HOSTARE UN SITO STATICO SU AWS: CLOUDFRONT È SEMPRE LA SCELTA GIUSTA?

Amazon CloudFront | Amazon S3 | Infrastructure as Code (IaC)

beSharp | 26 Giugno 2020

Introduzione

Il binomio di servizi **CloudFront** – **S3** fa oramai parte delle pratiche consolidate di molte aziende che hanno la necessità di ospitare un sito statico (o parte di esso) a **prezzi contenuti senza rinunciare però ai crismi di sicurezza** (come ad esempio la terminazione SSL sul proprio dominio). Ma è vero che è l'unica via percorribile? Esistono altri modi per conseguire il medesimo risultato che però rispondono ad esigenze differenti? In questo articolo si prova a dare una risposta!

Problema

Immaginiamo di dover rispondere alla seguente esigenza: il team di frontend della propria azienda richiede di poter testare ciò che ha appena sviluppato su un'infrastruttura simile a quella finale per mostrare anche solo a livello visivo che il risultato sia quello aspettato. In particolare, avrebbe piacere nell'avere a disposizione un nome di dominio differente per ogni feature in modo tale da non introdurre livelli di inconsistenza relativi ai path.

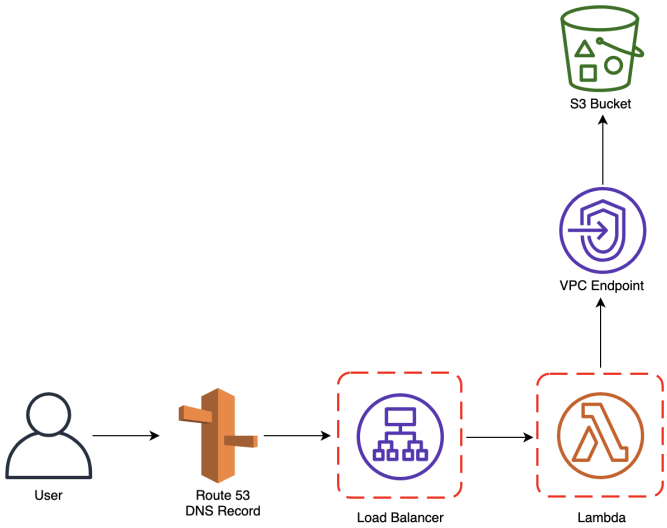
Anche senza prendere la calcolatrice in mano per fare i conti tentando di prevedere il traffico gestito, è subito chiaro che mantenere una differente distribuzione CloudFront per ogni feature e per ogni progetto frontend possa generare inutili complicazioni a livello di gestione dell'account AWS stesso. Vale la pena far notare che non è possibile creare due origin differenziate dalla path principale e collegarle a due domini differenti all'interno della stessa distribuzione CloudFront senza incorrere in almeno un redirect che porterebbe ai problemi di path sopra citati).

Soluzione

In prima istanza constatiamo la natura effimera di questi deploy. Ciò richiede che vi siano dei tempi relativamente ristretti di creazione e rimozione di questi elementi infrastrutturali. Per raggiungere questo obiettivo è utile quindi la condivisione delle medesime risorse da parte di più interlocutori in modo tale da minimizzare l'overhead introdotto.

La soluzione proposta è composta dai seguenti elementi infrastrutturali:

- 1 Bucket S3 privato (con Bucket policy)
- 1 LoadBalancer (con 2 Listener, 1 Target Group, 1 Security Group)
- 1 Lambda (in VPC con 1 Security Group)
- 1 S3 VPC Endpoint
- n DNS record su Route 53 (dove n è pari al numero di feature da validare in un determinato momento)



Creazione e configurazione del Bucket S3

Per ottenere il risultato desiderato è fondamentale la creazione del Bucket S3 con la configurazione corretta. Di seguito la definizione in CloudFormation:

```
S3Bucket:
 Type: AWS::S3::Bucket
 Properties:
   BucketName: 'subdomain.mydomain.com'
   CorsConfiguration:
     CorsRules:
       - AllowedHeaders:
           - '*'
         AllowedMethods:
           - GET
           - HEAD
           - POST
           - PUT
           - DELETE
         AllowedOrigins:
           - 'https://*.mydomain.com'
   PublicAccessBlockConfiguration:
     BlockPublicAcls: true
     BlockPublicPolicy: true
     IgnorePublicAcls: true
     RestrictPublicBuckets: true
   WebsiteConfiguration:
     ErrorDocument: error.html
```

```
      IndexDocument: index.html

  S3BucketPolicy:
   Type: AWS::S3::BucketPolicy
   Properties:
     Bucket: !Ref S3Bucket
     PolicyDocument:
       Version: '2012-10-17'
       Statement:
         - Sid: VPCEndpointReadGetObject
           Effect: Allow
           Principal: "*"
           Action: s3:GetObject
           Resource: !Sub '${S3Bucket.Arn}/*'
           Condition:
             StringEquals:
               aws:sourceVpce: !Ref S3VPCEndpointId
```

Come si può notare, è stata attivata la “website configuration” in modo tale da poterci interfacciare con il bucket tramite chiamate HTTP ma allo stesso tempo è presente anche una Bucket Policy che vieta il recupero di un qualsiasi oggetto a meno che la richiesta non passi dal VPC Endpoint di S3, garantendo quindi che solo gli interlocutori che passano dalla VPC dell’account possano accedere al Bucket stesso.

Creazione e configurazione del Load Balancer

Per permettere all’utente finale di visualizzare il sito statico ospitato su S3 occorre creare un Load Balancer (a seconda di chi possa essere questo utente finale, si può scegliere se rendere il Load Balancer privato o meno). In CloudFormation questo è l’insieme di risorse da creare:

```
LoadBalancer:
 Type: AWS::ElasticLoadBalancingV2::LoadBalancer
 Properties:
   Name: !Sub '${ProjectName}'
   LoadBalancerAttributes:
     - Key: 'idle_timeout.timeout_seconds'
       Value: '60'
     - Key: 'routing.http2.enabled'
       Value: 'true'
     - Key: 'access_logs.s3.enabled'
       Value: 'true'
     - Key: 'access_logs.s3.prefix'
       Value: loadbalancers
     - Key: 'access_logs.s3.bucket'
       Value: !Ref S3LogsBucketName
   Scheme: internet-facing
   SecurityGroups:
     - !Ref LoadBalancerSecurityGroup
   Subnets:
     - !Ref SubnetPublicAId
     - !Ref SubnetPublicBId
     - !Ref SubnetPublicCId
   Type: application

LoadBalancerSecurityGroup:
```

```
    Type: AWS::EC2::SecurityGroup
    Properties:
      GroupName: !Sub '${ProjectName}-alb'
      GroupDescription: !Sub '${ProjectName} Load Balancer Security Group'
      SecurityGroupIngress:
        - CidrIp: 0.0.0.0/0
          Description: ALB Ingress rule from world
          FromPort: 80
          ToPort: 80
          IpProtocol: tcp
        - CidrIp: 0.0.0.0/0
          Description: ALB Ingress rule from world
          FromPort: 443
          ToPort: 443
          IpProtocol: tcp
      Tags:
        - Key: Name
          Value: !Sub '${ProjectName}-alb'
        - Key: Environment
          Value: !Ref Environment
      VpcId: !Ref VPCId

  HttpListener:
    Type: AWS::ElasticLoadBalancingV2::Listener
    Properties:
      DefaultActions:
        - RedirectConfig:
            Port: '443'
            Protocol: HTTPS
            StatusCode: 'HTTP_301'
          Type: redirect
      LoadBalancerArn: !Ref LoadBalancer
      Port: 80
      Protocol: HTTP

  HttpsListener:
    Type: AWS::ElasticLoadBalancingV2::Listener
    Properties:
      Certificates:
        - CertificateArn: !Ref LoadBalancerCertificateArn
      DefaultActions:
        - Type: forward
          TargetGroupArn: !Ref TargetGroup
      LoadBalancerArn: !Ref LoadBalancer
      Port: 443
      Protocol: HTTPS

  TargetGroup:
    Type: AWS::ElasticLoadBalancingV2::TargetGroup
    Properties:
      Name: !Sub '${ProjectName}'
      HealthCheckEnabled: false
      TargetType: lambda
      Targets:
        - Id: !GetAtt Lambda.Arn
    DependsOn: LambdaPermission
```

Tramite questo template, viene deployato un Load Balancer pubblico con un listener che ascolta sulla porta 80 (HTTP) che effettua una redirect su 443 (HTTPS) su cui è presente un altro listener che però contatta un Target Group su cui è registrata una Lambda.

Codice della Lambda di routing

La parte di logica della soluzione è demandata alla Lambda. Qui di seguito trovate un esempio di codice a titolo dimostrativo (a tal ragione si consiglia di rivederlo e adattarlo per soluzioni production-ready):

```
import json
from boto3 import client as boto3_client
from os import environ as os_environ
import base64
from urllib3 import PoolManager

http = PoolManager()
s3 = boto3_client('s3')

def handler(event, context):
   try:
       print(event)
       print(context)

       host = event['headers']['host']
       print("Host:", host)

       feature = host.split('.')[0]
       feature = "-".join(feature.split('-')[1:])
       print("Feature:", feature)

       path = event['path'] if event['path'] != "/" else "/index.html"
       print("Path:", path)

       query_string_parameters = event['queryStringParameters']
       query_string_parameters = [f"{key}={value}" for key, value in event['queryStri
ngParameters'].items()]
       print("Query String Parameters:", query_string_parameters)

       http_method = event["httpMethod"]
       url = f"http://{os_environ['S3_BUCKET']}.s3-website-eu-west-1.amazonaws.com/{f
eature}{path}{'?' if [] != query_string_parameters else ''}{'&'.join(query_string_para
meters)}"
       print(url)

       headers = event['headers']
       headers.pop("host")
       print("Headers:", headers)

       body = event['body']
       print("Body:", body)

       r = http.request(http_method, url, headers=headers, body=body)
       print("Response:", r)
       print("Response Data:", r.data)

       try:
```

```
            decoded_response = base64.b64encode(r.data).decode('utf-8')
        except:
            decoded_response =  base64.b64encode(r.data)

        print("Decoded Response:", decoded_response)
        print("Headers Response:", dict(r.headers))
        return {
            'statusCode': 200,
            'body': decoded_response,
            "headers": dict(r.headers),
            "isBase64Encoded": True
        }
    except Exception as e:
        print(e)
        return {
          'statusCode': 400
        }
```

Nonostante non sia di immediata lettura, le operazioni effettuate sono molto semplici: partendo dal DNS name con cui l'utente ha raggiunto il Load Balancer, la Lambda gira la chiamata verso il Bucket S3 costruendo la sottocartella da contattare contenente una determinata feature. Per far si che tutto ciò funzioni bisogna chiaramente creare un DNS name per ciascuna feature.

Conclusione

Abbiamo visto come, grazie ad una opportuna configurazione delle risorse coinvolte, sia possibile mantenere un Bucket S3 privato nonostante il website hosting attivo, con lo scopo di avere una versione di un sito statico per ogni sottocartella definita, raggiungibile tramite differenti DNS name. Tale approccio è indubbiamente più veloce e agile quando le necessità sono legate alla verifica dell'aspetto visivo del sito e non della configurazione infrastrutturale, evitando quindi l'onere di tempo e di gestione di una distribuzione CloudFront.

Un buon risultato, vero? 🙂 Scriveteci per approfondire!

beSharp

Dal 2011 beSharp guida le aziende italiane sul Cloud. Dalla piccola impresa alla grande multinazionale, dal manifatturiero al terziario avanzato, aiutiamo le realtà più all'avanguardia a realizzare progetti innovativi in campo IT.

Get in touch

beSharp.it
proud2becloud@besharp.it