

AWS: Introducción a la Nube

AWS: Introducción a la Nube

1. Introducción: On Premise “Servidor de toda la vida” vs Cloud
2. Modelos de la nube: IaaS, PaaS, SaaS
3. Principales servicios de AWS
4. La profesión de Cloud Engineer (conocido como DevOps anteriormente)

On Premise “Servidor de toda la vida”



On Premise “Servidor de toda la vida”

- **On Premise** = En las propias instalaciones
- **Dentro del servidor** y la infraestructura de la empresa
- **Modelo tradicional** de aplicaciones empresariales
- **Empresa responsable** de seguridad, disponibilidad y gestión del software

Conceptos de OnPremise

- **CPD:** Centro de Procesamiento de Datos
- **Servidores:** máquinas que van a correr SOs, máquinas virtuales, aplicaciones...
- **Capa de networking:** cableado de red, switches, salida a Internet...
- **Virtualización:** utilización del software para imitar características y comportamiento de un hardware concreto
- **NAS:** Dispositivo de almacenamiento conectado a la red (backup)
- **SAI:** Sistema de Alimentación Ininterrumpida, sirve para proteger de caídas de corriente

CPD

- **Ubicación geográfica:**

- Coste económico: coste del terreno, impuestos municipales, seguros, coste eléctrico, coste de datos, etc.
- Infraestructuras disponibles en las cercanías: energía eléctrica, carreteras, acometidas de electricidad, centralitas de telecomunicaciones, bomberos, etc.
- Riesgo: posibilidad de inundaciones, incendios, robos, terremotos, etc.

- **Recursos internos:**

- Falsos suelos y falsos techos
- Cableado eléctrico y de datos redundante
- Sistemas de alimentación ininterrumpida
- Refrigeración
- Sistema anti-incendios: no agua, protección de equipos

- **Recursos externos:**

- Múltiple acometida eléctrica y de datos
- Seguridad en los accesos
- Medidas de seguridad en caso de incendio o inundación: drenajes, extintores, vías de evacuación, puertas ignífugas, etc.

CPD

- **CPDs duplicados**

- Un centro de respaldo es un centro de procesamiento de datos (CPD) específicamente diseñado para tomar el control de otro CPD principal en caso de contingencia

- **Racks**

- Anchura estándar 19" (48,26cm)
- Profundidad y altura no normalizada
- Altura medida en U=unidades de rack, 1,75" = 4,44cm
 - Encontramos equipos en formato de rack con altura de 1U, 2U, 3U... según el espacio que requieran
 - La electrónica de red también sigue este estándar

CPD

- **Virtualización**

- Una máquina física puede correr múltiples máquinas virtuales (VM)
- Las aplicaciones corren sin modificaciones sobre las VM
- Las VMs pueden migrar de una máquina física a otra según necesidades
- Permiten mayor flexibilidad y rapidez para la creación y actualización de servicios
- Existe hardware específico para poder correr múltiples instancias de VMs de forma más eficiente: Servidores blade
- Los servicios de computación en la nube suelen basarse en esquemas de virtualización



Problemas de las soluciones On Premise

- **Dependencia** del proveedor
- **Coste oculto:** coste de oportunidad
- **Riesgo** de la conexión a Internet
 - Abrir los servidores On Premise
 - Exigencia de movilidad por parte de clientes
- **Flexibilidad y escalabilidad muy limitada**
- **Alto coste** de instalación y mantenimiento
- **Vulnerabilidad** de las instalaciones
- **Cuello de botella** en la conexión

Llega la nube

- En pocas palabras: **cómputo en Internet**
- Frontend / Backend
 - Tu frontend
 - Su backend (alquilado)
- Un concepto no tan reciente
 - Desde que existe internet
 - 1997 → entorno académico



Características de la nube

- **No eres propietario** del hardware
- **Autoservicio** bajo demanda
- Acceso amplio a servicios
- Pool de recursos **dinámicos**
- Medición de uso dinámica



Algunos conceptos de la nube

- **Fault tolerance**
 - Habilidad de un sistema de continuar funcionando cuando algo falla
 - Zero Down-Time
 - Ejemplo: Netflix
- **High Availability**
 - Habilidad de minimizar la pérdida de servicio cuando algo falla
 - N 'nines' → %
 - Ejemplo: Google
- **Disaster Recovery**
 - Contra catástrofes
 - Plan de recuperación
 - Puntos y tiempos específicos
 - Ejemplo: [Videojuego Rust](#)

Algunos conceptos de la nube

- **Escalabilidad**

- Habilidad de escalar la capacidad de cómputo, tanto de forma **horizontal** (número de máquinas) como **vertical** (tamaño de las máquinas)

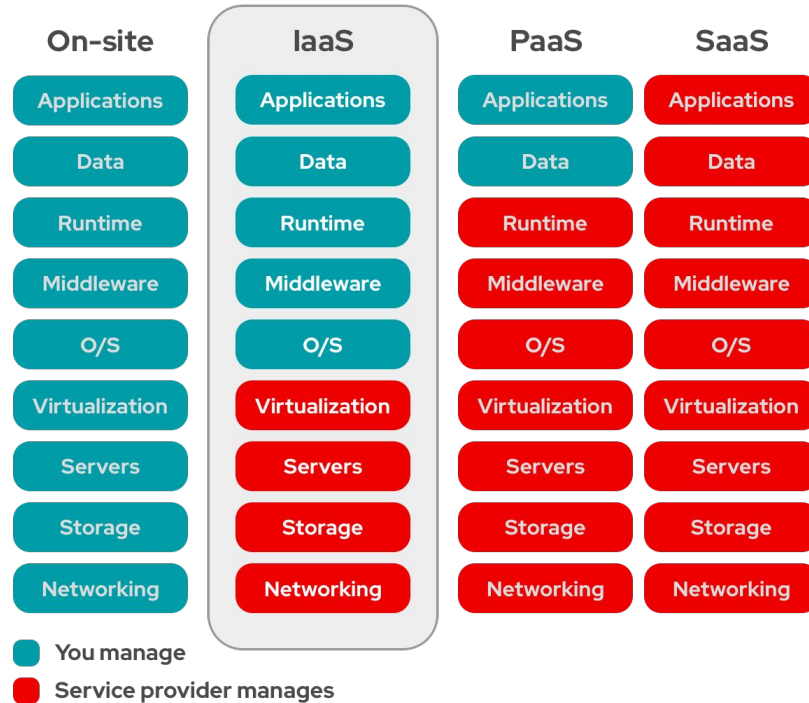
- **Elasticidad**

- Capacidad de aumentar o disminuir cualquier recurso con rapidez para responder a la demanda

- **Agilidad**

- Capacidad de desarrollar, testear e implementar rápidamente software y aplicaciones que aumentan el valor del negocio

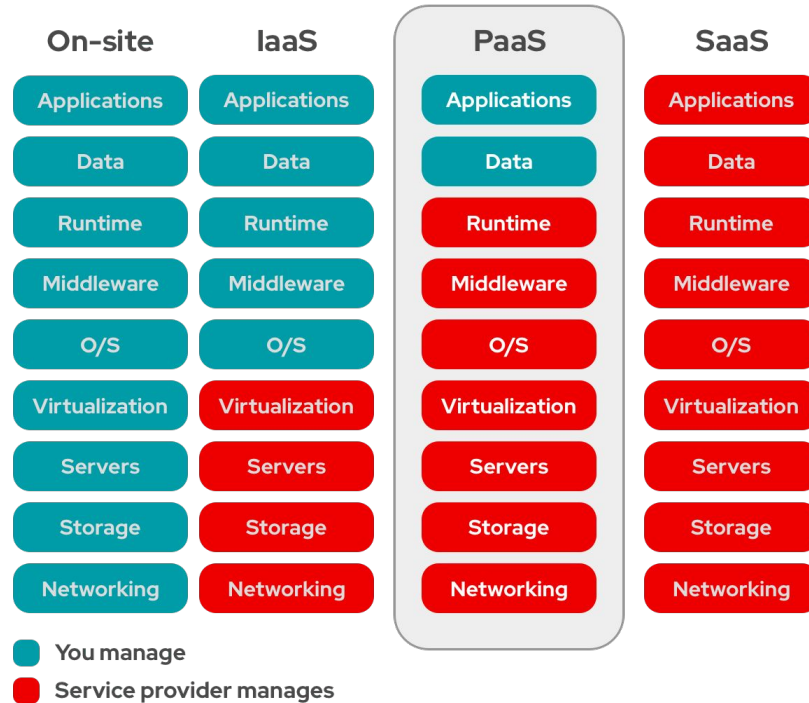
IaaS: Infraestructura como servicio



IaaS: Infraestructura como servicio

- **“Alquilar el Hardware”**
- Máquinas virtuales, firewalls, sistemas de backup, balanceadores de carga...
- Es la base: contiene los bloques fundamentales para la TI en la nube
- Ejemplos:
 - Amazon Web Services (EC2)
 - <https://fortune.com/longform/amazon-web-services-ceo-adam-selipsky-cloud-computing/>
 - Microsoft Azure
 - Google Cloud
 - <https://appinventiv.com/blog/aws-vs-azure-vs-google-cloud/>

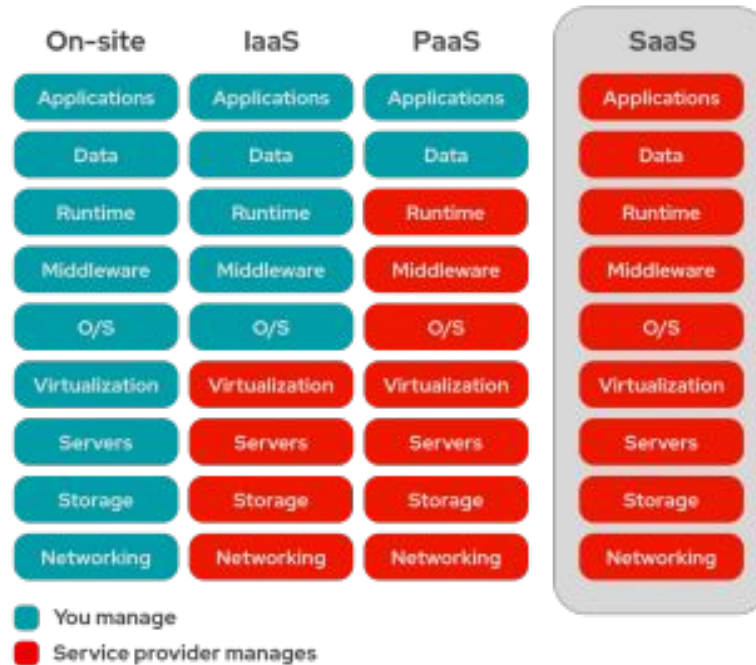
PaaS: Plataforma como servicio



PaaS: Plataforma como servicio

- **Plataformas sobre las que lanzar aplicaciones**, como bases de datos, middleware, herramientas de desarrollo...
- Ideal para desarrolladores que quieren centrarse en sus aplicaciones
 - Sin preocupaciones de los recursos
- Es más complicada de entender (es software, pero no es SaaS)
- Más conocida: Google App Engine <https://cloud.google.com/appengine>
- Otros ejemplos:
 - SAP Cloud
 - Heroku

SaaS: Software como servicio



SaaS: Software como servicio

- El **software** de una empresa está alojado en la nube y **se contrata su uso**.
- Requiere el **mínimo nivel de gestión**
 - Datos
 - Usuarios
- Ejemplos de SaaS:
 - Microsoft Office 365
 - Gmail, Google Drive...
 - Nuestros servicios (Veridas) ;)

Entonces cuáles son las diferencias entre ellas?

- **IaaS:** Servidores y redes sin configurar. Algo así como un alquiler del material necesario para desarrollar tus programas. Es como si fueses a la cocina de un amigo a hacerte tu la cena.
- **PaaS:** Servidores y redes configuradas. Un software instalado que te permite desarrollar tu proyecto con más facilidad. Es como si usases la cocina de tu amigo y este también te ayudase a hacer la cena.
- **SaaS:** Servidores y redes configuradas. Un software instalado que te soluciona parte o todo tu proyecto. Como si tu amigo te invitase a su casa a cenar.

IaaS: Infraestructura como servicio

Aspectos que deben considerarse a la hora de elegir un proveedor de IaaS

- **Flexibilidad:** adquiera únicamente los elementos que necesita para su caso práctico y amplíelos o redúzcalos en función de las necesidades empresariales.
- **Asequibilidad:** la IaaS es una opción asequible, ya que los gastos generales son bajos y no genera costos de mantenimiento.
- **Control:** el usuario controla la infraestructura.
- **Seguridad:** ¿tiene el proveedor una reputación confiable y posee los recursos necesarios para evitar y gestionar las amenazas de seguridad? ¿Existen protocolos documentados de recuperación ante desastres que garanticen la continuidad empresarial?
- **Servicio:** ¿cuáles son los acuerdos de nivel de servicio (SLA) del proveedor?
- **Confiabilidad:** el rendimiento y la velocidad dependen en gran medida del proveedor.

SaaS: Software como servicio

Aspectos que deben considerarse a la hora de elegir un proveedor de SaaS

- **Interoperabilidad:** La integración con tus otros programas tiene que ser posible. Si el SaaS no tiene una solución pensada, la integración puede ser más compleja.
- **Vendor lock-in:** Bloqueo de proveedor. Es fácil unirse a un servicio y difícil salir de él. Empezar a usar un SaaS puede crear una dependencia con este.
- **Seguridad:** Al utilizar un servicio externo cedes tu información a otra empresa. Eso puede ser un problema por temas de seguridad y/o confidencialidad
- **Falta de personalización:** En un SaaS no puedes decidir futuras funcionalidades. Si necesitas algo y tu proveedor no lo ofrece, solo te queda esperar (o pedirlo)
- **Rendimiento:** Obtienes el rendimiento que te ofrezcan. No puedes mejorarlo por tus propios medios, así que debes adaptarte a él

PaaS: Plataforma como servicio

Aspectos que deben considerarse a la hora de elegir un proveedor de PaaS

- **Básicamente una mezcla de los dos anteriores**
- Como hemos dicho el PaaS es un punto medio entre los otros dos servicios, y sus ventajas/inconvenientes son un punto medio entre los dos
- Los PaaS no tienen una características tan claras como los IaaS o los SaaS, por lo tanto sus ventajas e inconvenientes tampoco lo serán
- Deberíamos estudiarlo caso por caso

El SaaS de Veridas

Nosotros vendemos soluciones SaaS:

- **Document verification**
- **Face biometry**
- **Voice biometry**



El SaaS de Veridas

Nosotros vendemos soluciones SaaS:

- Document verification
- Face biometry
- Voice biometry

Que alojamos en soluciones IaaS de otros...



El SaaS de Veridas

Nosotros vendemos soluciones SaaS:

- Document verification
- Face biometry
- Voice biometry

Que alojamos en soluciones IaaS de otros...

Y algunos servicios los alojamos en nuestro CPD



El SaaS de IED

- **En este caso IED no tiene un SaaS como tal**
- **Tener una nube no significa tener un SaaS. Una nube también puede servir para**
 - **Gestionar tus empleados (acceso por AzureAD)**
 - **Gestionar recursos y herramientas internas**
 - **Almacenamiento para minería de datos, IA, observabilidad, etc.**
 - **POC's (proof of concept)**
- **Aunque si que estamos empezando a tener un poco de SaaS :)**

AWS: Principales Servicios

Como IaaS nos ofrece más de 200 servicios, pero los más famosos y útiles son:

- **EC2:** Básicamente el alquiler de servidores. Es el servicio core de su negocio y el más usado
- **S3:** Su servicio de almacenamiento. Otra pata clave en su negocio.
- **AWS Lambda:** Para poder ejecutar código sin usar un servidor
- **Bases de Datos (RDS, Aurora, etc.):** Se encargan de la gestión y tú solo consumes el servicio
- **Route 53:** Para gestionar dominios y rutas.
- **Amazon CloudWatch:** La solución de observabilidad integrada en AWS

Tener software en la nube no es una tarea fácil

- Hace falta tener conocimientos en redes, servidores y bases de datos para poder crear la base donde desplegar el software
- Necesitas un equipo que se encargue de desplegar estas aplicaciones con éxito y sin interrupción
- Hay que **mantener la nube**. Al fin y al cabo los servidores son máquinas en producción y estas pueden fallar. La monitorización es muy importante
- La nube puede llegar a costar mucho dinero si no se optimiza. Es necesario saber escalar y optimizar los despliegues
- Y mientras tanto otros equipos necesitarán ayuda para adaptar el software a la nube

Y.. quien hace todo esto?

- Los DevOps (o SRE, Cloud Architects, etc)
- Somos unos currelas
- Acabamos siempre pringando
- Estamos en mil fregaos y ninguno bueno
- Fuera bromas, es un puesto muy creativo y con muchas funciones y responsabilidades.
- **Interactuamos con muchos equipos y aprendemos muchísimo sobre tecnologías**

DevOps

- Realmente los “DevOps” no existen (sí la cultura DevOps), pero es una manera fácil de entender la manera de trabajar de la gente que se dedica a la nube.
 - Platform Engineer, Build Engineer, Release Manager, Site Reliability Engineer (SRE), Data Analyst, etc.
- Nuestras funciones son las de **crear**, **mantener** y **optimizar** todo lo que tenga que ver con infraestructura en la nube
- Nuestras tareas más comunes: desplegar nuevos microservicios, actualizar estos, mantener la nube con pequeñas tareas, optimizar la nube para reducir gastos y monitorear que todo funcione bien

Y como lo hacemos?

- **Automatizando.**
- **Todo.**
- Absolutamente todo.
- Algo no automatizado en producción va a ser nuestra mayor pesadilla.
- Y no lo hacemos porque somos unos quisquillosos. Si algo va a producción sin estar automatizado y falla podemos dejar a nuestros clientes sin servicio por mucho rato.
- Además sin automatizar somos más propensos a cometer **errores**.
- Son sistemas muy grandes y complejos y si algo falla encontrar el error puede costar mucho tiempo

Razones para automatizar

- Si algo falla, lo **arreglamos** mucho más **rápido**
- Cometeremos **menos errores**
- Automatizar es más complicado que hacerlo a mano, lo que nos ayuda a comprender mejor todo y estar preparados para futuras incidencias
- **No somos dependientes** de quien ha realizado el software.

Razones para no automatizar

- **Se tarda muchísimo más** en automatizar que en hacerlo manualmente.
- Un nuevo desarrollo puede que no tenga éxito y acabe abandonando.
- **Agilidad**

Herramientas básicas para automatizar

- **CI/CD:** Continuous Integration and Continuous Deployment. Integración y despliegue continuo
- **Tests:** Diferentes suites de tests para confirmar que todo funciona como debe
- **Métricas:** Nos indican cómo está respondiendo el sistema
- **Calidad:** Tenemos que construir una infraestructura de calidad para poder automatizar correctamente

Las incidencias

- Va a haber **incidencias**. Acéptalo.
- Y no hay que sentirse mal por ello, si no **aprender** de los errores.
- Es muy importante que después de una incidencia el equipo se reúna y estudie que ha pasado y cómo puede evitar el error.
- Nos podemos preparar para estas incidencias:
 - **Chaos Monkey**: Netflix saboteándose a si mismo.
 - **Disaster Recovery**: Como hacemos si todo sale mal.



ALBAÑILES DIGITALES

Developing your new career

