Escuela de Ingeniería en Computación

Proyecto 1

Redes

Profesor: Gerardo Nereo Campos Araya

Estudiantes:

Ángel Villalobos Peña - 2014015712

Sebastián Díaz Obando - 2020041942

Fernando Alvarez Olsen - 2019171657

Tania María Sánchez Irola - 2018138723

David Jose Espinoza Soto - 2016012024

Fecha de entrega: Viernes 13 de Octubre del 2023

Introducción

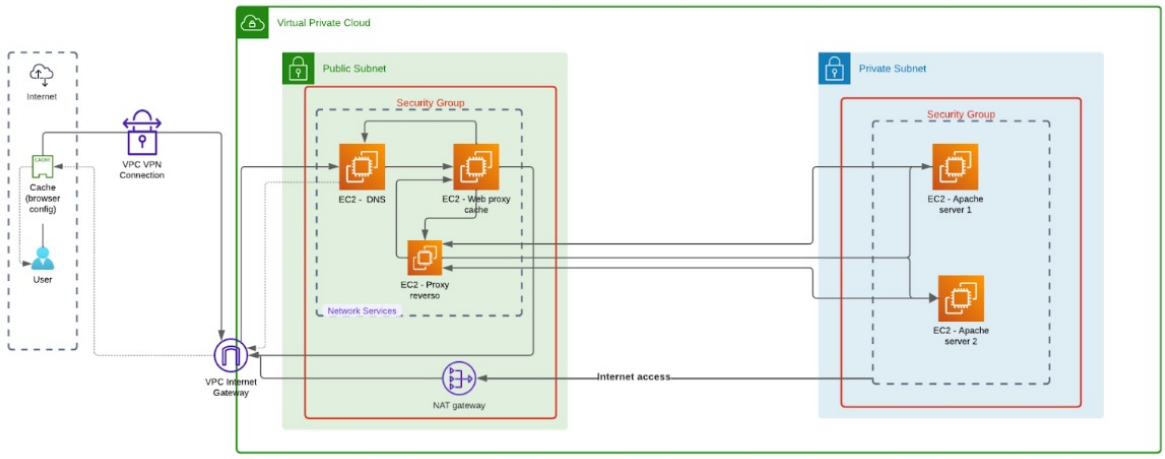
El proyecto se centra en la implementación de componentes de red en un entorno de Cloud Provider, con una serie de objetivos específicos, y requiere la creación de una red virtual, la instalación y configuración de servicios de capa de aplicación, la implementación de firewalls y security groups, un proxy reverso, configuración de un servicio VPN, un Web Cache, un servicio DNS, un router/NAT gateway en software, y la instalación de Web Servers para probar la red implementada.

Algunos aspectos clave del proyecto son:

- Creación de Red Virtual: Se debe crear una red virtual en un Cloud Provider utilizando Terraform. Esta red virtual debe incluir al menos dos subredes (una privada y una pública) con direcciones IP específicas y máquinas virtuales con reglas de seguridad.
- Servicio de DNS: Se debe implementar un servicio DNS en una máquina virtual que gestionará diferentes zonas DNS y configuraciones de resolución.
- Proxy Reverso: Se debe crear un proxy reverso utilizando NGINX junto con servidores Apache. El proxy redireccionará el tráfico a diferentes servicios según el PATH.
- Configuración de VPN: Se debe implementar un servicio VPN utilizando OpenVPN para enrutar el tráfico de red y permitir el acceso a la red interna.

- Web Cache: Se debe configurar un Web Proxy Cache transparente utilizando Squid Proxy Cache para mantener un caché de tráfico HTTP y HTTPS.
- Router/NAT Gateway: Se debe configurar un router en software que permita el acceso a Internet desde una red privada y reglas de NAT para el tráfico saliente.
- Automatización: La implementación debe estar completamente automatizada, utilizando herramientas como Chef Solo, Puppet, Ansible o equivalentes para la configuración.

Flujo



- Usuario se conecta a través del VPN (si es necesario)
    - Si el acceso debe ser a través de un VPN, el usuario primero establecerá una conexión VPN.
- Usuario hace una solicitud:
    - A través del navegador, el usuario hace una solicitud a una URL, por ejemplo www.asimov.io/server1.
    - La solicitud del usuario, desde el Internet público, llega a la VPC a través del IGW.
    - El DNS server resuelve la IP al Squid Web CachéSe debe configurar el proxy en el navegador, así que la solicitud irá al Squid Proxy Cache.
- Squid Web Proxy Cache:
    - Al recibir la solicitud, Squid primero verificará si tiene una versión almacenada (en caché) del recurso solicitado.
    - Si tiene una versión en caché que aún no ha expirado (según las políticas de expiración que configures), Squid responderá directamente al usuario sin tener que procesar la solicitud más adelante.
    - Si el recurso no se encuentra en caché o ha expirado, Squid pasará la solicitud al siguiente destino, que sería el DNS en este caso.
- Resolución DNS:
    - El servidor DNS resuelve la dirección IP del proxy inverso (NGINX).
- Proxy Inverso (NGINX):
    - NGINX revisa la ruta del PATH. Si es /server1, reenvía la solicitud al Apache 1. Si es /server2, la reenvía al Apache 2.
- Servidores Apache:
    - Reciben la solicitud desde NGINX y procesan la respuesta.
- Respuesta a través de Squid:

    - La respuesta de los servidores Apache se envía de regreso a NGINX, que a su vez la envía de vuelta a Squid. Si la respuesta es cacheable, Squid la almacenará y enviará la respuesta al usuario. Esto significa que futuras solicitudes similares pueden ser atendidas directamente desde el caché de Squid sin tener que pasar por todo el flujo de nuevo.
- Respuesta al usuario:
    - Finalmente, la respuesta llega al usuario. Si el usuario está conectado a través de VPN, la respuesta se envía a través de esa conexión VPN segura.

En este flujo, Squid actúa como una capa intermedia entre el usuario y los recursos en la infraestructura. Al configurar el proxy en el navegador, todo el tráfico del navegador pasa por Squid, permitiendo el almacenamiento en caché y la aceleración de las respuestas

Instrucciones para su Ejecución

Proceso de creación y conexión de las instancias con Terraform y SSH

Para poder establecer una conexión con la máquina virtual en AWS necesitamos descargar el archivo “mainkey.pub”, y pasarla al directorio llamado “.ssh”, eso se hace con los siguientes comandos:

Proceso de instalación y configuración de Nginx

1. Instalar Chef Solo.
2. Crear un repositorio de chef.
    1. chef generate repo chef-repo-name
3. Ingresar y descargar el Cookbook de Nginx desde Chef Supermarket en el directorio de cookbooks.
    1. cd chef-repo-name/
    2. knife supermarket download nginx
    3. cd cookbooks/
    4. tar xvf ../nginx-12.2.5.tar
4. Modificar la receta llamada “default” del cookbook nginx para establecer las configuraciones, este debe de tener las siguiente información.

```
root /var/www/html;
 include /etc/nginx/default.d/*.conf;

# Add index.php to the list if you are using PHP
index index.html index.htm index.nginx-debian.html;

server_name _;

location / {
        autoindex on;
        # First attempt to serve request as file, then
        # as directory, then fall back to displaying a 404.
        try_files $uri $uri/ =404;
}
location /server1/ {
       proxy_pass http://10.0.7.103/;
       proxy_set_header Host $host;
       proxy_set_header X-Real-IP $remote_addr;
       proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
}
location /server2/ {
       proxy_pass http://10.0.5.106/;
       proxy_set_header Host $host;
       proxy_set_header X-Real-IP $remote_addr;
       proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
}
location /server4/ {
       proxy_pass http://3.90.1.144;
       proxy_set_header Host $host;
       proxy_set_header X-Real-IP $remote_addr;
      proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
}
location /server5/ {
       proxy_pass http://10.0.5.165;
       proxy_set_header Host $host;
       proxy_set_header X-Real-IP $remote_addr;
      proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
}
```

1.

5. Configurar `solo.rb`: Crea un archivo `solo.rb` en el directorio raíz. Este archivo contiene la configuración para Chef Solo.

```
current_dir       = File.expand_path(File.dirname(__FILE__))
file_cache_path   "#{current_dir}"
cookbook_path     "#{current_dir}/cookbooks"
role_path         "#{current_dir}/roles"
data_bag_path     "#{current_dir}/data_bags"
```

1.

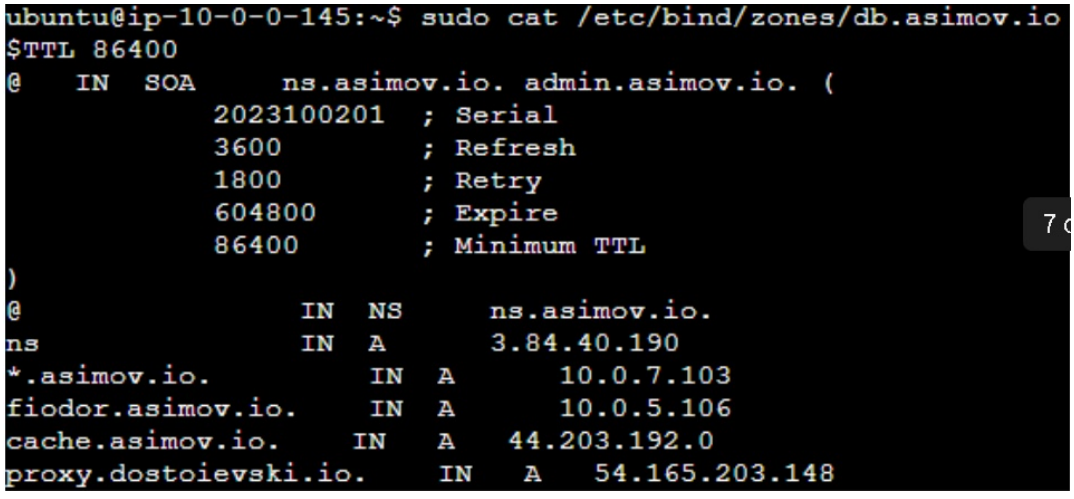
6. Crear un JSON de Configuración: Crea un archivo JSON que especifica qué recetas deseas aplicar.

```
{
  "run_list": [
    "recipe[nginx::default]"
  ]
}
```

1.

7. Ejecutar Chef Solo
    1. ''' chef-solo -c solo.rb -j nginx.json '''

Proceso de instalación y configuración de BIND DNS

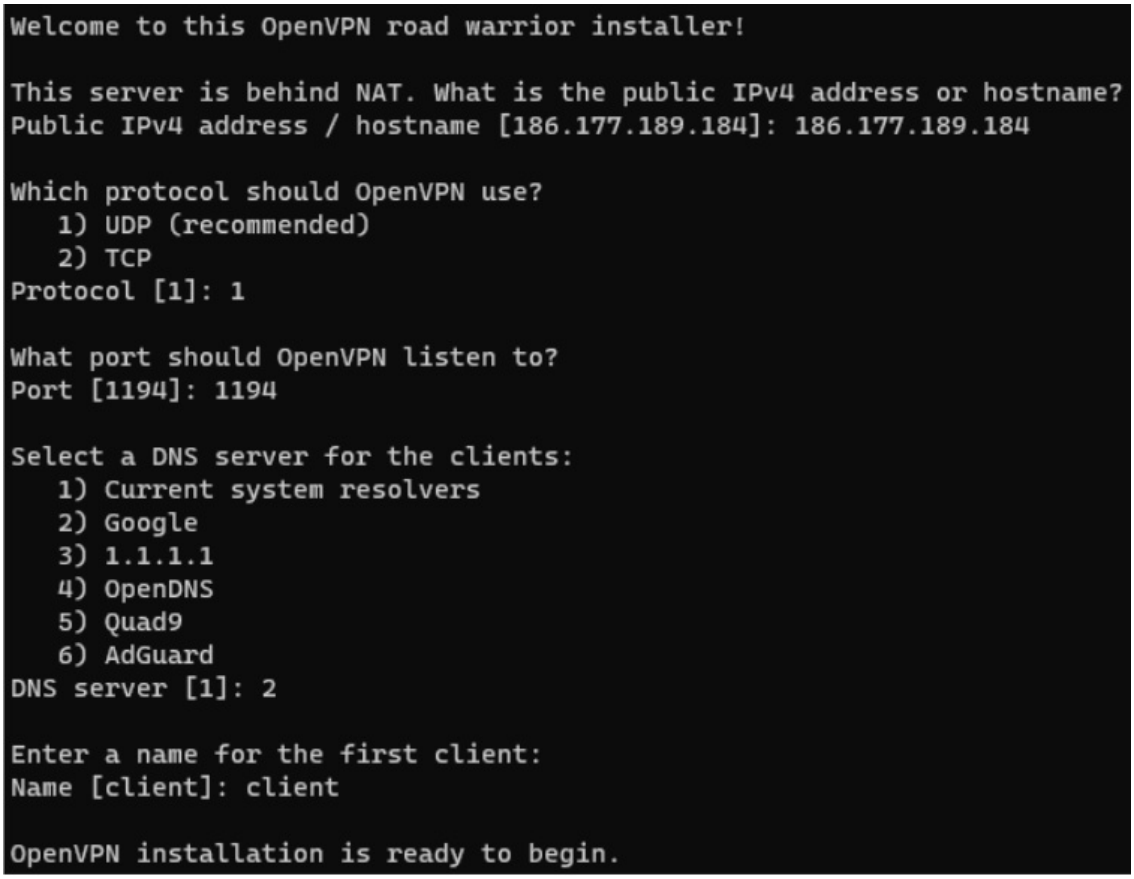
1. Actualizar e Instalar bind9
    1. sudo apt update && sudo apt upgrade -y
    2. sudo apt install bind9 bind9utils bind9-doc -y
2. Configuración en /etc/bind/named.conf.local
    1. sudo nano /etc/bind/named.conf.local
    2. Agregar configuración de zonas DNS específicas para dominios como "asimov.io," "dostoievski.io" y "google.com."
    3. 
```
ubuntu@ip-10-0-0-145:~$ sudo cat /etc/bind/named.conf.local
//
// Do any local configuration here
//

// Consider adding the 1918 zones here, if they are not used in your
// organization
//include "/etc/bind/zones.rfc1918";

zone "asimov.io" {
    type master;
    file "/etc/bind/zones/db.asimov.io";
};

zone "dostoievski.io" {
    type master;
    file "/etc/bind/zones/db.dostoievski.io";
};

zone "google.com" {
    type master;
    file "/etc/bind/zones/db.google.com";
};
```
3. Crear directorio de zonas
    1. sudo mkdir /etc/bind/zones
4. Configuración de las zonas: Editar archivos de zona DNS para configurar registros A y servidores de nombres (NS).
    1. sudo nano /etc/bind/zones/db.asimov.io
    2. 
```
ubuntu@ip-10-0-0-145:~$ sudo cat /etc/bind/zones/db.asimov.io
$TTL 86400
@   IN  SOA     ns.asimov.io. admin.asimov.io. (
              2023100201  ; Serial
              3600        ; Refresh
              1800        ; Retry
              604800      ; Expire
              86400       ; Minimum TTL
)
@               IN  NS      ns.asimov.io.
ns              IN  A       3.84.40.190
*.asimov.io.        IN  A       10.0.7.103
fiodor.asimov.io.   IN  A       10.0.5.106
cache.asimov.io.    IN  A   44.203.192.0
proxy.dostoievski.io.   IN  A   54.165.203.148
```
5. Repetir el mismo proceso para otros archivos de zona (db.dostoievski.io y db.google.com).
6. Configura los forwarders en BIND
    1. sudo nano /etc/bind/named.conf.options

2.
```
ubuntu@ip-10-0-0-145:~$ sudo cat /etc/bind/named.conf.options
options {
        directory "/var/cache/bind";

        // If there is a firewall between you and nameservers you want
        // to talk to, you may need to fix the firewall to allow multiple
        // ports to talk.  See http://www.kb.cert.org/vuls/id/800113

        // If your ISP provided one or more IP addresses for stable
        // nameservers, you probably want to use them as forwarders.
        // Uncomment the following block, and insert the addresses replacing
        // the all-0's placeholder.

         forwarders {
                0.0.0.0;
                8.8.8.8;
                8.8.4.4;
         };

        //========================================================================
        // If BIND logs error messages about the root key being expired,
        // you will need to update your keys.  See https://www.isc.org/bind-keys
        //========================================================================
        dnssec-validation auto;

        listen-on-v6 { any; };
};
```

7. Reinicio y verificación:
    1. sudo systemctl restart bind9
    2. named-checkconf # Verifica la configuración de BIND (no debería mostrar errores).
8. Pruebas locales
    1. dig @localhost www.google.com
    2. dig @localhost *.asimov.io
    3. dig @localhost fiodor.asimov.io
    4. dig @localhost *.dostoievski.io
    5. dig @localhost isaac.dostoievski.io
6.
```
ubuntu@ip-10-0-0-145:~$ dig @localhost www.google.com

; <<>> DiG 9.16.1-Ubuntu <<>> @localhost www.google.com
; (1 server found)
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 27993
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 2, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
; COOKIE: 713c33d3c0da544b0100000065248981436595 8cc126aad0 (good)
;; QUESTION SECTION:
;www.google.com.                        IN      A

;; ANSWER SECTION:
www.google.com.         86400   IN      A       10.0.5.106
www.google.com.         86400   IN      A       10.0.7.103

;; Query time: 0 msec
;; SERVER: 127.0.0.1#53(127.0.0.1)
;; WHEN: Mon Oct 09 23:15:13 UTC 2023
;; MSG SIZE  rcvd: 103
```

Proceso de instalación y configuración de OpenVPN

1. Actualizar paquetes e instalar OpenVPN.
    1. sudo apt-get update && sudo apt-get upgrade -y
    2. wget https://git.io/vpn -O openvpn-install.sh
2. Otorgar permisos de ejecución al script.

    1. chmod +x openvpn-install.sh
3. Ejecutar el script y brindar la información correspondiente para configurar OpenVPN.
    1. sudo ./openvpn-install.sh
4. 

5. Configurar usuarios adicionales según sea necesario.

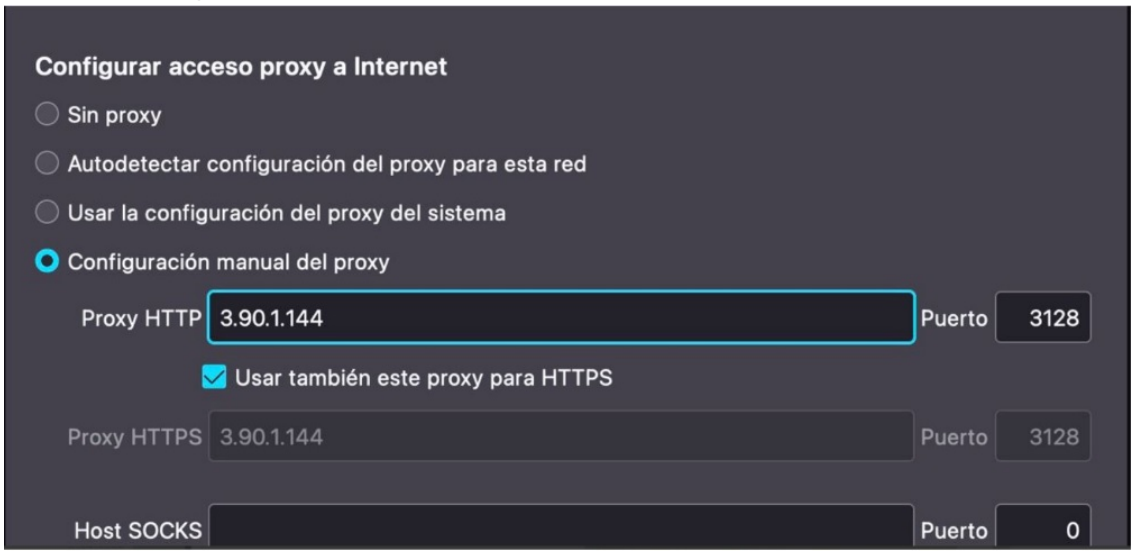
Nota: Durante la ejecución del script, se generará un archivo .ovpn que se utilizará para la conexión VPN.

Proceso de instalación y configuración de apache2 Web Servers

1. Instalar chef en las instancias privadas de EC2.
2. Se descarga el repositorio de apache2 proporcionado por el profesor para el proyecto opcional.
3. Agregar el archivo solo.rb para indicar donde se encuentran los folders de configuración y el node.json para indicar la receta que tiene que correr el runlist. En este caso es la receta de default-site.rb que viene en el repositorio.
4. Se sustituye el mensaje en HTML de Hello World a Apache1 y Apache 2 respectivamente para cada una de las instancias.
5. Se corre el repositorio de chef utilizando el comando chef-solo -c solo.rb -j node.json dentro de la carpeta.

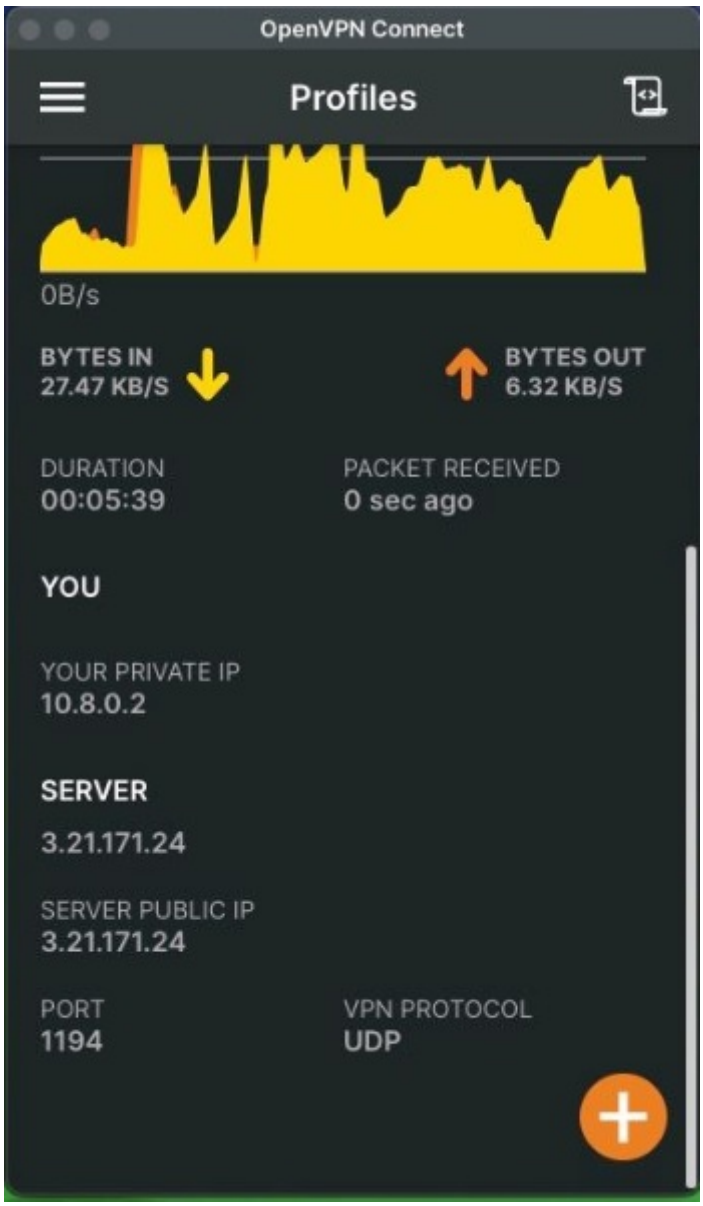
Proceso de instalación y configuración de Squid Web Cache

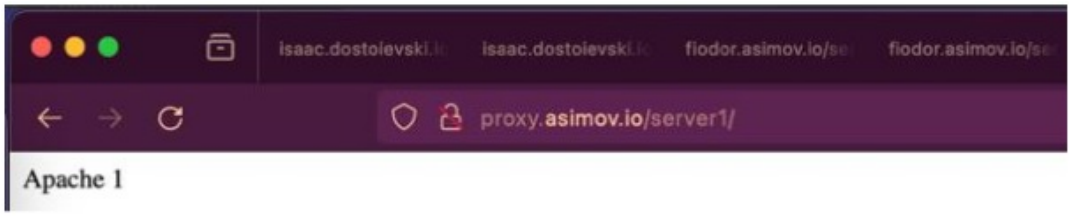
1. Instalar chef en la instancia privada de EC2.
2. Se descarga el repositorio de Squid desde el chef supermarket.
3. Se crea un archivo solo.rb para definir donde estan los folders y uno node.json para correr el runlist. Se utiliza la receta default.rb

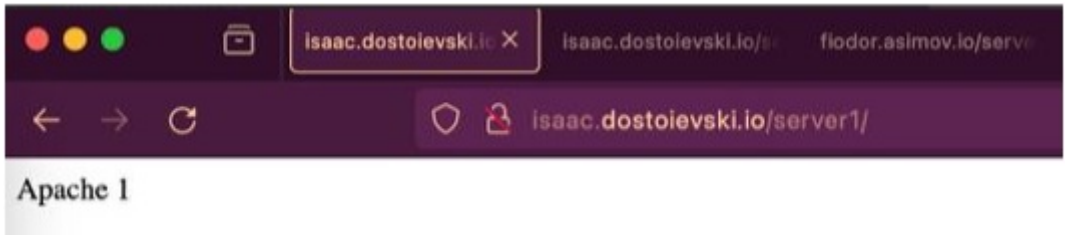
4. Se corre el repositorio de chef utilizando el comando chef-solo -c solo.rb -j node.json dentro de la carpeta.
5. Una vez está el servidor corriendo para probarlo se puede utilizar un navegador web que soporte conexiones proxy y setear el ip de la máquina y el puerto que se configuraron en la instalación de squid.
6. 

Pruebas del Proyecto

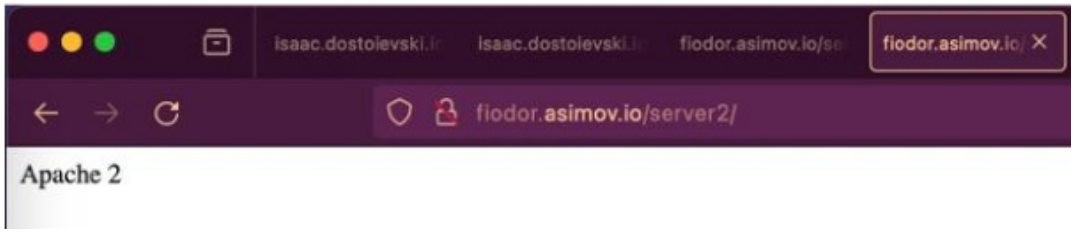
1. Ejecutar la herramienta y probar el funcionamiento del OpenVPN.
    1. Estado: Completado y funcional.

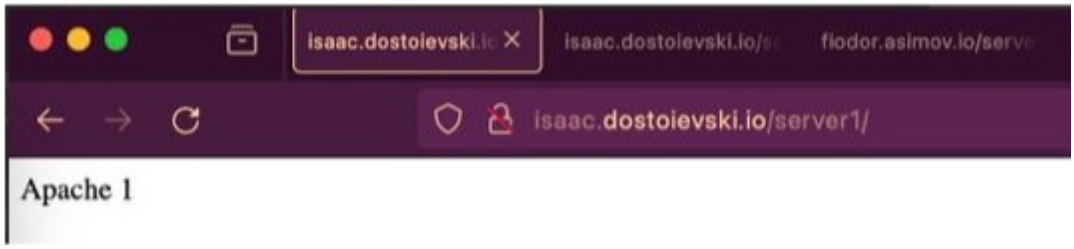
2. 

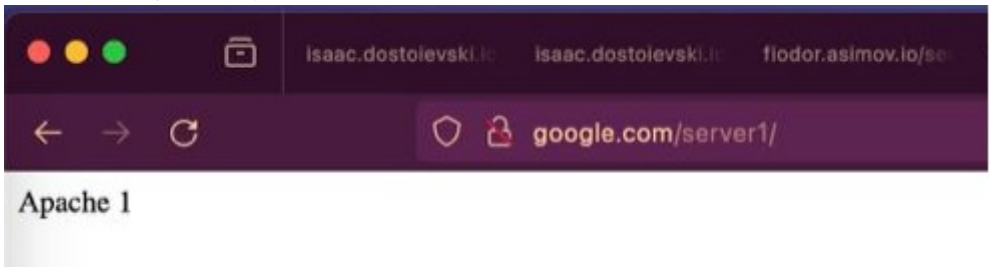
2. Prueba del DNS por medio de la dirección del proxy reverso
    1. Estado: Completado y funcional.
    2. 

3. Prueba del DNS por medio de la resolución directa.
    1. Estado: Completado y funcional.
    2. 

4. Prueba de visita a la zona asimov.io con la entrada fiodo.
    1. Estado: Completado y funcional.

2. 

5. Prueba de visita a la zona dostoievski.io
    1. Estado: Completado y funcional.
    2. 

6. Prueba de visita a la zona google.com
    1. Estado: Completado y funcional.
    2. 

Recomendaciones

1. Para proyectos donde tengamos que desarrollar este tipo de infraestructura en máquinas virtuales que todos los miembros utilizan, deberíamos asignar a un único encargado de levantarla, modificarla y/o probarla. Ya que 3 miembros distintos subieron su instancia de la infraestructura y AWS empezó a no reconocer las solicitudes.
2. Utilizar el sistema operativo linux para mayor facilidad de uso.
3. Redactar y actualizar constantemente las instrucciones de ejecución desde el momento en que se inicia para que así todos los miembros tengan acceso a ellas y puedan consultarlas cuando gusten.
4. Si se hace uso de repositorios en github o similares, manejar el proyecto de manera independiente a estos, para evitar colisiones entre los distintos repositorios.
5. Al crear la infraestructura y terminar de hacer el uso que se tenía planeado se debe destruir para evitar que se acumulen cobros indeseados.
6. Llevar una buena documentación del código.
7. Asesorarse acerca de las estructuras de pago de los distintos proveedores de servicios en la nube para asegurarse de cuál será el monto de la infraestructura que se desea instalar.
8. Almacenar de manera responsable los archivos .pub t .pem donde se guarda información sensible de las credenciales además de siempre tener un respaldo del mismo. Con esto se evita el crear una nueva y verse por la tarea de difundirlo a los demás miembros del equipo.
9. Al implementar un proxy reverso por medio de la herramienta nginx recuerde que este lo debe de configurar con los protocolos de seguridad de SSL y utilizando el puerto 443 para conexiones seguras mediante https.
10. Por último, se recomienda tener en cuenta el concepto de load Balance, ya que al distribuir las cargas de los servidores obtendremos mejores resultados para la red. Si bien por la magnitud de este proyecto no es necesario, es un concepto muy importante a considerar.

Conclusiones

1. Debemos asignar múltiples roles a los miembros del equipo, para mejorar la coordinación entre los miembros a la hora de probar el código y conectarlo con la versión final del mismo.
2. Sería buena idea coordinar reuniones cuando hayan dudas a las que ya les hayamos dado vueltas aunque nos parezcan tontas, ayuda bastante con el tiempo y se entiende bien lo que sucede.
3. Consultar las documentaciones oficiales de las herramientas de las cuales se están haciendo uso.
4. Es importante recordar a la hora de investigar cuáles herramientas se están utilizando en conjunto porque si se enfoca en una sola dejando el resto de lado lo que se encuentre o aprenda puede afectar negativamente o no ser lo que se necesita y causar confusión.
5. La implementación de una red virtual en un entorno de Cloud Provider utilizando herramientas como Terraform, Chef Solo/Puppet/Ansible y NGINX permitió una integración completa de servicios de red, proporcionando una experiencia para el grupo muy interesante ya que se trabajó un flujo de red seguro y con tecnologías de vanguardia.
6. La automatización de la configuración de los servidores es una estrategia sumamente útil para ahorrar tiempo y además en particular a nuestro grupo nos ha funcionado muy bien ya que evitamos errores de descarga mejorando la eficiencia a la hora de desarrollar.
7. La infraestructura de red que se realizó es sumamente enriquecedora ya que muestra cómo por medio de firewalls, Security Groups y VPN se puede optar por un nivel significativo de seguridad, limitando el acceso no autorizado y protegiendo los datos sensibles de la red, lo cual son prácticas sumamente solicitadas en el mundo laboral.
8. Consideramos como grupo que configurar un servidor DNS con zonas personalizadas como las de asimov, google y demás mejoró la gestión del sistema de nombres de dominio y también facilitó la resolución de nombres en la red.
9. Un objetivo sumamente interesante fue el uso de un Web Cache transparente con Squid, ya que este optimizó el ancho de banda gracias a su función de almacenar el tráfico HTTPS, de esta manera logramos reducir la carga en los servidores y mejorar la velocidad de acceso.
10. Por otro lado, la implementación del proxy reverso nos permitió utilizar múltiples servicios bajo el mismo nombre de dominio y puerto, lo cual nos simplifica la gestión de servicios y además es un gran aprendizaje acerca de herramientas para la resolución de direcciones.

Referencias

- Stack Overflow. (2017, 25 de enero). Nginx location 404 Not Found. https://stackoverflow.com/questions/41099318/nginx-location-404-not-found
- Hostinger. (s.f.). How to Set Up and Configure iptables on Linux. https://www.hostinger.com/tutorials/iptables-tutorial
- Terraform Registry. (s.f.). AWS EC2 Client VPN - Terraform Module. https://registry.terraform.io/modules/cloudposse/ec2-client-vpn/aws/latest
- Stack Overflow. (2020, 8 de julio). How to Deploy OpenVPN EC2 Instance via Terraform. https://stackoverflow.com/questions/63004852/how-to-deploy-openvpn-ec2-instance-via-terraform
- Chef Supermarket. (s.f.). OpenVPN Cookbook. https://supermarket.chef.io/cookbooks/openvpn
- Server Fault. (2016, 2 de septiembre). Nginx is giving 404 errors on all but the HTML pages. https://serverfault.com/questions/796930/nginx-is-giving-404-errors-on-all-but-the-html-pages

- Terraform by HashiCorp. (s.f.). Resource: aws_nat_gateway. Terraform Registry. https://registry.terraform.io/providers/hashicorp/aws/latest/docs/resources/nat_gateway
- OpenAI. (2016, 9 de octubre). ChatGPT [Modelo de lenguaje]. https://chat.openai.com/
- Stack Overflow. (2012, 5 de junio). Squid Proxy - Howto allow TCP Connect, getting TCP_DENIAL/400 with ERR_INVALID. https://stackoverflow.com/questions/10895711/squid-proxy-howto-allow-tcp-connect-getting-tcp-denial-400-with-err-invalid