Decentralized Virtual CDN With opportunistic offloading

António Santos
Network Engineering Master's Degree
Universidade do Porto
up201907156@up.pt

Hugo Rodrigues
Network Engineering Master's Degree
Universidade do Porto
up202302523@up.pt

I. Introduction

No âmbito da unidade curricular de Administração de Sistemas Cloud realizámos um projeto com o objetivo de criar um serviço descentralizado, utilizando as ferramentas da Google Cloud e Terraform. Este projeto visa proporcionar o fornecimento eficiente e seguro de conteúdo a clientes espalhados pelo mundo, atendendo às exigências regulatórias, como o GDPR e priorizando a privacidade dos dados.

Este projeto foca na implementação de uma CDN descentralizada utilizando ferramentas do Google Cloud e Terraform. Para otimizar a entrega de conteúdos de maneira eficiente e segura desenvolvemos uma arquitetura que garante baixa latência e alta disponibilidade. A nossa abordagem envolve a utilização de proxies HTTPS, balanceadores de carga, buckets e uma API REST para atender às requisições dos clientes. Este trabalho alinha-se às expectativas regulatórias, como o GDPR, oferecendo uma solução robusta que se adapta ao tráfego dinâmico dos clientes, distribuindo a carga de forma eficiente e segura.

II. Arquitetura

Para a nosssa solução de arquitetura utilizámos o Ceph como sistema de ficheiros distribuídos configurando uma CDN em uma máquina virtual como imagem CentOS e disponibilizando os vídeos através de URLs acessíveis por um balanceador de carga. Quando um cliente tenta acesar a um website, ele digita o domínio no browser http://blog.ns1video22world.com:443/. O servidor DNS responde com endereço IP associado ao domínio registo A.

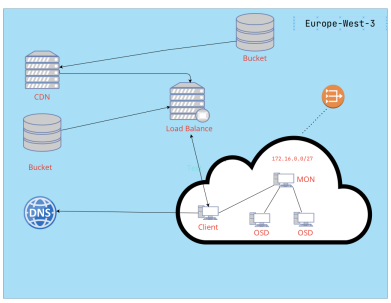

Fig. 1. Arquitetura do sistema

A tecnologia escolhida com base neste projeto foi o **Terraform** onde criamos e configuramos uma instância da máquina virtual no GCP, garantindo consistência e eficiência na gestão dos recursos da infraestrutura na nuvem.

A configuração da figura abaixo, utiliza a instância de VM uma imagem CentosOS Stream 8, inicializando um disco SSD persistente com 25GB de tamanho. Os discos foram escolhidos pela sua performance e operações de entrada e saída de modo a garantir desempenho rápido e responsivo. Cada Vm possui um endereço IP estático, obtido através do recurso `google_compute_address`. Além disso, o acesso à internet é gerenciado através da configuração NAT como um nível de serviço PREMIUM.

```
resource "google_compute_instance" "node01" {
  name         = local.host_name
  machine_type = "e2-custom-2-4096"
  zone         = var.google_cloud_zone

  can_ip_forward      = false
  deletion_protection = false
  enable_display      = true

  tags = var.google_cloud_tags

  You, 2 months ago | 1 author (You)
  labels = {
    goog-ec-src = "vm_add-tf"
  }

  You, 2 months ago | 1 author (You)
  boot_disk {
    auto_delete = true
    device_name = local.host_name

    You, 2 months ago | 1 author (You)
    initialize_params {
      image = "projects/centos-cloud/global/images/centos-stream-8-v20231115"
      size  = 40
      type  = "pd-ssd"
    }
    mode = "READ_WRITE"
  }
```

Fig. 2. Módulo da VM

III. Custo

O SLA do Google Cloud Build garante uma disponibilidade mensal de **99,95%**. Se essa meta não for atingida, o cliente pode receber créditos financeiros como compensação, desde que siga os procedimentos para solicitação desses créditos. Os créditos variam de **10%** a **50%** da fatura mensal, dependendo da percentagem de uptime. Para receber os créditos, o cliente

deve notificar o suporte técnico da Google dentro de 30 dias após a falhas. O SLA exclui serviços em pré-disponibilidade geral, erros fora do controle da Google, problemas causados por software ou hardware do cliente, abusos ou comportamento que violem o acordo e cotas aplicadas pelo sistema.

Porcentagem de tempo de atividade mensal	Porcentagem da fatura mensal do respectivo Serviço Coberto que não atende ao SLO que será creditada em futuras faturas mensais do Cliente
99,00% - < 99,95%	10%
95,0% - < 99,0%	25%
< 95,0%	50%

Fig. 3. Créditos SLA

Uso de uma Vm no Goole Cloud traz algumas vantagens, como a redução dos custos associados ao consumo de eletricidade e manutenção de hardware físico, além de proporcionar escalabilidade, alta disponibilidade e segurança.

Estimativa mensal

US$ 33,91

Cerca de US$ 0,05 por hora

Pague pelo que usar: faturamento por segundo e sem custos iniciais

Item	Estimativa mensal
2 vCPU + 4 GB memory	US$ 31,51
Disco permanente balanceado com 20 GB	US$ 2,40
Total	**US$ 33,91**

Fig. 4. Tabela de preço de cada VM

A. Rede privada

Na nossa arquitetura, implementamos uma rede privada configurada na subnet 172.16.0.0/27, composta por um cliente, uma máquina de monitoramento (MON), e duas máquinas de armazenamento de objetos (OSD). Esta configuração foi escolhida para utilizar o Ceph, um sistema de armazenamento distribuído, com o objetivo de gerenciar a distribuição e a redundância dos dados de forma eficiente e robusta.

1) No04: O "NO04" é responsável por enviar solicitações de dados para a rede simulando o comportamento de utilizadores finais acessando ao conteúdo da CDN. O cliente pode armazenar até 100 MB de dados em cache localmente. Este cache é usado para reduzir a latência ao acessar dados frequentemente solicitados.

2) MON: A máquina de monitoramento é um componente crucial na nossa rede privada desempenhando várias funções vitais para a operação do sistema. O MON gerencia o mapeamento dos dados no cluster, rastreando onde cada pedaço de dados está armazenado em qualquer momento. Outra função do MON é o monitoramento constante da saúde do cluster, verificando a integridade dos OSDs e outros componentes.

3) MGR: O MGR (Manager) monitora os serviços do cluster, verifica a resiliência das máquinas, identifica possíveis falhas, e avalia as relíquias dos objetos, determinando se estão em níveis altos ou baixos, garantindo assim a saúde e o desempenho do sistema. Além disso, o MGR fornece acesso a um dashboard para visualização e gerenciamento do status do sistema.

4) OSD: Na nossa arquitetura temos dois componentes OSDs que armazenam fisicamente os dados e são fundamentais para a resiliência e desempenho do sistema. Cada OSD armazena uma parte dos dados do cluster, distribuindo a carga de armazenamento e processamento entre várias máquinas.

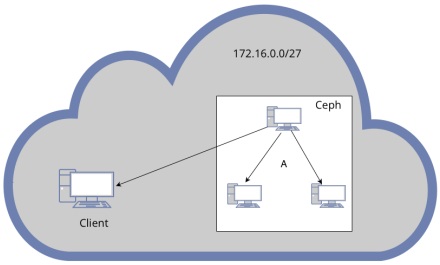

Fig. 5. Sistema de distribuição

B. Bucket

No nosso projeto, criamos dois buckets distintos no Google Cloud Storage para gerenciar o armazenamento e a distribuição de conteúdo.

O primeiro bucket denominado bucket_cloud_systems é utilizado para armazenar o conteúdo estático do nosso website. Localizado na região europe-west3 e utilizando o tipo de armazenamento "STANDARD" este bucket serve como deployment para a produção do website. O nosso balanceador de carga retransmite o conteúdo deste bucket para a máquinas cliente e garante assim uma entrega rápida e eficiente.

O segundo bucket é utilizado especificamente para armazenar os vídeos e permite um gerenciamento mais eficiente dos diferentes tipos de conteúdo.

Para garantir a segurança dos dados configuramos as permissões dos buckets para que apenas a nossa rede privada tenha acesso de leitura e escrita, evitando acessos não autorizados.

Esta configuração assegura que o primeiro bucket serve como base de deployment para o nosso website, enquanto o segundo bucket é dedicado ao armazenamento e distribuição

de vídeos via CDN. Dessa forma conseguimos entregar tanto o conteúdo estático quanto o conteúdo de mídia de maneira eficiente e segura aos nossos clientes.

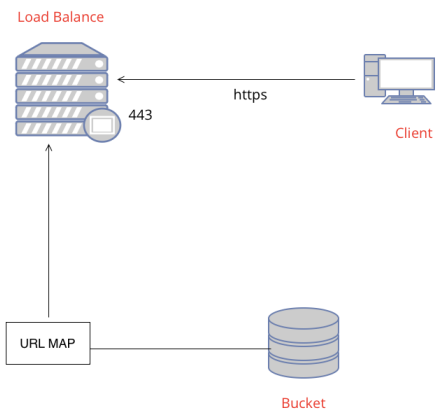

Fig. 6. Balanceamento de Carga

C. CDN

A CDN no nosso projeto está configurada especificamente para hospedar a nossa API Spring que é essencial para fornecer funcionalidades dinâmicas e interativas ao nosso site. Ao ser hospedada na CDN a API beneficia-se de baixa latência e alta disponibilidade, garantindo que as requisições dos usuários sejam processadas de maneira rápida e confiável.

A CDN no nosso projeto oferece vários benefícios significativos nomeadamente a redução de latencia e a distribuição geográfica do servidor garante alta disponibilidade da API, mesmo em caso de falha dos servidores.

Para implementar a CDN no nosso projeto configuramos o balanceador de carga NGINX para trabalhar em conjunto com a CDN na região Europe-West3-Frankfurt.

A CDN foi configurada numa máquina virtual que consiste em requisições via URL garantindo uma entrega rápida e segura das funcionalidades do site.

D. Balanceador de Carga

No nosso projeto utilizamos o NGINX para implementar o balanceamento de carga, garantindo a distribuição eficiente do tráfego entre os diferentes recursos.

Esta configuração assegura que as requisições dos clientes sejam atendidas de forma rápida e equilibrada de modo a melhorar a performance e a disponibilidade do sistema. O NGINX está configurado para gerenciar múltiplos servidores de backend de modo a distribuir as requisições conforme definido no arquivo de configuração. Por outro lado configuramos um backend bucket denominado ”website-backend” para armazenar os arquivos necessários para o website.

Também criamos um certificado SSL gerenciado ”website-cert” para assegurar conexões seguras configurado para os domínios relacionados ao nosso site e API. Utilizamos proxies HTTPS e HTTP para encaminhar o tráfego assegurando que todas as conexões sejam criptografadas e estabelecemos regras de encaminhamento globais para redirecionar o tráfego externo para os proxies. A configuração do NGINX distribui a carga entre várias instâncias de backend garantindo alta disponibilidade, segurança nas comunicações com HTTPS, flexibilidade e escalabilidade para ajustar a distribuição conforme o tráfego.

No nosso projeto, o balanceamento de carga é configurado para distribuir o tráfego HTTP(S) globalmente. As requisições dos clientes chegam ao balanceador de carga via HTTPS (porta 443) e são encaminhadas pelo proxy HTTPS `google_compute_target_https_proxy` utilizando certificados SSL gerenciados `google_compute_managed_ssl_certificate`. O tráfego é então direcionado pelo URL Map `google_compute_url_map` para os backend buckets apropriados que armazenam os arquivos necessários para o site e o CDN. As regras de encaminhamento globais (`google_compute_global_forwarding_rule`) garantem que o tráfego externo seja corretamente direcionado para os proxies configurados, proporcionando alta disponibilidade, segurança e eficiência na distribuição do tráfego.

E. DNS

Utilizamos o serviço de DNS do Google Cloud para gerenciar a resolução de nomes de domínio dos nossos vídeos e da API Spring onde configuramos uma zona gerenciada chamada ”videos-api-zone” com o domínio ”ns1video22world.com”. Dentro desta zona, criamos um registro do tipo ”A” para o subdomínio ”blog.ns1video22world.com.” que aponta para o endereço IP global da nossa CDN.

Esta configuração fornece aos clientes o acesso aos vídeos e a API de maneira intuitiva, utilizando URLs facilmente legiveis. A zona DNS é pública e garante que qualquer cliente possa resolver os nomes de domínio para acessar aos recursos do nosso site.

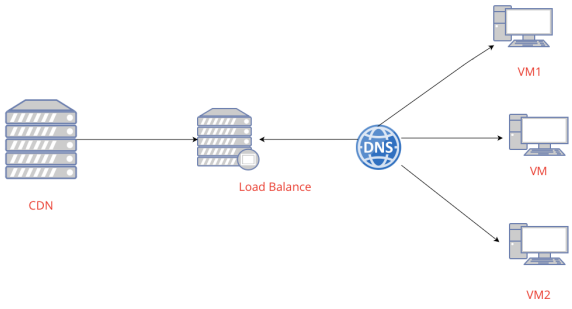

Fig. 7. Name Server

F. Firewall

No nosso projeto configuramos também uma firewall utilizando o Google Compute Engine para proteger a nossa infraestrutura de rede. A firewall, chamada ”cdn-firewall” está

associada à nossa rede interna e é configurada para permitir apenas o tráfego autorizado.

As regras de firewall permitem o tráfego ICMP para diagnósticos e tráfego TCP nas portas essenciais, como 22 (SSH), 80 (HTTP) e 443 (HTTPS) bem como portas específicas para a comunicação interna do Ceph e outros serviços necessários. O tráfego é permitido de qualquer endereço IP e os logs de metadados são ativados para monitoramento e auditoria aprofundados.

G. NAT

Configuramos uma NAT para permitir que as instâncias da nossa rede privada acessem a internet de maneira segura. Utilizamos o Google Cloud NAT para mapear endereços IP privados para públicos, gerenciando automaticamente a tradução e protegendo a rede garantindo que as VMs possam fazer solicitações externas sem expor seus IPs privados, evitando acessos não autorizados e mantendo a comunicação controlada com recursos externos.

IV. Conclusão

Em suma, o projeto desenvolvido demonstra um sistema eficaz e seguro de uma CDN descentralizada que garante baixa latência e alta disponibilidade.

A arquitetura implementada além de garantir resiliência e desempenho através do armazenamento distribuído com uso do Ceph utiliza um balanceador de carga que distribui eficientemente a carga e melhora significativamente a performance do nosso sistema. Adotamos medidas de segurança com o uso reforçado de firewall e NAT, e o uso de certificados SSL gerenciados e regras de encaminhamento globais assegura a segurança das comunicações.

Adicionalmente, configuramos buckets no Google Cloud Storage para armazenar e distribuir tanto o conteúdo estático quanto os vídeos garantindo uma gestão eficiente dos diferentes tipos de conteúdo. Utilizamos o serviço de DNS do Google Cloud para gerenciar a resolução de nomes de domínio de modo a assegurar que os clientes possam acessar os vídeos e a API de maneira intuitiva e segura.

Esta combinação de tecnologias e práticas assegura que a nossa CDN descentralizada é robusta, eficiente e preparada para atender às necessidades dos clientes mantendo a conformidade com as exigências regulatórias e garantindo a privacidade dos dados.

References

[1] Google Cloud, “Content Delivery Network Documentation,” [Online]. Available: https://cloud.google.com/cdn/docs/overview. [Accessed: 25-May-2024].

[2] Google Cloud, “Managing Infrastructure as Code with Terraform,” [Online]. Available: https://cloud.google.com/docs/terraform. [Accessed: 25-May-2024].

[3] Malek, A. (2019, May 9). Configuring Google Cloud CDN with Terraform. Medium. Retrieved from https://medium.com/cognite/configuring-google-cloud-cdn-with-terraform-ab65bb0456a9.

[4] Google Cloud, “Exemplos de Módulos do Terraform para Balanceamento de Carga HTTP(S) Externo,” [Online]. Available: https://cloud.google.com/load-balancing/docs/https/ext-http-lb-tf-module-examples?hl=pt-br. [Accessed: 25-May-2024].

[5] Google Cloud, “Set up Cloud DNS using Terraform,” Medium, 13-Feb-2020. [Online]. Available: https://medium.com/google-cloud/google-cloud-set-up-cloud-dns-using-terraform-20c12038e7fe. [Accessed: 25-May-2024].

[6] Chavhan, J. (2020, August 14). GCP: How to deploy Cloud NAT with Terraform. Medium. Retrieved from https://medium.com/google-cloud/gcp-how-to-deploy-cloud-nat-with-terraform-44745a4daaa8.