# Apuntes 5: Viernes 3 de noviembre

Diego Granados Retana 2022158363

Bases de datos II

3 de noviembre, 2023

**Apuntes de la clase:**

Seguridad en base de datos en cloud.

Configuración de bases de datos en un cloud provider.

AWS: Al igual que hacemos una red on premise, vamos a crear una cuenta de AWS. Esto es básicamente cobrar como un espacio virtual donde podemos poner cosas. Ahí ponemos un VPC, que es un Virtual PRivate Cloud. Esto es como si estuviéramos generando o instalando un switch, que tiene varios puertos, que es el dispositivo que ponen los ISPs en la casa. Hay que ponerle diferentes cosas para poder trabajar. Aquí ponemos redes. Al igual que on premise, podemos distinguir entre 3 tipos de redes. Amazon, igual que GCP y Azure, dice que se pueden manejar redes públicas, redes privadas y otra especial para bases de datos, que es un Database Subnet Group. Esto no implica que no podamos instalar una DB en una red privada o pública, pero Amazon tiene su tipo de subred que tiene ventajas para hacer análisis de tráfico y ataques en la base de datos.

Una vez que definimos las redes donde van a ir cada uno de los componentes, tenemos que tener varios componentes. Ahí tenemos un Internet Gateway, que es básicamente una salida hacia Internet. Esto es equivalente a contratar un servicio de Kolbi y conectarlo al gateway. Esto da acceso de entrada y salida. Desde internet una persona puede entrar por el gateway y entrar a la red, y desde la red se puede salir a Internet.

Vamos a tener un NAT gateway: Da salida hacia internet desde las redes privadas. Con un NAT gateway tenemos salida hacia internet, pero no tenemos entrada a Internet desde la red. Es equivalente a lo de tener una red privada con acceso a Internet de salida o sin ningún tipo de acceso. En AWS, si queremos crear una red con salida a internet, lo metemos al Internet Gateway y el NAT se conecta con el Internet Gateway.

La interacción entre las redes y las subredes se da entre un route table. Un Route Table, el cual existe su equivalente en hardware, va a establecer reglas de cómo se va a mover el tráfico entre las diferentes subredes y componentes. Esto quiere decir que si ocupamos que la red privada tenga saldia hacia internet, agregamos una entrada al route table y este se lo envía al Nat gateway. El nat se lo manda al Internet gateway y esta accede a Internet.

Para que la red pública pueda ver la red privada, como si tenemos un app en la red pública y una DB en la red privada, en el Route Table, definimos una ruta desde la red pública, a la tabla de ruteo, y después hacia la red privada.

Si tenemos una red de base de datos y queremos que salga a internet, es igual. Lo tiramos al route table y le decimos cómo salir. Para prevenir que un servidor no se pueda conectar a Internet, en el Route table no le definimos una forma de salir a Internet.

Hay otros controles que podemos poner. Toda red siempre va a venir con algo que se llama una ACL (Access Control List). En términos generales, es un NAC, Network Access Control. Va a limitar el tráfico que puede entrar a la red completa. Esto quiere decir que va a ser el primer Firewall que vamos a tener siempre en una red, pero tiene una característica interesante. Nosotros les decimos que es stateless. Esto se refiere a que cuando tenemos dos elementos comunicándose, como una base de datos y una aplicación, si la base de datos está dentro de una red privada y la base de datos se comunica con un agente externo, la base de datos origina la comunicación y de una red privada. Se va a generar un estado de lo que significa el inicio de la comunicación. Como la base de datos de cualquier elemento de la red le habla a un externo, en el momento que abro la conexión, le doy permiso de que me respondan. Esto tiene el peligro que la base de datos inicie la comunicación hacia un servidor de actualizaciones y este tenga algún tipo de backdoor. Como la base de datos inició la conversación, la información entra directamente. Aunque el Firewall no tenga abierto el puerto, el canal de comunicación queda abierto porque existe el estado. UN ACL es un network más rudo porque no cree en el estado. Como administrador de la red, tengo que explícitamente especificar que me voy a comunicar desde adentro y si quiero que me respondan, tengo que establecer una regla explícita permitiendo que me dé el Firewall de entrada.

Digamos que tenemos un servidor de DB con MySQL, el ACL va a permitir acceso de salida a la base de datos a Internet y de entrada a MySQL. Dentro del ACL, tengo que agregar una regla que va a decir, allow MySQL. MySQL tiene permitido enviar paquetes de vuelta. El estado para comunicar no es suficiente para definir el acceso de la comunicación. Tengo que decir que permito acceso a paquetes de MySQL, esto me da más control de la comunicación que estoy estableciendo.

Si algún servidor se conecta a un servidor comprometido y el último envía la información de vuelta, esta información va a ser bloqueada porque el ACL no tiene la regla que permite la comunicación en esa dirección. El ACL me permite controlar el acceso a redes inseguras, pero tengo que estar definiendo ese montón de reglas. Me ayuda a proteger a servidores que son sensibles.

¿Qué pasa en el caso de una configuración de un firewall normal? Si tengo Internet y servidor 1 y servidor 2. Explícitamente le estoy diciendo que permita que el servidor 1 ingrese al servicio por Internet. Si el servidor 2 origina la comunicación hacia la base de datos o el servicio, la comunicación no va a funcionar porque en el Firewall explícitamente no le he dicho que puede ingresar. Si por alguna razón alguien llegó al servidor en Internet y descarga algo, en este punto se genera un estado. Lo que sucede es que el servidor 2 puede venir a Internet y llegar al otro servicio. En un Firewall corriente, se permite cuando se inicia una configuración interna hacia un elemento que no está permitido. Un estado permite brincar la regla del Firewall.

Siempre la red va a tener Network ACLs. Una vez que los defino, defino dentro de las redes el concepto de Security Group. Un security group es literalmetne la definción de un Firewall corriente dentro de una red. El security group en el caso de una red cloud, va a tener una connotación pequeña o cosas extras. Un Firewall en su forma más básica, lo que hace es tener un IP Source, un IP destino, un Port Source y un Port Destino. Esto quiere decir que si defino una aplicación y tengo una base de datos y el network ACL me permite esa comunicación y tengo una ruta que sabe cómo mapear los ips. En el Firewall security group, puede delimitar reglas que eviten el acceso a mis servicios. Aunque el network ACL y la tabla de ruteo me deje mover información de la red pública a la privada, el security group me controla el tráfico a una aplicación específica.

En cualquier elemento de un cloud provider, podemos tener hasta 5 security groups. Podemos meterle hasta 5 Firewalls propios de cloud porque es muy conveniente tener Firewalls atómicos. Si para permitir tráfico HTTP/HTTPS necesitamos los puertos 443/ y 80 y para levantar NodeJS ocupamos el 3000, definimos Firewalls con configuraciones específicas para los servicios que vamos a tener. Si instalamos una aplicación que utiliza

Apache HTTP/HTTPS e instalamos un NodeJS, el servidor debería de incluir tanto el Firewall de HTTP como el de NodeJS. Si generamos una instalación de Apache más Tomcat, que usa el puerto 8081, incluimos las firewalls de HTTP y Tomcat. Si tengo varios Firewalls, puede definir configuraciones atómicas por cada uno y agrego los firewalls a los elementos que utilizan la red dependiendo a mis necesidades.

En un firewall común y corriente, tenemos IP y puertos fuente y destino. Pero supongamos que hay una base de datos y le pego un security group que dice que voy a permitir acceso en puerto 3306, con un filtro por IP que acepta conexiones desde el IP 10.10.10.10. Así, las aplicaciones con el IP 10.10.10.10, puede entrar a las bases de datos.

Un ataque común es el spoofing. Esto podría ser meter una máquina que se configure con el IP que tiene el servidor de aplicaciones. Empieza a enviar paquetes con la identificación de la aplicación. Esta es una debilidad de los Firewalls.

En los cloud providers, esto cambia un poco. En el momento en que tenemos un cloud provider, podemos definir Security Groups. Así, podemos definir “n” security groups. Luego, le decimos a las aplicaciones, a las VM que corren las aplicaciones, que usen los security groups. En este momento, la comunicación saliente de la aplicación va a llevar el IP y tags o etiquetas. Estas van a decir Security Group 1 y Security Group 2. Cada paquete viaja con los tags del security groups. Estos son objetos propios del cloud provider. Esto quiere decir que si alguien puso una máquina en el IP, para que puedan utilizar los security groups, tienen que hackear parte del cloud provider. Esto puede ocurrir si no configuramos bien el cloud provider. Si está bien configurado y entra una máquina maliciosa. Esta máquina no va a poder utilizar los security groups sin levantar todos los red flags por todos los sistemas de observabilidad.

En las reglas de firewall de bases de datos y redes, en cloud providers, agregamos security group source de donde viene el paquete. Cuando el paquete llega a la DB, revisamos que vengan del IP deseado, al puerto deseado y que vengan del security group deseado. Al hacer esto, tapamos esa posibilidad contra capa de que me puedan hackear.

Una vez tenemos los security groups, hay otro seguro que podemos tener. Es recomendado, pero no se usa en Costa Rica, es que el sistema operativo también tiene un Firewall. En Linux, IPTables es de los mejores que hay. Vamos a repetir reglas. Voy a tener primero tablas de ruteo, un bloqueo de acceso a Internet entrante a nivel de Nat gateway, un bloqueo de entradas enternads al network ACL sin estado, donde hay que definir reglas explícitas. La buena práctica es usar el Source del Security Group para prevenir Spoofing. Vamos a manejar la parte del Firewall para el sistema operativo.

AWS WAF: Web Application Firewall. Se sienta sobre la red a nivel de VPC y observa todo el tráfico. Hace un netflow. Agarra los paquetes que vienen de tráfico de red, los analiza y empieza a buscar si existe algún patrón extraño de ataques conocidos, de backdoors, de malware, etc. Todo el tráfico que entra a la red es filtrado. Después tenemos el guard, que básicamente va a empezar a analizar el tráfico tratando de identificar DOS attacks. Puede atajar DDOS normales y distribuidos.

El Guard DOS da la parte de AI y un equipo físico de personas expertas en ciberseguridad que están analizando patrones.

El VPC es una gestión meramente regional. Tiene una región geográfica donde hay 3 o más datacenters que están separados por 100 km. Si queremos maximizar la seguridad de la base de datos, en términos de disponibilidad, dentro de una misma región, voy a tener como mínimo los tres data centers separados. Si hacemos una red para la base de datos en un centro, esa red hay que replicarla en los otros, al menos tres

total, para poder asegurar máxima disponibilidad, para que se pueda escoger quién es el maestro. Tengo que crear redes en lo que se conoce como availability zones. Lo ideal es que tiremos un servidor una base de datos en cada availability zone y en el peor de los casos para que se nos metan a toda la base de datos, tienen que hackear tres availability zones de Amazon.

Cuando hackean un servicio en la nube, usualmente es por omisión de las personas que la administran.

Es importante tener presente el Responsability Matrix de AWS. Esto nos dice qué información es responsabilidad de Amazon, GCP o de Azure y qué es responsabilidad del cliente. Normalmente, AWS garantiza que las regiones, availability zones, edge locations (ubicación de datos de manera rápida), el hardware, el almacenamiento (discos), bases de datos como servicio, networking (red y que siempre esté arriba), estén arriba. A nosotros nos toca manejar el sistema y la seguridad. En un ambiente on premise, hay que ser responsable de todo el stack de tecnología. En el cloud, una parte es responsabilidad del CP y otra que es del administrador. La evolución de todo lo que es infraestructura ha ido subiendo, que hace que la matriz de responsabilidad se asigne más al cloud provider.

En la infraestructura como servicio, uno como cliente tiene responsabilidad de la encripción. Cuando nos movemos a un servicio con containers, lo que estamos haciendo es pasándonos a una plataforma como servicio. Transmitimos parte de la responsabilidad del cliente hacia el cloud provider. Después de eso apareció un managed service, donde nos dan todo listo para que yo solo envíe datos y saque datos de la base de datos.

La responsabilidad de todo el manejo de llaves de encripción de los datos en server side, en transit y eso, cae en el cloud provider. En la industria se usa más managed services que infrastructure, pero la razón por la cual pasa esto es porque si una empresa va a trabajar con una infraestructura de servicio y va a usar MySQL tiene que contratar a alguien experto en manejar servidores de base de datos. Si no hay suficiente tiempo de base de datos para esa persona, simplemente esa persona no hacía nada. En managed services, las empresas contratan a los desarrolladores y la administración se delega al cloud provider. Todos los cloud providers tienen bases de datos en managed services.

Existe otro enfoque que tenemos que tener presente: la seguridad en alta disponibilidad. ¿Cuánto es suficiente? Esto depende de la industria.

Una arquitectura más robusta que una base de datos distribuida en 3 availability zones es que también esté distribuida en diferentes zonas geográficas. Esto hace que para que falle la base de datos, tiene que haber un evento mundial. Aquí también entra en juego la geolocalización si los usuarios están cerca de los datos en la base de datos. Los costos de la red se reducen. La distribución multiregión puede ser para mejorar el acceso a los clientes y la alta disponibilidad.

Cuando tengo una red, Amazon, al igual que los otros cloud providers, va a utilizar la red que armamos como una red que no podemos armar y que pertenece explícitamente al cloud provider. Cuando estamos en una red on premise, el disco está conectado físicamente en al máquina. En un cloud provider, el disco se mueve a una red separada. Esto implica que la máquina corre en una red y a través de la red ingresa al disco. La ventaja es que tengo separados los datos de donde corren las aplicaciones. Así, puedo meter análisis, auditoría y puedo ver patrones de acceso al disco. Si tengo el disco en el servidor un atacante se metió, la base de datos está comprometida y no hay mucha forma de que pueda hacer algo en caso de que me hackeen. Así, pagamos el precio que por más buena que sea la red que tiene un cloud provider, nunca va a ser tan buena como tener los discos físicamente conectados a las máquinas que corren las bases de datos. Esto es una penalización muy significativa. Podría afectar en un 70% u 80%. Sin embargo, hay soluciones para

reducir esta pérdida de rendimiento de velocidad de acceso al disco. En el momento que nos movemos a aplicaciones que necesitan un alto rendimiento a nivel de hardware por sus operaciones críticas, se empieza a tener problemas. El disco siempre va a tener Server Side Encryption, el cual es manejado por un Key Management System. Este KMS va a permitir manejar las llaves de encriptación que pueden tener los diferentes elementos en una red en la nube. Esto nos garantiza trazabilidad: todo elemento que utilice una llave de encriptación va a grabar un evento. Estos eventos entran a una base de datos time series y puedo reconstruir todos los eventos hasta el punto en que alguien sacó algo de la base de datos. Cada llamada al API de amazon para pedir llaves de encriptaciónva dejando su rastro. después puedo hacer computación forense para reconstruir los elementos y las acciones para encontrar el punto donde perdí información. Me da trazabiildad y observabilidad. El manejo de llaves de encriptación no es abierto o gratuito. Existen políticas. Una política son permisos que voy a tener. Por ejemplo, el disco puede hacer gets de un key, o que el disco puede encriptar con un key específico, o que el disco puede desencriptar con una llave específica. En estas políticas, uno define la mínima cantidad de permisos que una aplicación o nube necesita para realizar su trabajo.

Normalmente junto con eso vamos a utilizar un rol. Los roles están asociados a un tipo de servicio. Si hay un disco, en Amazon se le llama EBS. El EBS va a manejar discos, es un Elastic Block Storage que es un servicio. Puede haber un rol asociado explícitamente con el servicio y luego con una política. Lo que pasa es que yo le estoy diciendo al rol que solo puede ser llamado por ese servicio y con la política, le digo que va a tener permisos para realizar esas acciones con el KMS. Lo puedo manejar y me da trazabilidad. Si alguien hackea el rol, y trata utilizarlo desde la base de datos, una máquina virtual como EC2 de Amazon es otro tipo de servicio. El rol no puede acceder a ese servicio porque no tiene el permiso y levanta todas las banderas que hay algo raro ahí.

Cuando tengo este tipo de comportamientos, creo roles con permisos específicos para cada uno de lso servicios dentro de la red. Voy a tener uan separación de datos at rest que van a vivir en una red separada donde las llaves de encriptación y la encriptación van a ser responsabilidad del cloud provider y no voy a tener que hacer el computing. Al igual que los otros componentes, van a tener un rol, que puede ser de EC2. Siempre tenemos que tener usuarios para la BD, nunca correr con un usuario administrador, el Firewall, monitoreo, security patches, service packs, aplicaciones contra malware, antivirus contra backdoors, que solo corren las aplicaciones que son, etc. La base de datos es una aplicación que necesita comer mucha memoria y CPU. Si tengo otra aplicación, voy a poner dos aplicaciones a competir, entonces la base de datos fuerza a que el sistema operativo empiece a paginar.

Hay otras ventajas con cloud que no tengo con una red on premise: la generación de snapshots a nivel de discos de red. Se copian los discos bit por bit, hago una copia exacta y lo saco y lo meto en un lugar seguro. Este lo maneja el cloud provider. Si alguien se hackea, roba y encripta los datos, tengo un respaldo de lso datos. Puedo haber tenido pérdida de información ya que lso snapshots se generan cada cierto tiempo, como cada 3 horas. Hay un concepto que se llama RPO y RTO que me dicen en qué momento creo snapshots y con qué frecuencia. Con más snapshots, peor el rendimiento y aumenta el costo. Si puedo generar snapshots a nivel de discos y alguien me hackea, apago el servidor de base de datos hackeado, lo borro y lo restauro. Ahí tengo un par de mecanismos más rápidos y seguros y administrados por el cloud provider.

Con las bases de datos tenemos ciertas peculiaridades en el momento de los backups. Si se levanta un disco que está inconsistente, ya que se quedó un commit de una transacción a medias, perdemos la consistencia. Usualmente, se ponen los discos en read only antes de hacer los inserts, pero esto hace que no se puedan escribir.

Cuando tenemos las bases de datos en cluster, tenemos que asegurar que la comunicación participen los servidores que tienen que hacerlo. Para garantizar esto, existen varias formas. Cada una tiene sus complejidades. Podemos definir un security group que acepte conexiones de un ip address. Si se muere uno de los servidores, se puede levantar otra réplica. En ese momento, lo que pasa es que el nuevo servidor no va a poder unirse al cluster porque notiene el IP en el security group, entonces hay que meterlo. Toda la parte de elasticidad, de recuperación de fallas, de alta disponibilidad, con un FIrewall, la tira al caño. Normalmente, uno define un security group, pero en lugar de hacer reglas basadas en IPs, hacemos reglas basadsa en el group. Como todos tienen el mismo security group, todos los paquetes saliendo de las máquinas tienen ese tag. Eso quiere decir que todos los servidores se van a poder comunicar.