TRABAJO DE GRADO
Opción Seminario-Diplomado.

Configuración de servicios de AWS (Amazon Web Services) para desplegar aplicaciones web

Corporación Universitaria Remington.
Nombre de la facultad: Ingeniería
Nombre del programa académico: Ingeniería de sistemas

Nombres de los estudiantes autores del trabajo de grado.
Jolausse Samir Alvarez Peña
Julian David Oñate Bolivar
Darguin Barbosa Lopez

Nombre del Tutor
Juan Pablo Berrio Lopez

Seminario AWS.
2025

Tabla de Contenidos

Resumen

En este documento se explica los diferentes conceptos abordados hacia la computación en la nube, las bases para conocer la evolución, el tecnicismo que involucra a los elementos contextuales necesarios para lograr un entendimiento básico de la implementación de este tipo de tecnologías.

Por otro lado, se explican los diferentes procesos de configuración de la consola de AWS (Amazon Web Services) con el fin de entender y poner en practica los diferentes conocimientos adquiridos en el seminario.

Palabras clave

Balanceador de carga (Elastic Load Balancing en AWS)

El load balancing en AWS es una forma de repartir el tráfico de red entre varios servidores o instancias (como las de EC2) para que ninguna se sobrecargue y la aplicación se mantenga disponible.

Auto-scaling

El auto Auto-scaling también conocido como (Amazon EC2 Auto Scaling) ayuda a gestionar la carga de la aplicación mediante la configuración de un número mínimo y máximo de instancias (de EC2), que se irán creando conforme el aplicativo disminuya o aumente su tráfico.

Instancias (Amazon EC2)

Una instancia (Amazon EC2) es una maquina virtual que se puede ejecutar desde la nube, esta se basa en una copia de una AMI (Amazon Machine Images), que se puede configurar y permite compartir y escalar recursos.

Virtual private clouds (VPC)

Un VPC es una red virtual privada similar a una red tradicional, donde se pueden crear subredes e instancias dependiendo la región y la configuración que se determine.

Docker

Es una plataforma de código abierto que nos permite compilar nuestro código fuente convirtiéndolo en imágenes que luego serán subidas ya accedidas desde la nube.

Marco conceptual y contextual

1. Marco conceptual

En el presente documento se abordan diversos conceptos fundamentales de la computación en la nube, que se entiende como acceso remoto a recursos informáticos a través de internet, los cuales fueron trabajados a lo largo del seminario, cuyos conceptos destacan: arquitectura cloud en AWS, infraestructura elástica, alta disponibilidad, escalabilidad, contenerización, redes virtuales, balanceo de carga, automatización de recursos y maquinas virtuales en la nube, que nos permite entender y realizar los procesos de configuración y prueba funcional de los servicios de AWS.

2. Marco contextual

Durante el desarrollo del seminario se exploraron servicios dentro de la consola de AWS, teniendo un enfoque en la configuración y utilización de VPC que permitió crear una red privada dentro de la infraestructura de AWS, lo que facilitó el lanzamiento de instancias y generó varias subredes con IP públicas y privadas para separar los recursos que necesitan ser accedidos de internet y los que no. Sobre esta infraestructura se utilizó el servicio de Amazon EC2 (Elastic Compute Cloud) que permite crear máquinas virtuales, utilizando Docker se crean contenedores para alojar la aplicación y exponer estos puertos a internet con la ayuda Apache y Nginx que son servidores web, además de configurar un gestor de carga (Auto-scaling) que ayudara a que la aplicación no se sature y ajustara automáticamente la cantidad de instancias según el tráfico junto con el load balancing (Balanceador de carga), para la distribución de ese trafico en las instancias creadas anteriormente para garantizar la alta disponibilidad

Desarrollo e implementación del aprendizaje

Computación en la nube

1.1. Historia de la computación en la nube y despliegues

Origen de la virtualización

La virtualización se entiende como un software para generar una capa de abstracción sobre el hardware del sistema abriendo la posibilidad de interactuar con los elementos de nuestra maquina (procesadores, memoria, almacenamiento, etc.), permitiendo simular maquinas virtualmente dentro de una maquina física, siendo esto todo un avance a lo largo de los años que ha tenido impacto en la infraestructura TI.

Esta tecnología tiene sus orígenes a finales de la década de los 60s e inicios de la década de los 70s, pues antes de que se conociera como tal la virtualización existía una problemática para los ingenieros o administradores que se enfocaban en operaciones de TI, ya que los equipos que usaban aparte de tener limitaciones en su hardware y software generaban costos muy altos y tenían tiempos de respuesta extensos hasta que en el año de 1964 se introdujo el primer sistema MAINFRAME CP-40 creado por IBM, cuyas características permitían a varios usuarios ejecutar procesos simultáneamente e independientemente además de implementar el primer HYPER VISOR lo que permitía correr varios sistemas operativos virtualizados en un único MAINFRAME, asentando las bases para lo que se conocería más adelante como servidores y cloud Computing.

Origen de la informática en la nube: los primeros esfuerzos

Durante los años 50, surgieron las primeras redes de computadoras, como el proyecto SAGE del gobierno de Estados Unidos, que se convirtió en uno de los primeros sistemas automatizados de defensa aérea que empleaba múltiples computadoras vinculadas en red para seguir el rastro de objetos voladores no detectados.

Después, en 1961, el Sistema Compartido de Computación del Instituto de Tecnología de Massachusetts aportó más a este concepto. Este sistema posibilitó que varios usuarios ingresaran a un solo ordenador central mediante terminales remotas y emplearan el mismo sistema operativo y las mismas aplicaciones.

A pesar de que no existe nadie que pueda ser reconocido como "padre/madre de la computación en la nube", hay varios científicos e ingenieros que aportaron al desarrollo del concepto de computación en la nube en los años 60. Algunos de los que incluyeron:

John McCarthy, un matemático y científico de la computación, fue el promotor del concepto de "Inteligencia Artificial" en la conferencia de Dartmouth en 1956, y en la conferencia del MIT en 1961, sugirió la noción de un "ordenador central" compartido por múltiples usuarios mediante terminales remotas.

Línea del Tiempo: Orígenes de la Computación en la Nube

Década de 1950

- 1955-1956: Se crea el concepto de "tiempo compartido" en computación, por la posibilidad de que múltiples usuarios utilicen una misma computadora para las operaciones de TI que se requieran simultáneamente.

Década de 1960

- 1969: Creación de ARPANET, la predecesora de lo que se conocería más adelante como Internet, permitiendo conectar las primeras computadoras vía remota.

Década de 1970

- 1970-1973: Creación de las primeras redes virtuales (VPN) y técnicas de virtualización.
- 1972: IBM desarrolla el Sistema de Máquina Virtual (VM), permitiendo que múltiples sistemas operativos corran en una misma máquina física.

Década de 1980

- 1983: ARPANET adopta TCP/IP, estableciendo el estándar para la comunicación de datos en redes.

Década de 1990

- 1991-1993: Se comercializa el Internet y se desarrolla la World Wide Web, facilitando el acceso global a servicios en red.
- 1993-1994: Desarrollo de tecnologías de navegación web graficas como Mosaic y Netscape.
- 1995: Aparición de las primeras tecnologías de virtualización moderna por parte de empresas como VMware.
- 1997: Ramnath Chellappa acuña el término "Cloud Computing" por primera vez en una conferencia académica.

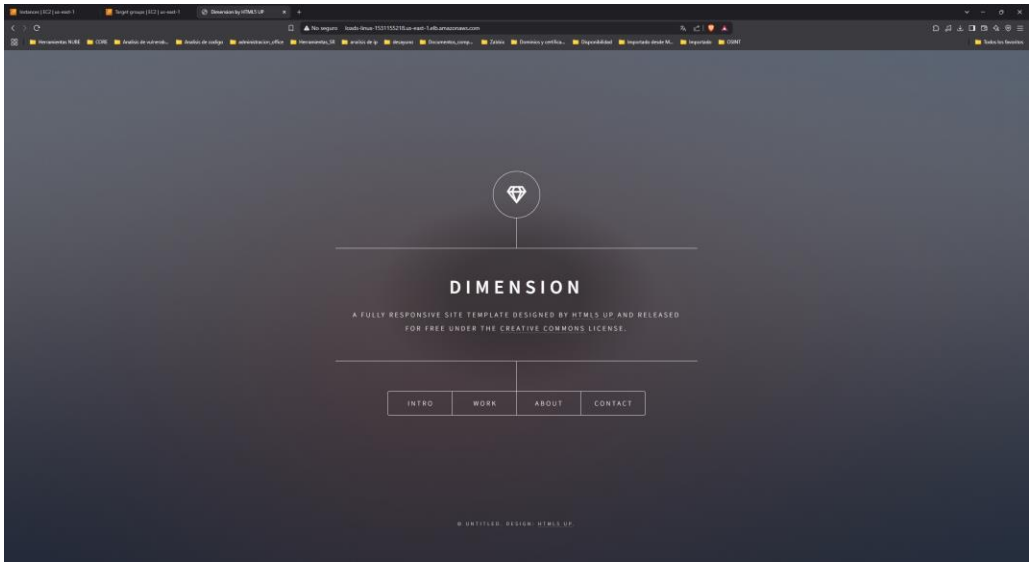
- 1999: Lanzamiento de Salesforce.com, primer servicio empresarial importante que demuestra que el software puede entregarse a través de Internet.

Década de 2000 (Nacimiento formal del Cloud Computing)

- 2002: Amazon Web Services (AWS) lanza sus primeros servicios web, creando la infraestructura inicial para futuras ofertas en la nube.
- 2006: Amazon lanza EC2 (Elastic Compute Cloud), permitiendo a empresas y desarrolladores alquilar computadoras virtuales para ejecutar sus aplicaciones.
- 2006: Amazon lanza S3 (Simple Storage Service), ofreciendo almacenamiento escalable en la nube.
- 2008: Lanzamiento de Eucalyptus, primera plataforma de código abierto compatible con AWS para desplegar infraestructuras de nube privada.
- 2008: Surge OpenNebula, uno de los primeros softwares para implementar nubes privadas e híbridas.

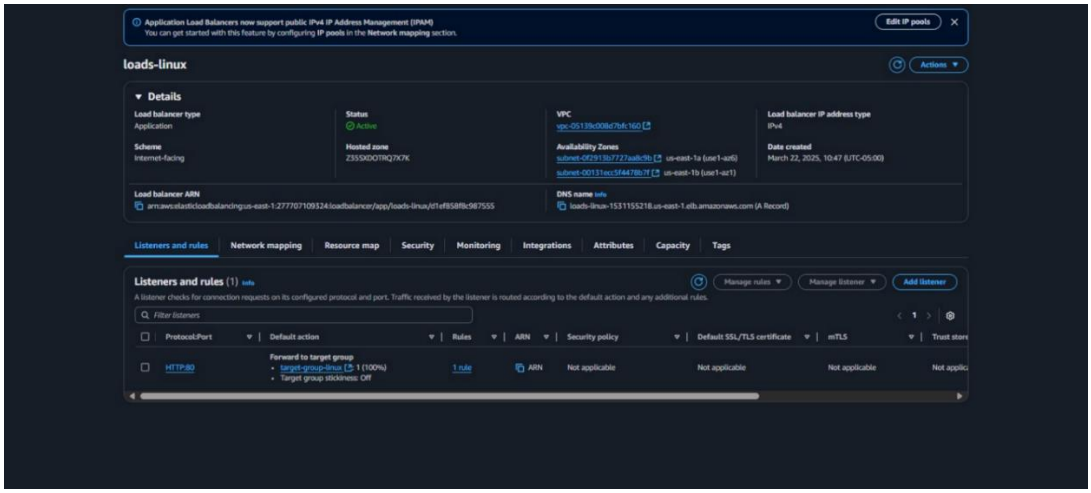
Implemente un balanceador de carga con un autoscaling, las instancias deben ser Linux, debe tener una aplicación web básica que cargue automáticamente al crearse la instancia. Se debe mostrar cómo funciona el autoscaling al terminar instancias manualmente.

Funcionamiento de instancias



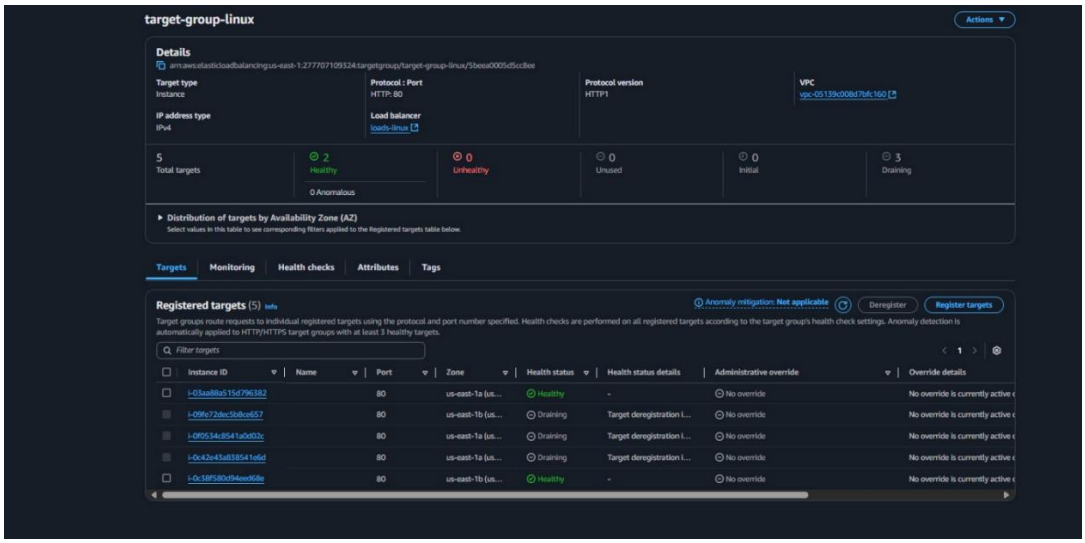
Se muestra la aplicación la cual esta corriendo en las instancias con el DNS público del balanceador.

Creación Load Balancer



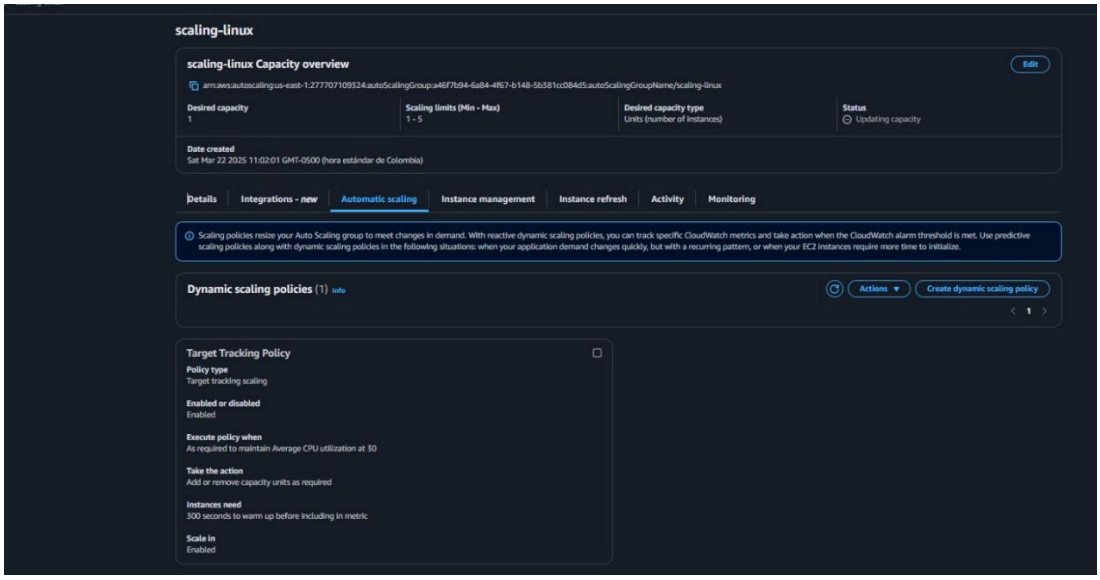
Se muestra la creación del load balancer el cual se va a encargar de distribuir el trafico de red para que no se afecten los recursos de las instancias.

Creación del Target groups

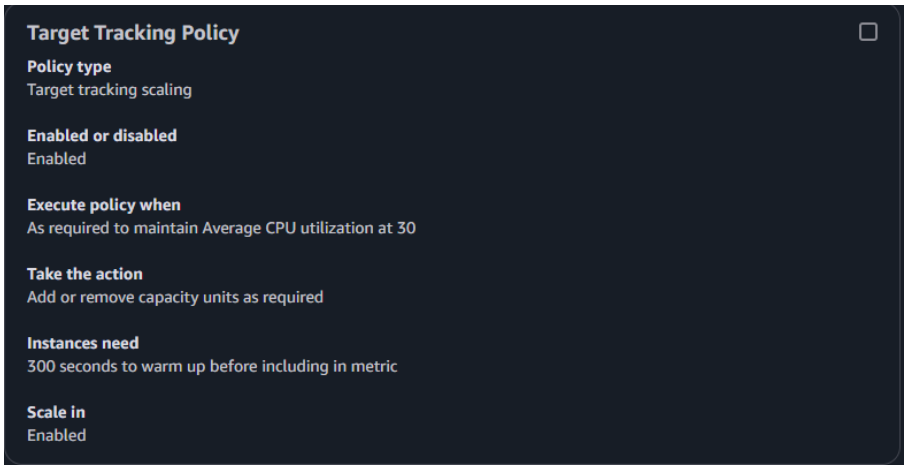


Se muestra las instancias en las cuales se van a distribuir el tráfico mediante el balanceador.

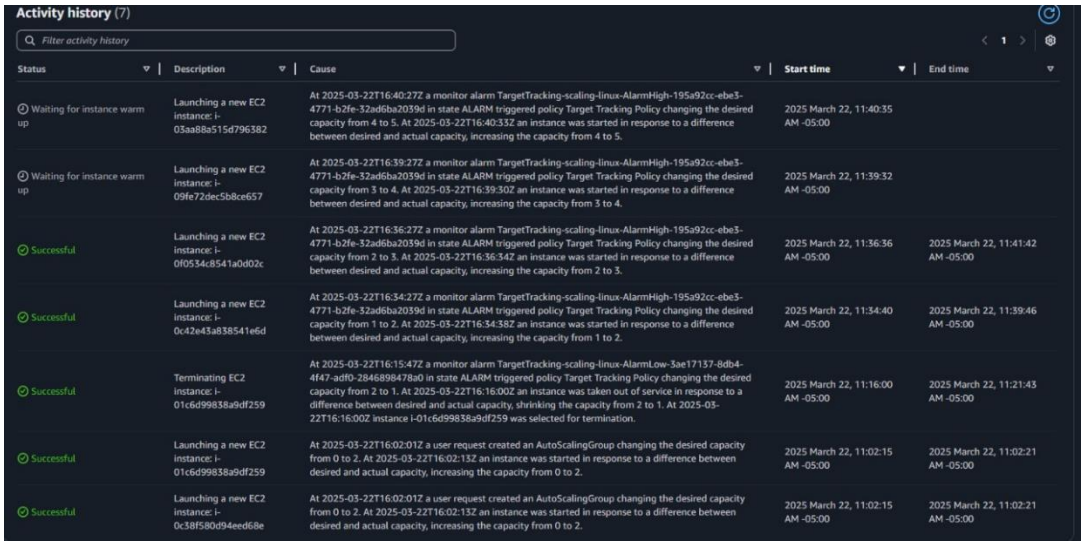
Creación de política



En esta parte nos muestra como se creó la política del auto scaling donde se configura que, cuando llegue al 30% de la cpu en una de las instancias que se encuentren el en el target groups cree mas instancias para regular los recursos de la aplicación el cual sería un máximo de 5 instancias a crear.



Prueba de funcionamiento del auto scaling



Acá podemos evidenciar como el auto scaling empieza a funcionar al momento que la cpu llega a su rango configurado que es del 30%, en las imágenes se muestra que la cpu de una de las instancias llega al 97% y crea instancias adicionales mediante va el crecimiento proporcional de la cpu.

Terminación de Instancias.

Successful	Terminating EC2 instance: i-03aa88a515d796382	At 2025-03-22T17:03:36Z a monitor alarm TargetTracking-scaling-linux-AlarmLow-3ae17137-8db4-4f47-adf0-2846898478a0 in state ALARM triggered policy Target Tracking Policy changing the desired capacity from 2 to 1. At 2025-03-22T17:03:46Z an instance was taken out of service in response to a difference between desired and actual capacity, shrinking the capacity from 2 to 1. At 2025-03-22T17:03:46Z instance i-03aa88a515d796382 was selected for termination.	2025 March 22, 12:03:46 PM -05:00	2025 March 22, 12:10:08 PM -05:00
Successful	Terminating EC2 instance: i-09fe72dec5b8ce657	At 2025-03-22T17:02:36Z a monitor alarm TargetTracking-scaling-linux-AlarmLow-3ae17137-8db4-4f47-adf0-2846898478a0 in state ALARM triggered policy Target Tracking Policy changing the desired capacity from 3 to 2. At 2025-03-22T17:02:43Z an instance was taken out of service in response to a difference between desired and actual capacity, shrinking the capacity from 3 to 2. At 2025-03-22T17:02:43Z instance i-09fe72dec5b8ce657 was selected for termination.	2025 March 22, 12:02:43 PM -05:00	2025 March 22, 12:08:46 PM -05:00
Successful	Terminating EC2 instance: i-0f0534c8541a0d02c	At 2025-03-22T17:01:47Z a monitor alarm TargetTracking-scaling-linux-AlarmLow-3ae17137-8db4-4f47-adf0-2846898478a0 in state ALARM triggered policy Target Tracking Policy changing the desired capacity from 4 to 3. At 2025-03-22T17:01:49Z an instance was taken out of service in response to a difference between desired and actual capacity, shrinking the capacity from 4 to 3. At 2025-03-22T17:01:50Z instance i-0f0534c8541a0d02c was selected for termination.	2025 March 22, 12:01:50 PM -05:00	2025 March 22, 12:07:52 PM -05:00
Successful	Terminating EC2 instance: i-0c42e43a838541e6d	At 2025-03-22T17:00:28Z a monitor alarm TargetTracking-scaling-linux-AlarmLow-3ae17137-8db4-4f47-adf0-2846898478a0 in state ALARM triggered policy Target Tracking Policy changing the desired capacity from 5 to 4. At 2025-03-22T17:00:35Z an instance was taken out of service in response to a difference between desired and actual capacity, shrinking the capacity from 5 to 4. At 2025-03-22T17:00:36Z instance i-0c42e43a838541e6d was selected for termination.	2025 March 22, 12:00:36 PM -05:00	2025 March 22, 12:05:59 PM -05:00

Acá mostramos que cuando la CPU de la instancia se regula a su normalidad va eliminando las instancias que se creo para regular el funcionamiento de la aplicación.

Figuras y tablas

Tabla 1. Comparación entre los diferentes servicios de AWS, Azure y Google

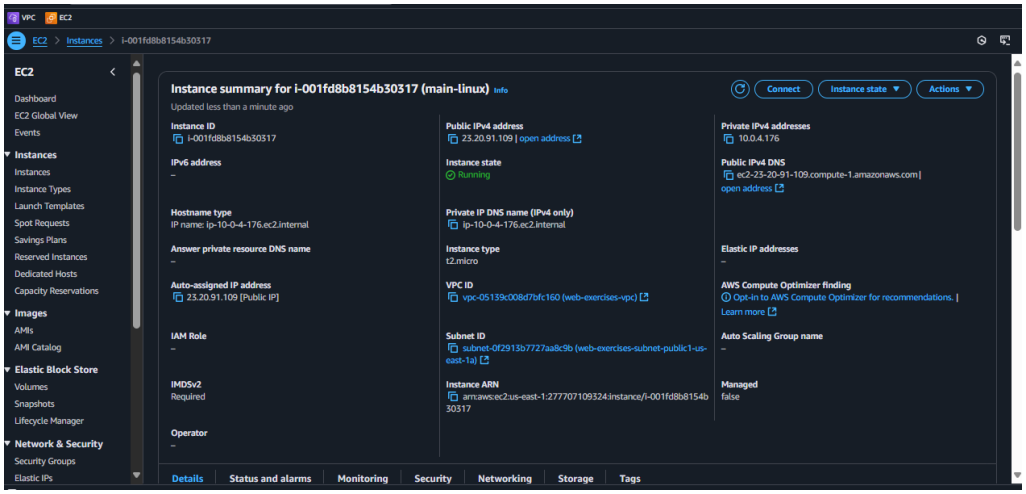
Servicios	AWS	Azure	GCP
Computación			
IaaS - Servidores Virtuales	Elastic Compute Cloud (EC2)	Azure Virtual Machines	Compute Engine
PaaS (Platform-as-a-Service)	Elastic Beanstalk	App Service, Cloud Services	App Engine Standard Environment, App Engine Flexible Environment
Servidores Virtuales Privados	Lightsail	Virtual Machine Images	-
Contenedores Gestionados	EC2 Container Service (ECS)	-	-
Kubernetes Gestionado	Elastic Kubernetes Service (EKS)	Azure Kubernetes Service (AKS)	Kubernetes Engine
Registro de Contenedores	EC2 Container Registry (ECR)	Azure Container Registry	Container Registry
Contenedores Serverless	Fargate	Container Instances	-
Orquestación de Microservicios	Service Fabric	App Engine	-
Serverless	Lambda	Functions	Cloud Functions
Computación por Lotes	AWS Batch	Azure Batch	-
Escalado Automático	AWS Auto Scaling	Virtual Machine Scale Sets, App Service Scale Capability (PAAS), AutoScaling	Instance Groups
Infraestructura Local	AWS Outposts	-	GKE On-Prem
Almacenamiento			
Almacenamiento de Objetos	Simple Storage Services (S3)	Blob Storage	Google Cloud Storage

Servicios	AWS	Azure	GCP
Almacenamiento de Archivado	S3 Infrequent Access (IA) Glacier	Storage (Cool) Storage (Archive)	Nearline Coldline
Disco para instancias	Elastic Block Store (EBS)	Disk Storage	Persistent Disk
Almacenamiento de ficheros	Elastic Block Store (EBS)	Disk Storage	Cloud Filestore
Transferir grandes cantidades de datos	AWS DataSync Snowball Snowmobile	Azure Data Box Import/Export	Storage Transfer Service
Backup	Glacier Storage Gateway	Azure Backup	Coldline
Almacenamiento Híbrido	Storage Gateway	StorSimple	-
Disaster Recovery	Site Recovery	-	-
Bases de Datos			
BD Relacionales	RDS, Amazon Aurora	SQL Database	Cloud SQL, Cloud Spanner
NoSQL - Almacenamiento de ocumentos	DynamoDB	Cosmos DB	Cloud Datastore, Bigtable
NoSQL - Clave-Valor	DynamoDB, SimpleDB	Table Storage	Cloud Datastore
Almacenamiento en caché	ElastiCache	Azure Redis Cache	Cloud Memorystore
Migración de BD	Database Migration Service	Azure Database Migration Service	-
Data Warehouse	Redshift	SQL Data Warehouse	BigQuery
Bases de Datos en Grafos	Neptune	Cosmos DB	-
Redes y Conectividad			

Servicios	AWS	Azure	GCP
Entornos virtuales de red aislados	Virtual Private Cloud	Virtual Network	Virtual Private Cloud
Conexión con entornos on-prem	AWS Managed VPN	VPN Gateway	Cloud VPN
DNS Administrado	Route 53	Azure DNS	Google Cloud DNS
Redirección del tráfico entrante para mejorar el rendimiento y la fiabilidad		Traffic Manager	
CDN	CloudFront	Content Delivery Network	Cloud CDN
Conexión Dedicada	Direct Connect	ExpressRoute	Cloud Interconnect
Balanceo de Carga	Elastic Load Balancing	Load Balancer	Cloud Load Balancing
Gestión y Control de la Cloud			
Asesoramiento Cloud	Trusted Advisor	Azure Advisor	Cloud Platform Security
Orquestación	OpsWorks, CloudFormation	Azure Automation, Resource Manager	Cloud Deployment Manager
Monitorización	CloudWatch, X-Ray	Azure Portal,Azure Monitor, Application Insights	Stackdriver Monitoring, Cloud Shell, Debugger, Trace, Error Reporting
Facturación	Usage and Billing	Azure Billing API	Cloud Billing
Administración	Systems Manager, Health Dashboard	Log Analytics, OMS, Resource Health	Cloud Console
Seguimiento de la actividad de los usuarios y el uso de las APIs	CloudTrail	Log Analytics, Audit Logging	Cloud Audit Logging

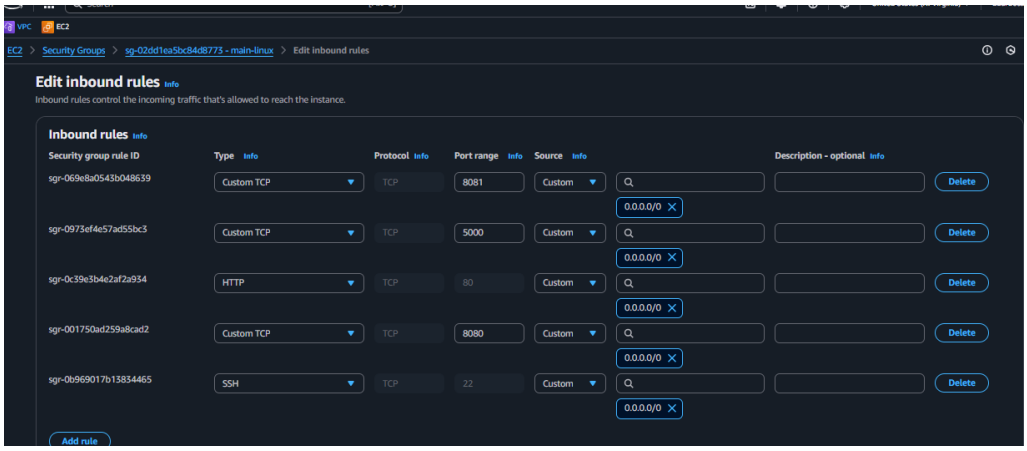
1.2. Creación de contenedores con Docker y configuración Nginx

Creación de instancia de AWS



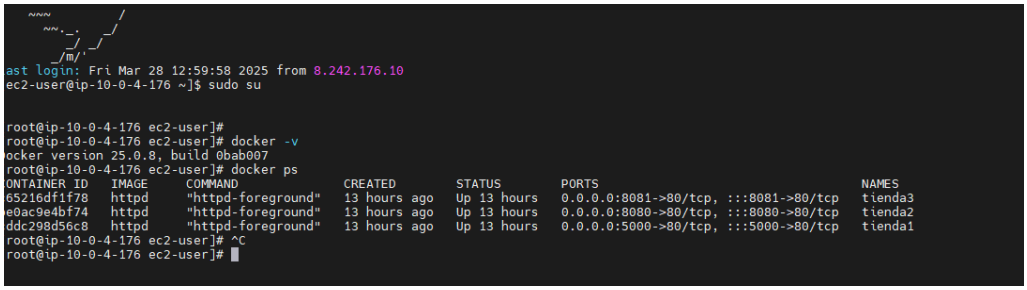
Se utiliza el servicio de Amazon EC2 para crear una instancia basada en Linux, la cual tiene una IP publica que nos deja acceder desde internet, gracias a esto nos va a permitir conectarnos ya sea por SSH y por otro protocolo que nos permita configurar nuestro servidor.

Evidencia de reglas



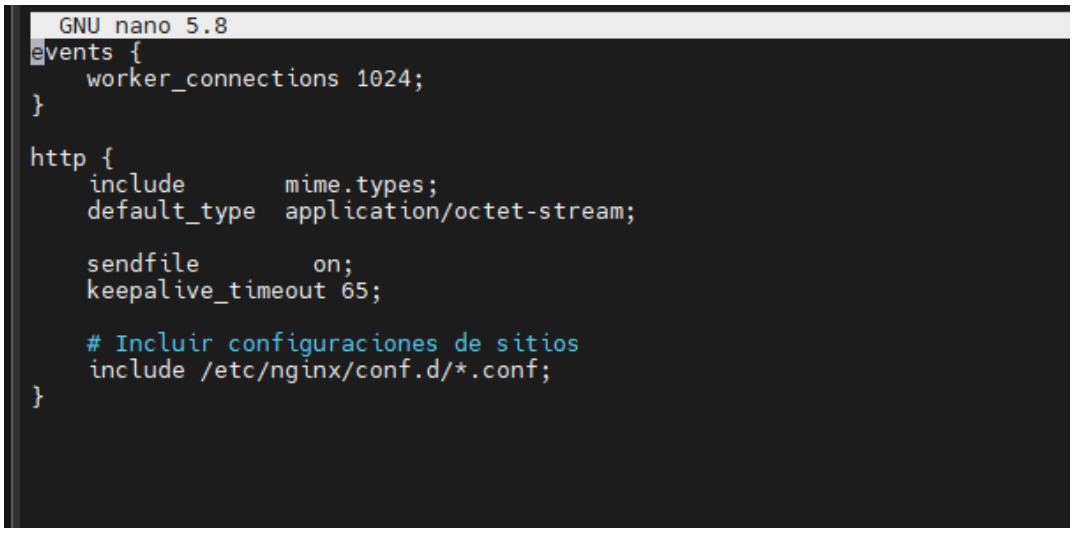
Se agregan reglas al security groups, para permitir el acceso desde cualquier IP a través de los puertos 8081, 5000, 80, 8080 y 22, como se evidencia en la imagen

Creación de contenedores



Se evidencia la creación de contenedores en la instancia de Linux, estos fueron creados a través del siguiente comando “docker run -dit –name tienda1 -d –resart always -p 5000:80 -v /home/ec2-user/tienda1/ : /usr/local/apache2/htdocs/ httpd”

Configuración de Nginx



El archivo “nginx.conf” el cual se encuentra en la ruta “/etc/nginx/” realiza una búsqueda de la carpeta “conf.d” donde se encuentran las configuraciones que nos

permitirán asignarle un DNS al enlace de la aplicación y redirigirá hacia el contenedor correspondiente

Configuración de aplicaciones

```
[root@ip-10-0-4-176 conf.d]# cat app1.conf
server {
    listen 80;
    server_name www.app1.com;

    location / {
        proxy_pass http://23.20.91.109:5000; # Redirige a la aplicación en el contenedor app1
        proxy_set_header Host $host;
        proxy_set_header X-Real-IP $remote_addr;
        proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
        proxy_set_header X-Forwarded-Proto $scheme;
    }
}
[root@ip-10-0-4-176 conf.d]#
```

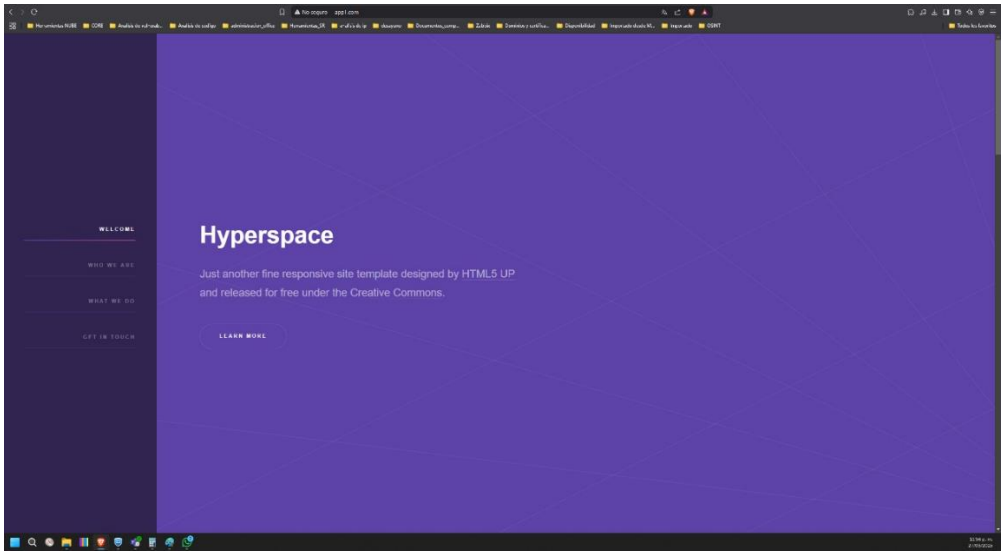
En los archivos de configuración de las aplicaciones en Nginx se configura el proxy reverso para que escuche desde el puerto 80, este nos permitirá especificar el dominio de la aplicación, por ejemplo, en este caso se especifica el dominio como www.app1.com si el usuario accede a este, va a redirigir todas las solicitudes hacia el contenedor http://23.20.91.109:5000

Configuración de Host maquina personal

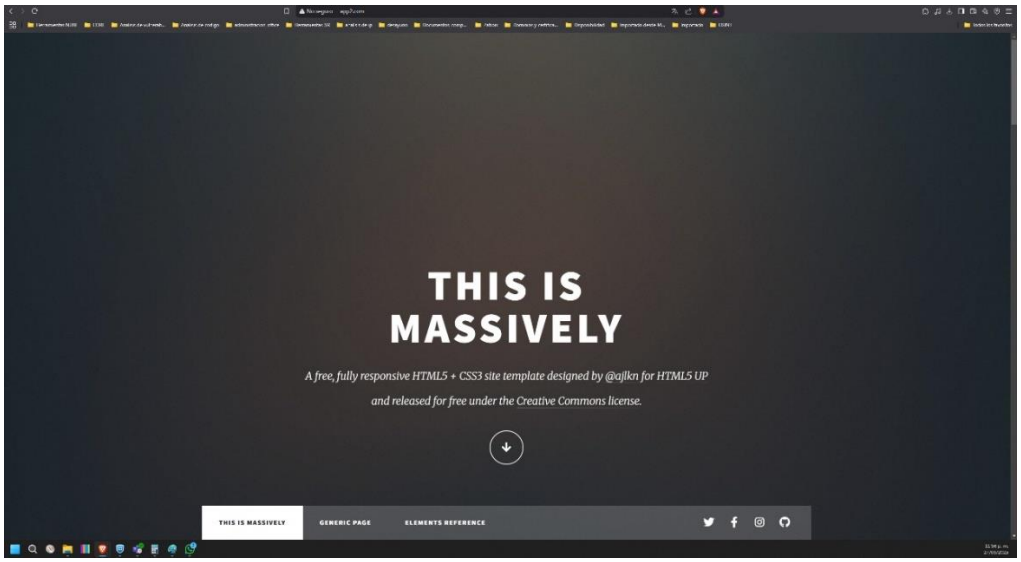
```
#       127.0.0.1       localhost
#       ::1             localhost
127.0.0.1 resilink.local
23.20.91.109 www.app1.com
23.20.91.109 www.app2.com
23.20.91.109 www.app3.com
# Added by Docker Desktop
192.168.13.34 host.docker.internal
```

Asociamos la IP publica de la instancia que creamos en EC2 con los DNS asignados a las aplicaciones alojadas en los contenedores, para que podamos ingresar a las mismas sin tener un DNS público, a este proceso se le llama “mapeo estático de DNS”

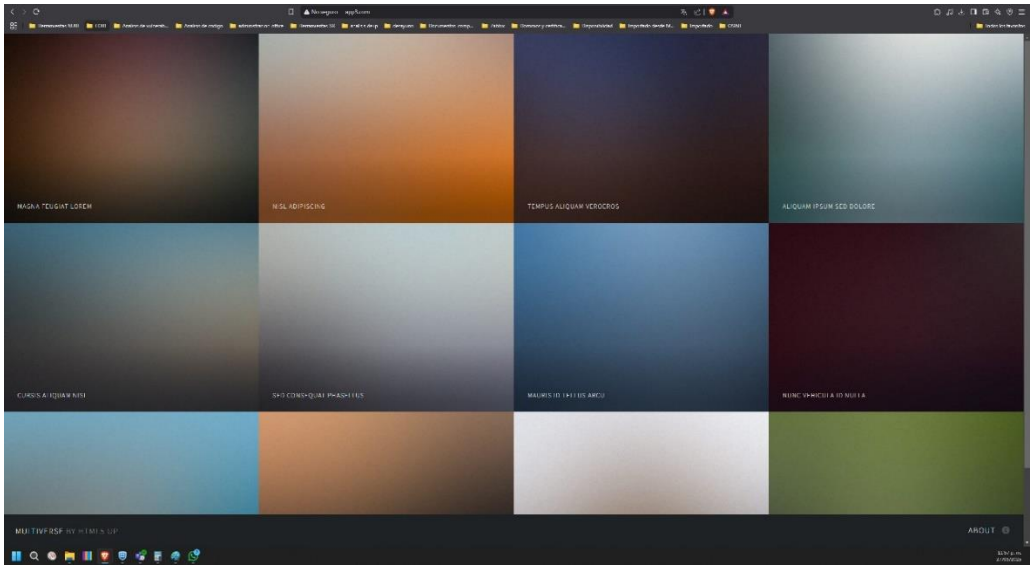
Funcionamiento



Accedemos a la app1



Accedemos a la app2



Accedemos a la app 3

Conclusiones

Evindeciamos la utilidad y el funcionamiento de la consola AWS y parte de los servicios que provee, dandonos a conocer la evolución, accesibilidad y varias posibilidades que se pueden aprovechar con este tipo de tecnologias, ademas de darnos un acercamiento tecnico – practico, que nos permite administrar recursos de AWS en nuestas aplicaciones, esto nos sirve como experiencia en el mundo de la computación en la nube

Referencias

Amazon Web Services. (s.f.). Amazon EC2. AWS Documentation. https://docs.aws.amazon.com/ec2/

Amazon Web Services. (s.f.). Amazon EC2 Auto Scaling. AWS Documentation. https://docs.aws.amazon.com/autoscaling/ec2/

Amazon Web Services. (s.f.). Amazon VPC. AWS Documentation. https://docs.aws.amazon.com/vpc/

Amazon Web Services. (s.f.). Elastic Load Balancing. AWS Documentation. https://docs.aws.amazon.com/elasticloadbalancing/

Docker Inc. (s.f.). Docker Documentation. https://docs.docker.com/

Amazon Web Services. (2014). EC2 Instance History. AWS News Blog. https://aws.amazon.com/blogs/aws/ec2-instance-history/

Arcitura Patterns. (s.f.). A brief history of cloud computing. https://patterns.arcitura.com/cloud-computing-patterns/basics/origins-and-influences/a_brief_history

Boulton, C. (2008). Eucalyptus: Open source for cloud computing. https://www.slideshare.net/clibou/eucalyptus-open-source-for-cloud-computing-presentation

Corbató, F. J., & Vyssotsky, V. A. (1965). Introduction and overview of the Multics system. Massachusetts Institute of Technology. https://people.csail.mit.edu/saltzer/Multics/CTSS-Documents/CTSS_50th_anniversary_web_03.pdf

Emory University. (s.f.). Ramnath K. Chellappa. Goizueta Business School. https://goizueta.emory.edu/faculty/profiles/ramnath-k-chellappa

IBM. (s.f.). VM History and Heritage References. https://www.vm.ibm.com/history/

IBM. (2015). 45 (Official) Years of Virtualization. https://www.vm.ibm.com/history/50th/45-virt.pdf

Muy Tecnológicos. (s.f.). ARPANET – Qué es, origen e historia. https://muytecnologicos.com/diccionario-tecnologico/arpanet

OpenNebula. (s.f.). History. https://archives.opennebula.org/about%3Ahistory

Salesforce. (s.f.). The history of Salesforce. https://www.salesforce.com/news/stories/the-history-of-salesforce/