Orquestación de Identidad

Antes de empezar a hablar sobre Orquestación de Identidad, es importante saber lo que es la Gestión de Identidad; es decir a que pueden acceder los usuarios en una aplicación; se puede resumir en 6 aspectos:

1. Autenticación, es decir, como compruebas **si el usuario es quien dice ser,** ya sea mediante contraseñas, tokens, etc.
2. Control de acceso, donde se determina **si puede acceder** a la aplicación o no.
3. Autorización, donde se determinan las **acciones que puede realizar.**
4. Atributos, toda **información adicional y privada asociada** a un usuario, por ejemplo, roles, permisos, etc.
5. Administración, todo lo relacionado con la **gestión** (creación, eliminación, asignación de permisos...)
6. Auditoria, el **registro de actividades** realizadas por los usuarios en el sistema.

Entonces, ¿Que es la orquestación de identidad? Es una forma de gestionar estas identidades en entornos distribuidos. **En lugar de tener muchos** sistemas de gestión de identidad **independientes**, se crea una **capa de abstracción** que actúa como **intermediario** entre las aplicaciones y los diferentes sistemas, lo que simplifica y facilita la gestión de identidad.

Algunas **ventajas** de Orquestación **en un entorno empresarial** son:

- **Modernización**: **Facilita la migración** de sistemas **a la nube** ya que no habrá necesidad de reescribir todo el código para adaptarse.
- **Fusiones y Adquisicione**s: Facilita la **integración de tecnologías diversas.**
- **Seguridad**: Facilita la **integración de nuevas y mejores formas de autenticación**, mejorando así la seguridad

Ahora vamos a ver qué diferencias hay entre los sistemas en la nube y las instalaciones locales, orientadas hacia la gestión y orquestación de identidades.

- En los sistemas en la nube hay mayor variedad de capacidades que en local.
- La gestión de infraestructura se externaliza, es decir, está en un servidor remoto, responsable de un proveedor externo.
- Mayor resiliencia/ capacidad para recuperarse o resistir frente a fallos e interrupciones; lo q los hace más robustos, aunque esto no garantiza que estén libres de problemas.

Una posibilidad de usar sistemas de identidad en la nube y en local mediante la Orquestación sería tener configurado un sistema principalmente en la nube, pero en el caso de que ocurriese alguna interrupción en el servicio, se cambiaria a otro sistema en local para que la aplicación pueda seguir funcionando como respaldo hasta que se reestablezca el servicio.

Algunas características que es recomendable que tengan los sistemas de gestión de identidad para facilitar la orquestación lo máximo posible serian la capacidad para admitir estándares, uso de APIs modernas, como REST, y uso de nuevas tecnologías y practicas actuales; todo esto va a permitir una integración más fácil y ágil.

Las compañías prefieren crear una tabla con los usuarios y sus contraseñas porque es más sencillo y barato de implementar. Esto no es nada recomendable porque genera una deuda técnica de cara al futuro y no es nada seguro puesto que las contraseñas se pueden filtrar.

La mejor opción es hacer uso de la capa de abstracción que nos proporciona la Orquestación de identidad con una autenticación multifactor, por ejemplo, una aplicación de autenticación o un código QR. Esto es mucho más seguro porque el ID del usuario está encriptado y depende de algo externo para la autenticación. La forma más rápida para implementar MFA si se tiene un sistema con contraseñas es Auth0.

La implementación de la Orquestación de identidad no debe confundir al usuario. Por tanto, debe ser lo más transparente posible y conllevar el menor número de cambios posibles. Si es imprescindible realizar cambios que afecten a la experiencia del usuario, se le debe avisar y educar para que no rechace esos cambios.

¿Es posible integrar varios sistemas de autenticación multifactor? La respuesta es sí. De hecho, cuantas más opciones mejor. La capacidad del "O" simboliza esto: Puedo hacer algo de esta manera O de esta otra O de otra distinta. La propia capa de abstracción se encarga de evitar confusiones entre los sistemas que coexistan.

Un ejemplo: Viajas en avión, por lo que no puedes autenticarte con un método que requiera de conexión a internet. Si tienes la alternativa de autenticarte con un token almacenado en tu hardware, problema solucionado. Tener más opciones es mejor.

No hay prácticamente diferencia entre los usuarios y APIs dado que estas acceden y se comportan de la misma forma que lo haría un usuario. Deben autenticarse utilizando certificados y no claves de API porque son igual que las contraseñas, con la falta de seguridad que implican.

En el ciclo de vida de las identidades se puede identificar dos puntos de vista: El administrativo y el en ejecución.

El primero corresponde a la creación, lectura, actualización y eliminación de cuentas (operaciones CRUD). Así como verificar la identidad en el proceso de creación e importar identidades ya existentes.

El segundo se refiere a todo lo que realiza el usuario durante la ejecución. Por ejemplo, login o acceder a una información/proceso que requiera que el sistema verifique su identidad.

No se debe hacer esperar al usuario mientras que se verifica su identidad. Se debe agilizar y disimular la verificación de su identidad dividiendo el proceso de crear una identidad en 3 fases:

1. Solicitar escalonadamente información al usuario.
2. Dividir dicha información para emplear solo la necesaria en verificar identidad.
3. Otorgar al usuario una credencial (no contraseña) para que pueda autenticarse en el futuro.

Un ejemplo: La creación de una cuenta bancaria. Lo mejor sería pedir información al cliente de forma escalonada y progresiva para que el sistema vaya preparando la verificación de su identidad. Se podría pedir al inicio su nacionalidad para saber qué sistema de identificación estatal se va a interactuar. Mientras el cliente sigue rellenando el formulario con otra información, nuestro sistema verifica su identidad sin que él se dé cuenta. Posteriormente si todo ha salido correctamente, le otorgamos una credencial de acceso a su nueva cuenta bancaria.

En cuanto a seguridad, el orquestador se podría pensar que es un punto crítico de ataques, de brechas de seguridad... Pero no es cierto ya que no contiene la información de los usuarios, simplemente utiliza estos datos y coordina los distintos sistemas de gestión de identidad.

Y, ¿La orquestación no añade más **complejidad al sistema**?
En primera instancia, **sí**. Puesto que hay que crear la orquestación y es una capa más por encima de los diferentes sistemas.
Pero poniendo, por **ejemplo**, que tenemos **5 sistemas diferentes que necesitan ser modificados** por cambios en políticas internas o externas, en un sistema normal, sin orquestación, se necesitaría modificar los 5 sistemas diferentes para adaptarnos al nuevo cambio. En cambio, si poseemos orquestación **el cambio se realizará desde un punto común**, **nos da consistencia y reduce las gestiones** a realizar a un solo punto.

Hay **diferentes formas de afrontar la orquestación**, pero la opcion del entrevistado fue separar entre:

- **Plano de control**: lugar donde estableces las **reglas** y políticas.
- **Plano de ejecución**: realiza las **acciones** necesarias segun las reglas establecidas en el plano de control.

Esta división se realiza ante la necesidad de cambios en las políticas o la forma que funciona el orquestador, **no tener que reiniciar el sistema completo**.

El orquestador puede tener varias **funciones**, las principales que nombra son:

- **Proxy**: intermediario de peticiones, es la forma más sencilla de entender la función del orquestador.
- **Translation layer**: paso entre diferentes protocolos de identificación.

Ante la necesidad de **tratar los datos de usuario de diferente forma (EUROPA VS EE. UU.)**, se utiliza la orquestación de la siguiente forma: Uso de **dos proveedores de identificación**, en este caso, uno Europa y otro en EE. UU., totalmente separados y respetando las regulaciones de cada región.
En caso de necesidad de **acceso a datos de la otra región se solicita al orquestador,** te la proporciona y luego la puedes utilizar; no hay necesidad de replicación.