4. Onboarding e Identidad

Comenzaré el capítulo viendo lo que se necesita para poner en marcha los fundamentos de nuestro control plane. Aquí es donde veremos el aprovisionamiento de la infraestructura y recursos que se necesitan para alojar los diversos servicios que se usarán para gestionar y operar tu arquitectura SaaS.

Creación de un entorno base (Baseline Environment)

Los servicios que soportan onboarding se ejecutan dentro del control plane, así que necesitamos comenzar poniendo en su lugar todas las piezas que se necesitan para ejecutar todos los microservicios del control plane que soportan onboarding e identity.

Esencialmente necesitamos crear los scripts y la automatización que nos permitirán poner en marcha todos los elementos que se necesitan para alojar nuestro entorno SaaS.

Aunque nuestro objetivo es poner en marcha onboarding e identity, el alcance del entorno base incluye todos los recursos que serían aprovisionados una sola vez para configurar nuestro entorno multi-tenant antes de que comencemos con el onboarding.

La creación real de nuestro entorno base se logra a través de un modelo DevOps clásico, usando herramientas de automatización de infraestructura para crear, configurar y desplegar todos los activos que son requeridos por nuestro entorno base.

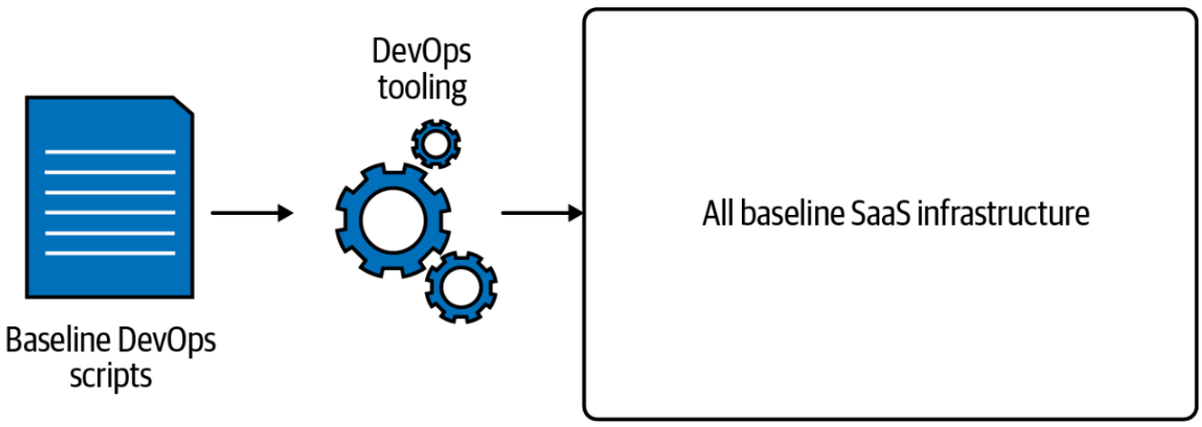

Figure 4-1. Automating creation of a baseline environment

Por supuesto, lo que realmente está en tu entorno base variará enormemente basándose en la naturaleza del stack tecnológico específico que estés usando para tu solución SaaS.

Creación de tu entorno base

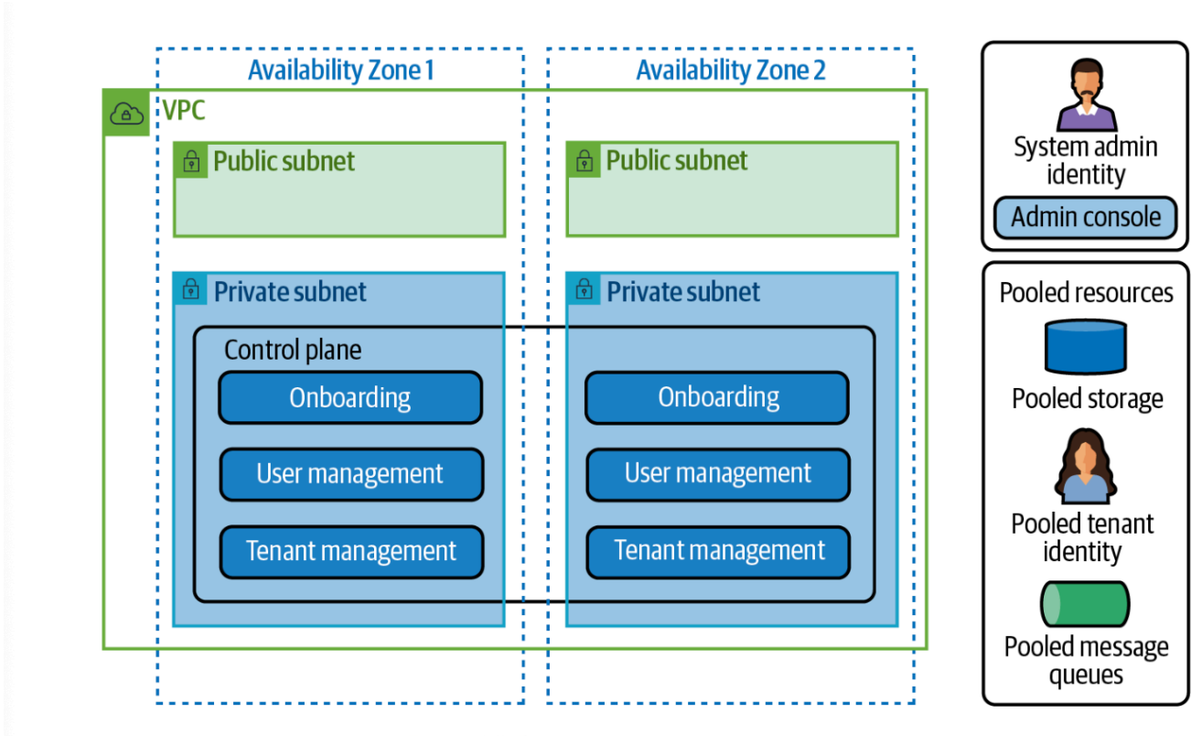

Figure 4-2. Provisioning a baseline environment

En el medio de la Figura 4-2, verás que he creado la infraestructura de redes fundamental que se necesita para alojar mi entorno SaaS multi-tenant.

La clave en esta etapa es solo enfocarse en el hecho de que la configuración y setup del entorno base requerirá que aprovisiones y configures todos los elementos de redes principales que serán usados por tu *control plane* y, potencialmente, tus tenants.

Dentro de esta red, **también he mostrado el despliegue del control plane.** Ya que el control plane es compartido por todos los tenants, puede ser configurado y desplegado como parte del aprovisionamiento de tu entorno base.

> Generalmente, si tienes recursos pool que serán compartidos por todos los tenants, puedes aprovisionarlos durante la configuración de tu entorno base (ya que no necesitarán ser creados durante el proceso de onboarding).

En la parte superior derecha, he mostrado marcadores de posición para la identidad de admin del sistema y la consola de administración. Esto representa a los usuarios que están iniciando sesión en la herramienta específica que has creado para soportar, actualizar, configurar y, en general, gestionar el estado de tu arquitectura multi-tenant.

La mayoría de equipos SaaS requieren su propia aplicación de admin personalizada que pueda abordar las necesidades multi-tenant específicas de su entorno.

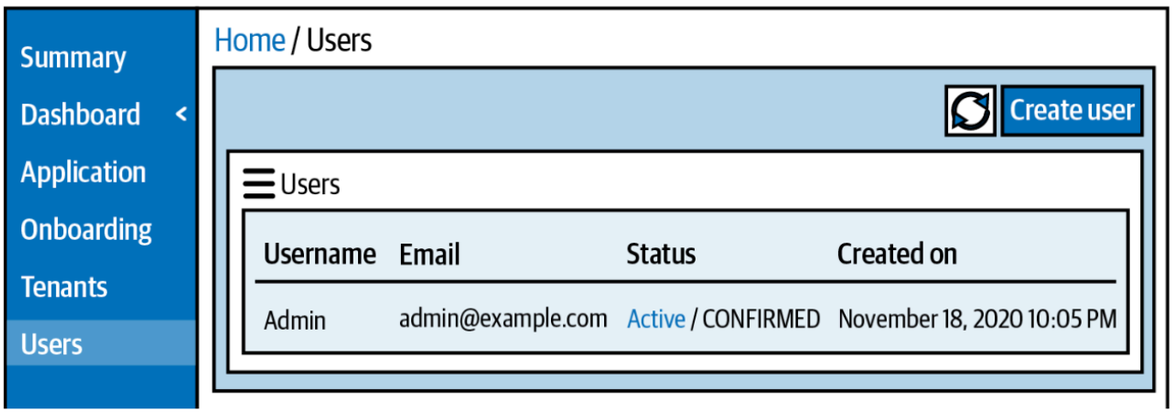

Figure 4-3. Creating and deploying a system admin console

Vale la pena notar que algunos equipos tienden a subinvertir en sus consolas de admin, prefiriendo soluciones listas en lugar de construir algo ellos mismos. **Generalmente, este trade-off rara vez parece valer la pena.** Aunque podrías ser capaz de usar soluciones de terceros para componer una experiencia de consola, hay operaciones específicas, perspectivas y opciones de configuración que solo pueden ser abordadas efectivamente a través de la creación de una experiencia dirigida.

Creación y gestión de identidades de Admin del Sistema

Como parte de configurar tu entorno base y configurar tu aplicación de administración, verás que tu proceso de aprovisionamiento también debe configurar tu modelo de identidad de admin del sistema.

Cada vez que actives la creación de un entorno base, se te requerirá proporcionar el perfil del usuario administrador inicial que podrá iniciar sesión en tu consola de admin.

La creación de esta identidad está completamente separada de la creación de una identidad de tenant.

Para soportar esta identidad de admin del sistema, necesitarás tener algún proveedor de identidad que posea y autentique a estos usuarios. El proveedor de identidad que uses aquí podría ser el mismo proveedor de identidad que se usará para tus identidades de tenant.

La conclusión clave es que necesitarás algunos pasos en tu automatización de aprovisionamiento del entorno base para crear y configurar tu modelo de identidad de administración del sistema. Esta automatización incluirá la creación y configuración del proveedor de identidad junto con la creación de los usuarios de admin del sistema iniciales.

Activación de onboarding desde la Consola de Admin

Una vez que hayas establecido tu usuario admin del sistema y tengas tu consola de admin en funcionamiento, tienes todas las piezas en su lugar para crear e incorporar tenants.

Ahora, en la versión final de tu oferta, tu onboarding podría ser invocado como parte de alguna experiencia de autoservicio, o podría ser impulsado por algún proceso interno.

Esto significaría tener alguna operación dentro de tu consola que recopile todos los datos necesarios para un nuevo tenant antes de invocar la operación de onboarding.

Opciones de aprovisionamiento del Control Plane

Mostré el *control plane* siendo desplegado en la misma infraestructura base donde tus tenants también aterrizarían. Esta es una opción perfectamente válida.

Sin embargo, vale la pena notar que cómo y dónde se coloca este control plane puede variar basándose en las necesidades de tu entorno y el stack tecnológico que se está usando para tu arquitectura multi-tenant.

También podría elegir colocar el control plane en una infraestructura completamente separada que esté dedicada al control plane.

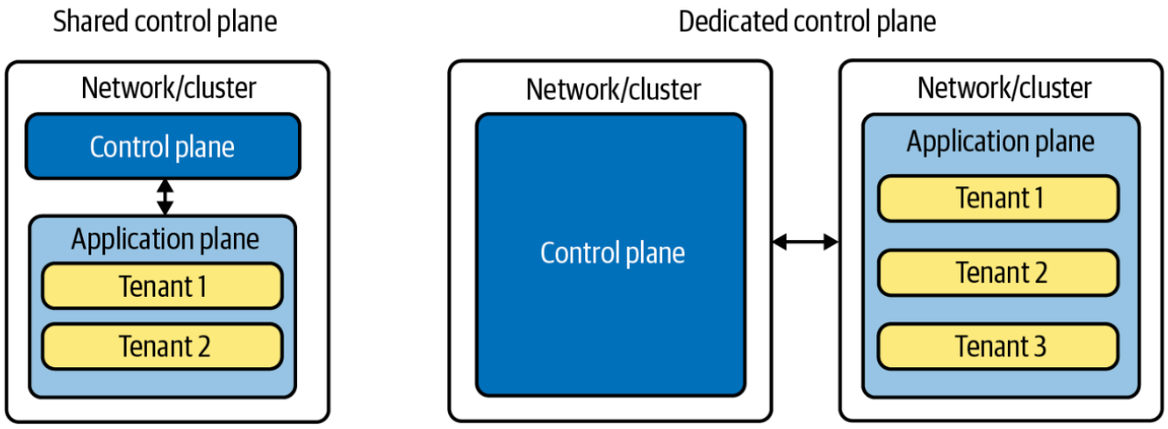

Figure 4-4. Picking a control plane deployment model

Los trade-offs de estas dos opciones son bastante directos. Podrías elegir tener un entorno de control plane dedicado si quieres escalar, gestionar y operar estos entornos completamente de manera independiente.

Poner el control plane en el mismo entorno con el *application plane* sí simplifica las cosas un poco. Reduce el número de partes móviles que tienes que gestionar, configurar y aprovisionar. También podría reducir tu huella de costos.

Algunos equipos, por ejemplo, podrían optar por diferentes stacks tecnológicos para el control plane y el application plane. Podría, por ejemplo, elegir serverless para el control plane y contenedores para el application plane.

La experiencia de onboarding

Es a través del onboarding que encontrarás que estás estableciendo y ejercitando algunos de los elementos más fundamentales de una arquitectura multi-tenant.

Comenzar aquí obliga a los equipos a responder muchas de las preguntas difíciles que influenciarán y darán forma al resto de su arquitectura SaaS. Onboarding no se trata solo de crear un tenant. **Se trata de crear y configurar todas las partes móviles de tu infraestructura que se necesitan para soportar ese nuevo tenant.**

Onboarding es parte de tu servicio

En un entorno SaaS, onboarding no se ve como algún script o automatización que esté de alguna manera fuera del alcance de tu oferta. En cambio, es uno de los componentes más fundamentales de tu experiencia SaaS y hacerlo bien debería ser clave para cualquier equipo que esté construyendo una solución multi-tenant.

La experiencia de onboarding también está directamente conectada a la noción de time to value (tiempo para generar valor), que examina cuánto tiempo le toma a un cliente moverse desde el registro hasta la productividad y valor real dentro de tu oferta SaaS.

Onboarding también es donde las estrategias de despliegue, identity, routing y tiering se ponen en acción. Cómo los tenants están aislados en silo y en pool, por ejemplo, necesitará ser expresado y realizado directamente a través de tu experiencia de onboarding.

En muchos aspectos, tu configuración de onboarding, automatización y código de despliegue estarán en el epicentro de realizar las estrategias multi-tenant que adoptes para tu entorno SaaS.

En realidad, onboarding representa uno de los elementos más fundamentales de un entorno multi-tenant. **Es a través del onboarding que puedes lograr los objetivos operacionales y de agilidad que son esenciales para un negocio SaaS.**

Los elementos fundamentales del onboarding

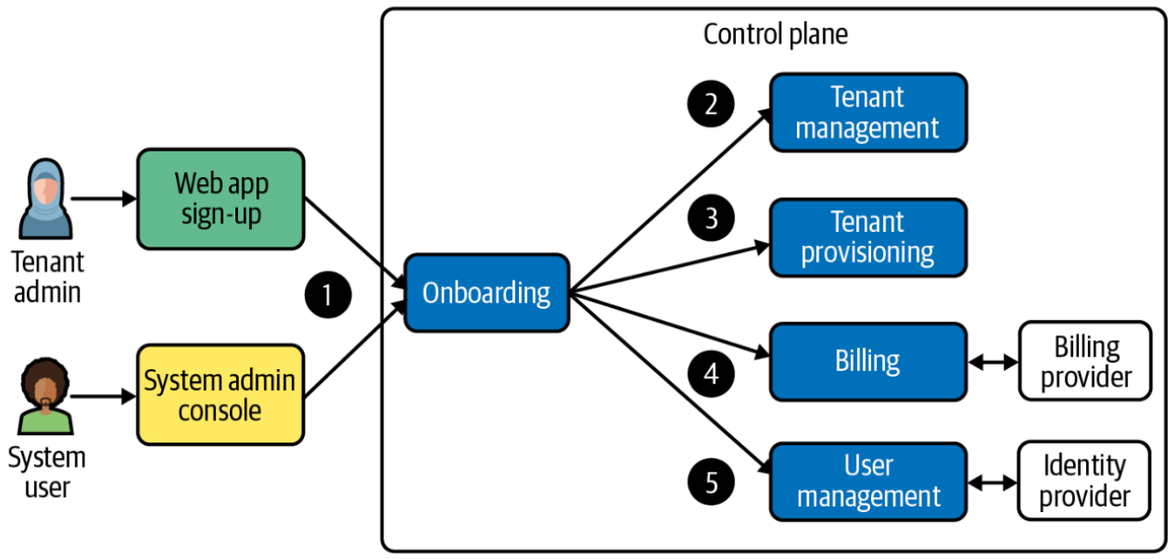

Figure 4-5. The fundamentals of tenant onboarding

Primero, he mostrado un administrador de tenant que se está incorporando a través de un proceso de registro autoservicio, presumiblemente una aplicación web que permite al tenant enviar su información, seleccionar un plan y proporcionar la información de configuración necesaria para establecerse como un nuevo tenant en el sistema.

También he mostrado un segundo flujo de incorporación que, en este ejemplo, es iniciado por un administrador del sistema. Esto representa algún rol interno en el proveedor SaaS que utiliza una consola de administración (u otra herramienta) para ingresar los datos de incorporación de un nuevo tenant e iniciar el proceso de onboarding.

Para la incorporación, generalmente prefiero tener un único servicio de onboarding que pueda gestionar toda la orquestación del proceso. Este servicio es responsable del ciclo de vida completo del proceso de incorporación, administrando y asegurando que todos los pasos del proceso se completen exitosamente.

El proceso de onboarding luego llama a una serie de servicios distribuidos que se utilizan para crear y configurar los ajustes del tenant y la infraestructura de soporte. La secuencia de este flujo de onboarding puede variar según la naturaleza de tu aplicación SaaS. **En general, el objetivo es crear y configurar todos los recursos requeridos del tenant antes de activar el tenant y/o notificar al usuario administrador del tenant que su cuenta está activa.**

Aunque hay múltiples formas de implementar este flujo de onboarding, necesitarás comenzar creando un identificador de tenant. En nuestro ejemplo, este identificador de tenant será creado enviando una solicitud de creación de tenant al servicio de Gestión de Tenants (paso 2), pasando toda la información sobre nuestro tenant (nombre de la compañía, configuración de identidad, nivel/tier, etc.). También generará el identificador único que será asociado con nuestro tenant.

> Los equipos a menudo usan un identificador único global (GUID) como valor para su identificador de tenant, **evitando la inclusión de cualquier atributo que pueda estar conectado al nombre u otra información identificable sobre el tenant.**

El siguiente paso en nuestro ejemplo de onboarding de tenant implica el aprovisionamiento de cualquier recurso de tenant que sea requerido (paso 3). Este paso de aprovisionamiento puede, para algunas arquitecturas multi-tenant, representar una de las piezas más significativas de tu implementación de onboarding.

A medida que profundizamos en ejemplos más trabajados, es posible que sorprenda descubrir cuánto código y automatización se dedica a esta experiencia de onboarding. De hecho, esta es a menudo un área donde los sistemas SaaS desdibujan los límites de DevOps. Los entornos SaaS pueden depender de la ejecución de código DevOps durante el onboarding de cada tenant individual.

Ahora podemos agregar este nuevo tenant al sistema de facturación (paso 4). Esto es esencialmente donde proporcionarás información al sistema de facturación que identifique al nuevo tenant e información que sea necesaria para caracterizar el modelo de facturación que debe aplicarse a este tenant en particular.

Para la pieza final de la experiencia de onboarding, necesitamos crear el usuario admin del tenant (paso 5). Si lo recuerdas, el rol de admin del tenant representa el primer usuario que se crea para un tenant determinado. Este tenant tendrá la capacidad de crear cualquier usuario adicional que pueda acceder al sistema.

Este proceso entonces obliga al usuario autenticado a ingresar una nueva contraseña como parte del flujo de inicio de sesión. El objetivo es impulsar gran parte de la automatización de este proceso de registro a tu proveedor de identidad.

Asumiendo que el onboarding sea exitoso, el servicio de Onboarding ahora puede llamar al servicio de *Tenant Management* y actualizar el estado activo a verdadero.

Seguimiento y visualización de estados de onboarding

Cuanto más complejo sea este proceso, más importante es tener información operacional útil y detallada en los diversos estados de tu flujo de onboarding.

Como mínimo, podrías imaginar tener un conjunto distinto de estados mapeados a cada uno de los pasos en nuestro flujo de onboarding. Entonces, podrías tener estados separados para TENANT_CREATED, TENANT_PROVISIONED, BILLING_INITIALIZED, USER_CREATED, y TENANT_ACTIVATED. Cada uno de estos estados podría ser visualizado a través de la vista de tenant en tu consola de administración, permitiéndote inspeccionar el onboarding de cualquier tenant en un momento determinado.

El valor real de asignar y visualizar estados de onboarding es proporcionar información operacional más rica sobre el estado de tu progreso de onboarding.

Esto es especialmente importante cuando tu proceso de onboarding incluye una cantidad significativa de aprovisionamiento de infraestructura y configuración. **En estos casos, podrías rastrear estados más granulares que te den información sobre los diversos pasos que están dentro de las partes móviles de tu proceso de aprovisionamiento.**

Onboarding basado en tiers

Este proceso de aprovisionamiento se vuelve un poco más interesante cuando consideras cómo los diferentes tiers de tenant podrían influir en cómo implementas tu ciclo de vida de aprovisionamiento.

Estas diferentes experiencias a menudo se traducen en una necesidad de infraestructura y configuraciones separadas basadas en el tier de tu sistema.

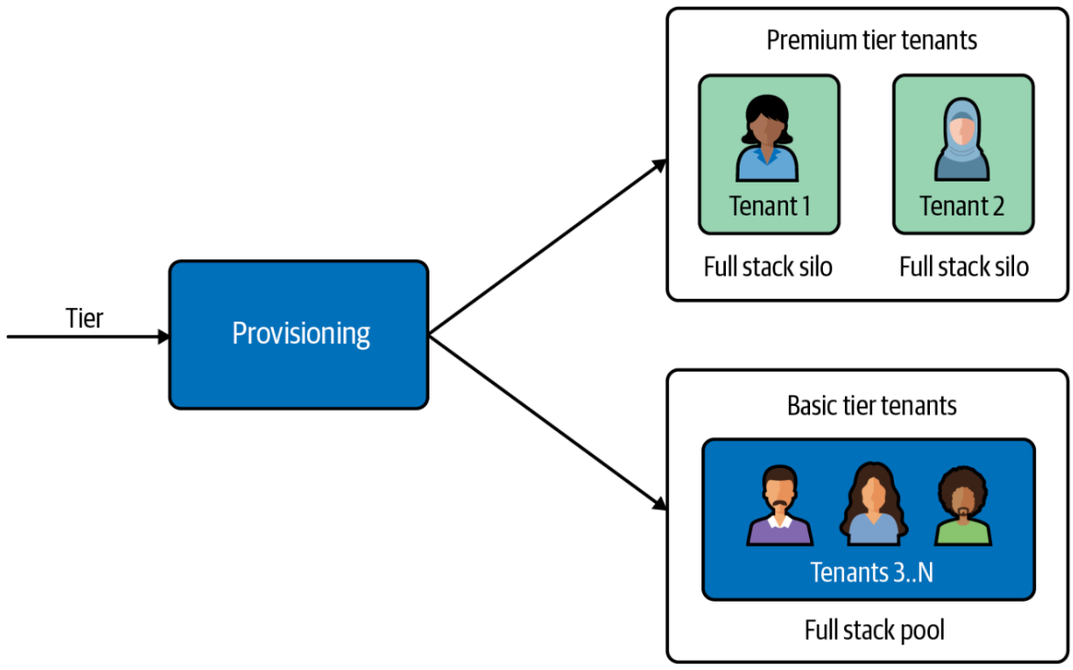

Figure 4-6. An example of tier-based onboarding

Cuando el servicio de Provisioning recibe esta solicitud, evaluará el tier y determinará cómo el tier seleccionado influenciará la configuración e infraestructura que será necesaria para soportar tu entorno de tenant.

Dónde se vuelve más interesante es cuando tienes un modelo de despliegue mixto. Con un despliegue en modo mixto, tus recursos están aislados en silo y en pool con granularidad más fina. Esto significa que tu proceso de onboarding necesitará aplicar políticas de onboarding basadas en tiers a cada recurso basándose en su configuración de silo o pool.

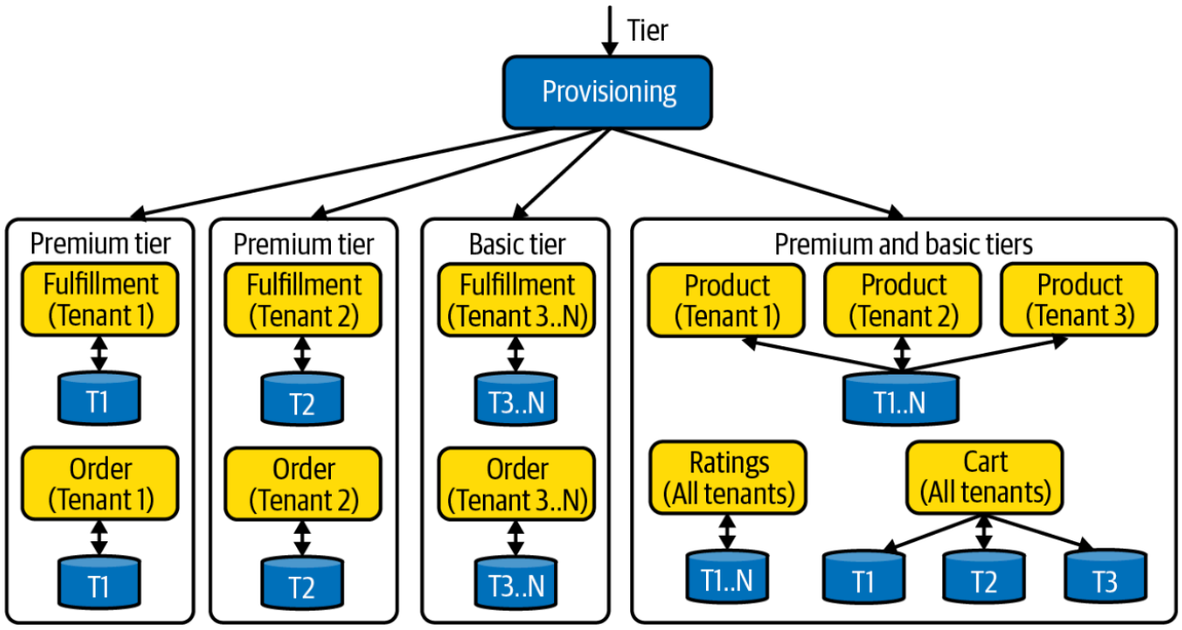

Figure 4-7. Tier-based onboarding with mixed mode deployments

Soportar el onboarding de estos servicios es entendeder cuáles son silos y cuales son pooled, además de disparar la creación y configuración de esos recursos dependiendo del contexto.

Una pregunta clave que suele surgir tiene que ver con el momento adecuado para aprovisionar los recursos compartidos durante el proceso de onboarding. **Dado que estos recursos se configuran y despliegan una sola vez, muchos prefieren preaprovisionarlos como parte de la configuración inicial de todo el entorno multi-tenant.**

Esto podría implicar que tu servicio de aprovisionamiento soporte una ruta independiente, invocada por tus herramientas de DevOps, para llevar a cabo la creación única de estos recursos.

La otra opción es retrasar la creación de estos recursos compartidos y activar su provisión durante el onboarding del primer tenant, de forma similar al patrón de *lazy loading*.

Seguimiento de recursos incorporados

Con el despliegue en modo mixto que tenemos aquí, no podemos simplemente desplegar en una ubicación estática para actualizar nuestro sistema. Imagina, por ejemplo, lanzar una nueva versión del servicio de Order. Para obtener el nuevo código desplegado, tu experiencia de DevOps necesitará encontrar los despliegues separados del servicio de Order que abarcan todos los diferentes recursos que fueron aprovisionados por la experiencia de onboarding.

Entonces, eso plantea la pregunta: ¿cómo sabría tu proceso de despliegue cómo manejar esto? ¿Cómo sabría qué recursos están aislados en silo para cada tenant? La única forma de que esto funcione es que tu experiencia de onboarding capture y registre la ubicación e identidad de estos recursos por tenant.

Lo principal es que si tu proceso de onboarding aprovisiona recursos dedicados de tenant, necesitarás capturar y registrar la información sobre estos recursos para que puedan ser referenciados por otras partes de tu experiencia de despliegue y operacional.

Manejo de fallos de onboarding

Aunque parte de tu confiabilidad aquí será extraída de la aplicación de prácticas de ingeniería sólidas, también hay áreas dentro del onboarding donde tus dependencias en sistemas externos pueden afectar la durabilidad de tu proceso de onboarding.

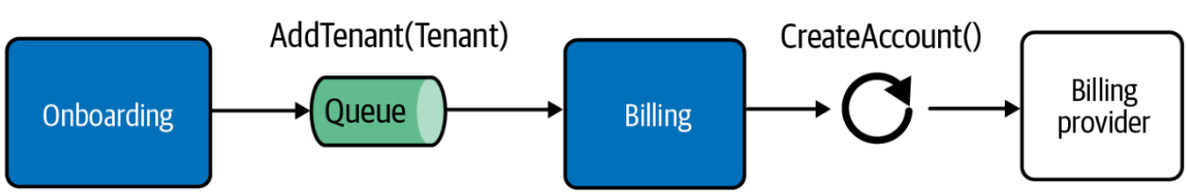

Figure 4-8. Fault-tolerant integration with a billing provider

Ahora, podrías simplemente presuponer que este es solo el riesgo asociado con el uso de soluciones de terceros. Sin embargo, en este escenario, tu sistema deberá ser capaz de continuar operando, incluso cuando el sistema de facturación esté fuera de línea.

Probar la experiencia de onboarding

La complejidad potencial y el número de partes móviles en este proceso puede hacerlo particularmente propenso a errores.

Demasiados equipos construyen un proceso de onboarding y simplemente se basan en la actividad de los clientes para descubrir cualquier cuello de botella o defectos de diseño que podrían estar impactando su experiencia de onboarding.

El objetivo es asegurar que el diseño, arquitectura y suposiciones de automatización de tu experiencia de onboarding se estén realizando completamente en tu solución de trabajo.

Creación de una identidad SaaS

Con identidad multi-tenant, tendrás que ir más allá de pensar en identity puramente como una herramienta para autenticar usuarios. **Debes ampliar tu visión de identity para incluir la idea de que cada usuario autenticado debe ser siempre autenticado en el contexto de un tenant.**

Así que esto significa que nuestro modelo de identidad debe ser expandido para cubrir tanto usuarios como tenants. El objetivo básico es crear una vinculación más estrecha entre usuarios y tenants que permita que sean accedidos, compartidos y gestionados como una única unidad.

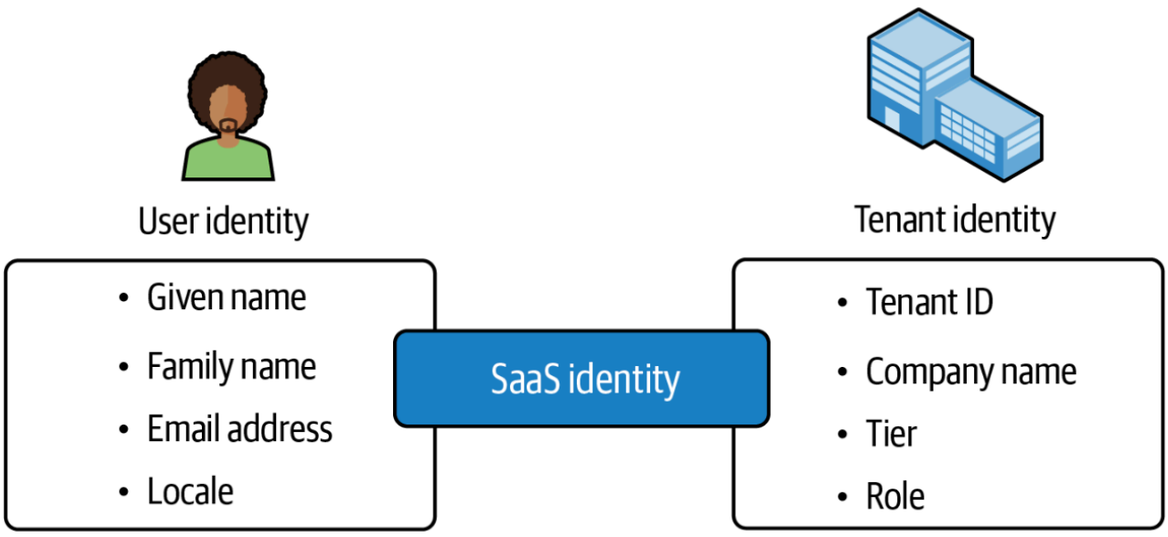

Figure 4-9. Creating a logical SaaS identity

Para entornos multi-tenant, estas dos nociones distintas de identidad se unen para crear lo que denomino una identidad SaaS.

Esta identidad SaaS debe ser introducida de una manera que le permita convertirse en un elemento de identidad de primera clase que se pasa a través de todas las capas de tu sistema.

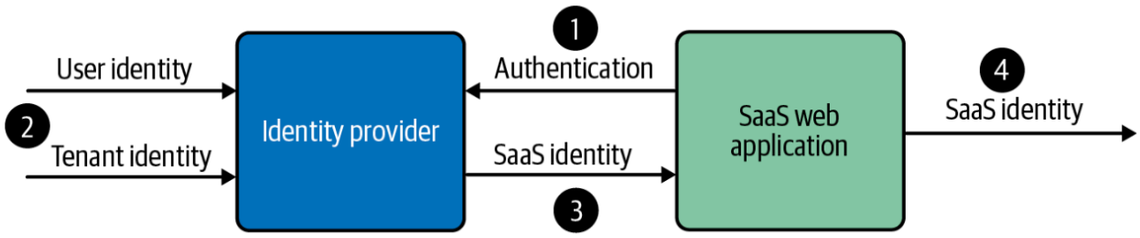

Figure 4-10. SaaS identity authentication flow

Subdominios, direcciones de correo electrónico o tablas de búsqueda, por ejemplo, podrían configurar cómo resuelves la ruta de un tenant hacia el proveedor de identidad correspondiente. **Al final, tu objetivo es resolver y crear esta identidad SaaS al inicio de este proceso y evitar empujar esta responsabilidad más adelante en los detalles de tu diseño e implementación.**

Vinculación de una identidad de tenant

Vinculación de una identidad de tenant

Para esta discusión, voy a enfocarme en cómo las especificaciones Open Authorization (OAuth) y OpenID Connect (OIDC) pueden ser usadas para crear y configurar una identidad SaaS.

Para vincular tenants con usuarios, primero necesitamos entender cómo la especificación OIDC empaqueta y transmite la información de autenticación de un usuario. Generalmente, al autenticar contra un proveedor de identidad compatible con OIDC, encontrarás que cada autenticación retorna tokens de identidad y acceso. Estos están representados como JSON Web Tokens (JWTs) que contienen todo el contexto de autenticación para ser usado en la autorización downstream. El token de identidad está pensado para transmitir información sobre un usuario, mientras que el token de acceso se usa para autorizar el acceso de ese usuario a diferentes recursos.

Dentro de estos JWTs, encontrarás un conjunto de propiedades y valores que proporcionan información más detallada sobre un usuario. Estos datos se denominan claims. Hay un conjunto predeterminado de claims que generalmente se incluyen con cada token para asegurar una representación estandarizada de atributos comunes. **Son estos JWTs los que se convierten en la moneda universal de nuestro modelo de identidad multi-tenant.**

La buena noticia con los JWTs es que permiten la introducción de custom claims. Esto crea la oportunidad de vincular datos contextuales del tenant a estos tokens.

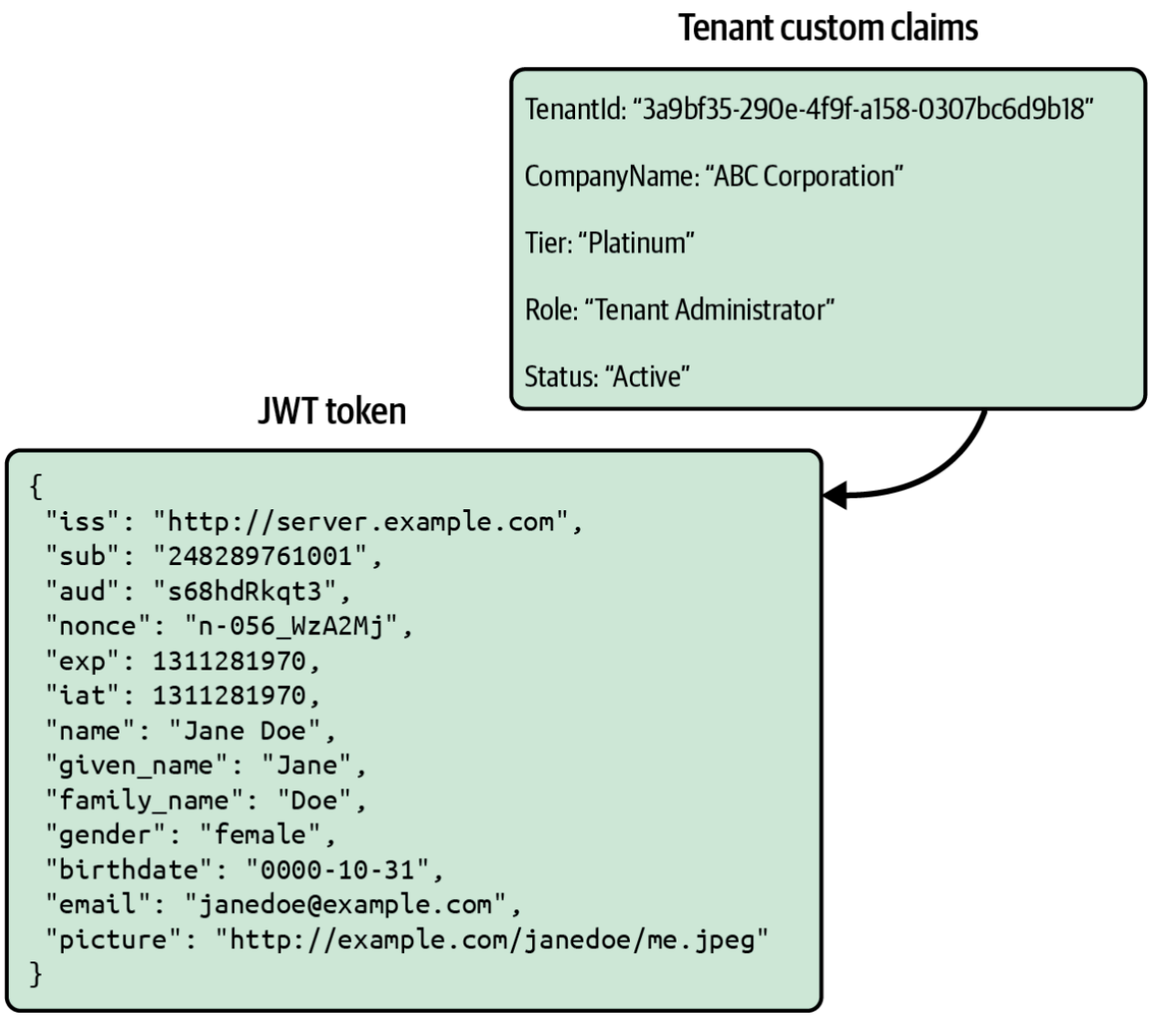

Figure 4-11. Adding tenant custom claims to a JWT

Aunque no hay nada mágico o elegante sobre este modelo, ser capaz de introducir estos custom claims como ciudadanos de primera clase proporciona una ventaja significativa.

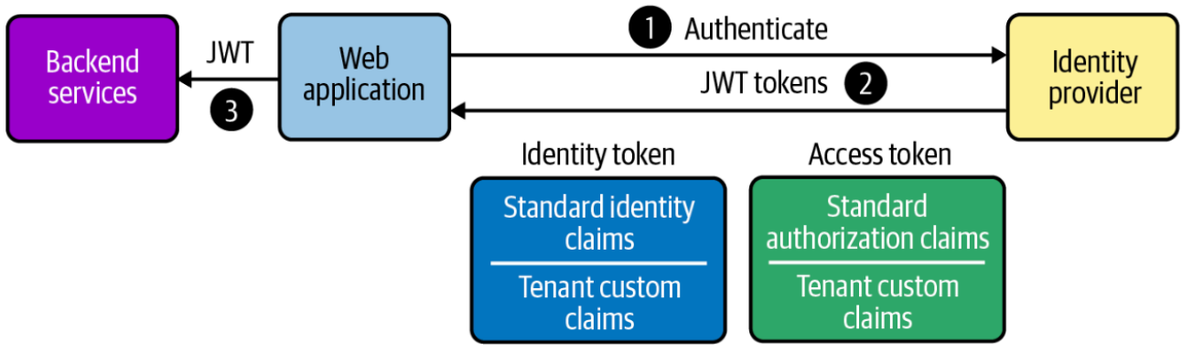

Figure 4-12. Authenticating with embedded tenant context

Ahora, estos tokens pueden ser inyectados como bearer tokens y enviados downstream a tus servicios de backend, heredando toda la seguridad, ciclo de vida y otros mecanismos que están integrados en las especificaciones OIDC y OAuth (paso 3).

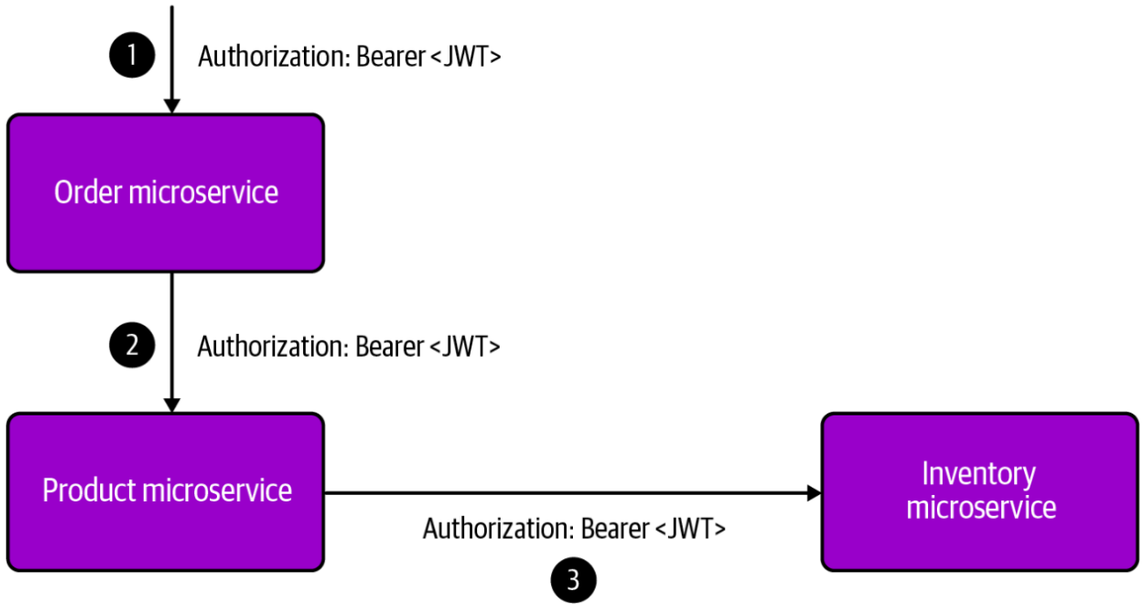

Figure 4-13. Passing tokens to downstream microservices

En este ejemplo, he asumido que puedo insertar el JWT en una solicitud HTTP como un bearer token. Sin embargo, incluso si estás usando otro protocolo aquí, probablemente haya maneras de inyectar este JWT como parte de tu contexto.

Los microservicios lo usarán para logging, métricas, facturación, aislamiento de tenant, particionamiento de datos y una gran cantidad de otras áreas. Tu arquitectura SaaS más amplia lo usará para tiering, throttling, enrutamiento y otros mecanismos globales que requieren contexto del tenant.

Poblamiento de custom claims durante onboarding

Hay dos pasos bastante directos asociados con agregar y poblar estos custom claims.

Primero, antes de incorporar cualquier tenant, típicamente necesitarás configurar tu proveedor de identidad, identificando cada uno de los atributos personalizados que te gustaría agregar a tu experiencia de autenticación. Aquí, definirás cada propiedad y tipo que querrás que termine en tus custom claims. Esto prepara tu proveedor de identidad para aceptar nuevos tenants que puedan almacenar y configurar sus tenants con los atributos adicionales.

La segunda mitad de este proceso se ejecuta durante el onboarding. Anteriormente, discutí la creación del usuario admin del tenant como parte del flujo general de onboarding. Sin embargo, lo que no mencioné fue el poblamiento de los custom claims para tu tenant recién creado. Mientras agregas la información sobre tu usuario (nombre, correo electrónico, etc.), también poblarás todos los campos de contexto del tenant para ese usuario (ID del tenant, rol, tier). Estos datos deben ser poblados para cada usuario dentro del proveedor de identidad, así que incluso después de que el onboarding se haya completado, la introducción de usuarios adicionales debe incluir el poblamiento de estos atributos personalizados.

Uso juicioso de custom claims

Los custom claims son un elemento útil para vincular contexto del tenant a tus tokens. En algunos casos, los equipos se apegan a este mecanismo y expanden su rol, usándolo para capturar y transmitir contexto de seguridad de la aplicación.

Muchas aplicaciones dependen de elementos de control de acceso para habilitar o deshabilitar el acceso a funcionalidad específica de la aplicación. Estos controles deben ser gestionados fuera del alcance de tu proveedor de identidad. **Generalmente, lo vería como un error inflar tus tokens con custom claims que son parte de tu estrategia tradicional de control de acceso de la aplicación.**

Puede haber momentos en que no esté claro si un atributo pertenece en un custom claim o tu modelo de control de acceso de la aplicación. Para mí, esto a menudo se resuelve basándose en el ciclo de vida y el rol del atributo.

Si el atributo tiende a estar evolucionando con la introducción de características, funciones y capacidades de la aplicación, entonces debería ser gestionado más a través de controles de acceso de la aplicación.

Generalmente, los atributos que terminan en tus custom claims es poco probable que estén cambiando a medida que tu aplicación cambia. **El contenido de tus tokens, por ejemplo, no debería estar cambiando semanalmente basándose en la adición de nuevas características o opciones de configuración de la aplicación.**

No usar servicios centralizados para resolver contexto del tenant

Algunos equipos intentan trazar una línea más dura entre la identidad de tenant y la identidad de usuario. En estos entornos, el proveedor de identidad solo se usa para autenticar usuarios.

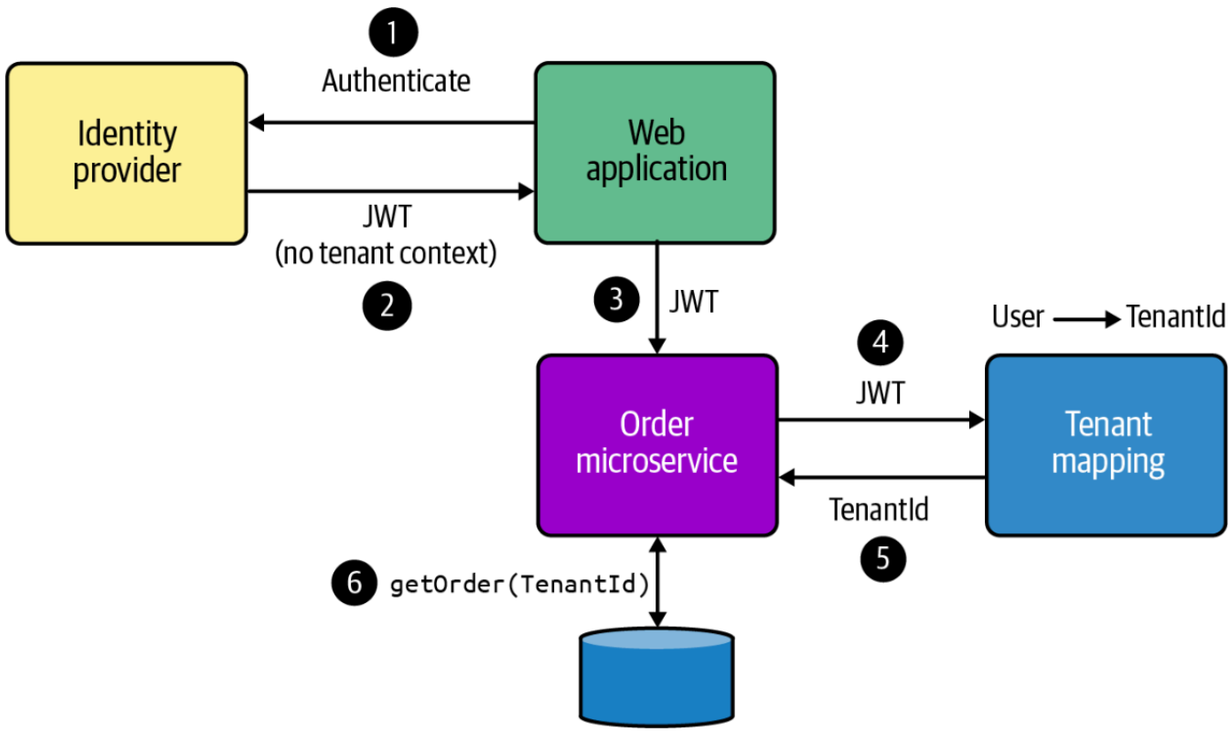

Figure 4-14. Using a separate user/tenant mapping service

En la superficie esto puede parecer una estrategia perfectamente válida. Sin embargo, en realidad presenta desafíos reales para muchos entornos SaaS.

El menor de los problemas aquí es que crea una separación dura entre el usuario y el tenant, requiriendo que los equipos gestionen el estado acoplado del usuario y el tenant de manera independiente.

El problema mayor, sin embargo, es que cada servicio en el sistema debe pasar por este mecanismo de mapeo centralizado para resolver el contexto del tenant.

Imagina este paso siendo realizado a través de cientos de servicios y miles de solicitudes. Muchos que adoptan este enfoque descubren rápidamente que este servicio de Mapeo de Tenant termina creando un cuello de botella significativo en su sistema.

Esta es otra razón por la cual es tan esencial que los contextos de usuario y tenant estén vinculados juntos y compartidos universalmente a través de toda la superficie de tu arquitectura multi-tenant.

Identidad SaaS federada

La mayor parte de lo que he descrito hasta ahora asume que tu sistema SaaS será capaz de ejecutarse con un único proveedor de identidad que esté bajo tu control. Aunque esto representa el escenario ideal y maximiza tus opciones, tampoco es práctico asumir que cada solución SaaS se construye con este modelo. **Algunos proveedores SaaS enfrentan necesidades de negocio, dominio o clientes que requieren que soporten un proveedor de identidad alojado por el cliente o por terceros.**

Un caso común que he visto es un escenario donde un cliente SaaS tiene alguna dependencia empresarial de un proveedor de identidad interno existente. Algunos de estos clientes pueden, como condición de su compra, requerir que un proveedor SaaS soporte autenticación desde estos proveedores de identidad internos.

Típicamente, esto se logra a través de algún nivel adicional de configuración de tenant donde tu onboarding de tenant agregará soporte adicional para configurar este proveedor de identidad alojado externamente.

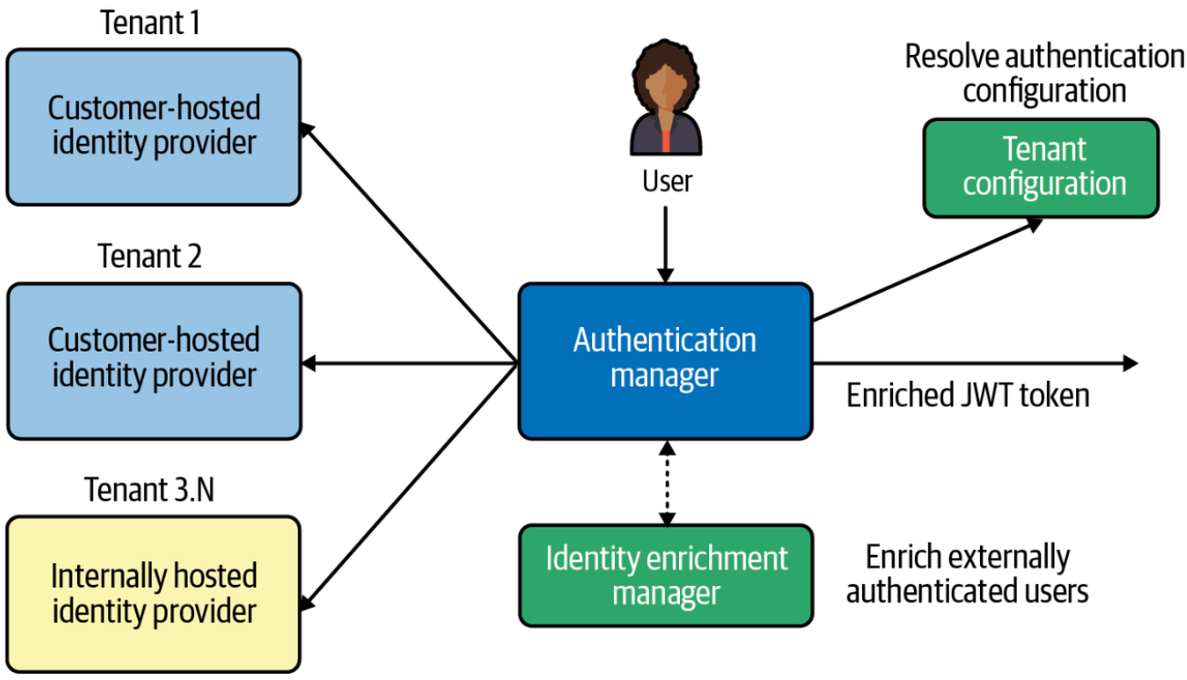

Figure 4-15. Supporting externally hosted identity providers

En el centro de este ejemplo, verás que tengo un authentication manager. Este es un marcador de posición conceptual para introducir algún servicio en tu flujo de autenticación que pueda soportar un conjunto más distribuido de proveedores de identidad.

Este modelo parece bastante directo. Sin embargo, el giro es que tu sistema no tiene control sobre estos proveedores de identidad externos. **Como tal, no puedes configurar los claims de estos proveedores o hacer que tu proceso de onboarding agregue contexto adicional del tenant a los datos de identidad que son gestionados por estos proveedores.**

Esto significa que los JWTs retornados desde tus solicitudes de autenticación no incluirán ninguno del contexto del tenant que es esencial para tu entorno multi-tenant.

Para resolver esto, tu solución necesitará introducir nueva funcionalidad que pueda enriquecer los tokens retornados desde estos proveedores de identidad externos, asumiendo la responsabilidad de enriquecer estos tokens con contexto del tenant que es gestionado dentro de tu entorno SaaS.

Generalmente, sin embargo, los modelos de federación del espacio de identidad a menudo te ofrecen diferentes técnicas para manejar este caso de uso.

He incluido este modelo porque representa un patrón inevitable que aparece en la realidad. Vale la pena notar que hay claras desventajas en este enfoque.

Elementos de agrupamiento/mapeo de tenant

Aunque los proveedores de identidad a menudo se conforman a especificaciones bien establecidas (OIDC, OAuth2), los elementos que se usan para organizar y gestionar identidades pueden variar de un proveedor de identidad a otro.

Estos proveedores ofrecen un rango de diferentes elementos para agrupar y organizar usuarios. Esto es especialmente importante en entornos multi-tenant donde podrías querer agrupar todos los usuarios que pertenecen a un tenant juntos.

En algunos casos, también podrías ser capaz de usar estos grupos para aplicar políticas de tiering a tenants para dar forma a su experiencia de autenticación y autorización.

Si miramos Amazon Cognito, por ejemplo, verás que ofrece múltiples formas de organizar tenants. Cognito introduce la idea de un User Pool. Estos User Pools se usan para contener una colección de usuarios, y pueden ser configurados individualmente, permitiendo que los pools ofrezcan experiencias de autenticación separadas.

La alternativa sería poner todos los tenants en un único User Pool y usar otros mecanismos (grupos, por ejemplo) para asociar usuarios con tenants.

Hay trade-offs que querrás considerar mientras eliges entre estos diferentes elementos de identidad. El número de tenants que tienes, por ejemplo, podría hacer impráctico tener User Pools separados para cada tenant. O puede que no necesites mucha variación entre tenants y prefieras tener todos los tenants configurados y gestionados colectivamente.

La escala, los requisitos de identidad y una serie de otras consideraciones van a dar forma a cómo eliges mapear tenants a cualesquiera elementos que sean soportados por tu proveedor de identidad.

La clave es que cuando comiences a diseñar tu estrategia de identidad SaaS, **querrás identificar las diferentes unidades de organización que pueden ser usadas para agrupar tus tenants** y determinar cómo estas darán forma a la escala, autenticación y configuración de tu experiencia de autenticación multi-tenant.

Con diferentes elementos organizacionales también vienen diferentes opciones de configuración de identidad. Los proveedores de identidad generalmente proporcionan un rango de opciones que pueden ser usadas para configurar tu experiencia de

autenticación.

- Multi-factor authentication (MFA), por ejemplo, se ofrece como una característica de identidad que puede ser habilitada o deshabilitada.
- También puedes configurar requisitos de formato de contraseña y políticas de expiración.

Las configuraciones para estas diferentes opciones de configuración no tienen que ser aplicadas globalmente a todos tus tenants. Puedes querer hacer diferentes características de identidad disponibles para diferentes tiers de tenant.

Compartir IDs de usuario entre tenants

Cada usuario de tu sistema SaaS tiene algún ID de usuario que identifica a ese usuario a un tenant. Este identificador de usuario a menudo está representado por una dirección de correo electrónico.

Hay momentos en que los proveedores SaaS tienen interés en asociar una única dirección de correo electrónico con muchos tenants. Esto, por supuesto, agrega un nivel de indirección a tu autenticación.

En algún lugar de tu flujo de inicio de sesión, tu sistema SaaS necesitará determinar a qué tenant estás accediendo.

La forma más de fuerza bruta que he visto es una que empuja la resolución del tenant al usuario final; durante el inicio de sesión, el sistema detectará que un usuario pertenece a múltiples tenants y solicitará al usuario que seleccione un tenant objetivo.

En el modelo, tendrías una tabla de mapeo que conecte usuarios con tenants y usarías esto como una búsqueda antes de iniciar el flujo de autenticación.

Un enfoque más limpio para esto sería depender de una experiencia de autenticación que proporcione contexto de manera más explícita. El mejor ejemplo probablemente sean dominios y subdominios. Si a cada uno de tus tenants se le asigna un subdominio (`tenant1.saasprovider.com`), tu proceso de autenticación puede usar este subdominio para adquirir el contexto del tenant.

Hay otras complicaciones en este escenario. Imagina, por ejemplo, que todos tus usuarios están ejecutándose en un elemento de proveedor de identidad compartido. En ese modo, el proveedor de identidad va a requerir que cada usuario sea único.

La autenticación de tenant no es aislamiento de tenant

Como parte de esta discusión sobre autenticación y JWTs, a veces encuentro que los equipos equiparan autenticación con aislamiento de tenant.

Esta es definitivamente un área de desconexión. Sí, la autenticación comienza la historia de aislamiento al emitir un JWT con contexto del tenant. Sin embargo, el código en tus microservicios aún puede incluir implementación que—incluso cuando trabaja en nombre de un usuario autenticado—puede acceder a los recursos de otro tenant.

El aislamiento de tenant se construye sobre el contexto del tenant que obtienes de un usuario autenticado, implementando una capa completamente separada de controles y medidas para asegurar que tu código no tenga permitido cruzar un límite de tenant.