Scelta di una strategia di persistenza per il tuo sistema di bilanciamento del carico

AWS Linee guida prescrittive

AWS Linee guida prescrittive: Scelta di una strategia di persistenza per il tuo sistema di bilanciamento del carico

Table of Contents

Scelta di una strategia di persistenza per il tuo sistema di bilanciamento del carico

Ryan Griffin, Amazon Web Services ()AWS

Luglio 2024 (cronologia dei documenti)

La persistenza è un termine utilizzato per descrivere la funzionalità di un sistema di bilanciamento del carico per instradare ripetutamente il traffico da un client a una singola destinazione, anziché bilanciare il traffico su più destinazioni. Ad esempio, il traffico proveniente dal client A può essere instradato continuamente verso un server specifico, in modo che il server possa mantenere i dati sullo stato della sessione. Se il traffico proveniente dal client A viene instradato verso due server distinti, a ciascun server potrebbero mancare informazioni importanti disponibili per l'altro server.

Pertanto, spesso è necessario mantenere una connessione client coerente tramite un sistema di bilanciamento del carico. Esistono due tipi di viscosità: le sessioni persistenti e la persistenza del gruppo target.

- Sessioni permanenti: gestione dei dati della sessione locale in un'istanza Amazon Elastic Compute Cloud EC2 (Amazon) per semplificare l'architettura dell'applicazione o migliorare le prestazioni dell'applicazione, poiché l'istanza può mantenere o memorizzare nella cache le informazioni sullo stato della sessione localmente. AWS attualmente offre due tipi di sessioni permanenti, che questa guida illustra in dettaglio: i cookie delle applicazioni e i cookie di bilanciamento del carico.
- Visibilità del gruppo target: nelle distribuzioni blu/verdi è possibile che siano distribuite più versioni di un'applicazione e si potrebbe desiderare che il client continui a utilizzare la stessa versione dell'applicazione durante la sessione. In questo caso, puoi utilizzare la fedeltà del gruppo target per indirizzare tutte le comunicazioni dal client allo stesso gruppo target anziché alla stessa istanza. EC2

È possibile utilizzare queste due strategie di adesività separatamente o insieme.

Questa guida descrive i diversi tipi di rigidità del sistema di bilanciamento del carico e i casi d'uso applicabili, per aiutarti a scegliere una strategia. La guida include AWS CloudFormation modelli che illustrano ciascuna strategia.

Codice di esempio

Questa guida fornisce un file.zip allegato che include quattro AWS CloudFormation modelli che è possibile implementare per creare un'architettura di base e provare ogni strategia di adesività. Ti consigliamo di distribuire questi modelli in un ambiente di laboratorio per testare ogni approccio.

Scarica il codice di esempio

Il download include i seguenti modelli:

- `basic.yml`— Configura un Application Load Balancer senza appiccicosità.
- `targetgroupstickiness.yml`— Dimostra la persistenza in base ai gruppi target.
- `stickysessionslb.yml`— Dimostra sessioni persistenti con cookie generati dal load balancer.
- `stickysessionsapp.yml`— Dimostra sessioni permanenti con cookie basati su applicazioni.

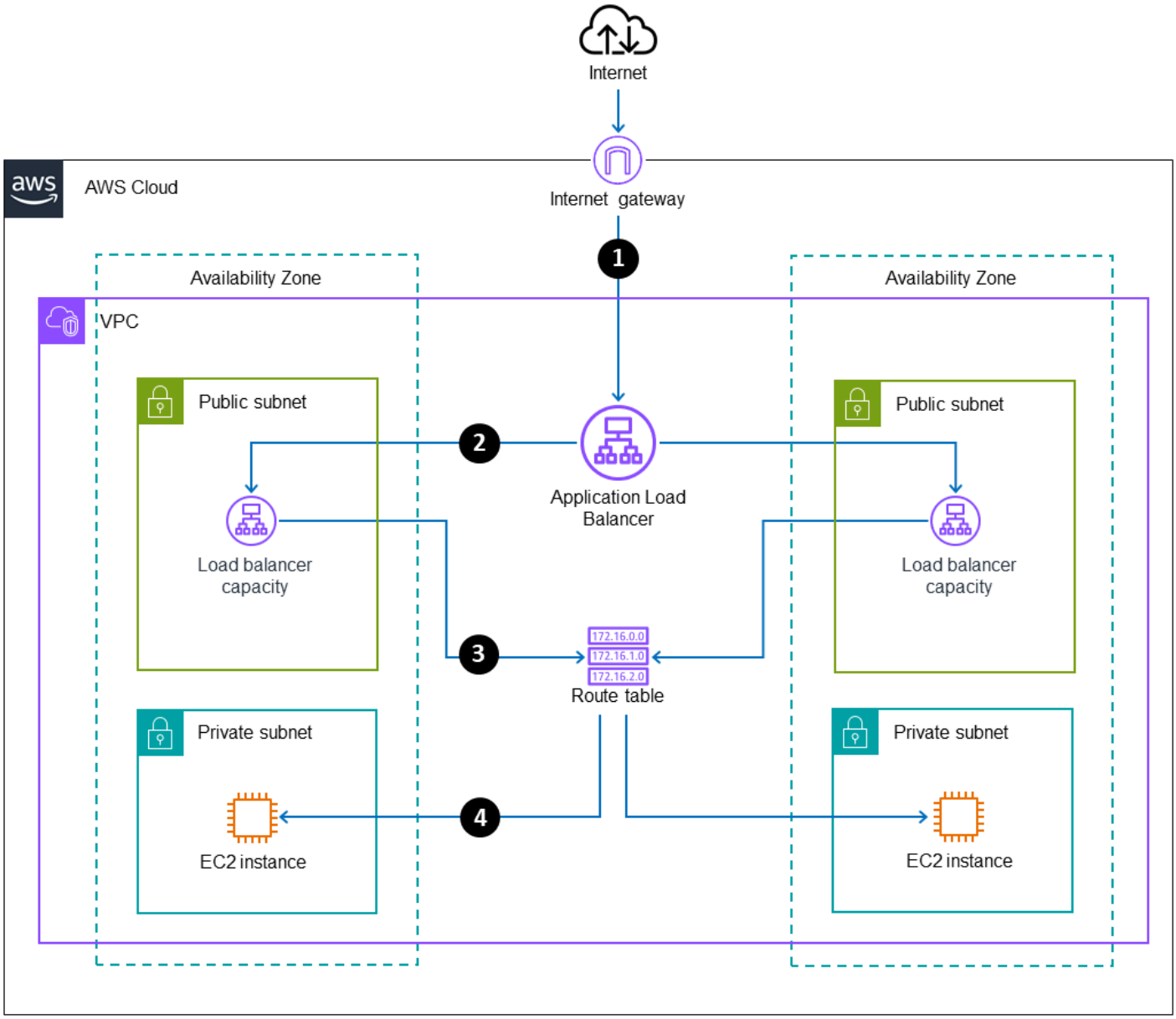
Per distribuire questi modelli, avrai bisogno di un AWS account: attivo e di un accesso alla console.CloudFormation Per step-by-steps istruzioni sulla distribuzione di un CloudFormation modello, consulta Creazione di uno stack nella documentazione. CloudFormation

Sottoreti e routing del sistema di bilanciamento del carico

Iniziamo con un'illustrazione del flusso di traffico quando configuri un'istanza di Application Load Balancer che si affaccia su Internet, verso le istanze di Amazon Elastic Compute Cloud (Amazon EC2) in una sottorete privata. Questa architettura riflette le migliori pratiche per l'implementazione di un Application Load Balancer aperto a Internet.

Percorso del traffico in entrata

Il diagramma seguente illustra le sottoreti e il routing del cloud privato virtuale (VPC) associati al flusso di traffico in entrata, con il traffico di ritorno rimosso dal diagramma per maggiore chiarezza.

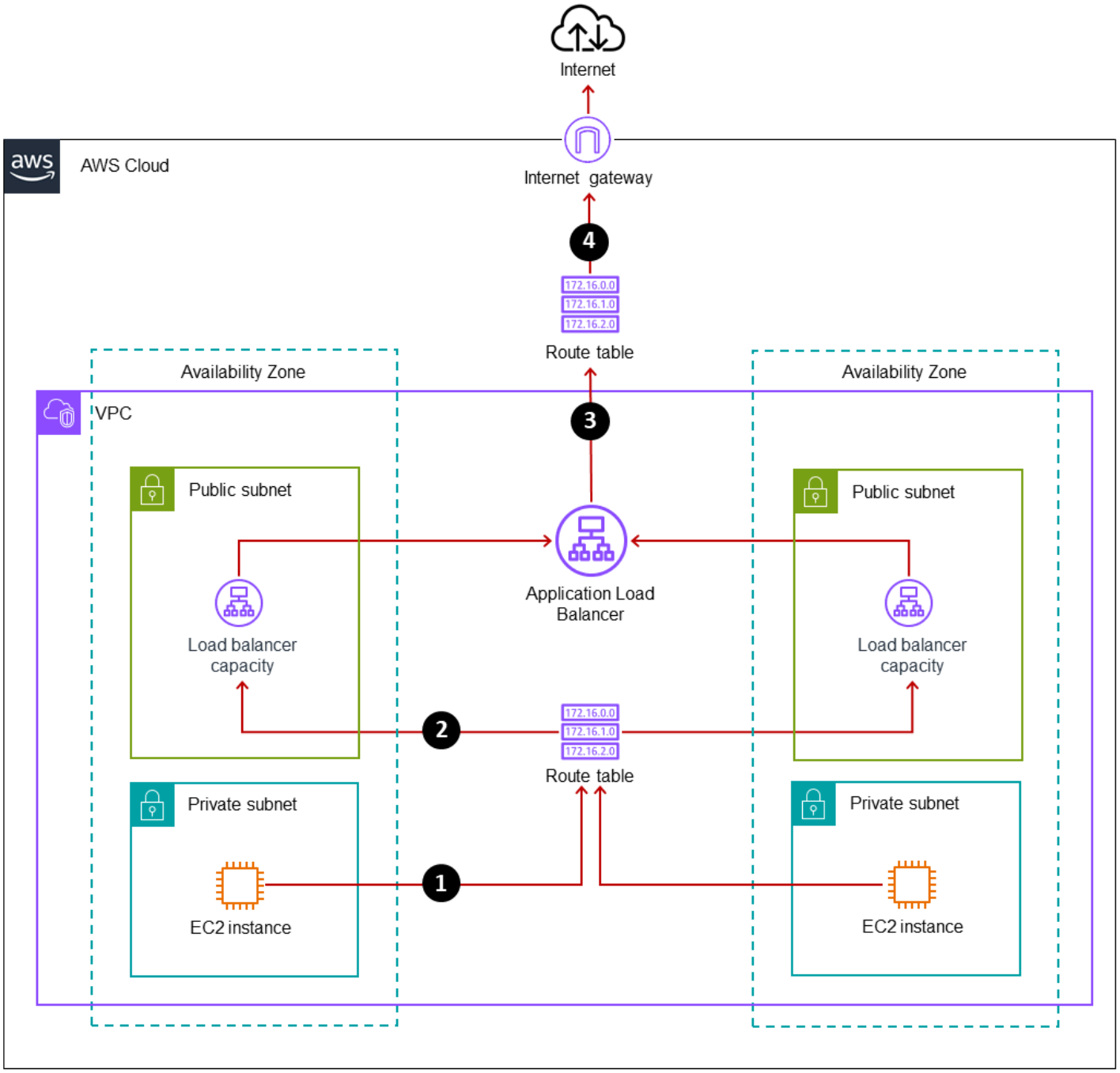


1. Il traffico proveniente da Internet confluisce nel nome DNS dell'Application Load Balancer.
2. L'Application Load Balancer è associato a due sottoreti pubbliche nello scenario illustrato. Il servizio Elastic Load Balancing crea capacità di bilanciamento del carico in ogni zona di disponibilità abilitata e invia il traffico attraverso la zona di disponibilità per determinare la logica di routing appropriata. L'Application Load Balancer utilizza la sua logica interna per determinare a quale gruppo di destinazione e istanza indirizzare il traffico.
3. L'Application Load Balancer indirizza la richiesta all'istanza EC2 attraverso un nodo associato alla sottorete pubblica nella stessa zona di disponibilità. (Questi nodi sono configurati, gestiti e scalati dal servizio Elastic Load Balancing e non sono visibili agli utenti.)

4. La tabella delle rotte indirizza il traffico localmente all'interno del VPC, tra la sottorete pubblica e la sottorete privata e verso l'istanza EC2.

Percorso del traffico di ritorno

Il diagramma seguente illustra le sottoreti VPC e il routing associati al percorso del traffico di ritorno a Internet, con il traffico in entrata rimosso dal diagramma per maggiore chiarezza.

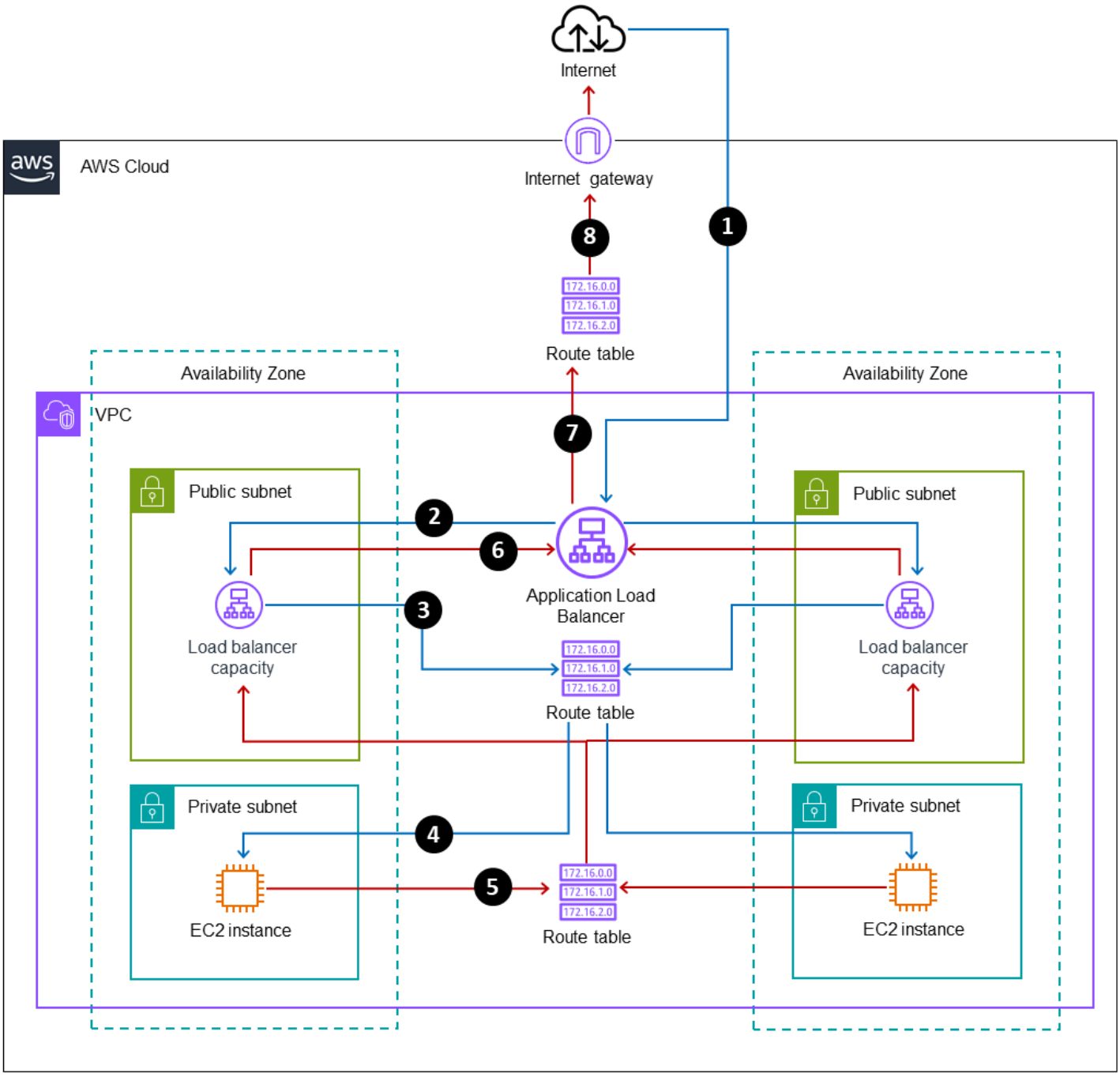


1. L'istanza EC2 nella sottorete privata indirizza il traffico in uscita attraverso la tabella delle rotte.
2. La tabella delle rotte ha un percorso locale verso la sottorete pubblica. Raggiunge la capacità dell'Application Load Balancer su cui è entrato il traffico, nella sottorete pubblica corrispondente, seguendo il percorso a ritroso da dove è entrato il traffico.
3. L'Application Load Balancer indirizza il traffico in uscita attraverso la sua interfaccia pubblica.

4. La tabella di routing della sottorete pubblica ha una route predefinita che punta a un gateway Internet, che reindirizza il traffico verso Internet.

Diagramma completo del flusso di traffico

Il diagramma seguente combina i flussi di traffico in entrata e di ritorno per fornire un'illustrazione completa del routing del sistema di bilanciamento del carico.



1. Il traffico proveniente da Internet confluisce nel nome DNS dell'Application Load Balancer.
2. L'Application Load Balancer è associato a due sottoreti pubbliche nello scenario illustrato. Il servizio Elastic Load Balancing crea capacità di bilanciamento del carico in ogni zona di disponibilità abilitata e invia il traffico attraverso la zona di disponibilità per determinare la logica di routing appropriata. L'Application Load Balancer utilizza la sua logica interna per determinare a quale gruppo di destinazione e istanza indirizzare il traffico.

3. L'Application Load Balancer indirizza la richiesta all'istanza EC2 attraverso un nodo associato alla sottorete pubblica nella stessa zona di disponibilità. (Questi nodi sono configurati, gestiti e scalati dal servizio Elastic Load Balancing e non sono visibili agli utenti.)
4. La tabella delle rotte indirizza il traffico localmente all'interno del VPC, tra la sottorete pubblica e la sottorete privata e verso l'istanza EC2.
5. L'istanza EC2 nella sottorete privata indirizza il traffico in uscita attraverso la tabella delle rotte.
6. La tabella delle rotte ha un percorso locale verso la sottorete pubblica. Raggiunge la capacità dell'Application Load Balancer su cui è entrato il traffico, nella sottorete pubblica corrispondente, seguendo il percorso a ritroso da dove è entrato il traffico.
7. L'Application Load Balancer indirizza il traffico in uscita attraverso la sua interfaccia pubblica.
8. La tabella di routing della sottorete pubblica ha una route predefinita che punta a un gateway Internet, che reindirizza il traffico verso Internet.

Qual è la strategia di viscosità giusta per te?

Rispondere alle seguenti domande vi aiuterà a determinare la vostra strategia di persistenza del sistema di bilanciamento del carico.

Stai usando? WebSockets

- Sì: WebSockets sono intrinsecamente appiccicosi, quindi non è necessario utilizzare alcuna strategia.
- No: passa alla domanda successiva.

Stai pianificando una distribuzione blu/verde?

- Sì: come minimo, utilizzate la fedeltà del gruppo target, ma passate alla domanda successiva.
- No: passa alla domanda successiva.

È necessario che parte o tutto il traffico proveniente da un client venga indirizzato ripetutamente alla stessa EC2 istanza?

- Sì: passa alla domanda successiva.
- No: non è necessario utilizzare sessioni permanenti.

Vuoi che il codice dell'applicazione o il client controllino gli attributi dello stato e la durata utilizzando i cookie?

- Sì: utilizza i cookie basati sulle applicazioni per abilitare sessioni permanenti basate sull'applicazione.
- No: utilizza i cookie generati dal sistema di bilanciamento del carico per abilitare sessioni permanenti basate sulla durata.

Application Load Balancer senza viscosità

Quando si utilizza un Application Load Balancer senza alcuna forma di persistenza, per impostazione predefinita, il load balancer utilizza il metodo round robin per determinare l' EC2 istanza verso cui indirizzare il traffico.

Modello: utilizzate il CloudFormation modello `basic.yml` (incluso nel file di esempio .zip) per provare questa funzionalità.

> **Note**
>
> Tutti i CloudFormation modelli inclusi in questa guida implementano un VPC personalizzato, tabelle di routing, percorsi, un gateway Internet, un Application Load Balancer, gruppi target, ascoltatori e istanze, per illustrare una strategia EC2 specifica di solidità del load balancer.

Casi di utilizzo comune

Utilizzate un Application Load Balancer senza problemi in questi scenari:

- Hai un elenco di destinazioni verso cui indirizzare il traffico, ma le destinazioni non mantengono lo stato della sessione.
- Stai utilizzando server Web che non mantengono lo stato della sessione.
- Stai utilizzando server di applicazioni che non mantengono lo stato della sessione.

Fasi

> **Note**
>
> - I gateway NAT hanno un costo ridotto.
> - Più EC2 istanze consumano le ore del piano gratuito più velocemente di una singola istanza. EC2

1. Implementa il CloudFormation modello `basic.yml` in un ambiente di laboratorio.

2. Attendi che lo stato di salute delle istanze del gruppo target passi da iniziale a integro.
3. Accedere all'URL dell'Application Load Balancer in un browser Web, utilizzando HTTP (TCP/80).

   Ad esempio: `http://alb-123456789.us-east-1.elb.amazonaws.com/`

   La pagina Web visualizza l'istanza 1 o l'istanza 2. TG1 TG1
4. Aggiorna la pagina più volte.

Risultati attesi

L'istanza che carica la pagina Web (Istanza 1 o Istanza 2) dovrebbe cambiare ogni volta, come indicato nel testo della pagina. La logica del bilanciamento del carico gestisce l'ultima destinazione su più nodi interni, il che può causare un ritardo di sincronizzazione, quindi esiste la possibilità che l'utente venga indirizzato alla stessa destinazione.

Come funziona

- In questo esempio, due EC2 istanze vengono assegnate a un singolo gruppo target. Sulle EC2 istanze è installato un server web Apache (`httpd`) e il testo della `index.html` pagina su ciascuna EC2 istanza è codificato per identificare l'istanza.
- L'Application Load Balancer esegue la sua logica round robin interna per determinare quale EC2 istanza deve ricevere il traffico.
- Ogni volta che ricarichi la pagina Web, Application Load Balancer esegue la sua logica di routing e nella pagina viene visualizzata l'istanza 1 o l'istanza 2 TG1. TG1

Viscosità del gruppo bersaglio

Quando si utilizza un Application Load Balancer con fedeltà del gruppo target:

- L'Application Load Balancer utilizza il peso del gruppo target per determinare come bilanciare il traffico in entrata tra i gruppi target.
- Per impostazione predefinita, Application Load Balancer utilizza il metodo round robin per indirizzare le richieste alle EC2 istanze del gruppo target di destinazione.

Modello: utilizza il CloudFormation modello `targetgroupstickiness.yml` (incluso nel file.zip del codice di esempio) per provare la fedeltà del gruppo target.

Casi di utilizzo comune

Utilizza la fedeltà del gruppo target in questi scenari:

- Al load balancer sono assegnati più gruppi target e il traffico proveniente da un client deve essere instradato in modo coerente verso le istanze all'interno di quel gruppo target.
- Implementazioni blu/verdi.

Modifiche al codice da basic.yml

È stata apportata un'unica modifica al listener: abbiamo modificato le azioni predefinite di Application Load Balancer per specificare due gruppi target `TG1` (`TG2e`) di uguale peso, con una configurazione di adesività.

basic.yml

```
Listener1:
  Type: 'AWS::ElasticLoadBalancingV
2::Listener'
  Properties:
    LoadBalancerArn: !Ref ALB
    Protocol: HTTP
    Port: 80
```

targetgroupstickiness.yml

```
Listener1:
  Type: 'AWS::ElasticLoadBalancingV
2::Listener'
  Properties:
    LoadBalancerArn: !Ref ALB
    Protocol: HTTP
    Port: 80
```

```
    DefaultActions:
      - TargetGroupArn: !Ref TG1
        Type: forward
```

```
    DefaultActions:
      - ForwardConfig:
          TargetGroups:
            - TargetGroupArn: !Ref TG1
              Weight: 1
            - TargetGroupArn: !Ref TG2
              Weight: 1
          TargetGroupStickinessConfig
:
              DurationSeconds: 10
              Enabled: true
        Type: forward
```

Fasi

> **Note**
>
> - I gateway NAT hanno un costo ridotto.
> - Più EC2 istanze consumano le ore del piano gratuito più velocemente di una singola istanza. EC2

1. Implementa il CloudFormation modello `targetgroupstickiness.yml` in un ambiente di laboratorio.
2. Attendi che lo stato di salute delle istanze del gruppo target passi da iniziale a integro.
3. Accedere all'URL dell'Application Load Balancer in un browser Web, utilizzando HTTP (TCP/80).

   Ad esempio: `http://alb-123456789.us-east-1.elb.amazonaws.com/`

   La pagina Web mostra una delle seguenti opzioni: Istanza 1 - TG1, Istanza 2 -, Istanza 3 - TG1 o Istanza 4 - TG2. TG2
4. Aggiorna la pagina più volte.

Risultati attesi

> **Note**
>
> Il CloudFormation modello in questo esempio configura la viscosità in modo che duri 10 secondi.

Le istanze che caricano la pagina Web devono rimanere all'interno del gruppo target (TG1 o TG2) entro la durata di 10 secondi, come indicato nel testo della pagina.

Dopo circa 10 secondi, la persistenza si attenua e il set di istanze del gruppo target potrebbe cambiare.

Come funziona

- In questo esempio, quattro EC2 istanze sono suddivise in due gruppi target, con due istanze per gruppo target. Sulle EC2 istanze è installato un server web Apache (`httpd`) e il testo della `index.html` pagina su ciascuna EC2 istanza è codificato in modo da essere distinto.
- L'Application Load Balancer crea un'associazione per la sessione dell'utente verso il gruppo target di destinazione, con una scadenza.
- Quando ricaricate la pagina, l'Application Load Balancer verifica se l'associazione esiste e non è scaduta.
  - Se l'associazione è scaduta o non esiste, Application Load Balancer esegue la sua logica di routing e determina il gruppo target di destinazione.
  - Se l'associazione non è scaduta, l'Application Load Balancer indirizza il traffico verso lo stesso gruppo di destinazione, ma non necessariamente verso la EC2 stessa istanza.

Sessioni permanenti con cookie generati dal sistema di bilanciamento del carico

Quando si utilizza un Application Load Balancer con un cookie generato dal load balancer:

- L'Application Load Balancer utilizza il peso del gruppo target per determinare come bilanciare il traffico in entrata tra i gruppi target.
- Per impostazione predefinita, Application Load Balancer utilizza il metodo round robin per indirizzare le richieste alle EC2 istanze del gruppo target di destinazione.

  Dopo che il traffico è stato inizialmente indirizzato a un'istanza, il traffico successivo rimarrà su quell' EC2 istanza per una durata specificata.

Modello: utilizza il CloudFormation modello `stickysessionslb.yml` (incluso nel file.zip con codice di esempio) per provare sessioni permanenti con cookie generati dal sistema di bilanciamento del carico.

Casi di utilizzo comune

Utilizza sessioni permanenti con cookie generati dal load balancer in questi scenari:

- Server web PHP
- Server che conservano dati di sessione temporanei come registri, carrelli della spesa o conversazioni in chat

Modifiche al codice da basic.yml

Le modifiche al codice rilevanti riguardano la configurazione del gruppo target, per impostare il tipo di adesività `lb_cookie` e la durata su 10 secondi.

basic.yml	**stickysessionslb.yml**
`TG1:` `  Type: 'AWS::ElasticLoadBalancingV2::TargetGroup'`	`TG1:` `  Type: 'AWS::ElasticLoadBalancingV2::TargetGroup'`

```
    Properties:
      Name: TG1
      Protocol: HTTP
      Port: 80
      TargetType: instance
      Targets:
        - Id: !Ref Instance1
        - Id: !Ref Instance2
VpcId: !Ref CustomVPC
```

```
    Properties:
      Name: TG1
      Protocol: HTTP
      Port: 80
      TargetType: instance
      Targets:
        - Id: !Ref Instance1
        - Id: !Ref Instance2
      VpcId: !Ref CustomVPC
      TargetGroupAttributes:
        - Key: stickiness.enabled
          Value: true
        - Key: stickiness.type
          Value: lb_cookie
        - Key: stickiness.lb_cook
ie.duration_seconds
Value: 10
```

Fasi

> **Note**
>
> - I gateway NAT hanno un piccolo costo.
> - Più EC2 istanze consumeranno le ore del piano gratuito più velocemente di una singola istanza. EC2

1. Implementa il CloudFormation modello `stickysessionslb.yml` in un ambiente di laboratorio.
2. Attendi che lo stato di salute delle istanze del gruppo target passi da iniziale a integro.
3. Accedere all'URL dell'Application Load Balancer in un browser Web, utilizzando HTTP (TCP/80).

   Ad esempio: `http://alb-123456789.us-east-1.elb.amazonaws.com/`

   La pagina Web mostra una delle seguenti opzioni: Istanza 1 - TG1, Istanza 2 -, Istanza 3 - TG1 o Istanza 4 - TG2. TG1
4. Aggiorna la pagina più volte.

Risultati attesi

> **Note**
>
> Il CloudFormation modello in questo esempio configura la viscosità in modo che duri 10 secondi.

L'istanza che carica la pagina Web deve rimanere invariata entro la durata di 10 secondi, come indicato nel testo della pagina. Dopo circa 10 secondi, la persistenza viene rilasciata e l'istanza di destinazione potrebbe cambiare.

Come funziona

- In questo esempio, due EC2 istanze sono presenti in un gruppo target. Nelle EC2 istanze è installato un server web Apache (`httpd`) e il testo della `index.html` pagina su ciascuna EC2 istanza è codificato in modo da essere distinto.
- L'Application Load Balancer crea un'associazione per la sessione dell'utente, che si collega alla destinazione, con una data di scadenza.
- Quando ricaricate la pagina, l'Application Load Balancer verifica se l'associazione esiste e non è scaduta.
  - Se l'associazione è scaduta o non esiste, Application Load Balancer esegue la logica di routing e determina l'istanza di destinazione.
  - Se l'associazione non è scaduta, l'Application Load Balancer indirizza il traffico verso la stessa istanza di destinazione.

Sessioni permanenti con cookie basati su applicazioni

Quando si utilizza un Application Load Balancer con un cookie basato sull'applicazione:

- L'Application Load Balancer utilizza il peso del gruppo target per determinare come bilanciare il traffico in entrata tra i gruppi target.
- Per impostazione predefinita, Application Load Balancer utilizza il metodo round robin per indirizzare le richieste alle EC2 istanze del gruppo target di destinazione.
- Dopo che il traffico è stato inizialmente indirizzato a un' EC2 istanza, la risposta dell'applicazione dell' EC2 istanza deve contenere un cookie dell'applicazione personalizzato, che viene inviato al client insieme a un cookie Automatico di Application Load Balancer.
- Il traffico successivo si limiterà all' EC2 istanza se il client restituisce il cookie dell'applicazione e il cookie Application Load Balancer.
- Il cookie basato sull'applicazione scade dopo il mancato utilizzo per la durata configurata.

Modello: utilizza il CloudFormation modello `stickysessionsapp.yml` (incluso nel file zip con codice di esempio) per provare sessioni permanenti con cookie basati sull'applicazione.

Casi di utilizzo comune

Utilizza sessioni permanenti con cookie generati dall'applicazione quando desideri un controllo aggiuntivo in questi scenari:

- Server web PHP
- Server che conservano dati di sessione temporanei come registri, carrelli della spesa o conversazioni in chat

Modifiche al codice da basic.yml

L'unica modifica al codice è nella configurazione del gruppo target. Abbiamo aggiunto una configurazione di stickiness agli attributi dell'Application Load Balancer e del gruppo target. La durata del cookie dell'applicazione è specificata e il gruppo target ha abilitato la persistenza dei cookie dell'applicazione.

basic.yml

```
TG1:
   Type: 'AWS::ElasticLoadBalancingV
2::TargetGroup'
   Properties:
     Name: TG1
     Protocol: HTTP
     Port: 80
     TargetType: instance
     Targets:
       - Id: !Ref Instance1
       - Id: !Ref Instance2
VpcId: !Ref CustomVPC
```

stickysessionsapp.yml

```
TG1:
  Type: 'AWS::ElasticLoadBalancingV
2::TargetGroup'
  Properties:
    Name: TG1
    Protocol: HTTP
    Port: 80
    TargetType: instance
    Targets:
      - Id: !Ref Instance1
      - Id: !Ref Instance2
    VpcId: !Ref CustomVPC
    TargetGroupAttributes:
      - Key: stickiness.enabled
        Value: true
      - Key: stickiness.type
        Value: app_cookie
      - Key: stickiness.app_coo
kie.duration_seconds
        Value: 10
      - Key: stickiness.app_coo
kie.cookie_name
        Value: TESTCOOKIE
```

Fasi

> **Note**
>
> - I gateway NAT hanno un piccolo costo.
> - Più EC2 istanze consumeranno le ore del piano gratuito più velocemente di una singola istanza. EC2

1. Implementa il CloudFormation modello `stickysessionslb.yml` in un ambiente di laboratorio.
2. Attendi che lo stato di salute delle istanze del gruppo target passi da iniziale a integro.

3. Accedere all'URL dell'Application Load Balancer in un browser Web, utilizzando HTTP (TCP/80).

   Ad esempio: `http://alb-123456789.us-east-1.elb.amazonaws.com/`

   La pagina Web mostra una delle seguenti opzioni: Istanza 1 - TG1, Istanza 2 -, Istanza 3 - TG1 o Istanza 4 - TG2. TG2
4. Aggiorna la pagina più volte.

Risultati attesi

> **Note**
>
> Il CloudFormation modello in questo esempio ha configurato la viscosità in modo che duri 10 secondi. La configurazione valida della durata della viscosità è compresa tra 1 secondo e 1 settimana.

L'istanza che carica la pagina Web dovrebbe rimanere la stessa, come indicato dal testo della pagina.

La durata della permanenza non si aggiorna, ma si basa sulla scadenza configurata nell'Application Load Balancer per il cookie dell'applicazione generato dall'istanza. EC2

Esempio 1: attendi 5 secondi per aggiornare la pagina. La stessa istanza verrà caricata e la viscosità verrà aggiornata per altri 10 secondi.

Esempio 2: Attendi più di 10 secondi per ricaricare la pagina. Il cookie dell'applicazione scade e l'utente viene indirizzato a un'istanza diversa. EC2 Questa nuova istanza genera un altro cookie dell'applicazione con una durata di 10 secondi.

Come funziona

- In questo esempio sulle EC2 istanze è installato un server web Apache ()`httpd`. Il `httpd.conf` file è configurato per restituire un `Set-Cookie` valore statico al client (il browser Web). Il `Set-Cookie` valore è codificato per essere. `TESTCOOKIE=<somevalue>`
- Apri l'opzione Inspect Element del browser, scegli la scheda Rete, quindi scegli il metodo Get, che carica la pagina. Verrà visualizzata la scheda Cookie.

- Il browser è un'applicazione client configurata automaticamente per restituire eventuali aggiornamenti successivi al server con i cookie che riceve nella `Set-Cookie` risposta del server.
- Quando ricarichi la pagina, i cookie ricevuti durante il caricamento iniziale della pagina vengono automaticamente rispediti all'Application Load Balancer.
  - Se il cookie è scaduto (ovvero sono trascorsi 10 secondi dall'ultima chiamata), Application Load Balancer utilizza una nuova logica per determinare a EC2 quale istanza indirizzare il traffico.
  - Se il cookie non è scaduto, l'Application Load Balancer indirizza il traffico verso la EC2 stessa istanza.

Resources

- Sessioni permanenti per il tuo Application Load Balancer (documentazione Elastic Load Balancing)

Cronologia dei documenti

La tabella seguente descrive le modifiche significative apportate a questa guida. Per ricevere notifiche sugli aggiornamenti futuri, puoi abbonarti a un feed RSS.

Modifica	Descrizione	Data
Sezione aggiornata	È stata aggiornata la sezione relativa alle sottoreti e al routing del Load Balancer con nuovi diagrammi e chiarimenti.	5 luglio 2024
Aggiornamenti e correzioni	Sono stati aggiornati il codice, i diagrammi e gli esempi.	31 maggio 2023
Sezioni aggiornate	Sono state aggiornate le sezioni Come funziona in Target group stickiness e Sticky sessions with load balancer per fornire informazioni più accurate e dettagliate.	18 luglio 2022
—	Pubblicazione iniziale	5 agosto 2021

Le traduzioni sono generate tramite traduzione automatica. In caso di conflitto tra il contenuto di una traduzione e la versione originale in Inglese, quest'ultima prevarrà.