SEAMLESS TENANT ONBOARDING USING AWS

REBURA
A Tenth Revolution Group Company

When thinking of moving towards a SaaS delivery mode, a key area to consider is how new tenants will onboard onto the product. Making the tenant onboarding process seamless not only increases customer satisfaction, but can lead to better product adoption, where free tiers / trials exist.

A tenant is the main construct of a SaaS environment and as a SaaS provider building an application, the onboarding needs to be a fluid and streamlined process. The application would often be architected to support a range of market segments, provide separate pricing and experiences to a wide range of customer profiles, often referred to as tiers. These tiering models can influence the cost, operations, management and reliability footprint of a SaaS solution.

In order to introduce new tenants to their environment, it requires an effortless model with multiple components to successfully provision and combine all the elements. During this process it would involve a number of components to successfully provision and configure all the necessary areas to create a new tenant. This will be initiated either directly by the tenants or as part of a provider-managed process.

Having an automated, low-friction onboarding process is key to allowing SaaS providers to have a scalable mechanism that is repeatable for introducing new tenants. Implementing automated scripts that provision all the elements, providing a self-service user experience and single automated process to onboard tenants are all vital to create a fluid onboarding process. In order to do this, we would look at using serverless services within AWS and we can break this down into a few key areas.

1. REGISTRATION

Tenants register themselves using the sign-up web application, the flow of this would vary slightly based on the tier the tenant selects during the sign-up process. Once the tenant provides their details, the registration service would examine this and flag it to be onboarded. The registration would generate a tenant id which will be used in the management and provisioning. The user will send the request with the new tenant payload (tenant name, description and ID) in a JSON format to a REST API hosted by Amazon API Gateway which processes the request and forwards it to the backed Lambda function for tenant on-boarding. The registration service makes use of the API Gateway Resource Policies to authenticate against the User and Tenant Management services. This would then invoke the user management service to create a new tenant user.

2. USER MANAGEMENT

The tenant would be provisioned into the user pool dedicated to the tier selected by the user at registration, for this we would use Amazon Cognito as our identity provider. Other tiers would share a common user pool but are assigned different Cognito groups within that single user pool. The Tenant Id and user role are stored as a custom claim inside Cognito. Using Amazon IAM a role would be used for the Lambda function to on-board the tenant.

SEAMLESS TENANT ONBOARDING USING AWS

REBURA
A Tenth Revolution Group Company

3. TENANT MANAGEMENT

The Tenant Management will look to store the tenant details, which includes the tenant name, generated tenant universally unique identifier (UUID), tenant description and settings inside of the managed service Amazon DynamoDB. Once this is done a DynamoDB stream initiates the downstream Lambda function to build the tenant infrastructure.

4. TENANT PROVISIONING

The last part of the process is the provisioning service which invokes the tenant pipeline to provision the specific infrastructure. The Lambda function acts based on the received stream from the DynamoDB which in this case is an INSERT event. Using the stream's NewImage section, this invokes the CloudFormation to create a new tenant infrastructure using the template stored in an S3 bucket. The CloudFormation creates the tenant infrastructure based on the template and input parameters then kicks off the provisioning of the application services using AWS CodePipeline and AWS CodeBuild to deploy the tenant's resources. The tenant isolation security policies are applied at the network and data level. The AWS CodePipeline is used to manage the deployment of the application services using a CI/CD approach, when the code is merged to the main branch it will trigger automatically to build, source and deploy the services. Each tenant infrastructure setup would include a CloudWatch alarm, billing alarm and an alarm event.

CONCLUSION

Overall, onboarding tenants on AWS can result in a more efficient, cost-effective, and secure environment that benefits both tenants and the hosting organisation. This will enable them to focus on their core competencies and deliver better services to their customers. Using an automated, repeatable process to introduce new tenants into your system as well as including the provisioning of infrastructure, isolation polices, billing and any tenant configuration will reduce friction which promotes operational efficiency and organisational agility.
AWS offers several benefits for businesses and organisations, especially those that provide software-as-a-service (SaaS) or multi-tenant applications.