch métrique ne précise son emplacement. Lorsque vous voyez cette alarme s'activer, vous devez l'utiliser pour signaler un problème éventuel avec la ressource et lancer un basculement vers une autre zone de disponibilité, si cela n'a pas été fait automatiquement. Après avoir déplacé la ressource vers une autre zone de disponibilité, vous pouvez attendre de voir si votre alarme de zone de disponibilité isolée est activée, ou vous pouvez choisir d'invoquer de manière préventive votre plan d'évacuation de zone de disponibilité.

Récapitulatif

Cette section décrit trois approches permettant d'identifier les déficiences liées à une seule zone de disponibilité. Chaque approche doit être utilisée conjointement pour fournir une vision globale de l'état de votre charge de travail.

L'approche d'alarme CloudWatch composite vous permet de détecter les problèmes pour lesquels l'écart de disponibilité n'est pas statistiquement significatif, par exemple des disponibilités de 98 % (zone de disponibilité altérée), 100 % et 99,99 %, qui ne sont pas causés par une seule ressource partagée.

La détection des valeurs aberrantes permet de détecter les défaillances d'une seule zone de disponibilité lorsque des erreurs non corrélées se produisent dans plusieurs zones de disponibilité qui dépassent toutes votre seuil d'alarme.

Enfin, l'identification de la dégradation d'une ressource zonale d'une instance unique permet de découvrir quand une altération d'une zone de disponibilité affecte une ressource partagée entre les zones de disponibilité.

Les alarmes résultant de chacun de ces modèles peuvent être combinées dans une hiérarchie d'alarmes CloudWatch composite afin de détecter les défaillances d'une seule zone de disponibilité qui ont un impact sur la disponibilité ou la latence de votre charge de travail.

Schémas d'évacuation des zones de disponibilité

Après avoir détecté un impact dans une seule zone de disponibilité, l'étape suivante consiste à évacuer cette zone de disponibilité. L'évacuation doit atteindre deux résultats.

Tout d'abord, vous souhaitez arrêter d'envoyer du travail vers la zone de disponibilité concernée. Cela peut signifier différentes choses selon les architectures. Dans le cas d'une charge de travail de type requête/réponse, cela signifierait empêcher que des requêtes HTTP ou gRPC provenant de vos clients ne soient envoyées à l'équilibreur de charge ou à d'autres ressources de la zone de disponibilité. Dans un système de traitement par lots ou de files d'attente, cela peut signifier l'arrêt des ressources informatiques pour traiter le travail dans la zone de disponibilité concernée. Vous devrez également empêcher les ressources des zones de disponibilité non affectées d'interagir avec les ressources de la zone de disponibilité affectée, par exemple, une instance EC2 envoyant du trafic vers unpoint de terminaison VPC de l'interfacedans la zone de disponibilité concernée ou en se connectant à l'instance principale d'une base de données.

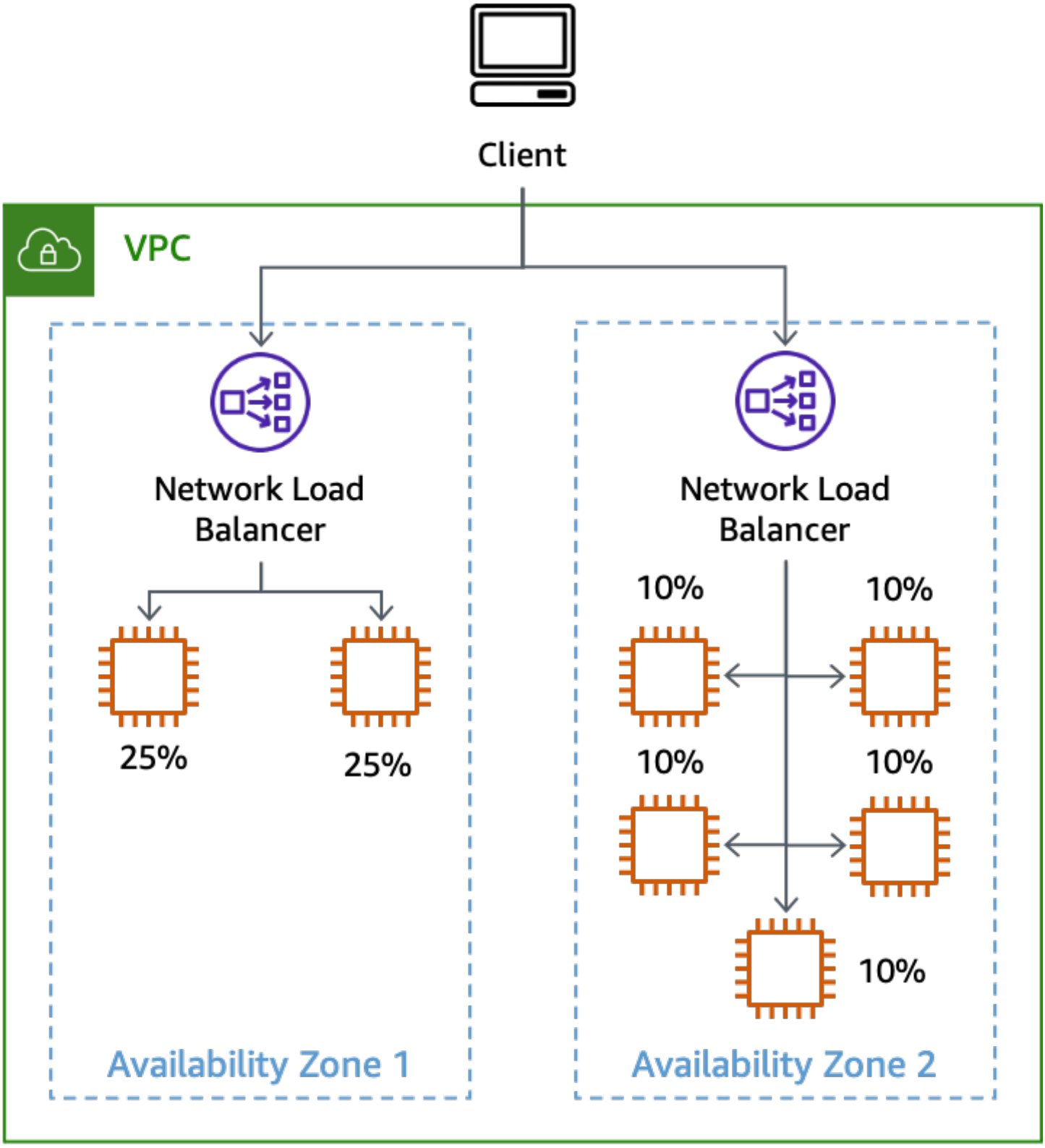
Le deuxième résultat est d'empêcher le provisionnement de nouvelles capacités dans la zone de disponibilité affectée. Cela est important car les nouvelles ressources, telles que les instances ou les conteneurs EC2, mises à disposition dans la zone de disponibilité affectée sont susceptibles d'avoir le même impact que les ressources existantes. De plus, étant donné que le premier résultat empêche de leur envoyer du travail, ils ne peuvent pas absorber la charge pour laquelle ils ont été provisionnés. Cela entraîne une charge accrue sur les ressources existantes, ce qui peut finalement conduire àbrunirou une indisponibilité totale de la charge de travail. Plusieurs services de mise à l'échelle automatique sont disponibles dansAWSle cas échéant :Mise à l'échelle automatique d'Amazon EC2,Mise à l'échelle automatique des applications, etAWS Auto Scaling. En outre, des services tels qu'Amazon ECS, Amazon EKS etAWS Batchpeuvent planifier le travail sur les hôtes dans les zones de disponibilité d'un VPC dans le cadre de leur fonctionnement normal.

Rubriques

Indépendance de la zone de disponibilité

Pour obtenir le premier résultat, à savoir arrêter d'envoyer du travail vers la zone de disponibilité concernée, l'évacuation nécessite que vous mettiez en œuvreIndépendance de la zone de disponibilité(AZI), aussi parfois appeléAffinité des zones de disponibilité. Ce modèle architectural isole les ressources au sein d'une zone de disponibilité et empêche toute interaction entre les ressources des différentes zones de disponibilité, sauf lorsque cela est absolument nécessaire, par exemple pour se connecter à une instance de base de données principale dans une autre zone de disponibilité.

Dans une charge de travail de type requête/réponse, la mise en œuvre de l'AZI vous oblige àdésactiver l'équilibrage de charge entre zones pour les équilibreurs de charge d'applications(ALB),Équilibreurs de charge classiques(CLB), etÉquilibreurs de charge réseau(NLB) (l'équilibrage de charge entre zones est désactivé par défaut pour les NLB). La désactivation de l'équilibrage de charge entre zones comporte quelques inconvénients. Lorsque vous désactivez l'équilibrage de charge entre zones,le trafic est réparti de manière égale entre chaque zone de disponibilitéquel que soit le nombre d'instances présentes dans chacune d'elles. Si vous avez des ressources déséquilibrées ou des groupes Auto Scaling, cela peut entraîner une charge supplémentaire sur les ressources d'une zone de disponibilité qui possède moins de ressources que les autres. Cela est illustré dans la figure suivante où deux instances de la zone de disponibilité 1 reçoivent chacune 25 % de la charge et les cinq instances de la zone de disponibilité 2 reçoivent chacune 10 % de la charge.



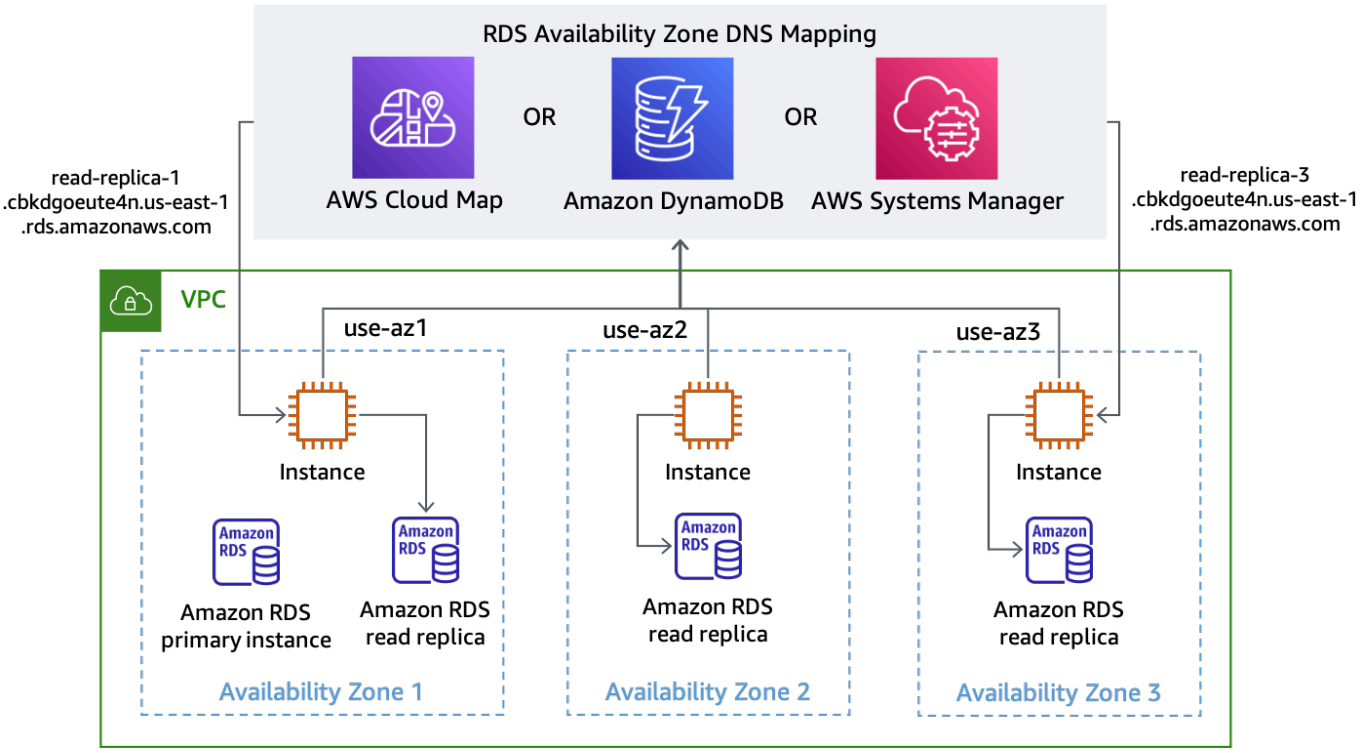
L'effet de la désactivation de l'équilibrage de charge entre zones avec des instances non équilibrées

Les autres services zonaux que vous utilisez devront également être implémentés à l'aide de modèles AZI pour permettre une évacuation efficace de la zone de disponibilité. Par exemple, les points de terminaison d'interface VPC fournissentnoms DNS spécifiques pour chaque zone de disponibilitéle point de terminaison de l'interface est disponible dans.

L'un des défis liés à la mise en œuvre de l'AZI concerne les bases de données, en particulier parce que la plupart des bases de données relationnelles ne prennent en charge qu'un seul rédacteur principal à la fois. Lorsque vous communiquez avec l'instance principale, il se peut que vous deviez franchir les limites d'une zone de disponibilité. De nombreuxAWSles services de base de données prennent en charge une configuration multi-AZ définie par l'utilisateur et disposent d'une fonction de basculement multi-AZ intégrée, telle queAmazon RDSouAmazon Aurore. Dans de nombreux scénarios de défaillance, le service peut détecter l'impact et faire basculer automatiquement la base de données vers une autre zone de disponibilité en cas de problème. Toutefois, en cas de panne grise, le service peut ne pas détecter l'impact qui affecte votre charge de travail, ou l'impact peut ne pas être lié du tout à la base de données. Dans ces cas, une fois que vous avez détecté un impact dans une zone de disponibilité, vous pouvez invoquer manuellement un basculement pour déplacer la base de données principale. Cela vous permet de réagir efficacement en cas de défaillance d'une seule zone de disponibilité.

Si vous utilisez des répliques en lecture avec ces bases de données, vous souhaiterez peut-être également implémenter l'AZI pour celles-ci, car vous ne pouvez pas basculer une réplique en lecture vers une autre zone de disponibilité comme vous le feriez pour la base de données principale. Si vous disposez d'une réplique à lecture unique dans la zone de disponibilité 1 et que des instances de trois zones de disponibilité sont configurées pour l'utiliser, une défaillance affectant la zone de disponibilité 1 aura également un impact sur les opérations dans les deux autres zones de disponibilité. C'est l'impact que vous voulez éviter.

Pour les instances RDS, vous recevez un point de terminaison DNS pour accéder à la réplique dans une zone de disponibilité spécifique. Pour obtenir l'AZI, vous aurez besoin d'une réplique en lecture par zone de disponibilité et d'un moyen pour votre application de savoir quel point de terminaison de réplication utiliser pour la zone de disponibilité dans laquelle il se trouve. Une approche que vous pouvez adopter consiste à utiliser l'ID de la zone de disponibilité dans le cadre de l'identifiant de base de données, quelque chose comme`use1-az1-read-replica.cbkdgoeute4n.us-east-1.rds.amazonaws.com`. Vous pouvez également le faire à l'aide de la découverte des services (par exemple avecAWS Cloud Map) ou en recherchant une carte simple enregistrée dansAWSMagasin de paramètres du gestionnaire de systèmesou une table DynamoDB. Ce concept est illustré dans la figure suivante.

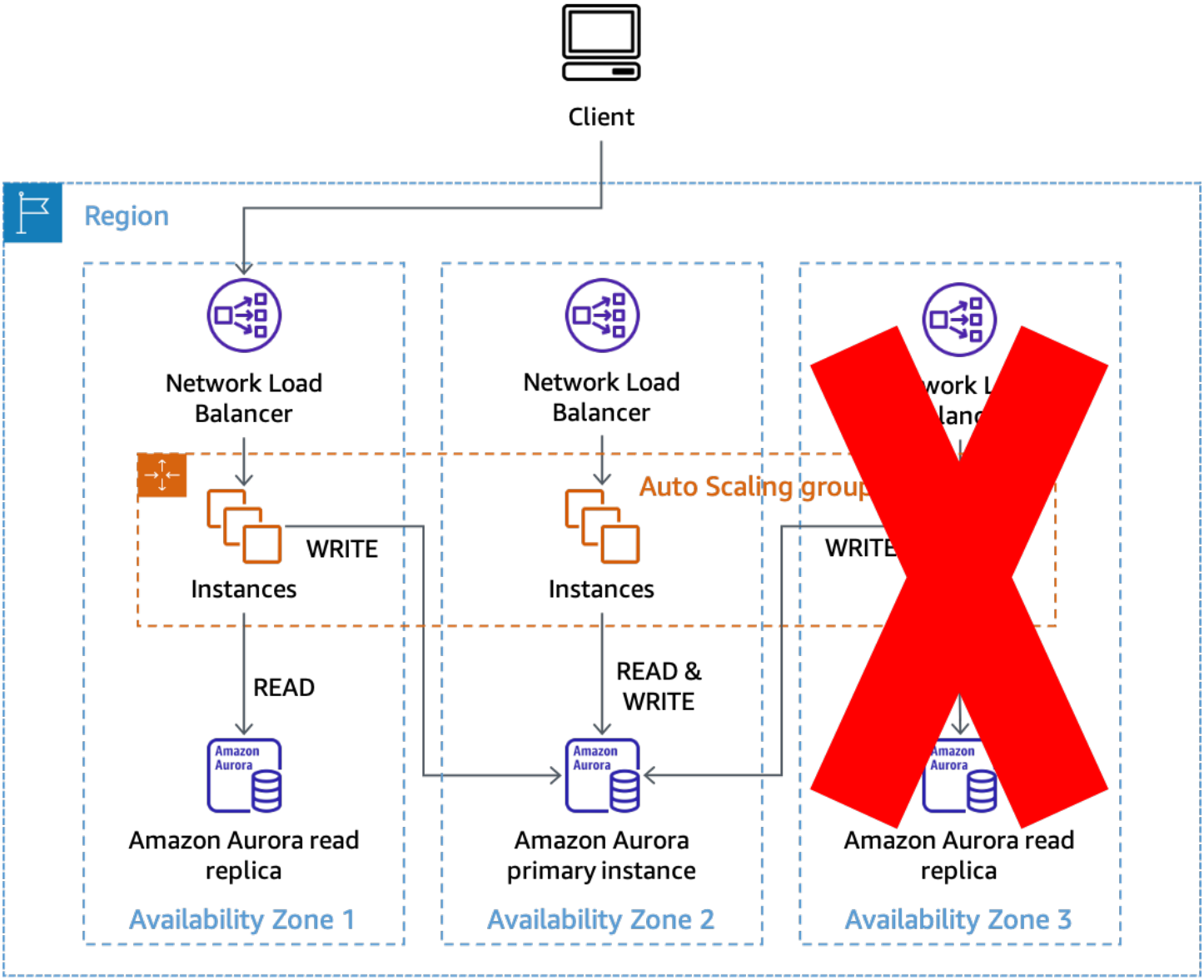


Découverte des noms DNS des points de terminaison RDS pour chaque zone de disponibilité

La configuration par défaut d'Amazon Aurora consiste à fournir unpoint de terminaison à lecteur uniquequi équilibre la charge des demandes entre les répliques de lecture disponibles. Pour implémenter l'AZI à l'aide d'Aurora, vous pouvez utiliserpoint de terminaison personnalisépour chaque réplique lue à l'aide duANYtype (afin que vous puissiez promouvoir une réplique en lecture si nécessaire). Nommez le point de terminaison personnalisé en fonction de l'ID de zone de disponibilité où la réplique est déployée. Vous pouvez ensuite utiliser le nom DNS fourni par le point de terminaison personnalisé pour vous connecter à une réplique en lecture spécifique dans une zone de disponibilité spécifique, comme illustré dans la figure suivante.

Utilisation d'un point de terminaison personnalisé pour une réplique en lecture d'Aurora

Lorsque votre système est conçu de cette façon, l'évacuation de la zone de disponibilité est beaucoup plus simple. Par exemple, dans la figure suivante, lorsqu'une défaillance affecte la zone de disponibilité 3, les opérations de lecture et d'écriture dans les zones de disponibilité 1 et 2 ne sont pas affectées.



Utilisation de l'AZI pour éviter tout impact avec les répliques en lecture d'Amazon Aurora

Par ailleurs, si la zone de disponibilité 2 était affectée, les opérations de lecture continueraient de réussir dans les zones de disponibilité 1 et 3. Ensuite, si Amazon Aurora n'a pas automatiquement basculé sur la base de données principale, vous pouvez invoquer manuellement un basculement vers une autre zone de disponibilité afin de rétablir la capacité de traitement des écritures. Cette approche évite d'avoir à apporter des modifications à la configuration de vos connexions à la base de données lorsque vous devez évacuer une zone de disponibilité. Minimiser les modifications requises et garder le processus aussi simple que possible le rendra plus fiable.

Plans de contrôle et plans de données

Avant d'aborder les modèles réels que vous pouvez utiliser pour effectuer une évacuation de la zone de disponibilité, nous devons aborder les concepts de plans de contrôle et de plans de données.AWSfait la distinction entre les plans de contrôle et les plans de données dans nos services. Les plans de contrôle sont les mécanismes qui permettent d'apporter des modifications à un système (ajout de ressources, suppression de ressources, modification de ressources) et de propagation de ces modifications là où elles doivent être appliquées, par exemple en mettant à jour la configuration réseau d'un ALB ou en créant unAWS Lambdafonction.

Les plans de données constituent la fonction principale de ces ressources, par exemple l'instance EC2 en cours d'exécution ou l'obtention ou l'ajout d'éléments dans une table Amazon DynamoDB. Pour une présentation plus détaillée des plans de contrôle et des plans de données, reportez-vous àStabilité statique à l'aide des zones de disponibilitéetAWSLimites d'isolation des pannes.

Aux fins du présent document, considérez que les plans de contrôle ont tendance à comporter davantage de pièces mobiles et de dépendances que les plans de données. Cela rend statistiquement plus probable que le plan de contrôle soit altéré par rapport au plan de données. Cela est particulièrement pertinent pour les services qui fournissent l'AZI, tels qu'Amazon EC2 et EBS, car certains de ces services disposent de plans de contrôle qui sont également indépendants de la zone et peuvent être affectés lors d'un événement impliquant une seule AZ.

Bien que les actions du plan de contrôle puissent être utilisées pour effectuer une évacuation de la zone de repos, selon les informations précédentes, elles peuvent avoir une probabilité de succès plus faible, en particulier en cas de défaillance. Pour augmenter la probabilité de réussir à atténuer l'impact, vous pouvez utiliser deux modèles différents. Le premier modèle repose uniquement sur les actions du plan de données pour initialement atténuer l'impact en empêchant le routage du travail vers la zone de disponibilité affectée ou en interrompant son exécution dans la zone de disponibilité affectée. Ensuite, le second modèle peut être tenté pour mettre à jour la configuration des ressources à l'aide d'actions du plan de contrôle visant à la fois à empêcher le provisionnement de la capacité dans la zone de disponibilité affectée et à arrêter la communication entre la zone de disponibilité et cette zone de disponibilité.

Les modèles de restauration présentés dans cette section sontgros boutons rouges. Ce sont les mécanismes que vous utilisez pour prendre des mesures à grande échelle, rapidement, comme si vous tiriez unCordon Andon sur une chaîne de montage. Ils supposent que les charges de travail ont déjà essayé des stratégies telles queréessayer avec recul exponentiel avec gigue dans leur code pour surmonter les erreurs transitoires. Cela signifie que lorsqu'un impact isolé sur une zone de

disponibilité est détecté, ses effets sur la disponibilité ou la latence sont suffisamment graves pour nécessiter l'évacuation de la zone de disponibilité afin de les atténuer efficacement.

Évacuation contrôlée par avion de données

Il existe plusieurs solutions que vous pouvez mettre en œuvre pour effectuer une évacuation de la zone de disponibilité à l'aide d'actions relevant uniquement du plan de données. Cette section décrit trois d'entre eux et les cas d'utilisation dans lesquels vous pouvez choisir l'un plutôt que l'autre.

Lorsque vous utilisez l'une de ces solutions, vous devez vous assurer de disposer d'une capacité suffisante dans les zones de disponibilité restantes pour gérer la charge de la zone de disponibilité que vous quittez. Pour ce faire, la solution la plus résiliente consiste à préprovisionner la capacité requise dans chaque zone de disponibilité. Si vous utilisez trois zones de disponibilité, vous disposerez de 50 % de la capacité requise pour gérer votre charge de pointe déployée dans chacune d'entre elles, de sorte que la perte d'une seule zone de disponibilité vous laisserait tout de même 100 % de la capacité requise sans avoir à vous fier à un plan de contrôle pour en provisionner davantage.

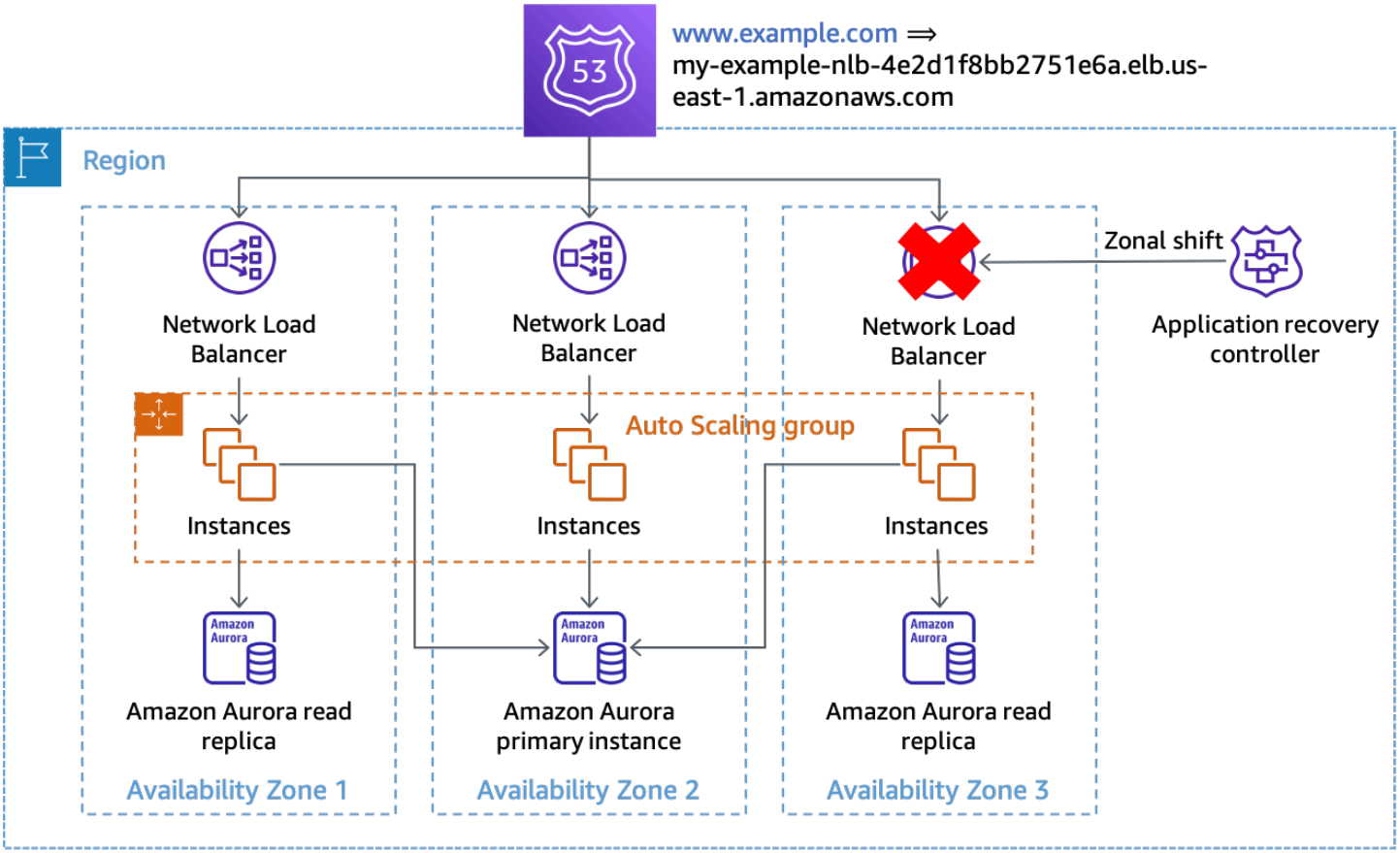
En outre, si vous utilisez EC2 Auto Scaling, assurez-vous que votre groupe Auto Scaling (ASG) n'évolue pas pendant le quart de travail, de sorte qu'à la fin du quart de travail, vous disposiez toujours d'une capacité suffisante dans le groupe pour gérer le trafic de vos clients. Pour ce faire, vous devez vous assurer que la capacité minimale souhaitée de votre ASG est en mesure de faire face à votre charge de clientèle actuelle. Vous pouvez également vous assurer que votre ASG n'évolue pas par inadvertance en utilisant des moyennes dans vos mesures plutôt que des mesures percentiles aberrantes telles que P90 ou P99.

Pendant un quart de travail, les ressources qui ne desservent plus le trafic devraient être très peu utilisées, mais les autres ressources augmenteront leur utilisation en fonction du nouveau trafic, en maintenant la moyenne relativement constante, ce qui empêcherait toute action de mise à l'échelle. Enfin, vous pouvez également utiliser les paramètres de santé du groupe cible pourALBetNLBpour spécifier le basculement DNS avec un pourcentage ou un nombre d'hôtes sains. Cela empêche le trafic d'être acheminé vers une zone de disponibilité ne disposant pas d'un nombre suffisant d'hôtes sains.

Changement zonal dans le contrôleur de restauration des applications (ARC) Route 53

La première solution pour les utilisations d'évacuation des zones de disponibilitéchangement de zone sur la Route 53 ARC. Cette solution peut être utilisée pour les charges de travail de requête/réponse qui utilisent un NLB ou un ALB comme point d'entrée pour le trafic client.

Lorsque vous détectez qu'une zone de disponibilité est altérée, vous pouvez initier un changement de zone avec Route 53 ARC. Une fois que cette opération est terminée et que les réponses DNS mises en cache existantes expirent, toutes les nouvelles demandes sont uniquement acheminées vers les ressources des zones de disponibilité restantes. La figure suivante montre le fonctionnement du décalage zonal. Dans la figure suivante, nous avons un enregistrement d'alias Route 53 pour`www.example.com`qui pointe vers`my-example-nlb-4e2d1f8bb2751e6a.elb.us-east-1.amazonaws.com`. Le décalage zonal est effectué pour la zone de disponibilité 3.



Déplacement zonal

Dans l'exemple, si l'instance de base de données principale ne se trouve pas dans la zone de disponibilité 3, l'exécution du changement de zone est la seule action requise pour obtenir le

premier résultat en matière d'évacuation, à savoir empêcher le traitement du travail dans la zone de disponibilité concernée. Si le nœud principal se trouvait dans la zone de disponibilité 3, vous pouviez effectuer un basculement lancé manuellement (qui repose sur le plan de contrôle Amazon RDS) en coordination avec le décalage zonal, si Amazon RDS ne le faisait pas déjà automatiquement. Cela sera vrai pour toutes les solutions contrôlées par plan de données présentées dans cette section.

Vous devez initier le changement de zone à l'aide des commandes CLI ou de l'API afin de minimiser les dépendances requises pour démarrer l'évacuation. Plus le processus d'évacuation est simple, plus il sera fiable. Les commandes spécifiques peuvent être stockées dans un runbook local auquel les ingénieurs de garde peuvent facilement accéder. Le décalage zonal est la solution la plus recommandée et la plus simple pour évacuer une zone de disponibilité.

Route 53 ARC

La seconde solution utilise les fonctionnalités de Route 53 ARC pour spécifier manuellement l'état de santé de certains enregistrements DNS. Cette solution présente l'avantage d'utiliser le plan de données du cluster Route 53 ARC à haute disponibilité, ce qui le rend résilient à la dégradation de deux plans différentsRégions AWS. Cela implique un coût supplémentaire et nécessite une configuration supplémentaire des enregistrements DNS. Pour implémenter ce modèle, vous devez créer des enregistrements d'alias pourNoms DNS spécifiques aux zones de disponibilitéfourni par l'équilibreur de charge (ALB ou NLB). Cela est indiqué dans le tableau suivant.

Tableau 3 : Enregistrements d'alias Route 53 configurés pour les noms DNS zonaux de l'équilibreur de charge

Politique de routage: pondéré	Politique de routage :pondérée	Politique de routage :pondérée
Nom: `www.example.com`	Nom: `www.example.com`	Nom: `www.example.com`
Type:`A`(alias)	Type : `A`(alias)	Type : `A`(alias)
Value (Valeur) : `us-east-1b.load-balancer-name.elb.us-east-1.amazonaws.com`	Valeur : `us-east-1a.load-balancer-name.elb.us-east-1.amazonaws.com`	Valeur : `us-east-1c.load-balancer-name.elb.us-east-1.amazonaws.com`
Poids:`100`	Poids : `100`	Poids : `100`

Évaluer la santé de la cible: vrai	Évaluez l'état de santé cible : `true`	Évaluez l'état de santé cible : `true`

Pour chacun de ces enregistrements DNS, vous devez configurer un contrôle de santé de Route 53 associé à un ARC de Route 53contrôle de routage. Lorsque vous souhaitez lancer une évacuation de la zone de disponibilité, définissez l'état du contrôle du routage sur`Off`.AWSvous recommande de le faire à l'aide de l'interface de ligne de commande ou de l'API afin de minimiser les dépendances requises pour démarrer l'évacuation de la zone de disponibilité. En tant quemeilleure pratique, vous devez conserver une copie locale des points de terminaison du cluster Route 53 ARC afin de ne pas avoir à les récupérer depuis le plan de contrôle ARC lorsque vous devez effectuer une évacuation.

Pour minimiser les coûts lors de l'utilisation de cette approche, vous pouvez créer un cluster ARC Route 53 unique et effectuer des contrôles de santé dans un seul cluster.Compte AWSetpartager les bilans de santé avec d'autresComptes AWSdans votre organisation. Lorsque vous adoptez cette approche, vous devez utiliserID de zone de disponibilité(AZ-ID) (par exemple,`use1-az1`) au lieu du nom de la zone de disponibilité (par exemple,`us-east-1a`) pour vos commandes de routage. Parce queAWSfait correspondre la zone de disponibilité physique de manière aléatoire aux noms des zones de disponibilité pour chacuneCompte AWS, l'utilisation de l'AZ-ID fournit un moyen cohérent de faire référence aux mêmes emplacements physiques. Lorsque vous lancez l'évacuation d'une zone de disponibilité, dites pour`use1-az2`, les records de la Route 53 établis dans chaqueCompte AWSdoivent s'assurer qu'ils utilisent le mappage AZ-ID pour configurer le bon bilan de santé pour chaque enregistrement NLB.

Par exemple, supposons que nous ayons un bilan de santé de Route 53 associé à un contrôle de routage Route 53 ARC pour`use1-az2`, avec un identifiant de`0385ed2d-d65c-4f63-a19b-2412a31ef431`. Si c'est dans un autreCompte AWSqui souhaite utiliser ce bilan de santé,`us-east-1c`a été mappé à`use1-az2`, vous devez utiliser`use1-az2`bilan de santé pour le dossier`us-east-1c.load-balancer-name.elb.us-east-1.amazonaws.com`. Vous utiliseriez l'identifiant du bilan de santé`0385ed2d-d65c-4f63-a19b-2412a31ef431`avec ce record de ressources défini.

Utilisation d'un point de terminaison HTTP autogéré

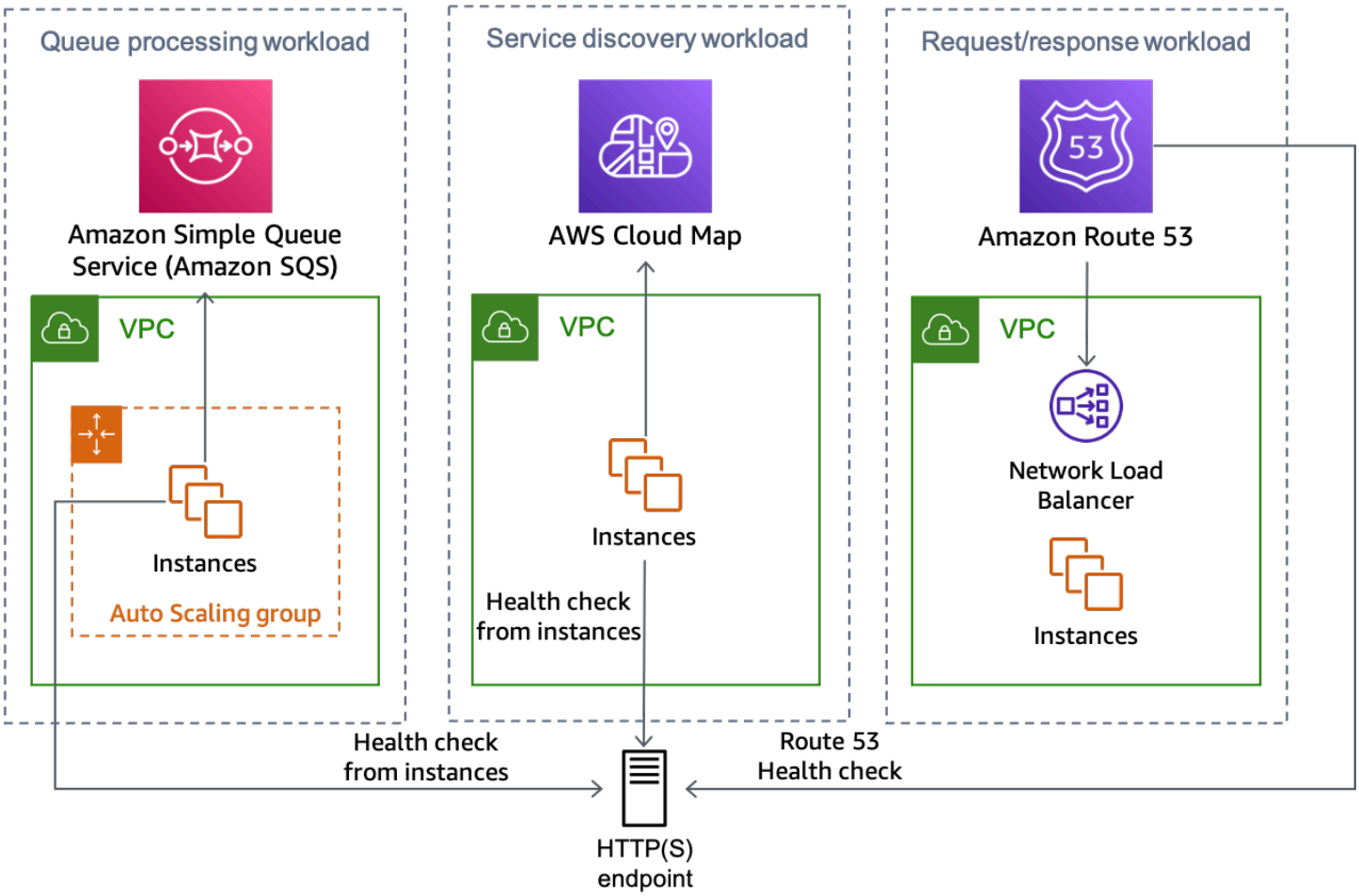
Vous pouvez également implémenter cette solution en gérant votre propre point de terminaison HTTP qui indique l'état d'une zone de disponibilité particulière. Il vous permet de spécifier manuellement quand une zone de disponibilité n'est pas saine en fonction de la réponse du point de terminaison HTTP. Cette solution coûte moins cher que l'utilisation de Route 53 ARC, mais elle est plus coûteuse

que le décalage zonal et nécessite la gestion d'une infrastructure supplémentaire. Il présente l'avantage d'être beaucoup plus flexible pour différents scénarios.

Le modèle peut être utilisé avec les architectures NLB ou ALB et les contrôles de santé de Route 53. Il peut également être utilisé dans des architectures sans équilibrage de charge, telles que les systèmes de découverte de services ou de traitement de files d'attente dans lesquels les nœuds de travail effectuent leurs propres contrôles de santé. Dans ces scénarios, les hôtes peuvent utiliser un fil d'arrière-plan dans lequel ils adressent régulièrement une requête au point de terminaison HTTP avec leur identifiant AZ (voirAnnexe A — Obtenir l'ID de la zone de disponibilité pour savoir comment le trouver) et recevoir en retour une réponse concernant l'état de santé de la zone de disponibilité.

Si la zone de disponibilité a été déclarée insalubre, plusieurs options s'offrent à elle pour y répondre. Ils peuvent choisir d'échouer à un contrôle de santé externe provenant de sources telles que ELB, Route 53 ou à des contrôles d'intégrité personnalisés dans des architectures de découverte de services afin de les rendre malsains aux yeux de ces services. Ils peuvent également répondre immédiatement en signalant une erreur s'ils reçoivent une demande, ce qui permet au client de revenir en arrière et de réessayer. Dans les architectures pilotées par les événements, les nœuds peuvent intentionnellement échouer à traiter le travail, par exemple en renvoyant intentionnellement un message SQS dans la file d'attente. Dans les architectures de routeurs professionnels où un service central planifie le travail sur des hôtes spécifiques, vous pouvez également utiliser ce modèle. Le routeur peut vérifier l'état d'une zone de disponibilité avant de sélectionner un travailleur, un point de terminaison ou une cellule. Dans les architectures de découverte de services qui utilisentAWS Cloud Map, tu peuxdécouvrez les points de terminaison en fournissant un filtre dans votre demande, tel qu'un AZ-ID.

La figure suivante montre comment cette approche peut être utilisée pour plusieurs types de charges de travail.



Plusieurs types de charge de travail peuvent tous utiliser la solution de point de terminaison HTTP

Il existe plusieurs manières de mettre en œuvre l'approche des points de terminaison HTTP ; deux d'entre elles sont décrites ci-dessous.

Utilisation d'Amazon S3

Ce modèle a été initialement présenté dans cearticle de blogpour la reprise après sinistre multirégionale. Vous pouvez utiliser le même schéma pour l'évacuation de la zone de disponibilité.

Dans ce scénario, vous créeriez des ensembles d'enregistrements de ressources DNS Route 53 pour chaque enregistrement DNS zonal, de la même manière que leItinéraire 53 ARCscénario ci-dessus ainsi que les bilans de santé associés. Toutefois, pour cette implémentation, au lieu d'associer les contrôles de santé aux contrôles de routage ARC Route 53, ils sont configurés pour utiliser unPoint de terminaison HTTPet sont inversés pour éviter qu'une défaillance d'Amazon S3 ne déclenche accidentellement une évacuation. Le bilan de santé est pris en compteen bonne santélorsque l'objet

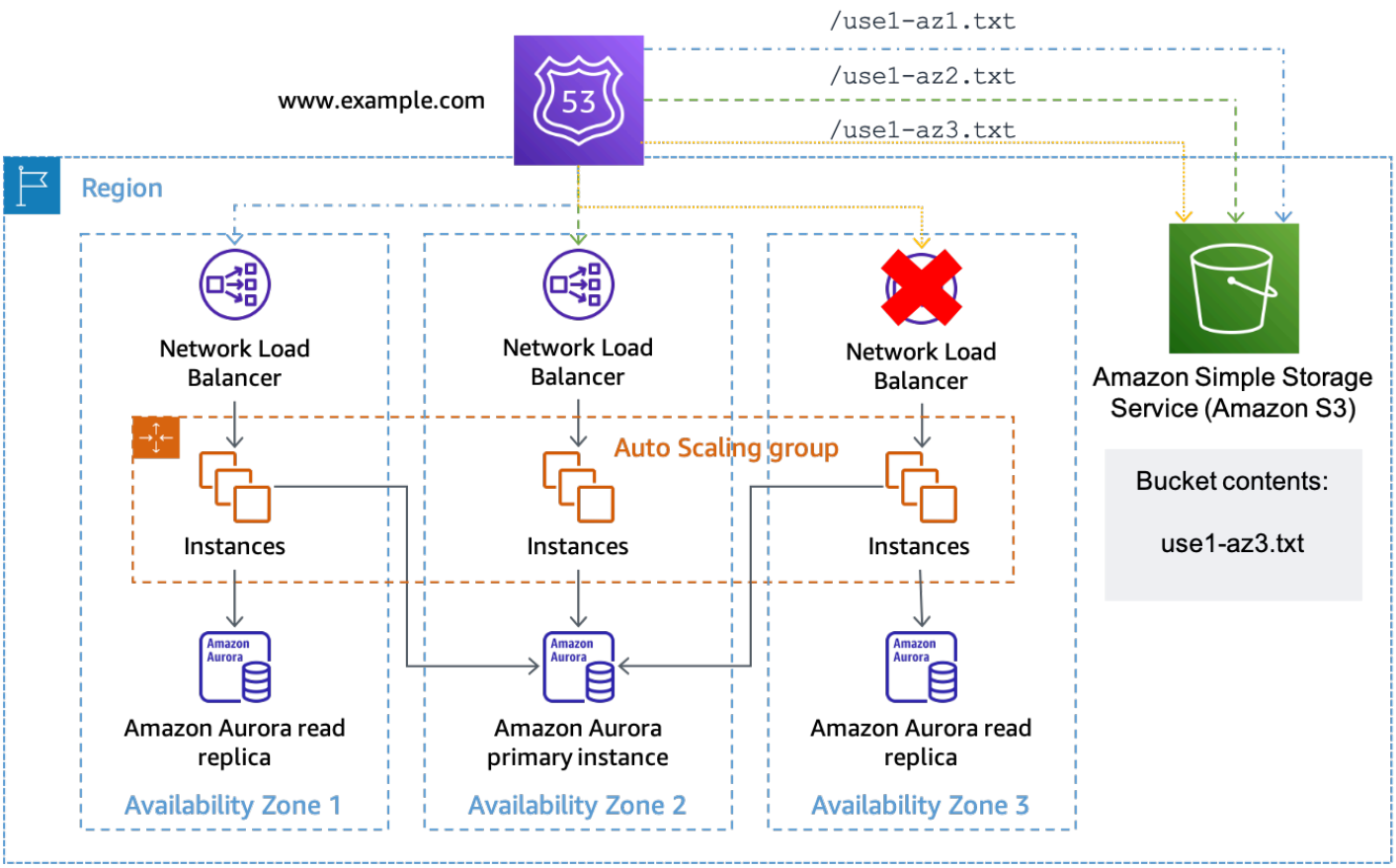
est absent etmauvais pour la santélorsque l'objet est présent. Cette configuration est présentée dans le tableau suivant.

Tableau 4 : Configuration des enregistrements DNS pour l'utilisation des contrôles de santé de Route 53 par zone de disponibilité

Type de bilan de santé: surveiller un point de terminaison Protocole : HTTPS IDENTIFIANT:dddd-4444 URL:https://*bucketname*.s3.us-east-1.amazonaws.com/use1-az1.txt	Type de bilan de santé: surveiller un point de terminaison Protocole : HTTPS IDENTIFIANT:eeee-5555 URL: https://*bucketname*.s3.us-east-1.amazonaws.com/use1-az3.txt	Type de bilan de santé: surveiller un point de terminaison Protocole : HTTPS IDENTIFIANT:ffff-6666 URL:https://*bucketname*.s3.us-east-1.amazonaws.com/use1-az2.txt	←	Contrôles de santé
↑	↑	↑		
Politique de routage: pondéré Nom: www.example.com Type:A(alias) Value (Valeur) : us-east-1	Politique de routage :pondérée Nom: www.example.com Type : A(alias) Value (Valeur) : us-east-1	Politique de routage :pondérée Nom: www.example.com Type : A(alias) Value (Valeur) : us-east-1	←	Les enregistrements d'alias A de niveau supérieur et à pondération uniforme pointent vers des points de terminaison

`b.load-balancer-name.elb.us-east-1.amazonaws.com` Poids:`100` Évaluer la santé de la cible:`true`	`a.load-balancer-name.elb.us-east-1.amazonaws.com` Poids : `100` Évaluez l'état de santé cible : `true`	`c.load-balancer-name.elb.us-east-1.amazonaws.com` Poids : `100` Évaluez l'état de santé cible : `true`		spécifiques à NLB AZ

Supposons que la zone de disponibilité`us-east-1a`est mappé à`use1-az3`dans le compte où nous avons une charge de travail où nous voulons effectuer une évacuation de la zone de disponibilité. Pour le jeu d'enregistrements de ressources créé pour`us-east-1a.load-balancer-name.elb.us-east-1.amazonaws.com`associerait un bilan de santé qui teste l'URL`https://`*`bucket-name`*`.s3.us-east-1.amazonaws.com/use1-az3.txt`. Lorsque vous souhaitez initier une évacuation de la zone de disponibilité pour`use1-az3`, chargez un fichier nommé`use1-az3.txt`au bucket à l'aide de l'interface de ligne de commande ou de l'API. Le fichier n'a pas besoin de contenir de contenu, mais il doit être public pour que le bilan de santé de Route 53 puisse y accéder. La figure suivante montre que cette implémentation est utilisée pour évacuer`use1-az3`.



Utilisation d'Amazon S3 comme cible pour un bilan de santé de Route 53

Utilisation d'API Gateway et de DynamoDB

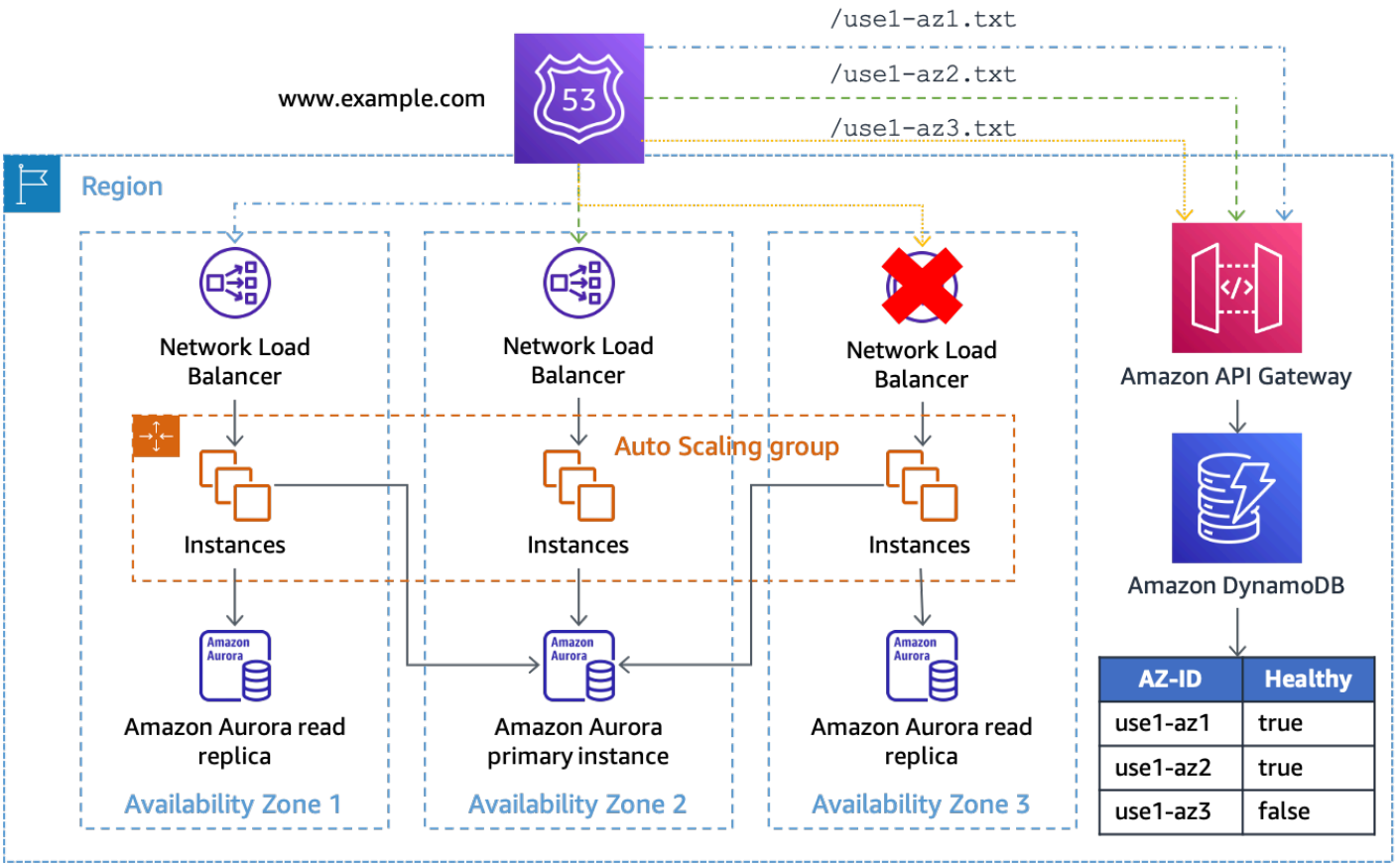
La deuxième implémentation de ce modèle utilise unPasserelle d'API Amazon API REST. L'API est configurée avecintégration des servicesvers Amazon DynamoDB où l'état de chaque zone de disponibilité en cours d'utilisation est enregistré. Cette mise en œuvre est plus flexible que l'approche Amazon S3, mais nécessite de créer, d'exploiter et de surveiller davantage d'infrastructures. Il peut également être utilisé à la fois pour les contrôles de santé de Route 53 et pour les contrôles de santé effectués par des hôtes individuels.

Si vous utilisez cette solution avec une architecture NLB ou ALB, configurez vos enregistrements DNS de la même manière que dans l'exemple Amazon S3 ci-dessus, sauf en modifiant le chemin de vérification de l'état pour utiliser le point de terminaison API Gateway et en fournissant le`AZ-ID`dans le chemin de l'URL. Par exemple, si la passerelle d'API est configurée avec un domaine personnalisé de`az-status.example.com`, la demande complète pour`use1-az1`ressemblerait à`https://az-status.example.com/status/use1-az1`. Lorsque vous souhaitez lancer l'évacuation d'une zone de disponibilité, vous pouvez créer ou mettre à jour un élément DynamoDB à l'aide de l'interface

de ligne de commande ou de l'API. L'article utilise le`AZ-ID`comme clé primaire, puis possède un attribut booléen appelé`Healthy`qui est utilisé pour indiquer comment API Gateway répond. Voici un exemple de code utilisé dans la configuration d'API Gateway pour effectuer cette détermination.

```
#set($inputRoot = $input.path('$'))
#if ($inputRoot.Item.Healthy['BOOL'] == (false))
    #set($context.responseOverride.status = 500)
#end
```

Si l'attribut est`true`(ou n'est pas présent), API Gateway répond au contrôle de santé par un HTTP 200 ; si ce paramètre est faux, il répond par un HTTP 500. Cette implémentation est illustrée dans la figure suivante.



Utilisation d'API Gateway et de DynamoDB comme cible des contrôles de santé de Route 53

Dans cette solution, vous devez utiliser API Gateway en face de DynamoDB afin de rendre le point de terminaison accessible au public et de manipuler l'URL de la demande pour en faire un`GetItem`demande pour DynamoDB. La solution offre également de la flexibilité si vous souhaitez inclure des données supplémentaires dans la demande. Par exemple, si vous souhaitez créer des

statuts plus précis, par exemple par application, vous pouvez configurer l'URL de vérification de l'état de manière à fournir un ID d'application dans le chemin ou la chaîne de requête qui soit également mis en correspondance avec l'élément DynamoDB.

Le point de terminaison de l'état de la zone de disponibilité peut être déployé de manière centralisée afin que plusieurs ressources puissent vérifier l'état de santé deComptes AWSpeuvent tous utiliser la même vue cohérente de l'état de la zone de disponibilité (en veillant à ce que votre API REST API Gateway et votre table DynamoDB soient adaptées à la charge) et éliminent le besoin de partager les contrôles de santé de Route 53.

La solution peut également être étendue à plusieursRégions AWSà l'aide d'unTableau global Amazon DynamoDBet une copie de l'API REST d'API Gateway dans chaque région. Cela évite que cette solution ne dépende d'une seule région et augmente sa disponibilité. Vous pouvez déployer la solution dans trois ou cinq régions et interroger chacune d'elles pour connaître l'état de la zone de disponibilité, en utilisant le résultat de la majorité des points de terminaison pour garantir le quorum. Cela permet à terme de répliquer les mises à jour de manière cohérente dans le tableau global et d'atténuer les déficiences susceptibles d'empêcher un terminal de répondre. Par exemple, si vous utilisez cinq régions et que trois points de terminaison signalent qu'une zone de disponibilité est défectueuse, qu'un point de terminaison signale que la zone de disponibilité est saine et qu'un point de terminaison ne répond pas, vous pouvez choisir de traiter la zone de disponibilité comme étant défectueuse. Vous pouvez également créer unBilan de santé calculé de Route 53à l'aide d'unm de ncalculpour exécuter cette logique afin de déterminer l'état de la zone de disponibilité.

Si vous développiez une solution permettant à des hôtes individuels de déterminer l'état de santé de leur zone de disponibilité, vous pouvez utiliser des notifications push au lieu de fournir un mécanisme d'extraction pour les contrôles de santé. Pour ce faire, vous pouvez notamment créer une rubrique SNS à laquelle vos clients s'abonnent. Lorsque vous souhaitez déclencher le disjoncteur, publiez un message dans la rubrique SNS indiquant quelle zone de disponibilité est affectée. Cette approche fait des compromis avec la première. Il n'est plus nécessaire de créer et d'exploiter l'infrastructure API Gateway et d'effectuer la gestion des capacités. Cela peut également permettre une convergence plus rapide de l'état de la zone de disponibilité. Toutefois, il supprime la possibilité d'effectuer des requêtes ad hoc et s'appuie surPolitique relative aux nouvelles tentatives de livraison par SNSpour s'assurer que chaque point de terminaison reçoit la notification. Cela nécessite également que chaque charge de travail ou service crée un moyen de recevoir la notification SNS et d'agir en conséquence.

Par exemple, chaque nouvelle instance ou conteneur EC2 lancé devra s'abonner à la rubrique avec un point de terminaison HTTP lors de son démarrage. Ensuite, chaque instance doit implémenter

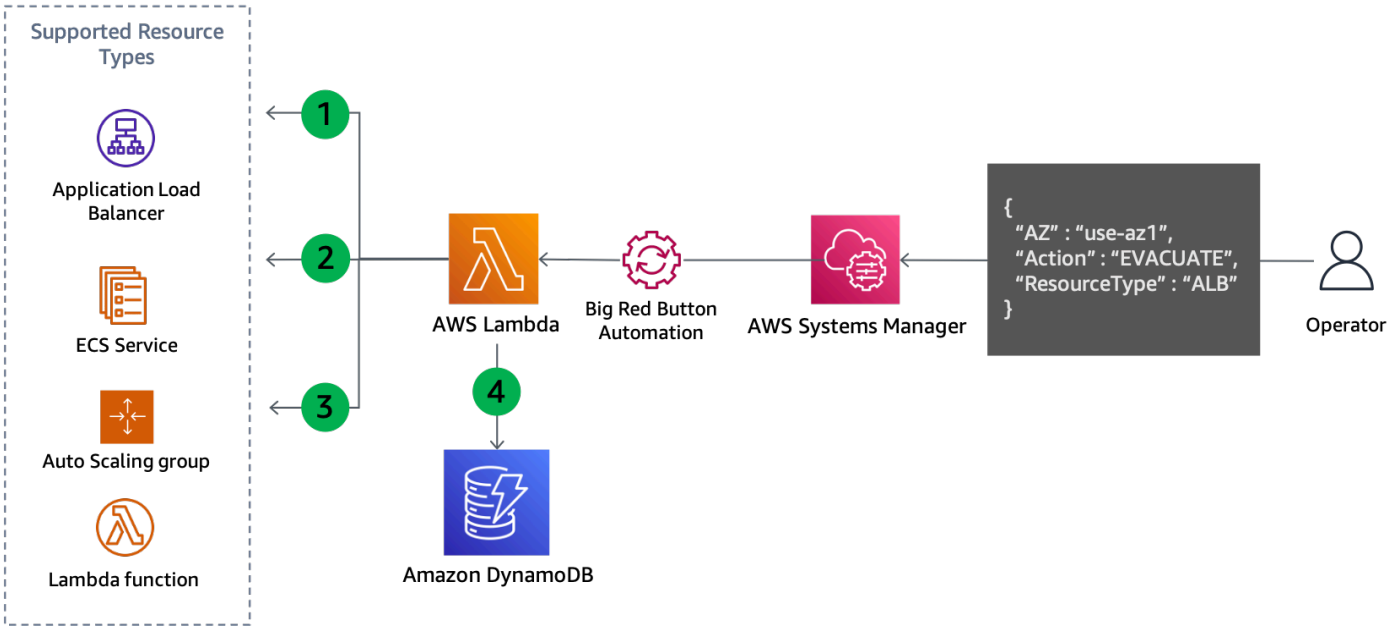
un logiciel qui écoute ce point de terminaison où la notification est envoyée. En outre, si l'instance est affectée par l'événement, elle risque de ne pas recevoir la notification push et de continuer à fonctionner. En revanche, avec une notification d'extraction, l'instance saura si sa demande d'extraction échoue et pourra choisir les mesures à prendre en réponse.

Une deuxième façon d'envoyer des notifications push consiste à utiliser LonglivedWebSocketconnexions. Amazon API Gateway peut être utilisé pour fournir unWebSocketAPIauquel les consommateurs peuvent se connecter et recevoir un message lorsqueenvoyé par le backend. Avec unWebSocket, les instances peuvent à la fois effectuer des extractions périodiques pour s'assurer que leur connexion est saine et recevoir des notifications push à faible latence.

Évacuation contrôlée par le plan de commande

Le premier modèle utilise les opérations du plan de données pour empêcher d'effectuer des travaux dans une zone de disponibilité affectée afin d'atténuer l'impact d'un événement. Toutefois, vous utilisez peut-être une architecture qui n'utilise pas d'équilibreurs de charge ou pour laquelle il n'est pas possible de configurer un bilan de santé par hôte. Vous pouvez également souhaiter empêcher le déploiement de nouvelles capacités dans la zone de disponibilité concernée par le biais d'une mise à l'échelle automatique ou d'une planification normale du travail.

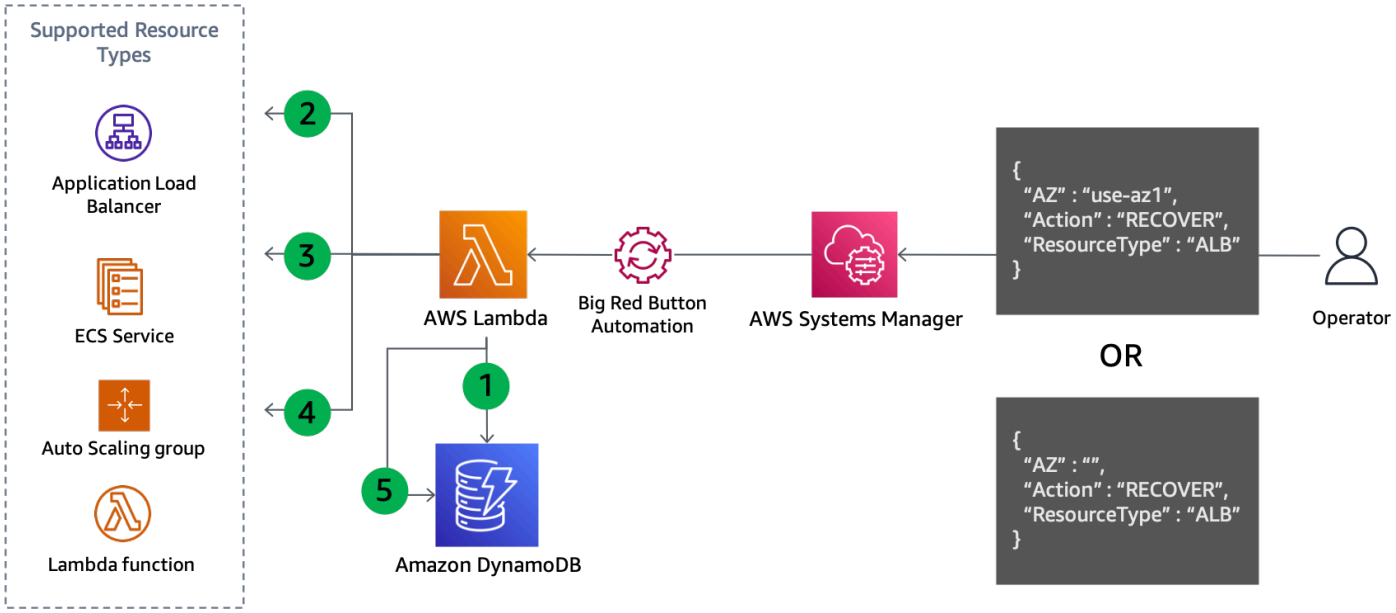
Pour résoudre les deux situations, des actions du plan de contrôle sont nécessaires pour mettre à jour la configuration de la ressource. Le modèle fonctionnera pour tous les services dont la configuration réseau peut être mise à jour, par exemple EC2 Auto Scaling, Amazon ECS, Lambda, etc. Cela nécessite d'écrire du code pour chaque service, mais la logique métier suit un modèle standard. Le code doit être exécuté localement par un opérateur répondant à l'événement afin de minimiser les dépendances requises. Le flux de base de la logique du script est illustré dans la figure suivante.



Mise à jour du plan de contrôle pour évacuer une zone de disponibilité

1. Le script répertorie toutes les ressources du type spécifié, telles que le groupe Auto Scaling, le service ECS ou la fonction Lambda, et extrait leurs sous-réseaux à partir des informations sur les ressources. Les ressources prises en charge dépendent de ce que le script a été configuré pour prendre en charge.
2. Il détermine quels sous-réseaux doivent être supprimés en comparant le nom de zone de disponibilité de chaque sous-réseau à son ID de zone de disponibilité mappé qui a été fourni en tant que paramètre d'entrée.
3. La configuration réseau de la ressource est mise à jour pour supprimer les sous-réseaux identifiés.
4. Les détails de la mise à jour sont enregistrés dans une table DynamoDB. L'ID de la zone de disponibilité est enregistré sous la formeclé de partitionet l'ARN ou le nom de la ressource est stocké sous la formeclé de tri. Les sous-réseaux qui ont été supprimés sont stockés sous forme de tableau de chaînes. Enfin, le type de ressource est également stocké et utilisé comme clé de hachage pourIndice secondaire mondial(SIG).

Étant donné que la quatrième étape enregistre les mises à jour effectuées, cette approche se prête également à être facilement réversible lorsque vous êtes prêt à effectuer la restauration, comme le montre la figure suivante.



Mise à jour du plan de contrôle en cas d'évacuation de la zone de disponibilité

Étapes de restauration :

1. Interrogez le GSI pour supprimer les sous-réseaux pour chaque ressource du type spécifié dans la zone de disponibilité spécifiée (ou toutes les zones de disponibilité si aucune n'est spécifiée).
2. Décrivez chaque ressource trouvée dans la requête DynamoDB pour obtenir sa configuration réseau actuelle.
3. Combinez les sous-réseaux de la configuration réseau actuelle avec ceux extraits de la requête DynamoDB.
4. Mettez à jour la configuration réseau de la ressource avec le nouvel ensemble de sous-réseaux.
5. Supprimez l'enregistrement de la table DynamoDB une fois la mise à jour terminée avec succès.

Ce schéma généralisé empêche à la fois les travaux de routage vers la zone de disponibilité concernée et empêche le déploiement de nouvelles capacités dans cette zone. Vous trouverez ci-dessous des exemples de la manière dont cela est réalisé pour différents services.

- Lambda— Met à jour les fonctionsConfiguration du VPCpour supprimer les sous-réseaux de la zone de disponibilité spécifiée.
- Groupe Auto Scaling—Supprimer les sous-réseaux de la configuration ASGqui remplacera cette capacité dans les zones de disponibilité restantes.
- Amazon ECS—Mettre à jour la configuration VPC du service ECSpour supprimer les sous-réseaux.

- Amazon EKS— Postulezsouillureaux nœuds de la zone de disponibilité affectée pour expulser des pods existants et empêcher la planification de nouveaux pods dans cette zone.

Chaque service réagira différemment à la mise à jour de configuration. Par exemple, Amazon ECS suivra leconfiguration du déploiement du service après une mise à jouret déclenchez un déploiement continu ou un déploiement bleu/vert de nouvelles tâches.

Ces mises à jour peuvent transférer trop rapidement le travail vers les zones de disponibilité saines pour certaines charges de travail. Tout en étant configuré pour être statiquement stable en cas de panne (en disposant de suffisamment de capacité préprovisionnée dans les zones de disponibilité restantes pour gérer le travail de la zone de disponibilité affectée), vous pouvez également souhaiter supprimer progressivement la capacité de la zone de disponibilité affectée.

Si vous envisagez de mettre à jour la configuration réseau de votre groupe Auto Scaling, qui est le groupe cible d'un équilibreur de charge avec équilibrage de charge entre zoneshandicapé, suivez ces instructions.

Auto Scaling réagit à ce changement en utilisantLogique de rééquilibrage des zones de disponibilité. Il lancera des instances dans les autres zones de disponibilité pour atteindre la capacité souhaitée et mettra fin aux instances de la zone de disponibilité que vous avez supprimée. Toutefois, l'équilibreur de charge continuera à répartir le trafic de manière égale entre chaque zone de disponibilité, y compris celle que vous avez supprimée de l'ASG, pendant la résiliation des instances. Cela pourrait entraîner une baisse de la capacité restante dans cette zone de disponibilité jusqu'à ce que toutes les instances y soient clôturées avec succès. Il s'agit du même problème décrit dansIndépendance de la zone de disponibilitéconcernant le déséquilibre des zones de disponibilité lorsque l'équilibrage de charge entre zones est désactivé. Pour éviter que cela ne se produise, vous pouvez :

- Effectuez toujours l'évacuation de votre zone de disponibilité en premier afin que le trafic ne soit réparti qu'entre les zones de disponibilité restantes
- Spécifiez unnombre minimal de cibles saines avec basculement DNSpour atteindre le nombre minimum d'objectifs requis pour cette zone de disponibilité.

Cela permettra de garantir que le trafic n'est pas envoyé vers la zone de disponibilité que vous avez supprimée après le début de la fermeture des instances.

Récapitulatif

Le tableau suivant résume les avantages et les inconvénients des modèles d'évacuation décrits.

Tableau 5 : Avantages et inconvénients du schéma d'évacuation

Approche	Avantages	Inconvénients
Évacuation contrôlée par avion de données	S'appuie uniquement sur les actions du plan de données Empêche rapidement le travail d'être effectué dans la zone de disponibilité concernée Approche flexible pour une vue centralisée de l'état de la zone de disponibilité	N'empêche pas le déploiement de la capacité dans une zone de disponibilité affectée Tous les types de charge de travail ne peuvent pas facilement utiliser cette approche
Évacuation contrôlée par le plan de commande	Empêche le déploieme nt de nouvelles capacités dans la zone de disponibilité concernée Supprime la capacité existante de la zone de disponibilité affectée	S'appuie sur le plan de contrôle de chaque service Nécessite l'écriture d'un code pour chaque service Doit être complété service par service Il faut faire attention à ne pas surcharger la capacité pendant la mise à jour

Vous utiliserez probablement les deux approches ensemble dans le cadre d'un plan d'évacuation de la zone de disponibilité. Commencez par les actions d'évacuation contrôlées par le plan de données qui ont le plus de chances de réussir pour arrêter rapidement le travail de traitement dans la zone de disponibilité concernée. Ensuite, une fois que l'impact initial est atténué, effectuez un suivi avec les mesures d'évacuation contrôlées par le plan de contrôle, si vous le jugez nécessaire.

Conclusion

Cet article fournit une vue d'ensemble des défaillances grises, de la façon dont elles se manifestent et explique pourquoi vous devez créer des outils d'observabilité et d'évacuation pour atténuer ces types d'événements lorsqu'ils se produisent. Dans la section suivante, vous avez examiné l'observabilité multi-AZ et les trois approches que vous pouvez mettre en œuvre pour détecter l'impact d'une seule zone de disponibilité. Dans la dernière section, cet article a présenté deux approches générales pour effectuer l'évacuation de la zone de disponibilité. La première approche utilise des actions du plan de données pour empêcher le routage du travail vers la zone de disponibilité affectée, tandis que la seconde approche utilise des actions du plan de contrôle pour empêcher le provisionnement de la capacité dans la zone de disponibilité affectée. Ensemble, ces deux approches permettent d'atteindre les deux objectifs visés par l'évacuation de la zone de disponibilité.

Les modèles de restauration décrits dans cet article feront probablement partie d'une solution plus large de surveillance et de restauration après panne. Cette approche visant à traiter les pannes grises dans une zone de disponibilité unique nécessite des travaux d'ingénierie visant à mettre au point l'instrumentation nécessaire pour les détecter ainsi que les outils nécessaires pour y répondre. Toutefois, pour de nombreuses charges de travail, cette approche peut constituer une alternative plus simple et moins coûteuse à la création d'architectures multirégionales. En outre, il peut aider à réduire les RPO et les RTO (ce qui augmente la disponibilité de la charge de travail) par rapport à une DR multirégionale.

Annexe A — Obtenir l'ID de la zone de disponibilité

Si vous utilisez leAWS.NET SDK (ainsi que d'autres commeJavaScript) ou en exécutant votre système sur une instance EC2 (y compris Amazon ECS et Amazon EKS), vous pouvez obtenir directement l'ID de la zone de disponibilité.

- AWSSDK .NET

```
Amazon.Util.EC2InstanceMetadata.GetData(“/placement/availability-zone-id”)
```

- Service de métadonnées d'instance EC2

```
curl http://169.254.169.254/latest/meta-data/placement/availability-zone-id
```

Sur d'autres plateformes, telles que Lambda et Fargate, vous devrez récupérer le nom de la zone de disponibilité, puis rechercher le mappage avec l'ID de la zone de disponibilité. Avec le nom de la zone de disponibilité, vous pouvez trouver l'ID de la zone de disponibilité comme suit :

```
aws ec2 describe-availability-zones --zone-names $AZ --output json
    --query ‘AvailabilityZones[0].ZoneId’
```

Les exemples suivants permettant de trouver le nom de la zone de disponibilité à utiliser dans l'exemple ci-dessus sont écrits dans bash à l'aide duAWS CLIet le packagejq. Ils devront être convertis dans le langage de programmation utilisé pour votre charge de travail.

- Amazon ECS- Si le service de métadonnées d'instance (IMDS) est bloqué par l'hôte, vous pouvez utiliser le fichier de métadonnées du conteneur à la place.

```
AZ=$(cat $ECS_CONTAINER_METADATA_FILE | jq --raw-output
    .AvailabilityZone)
```

- Fargate(version 1.4 ou ultérieure de la plateforme)

```
AZ=$(curl $ECS_CONTAINER_METADATA_URI_V4/task | jq --raw-output
    .AvailabilityZone)
```

- Lambda— La zone de disponibilité n'est pas directement exposée à la fonction. Pour le trouver, vous devez suivre plusieurs étapes. Pour ce faire, vous devez créer un point de terminaison REST privé pour la passerelle API qui renvoie l'adresse IP du demandeur. Cela permettra d'identifier l'adresse IP privée attribuée à l'interface réseau élastique utilisée par la fonction.
  - Appelez le Lambda`GetFunction`API permettant de trouver l'ID VPC de la fonction.
  - Appelez le service API Gateway pour obtenir l'adresse IP de la fonction.
  - À l'aide de l'IP et de l'ID VPC, recherchez l'interface réseau associée et extrayez la zone de disponibilité.

```
VPC_ID=$(aws lambda get-function --function-name $ AWS_LAMBDA_FUNCTION_NAME --
region $AWS_REGION --output json --query ‘Configuration.VpcConfig.VpcId’)

MY_IP=$(curl http://whats-my-private-ip.internal)

AZ=$(aws ec2 describe-network-interfaces --filters Name=private-ip-address,Values=
$MY_IP Name=vpc-id,Values=$VPC_ID --region $AWS_REGION --output json –query
 ‘NetworkInterfaces[0].AvailabilityZone’)
```

Annexe B — Exemple de calcul du Khi deux

Voici un exemple de collecte de mesures d'erreur et d'exécution d'un test du Khi deux sur les données. Le code n'est pas prêt pour la production et n'effectue pas la gestion des erreurs nécessaire, mais fournit une preuve de concept sur le fonctionnement de la logique. Vous devez mettre à jour cet exemple en fonction de vos besoins.

Tout d'abord, une fonction Lambda est invoquée chaque minute par un AmazonEventBridgeévénement prévu. Le contenu de l'événement est configuré avec les données suivantes :

```
{
  "timestamp": "2023-03-15T15:26:37.527Z",
  "namespace": "multi-az/frontend",
  "metricName": "5xx",
  "dimensions": [
    { "Name": "Region", "Value": "us-east-1" },
    { "Name": "Controller", "Value": "Home" },
    { "Name": "Action", "Value": "Index" }
  ],
  "period": 60,
  "stat": "Sum",
  "unit": "Count",
  "chiSquareMetricName": "multi-az/chi-squared",
  "azs": [ "use1-az2", "use1-az4", "use1-az6" ]
}
```

Les données sont utilisées pour spécifier les données communes nécessaires pour récupérer les informations appropriéesCloudWatchdes métriques (telles que l'espace de nommage, le nom de la métrique et les dimensions), puis publiez les résultats au Khi deux pour chaque zone de disponibilité. Le code de la fonction Lambda ressemble au suivant avec Python 3.9. À un niveau élevé, il collecte les informations spécifiéesCloudWatchmétriques de la minute précédente, exécute le test du Khi deux sur ces données, puis publieCloudWatchdes mesures concernant le résultat du test pour chaque zone de disponibilité spécifiée.

```
import os
import boto3
import datetime
import copy
```

```
import json
from datetime import timedelta
from scipy.stats import chisquare
from aws_embedded_metrics import metric_scope

cw_client = boto3.client("cloudwatch", os.environ.get("AWS_REGION", "us-east-1"))

@metric_scope
def handler(event, context, metrics):
    metrics.set_property("Event", json.loads(json.dumps(event, default = str)))
    time = datetime.datetime.strptime(event["timestamp"], "%Y-%m-%dT%H:%M:%S.%fZ")

    # Round down to the previous minute
    end: datetime = roundTime(time)

    # Subtract a minute for the start
    start: datetime = end - timedelta(minutes = 1)

    # Get all the metrics that match the query
    results = get_all_metrics(event, start, end, metrics)
    metrics.set_property("MetricCounts", results)

    # Calculate the chi squared result
    chi_sq_result = chisquare(list(results.values()))
    expected = sum(list(results.values())) / len(results.values())
    metrics.set_property("ChiSquaredResult", chi_sq_result)

    # Put the chi square metrics into CloudWatch
    put_all_metrics(event, results, chi_sq_result[1], expected, start, metrics)

def get_all_metrics(detail: dict, start: datetime, end: datetime, metrics):
    """
    Gets all of the error metrics for each AZ specified
    """
    metric_query = {
        "MetricDataQueries": [
        ],
        "StartTime": start,
        "EndTime": end
    }

    for az in detail["azs"]:

        dim = copy.deepcopy(detail["dimensions"])
```

```
        dim.append({"Name": "AZ-ID", "Value": az})

        query = {
            "Id": az.replace("-", "_"),
            "MetricStat": {
                "Metric": {
                    "Namespace": detail["namespace"],
                    "MetricName": detail["metricName"],
                    "Dimensions": dim
                },
                "Period": int(detail["period"]),
                "Stat": detail["stat"],
                "Unit": detail["unit"]
            },
            "Label": az,
            "ReturnData": True
        }

        metric_query["MetricDataQueries"].append(query)

    metrics.set_property("GetMetricRequest", json.loads(json.dumps(metric_query,
default=str)))
    next_token: str = None
    results = {}

    while True:
        if next_token is not None:
            metric_query["NextToken"] = next_token

        data = cw_client.get_metric_data(**metric_query)

        if next_token is not None:
            metrics.set_property("GetMetricResult::" + next_token,
json.loads(json.dumps(data, default = str)))
        else:
            metrics.set_property("GetMetricResult", json.loads(json.dumps(data, default
= str)))

        for item in data["MetricDataResults"]:
            key = item["Id"].replace("_", "-")
            if key not in results:
              results[key] = 0

            results[key] += sum(item["Values"])
```

```
        if "NextToken" in data:
            next_token = data["NextToken"]

        if next_token is None:
            break

    return results

def put_all_metrics(detail: dict, results: dict, chi_sq_value: float, expected: float,
 timestamp: datetime, metrics):
    """
    Adds the chi squared metric for all AZs to CloudWatch
    """
    farthest_from_expected = None
    if len(results) > 0:
        keys = list(results.keys())
        farthest_from_expected = keys[0]

        for key in keys:
            if abs(results[key] - expected) > abs(results[farthest_from_expected] -
 expected):
                farthest_from_expected = key

    metric_query = {
        "Namespace": detail["namespace"],
        "MetricData": []
    }

    for az in detail["azs"]:
        dim = copy.deepcopy(detail["dimensions"])
        dim.append({"Name": "AZ-ID", "Value": az})

        query = {
            "MetricName": detail["chiSquareMetricName"],
            "Dimensions": dim,
            "Timestamp": timestamp,
        }

        if chi_sq_value <= 0.05 and az == farthest_from_expected:
            query["Value"] = 1
        else:
            query["Value"] = 0
```

```
        metric_query["MetricData"].append(query)

    metrics.set_property("PutMetricRequest", json.loads(json.dumps(metric_query,
 default = str)))

    cw_client.put_metric_data(**metric_query)

def roundTime(dt=None, roundTo=60):
   """Round a datetime object to any time lapse in seconds
   dt : datetime.datetime object, default now.
   roundTo : Closest number of seconds to round to, default 1 minute.
   """
   if dt == None : dt = datetime.datetime.now()
   seconds = (dt.replace(tzinfo=None) - dt.min).seconds
   rounding = (seconds+roundTo/2) // roundTo * roundTo
   return dt + datetime.timedelta(0,rounding-seconds,-dt.microsecond)
```

Vous pouvez ensuite créer une alarme par AZ. L'exemple suivant concerne`use1-az2`et des alarmes pour trois points de données consécutifs d'une minute dont la valeur maximale est égale à 1 (1 est la métrique publiée lorsque le test du Khi deux détermine un biais statistiquement significatif du taux d'erreur).

```
{
    "Type": "AWS::CloudWatch::Alarm",
    "Properties": {
        "AlarmName": "use1-az2-chi-squared",
        "ActionsEnabled": true,
        "OKActions": [],
        "AlarmActions": [],
        "InsufficientDataActions": [],
        "MetricName": "multi-az/chi-squared",
        "Namespace": "multi-az/frontend",
        "Statistic": "Maximum",
        "Dimensions": [
            {
                "Name": "AZ-ID",
                "Value": "use1-az2"
            },
            {
                "Name": "Action",
                "Value": "Index"
```

```
            },
            {
                "Name": "Region",
                "Value": "us-east-1"
            },
            {
                "Name": "Controller",
                "Value": "Home"
            }
        ],
        "Period": 60,
        "EvaluationPeriods": 3,
        "DatapointsToAlarm": 3,
        "Threshold": 1,
        "ComparisonOperator": "GreaterThanOrEqualToThreshold",
        "TreatMissingData": "missing"
    }
}
```

Vous pouvez également créer unm-of-nalarme et combinez ces deux alarmes avec une alarme composite. Vous devez également créer les mêmes alarmes pour chaque combinaison contrôleur/action ou microservice que vous avez dans chaque zone de disponibilité. Enfin, vous pouvez ajouter l'alarme composite au Khi deux à l'alarme spécifique à la zone de disponibilité pour chaque combinaison contrôleur/action, comme indiqué dans[Détection des défaillances grâce à la détection des valeurs aberrantes](#).

Collaborateurs

Les contributeurs à ce document incluent :

- Michael Haken, architecte de solutions principal, Amazon Web Services

Révisions du document

Pour être informé des mises à jour de ce livre blanc, abonnez-vous au fil RSS.

Modification	Description	Date
Livre blanc mis à jour	Mise à jour avec des instructions d'observabilité supplémentaires et pour utiliser la nouvelle fonctionnalité de décalage zonal.	11 juillet 2023
Publication initiale	Le livre blanc a été publié pour la première fois.	2 mars 2022

> **Note**
>
> Pour vous abonner aux mises à jour RSS, un plug-in RSS doit être activé pour le navigateur que vous utilisez.

Avis

Les clients sont tenus de faire leur propre évaluation indépendante des informations contenues dans ce document. Ce document : (a) est à titre informatif uniquement, (b) représente le courantAWSles offres de produits et les pratiques, qui peuvent être modifiées sans préavis, et (c) ne créent aucun engagement ni aucune garantie deAWSet ses filiales, fournisseurs ou concédants de licence.AWSles produits ou services sont fournis « tels quels » sans garantie, représentation ou condition d'aucune sorte, qu'elle soit expresse ou implicite. Les responsabilités et les obligations deAWSà ses clients sont contrôlés parAWSaccords, et ce document ne fait partie d'aucun accord entre et ne le modifie pasAWSet ses clients.

Glossaire AWS

Pour connaître la terminologie la plus récente d'AWS, consultez le Glossaire AWS dans la Référence Glossaire AWS.

Les traductions sont fournies par des outils de traduction automatique. En cas de conflit entre le contenu d'une traduction et celui de la version originale en anglais, la version anglaise prévaudra.