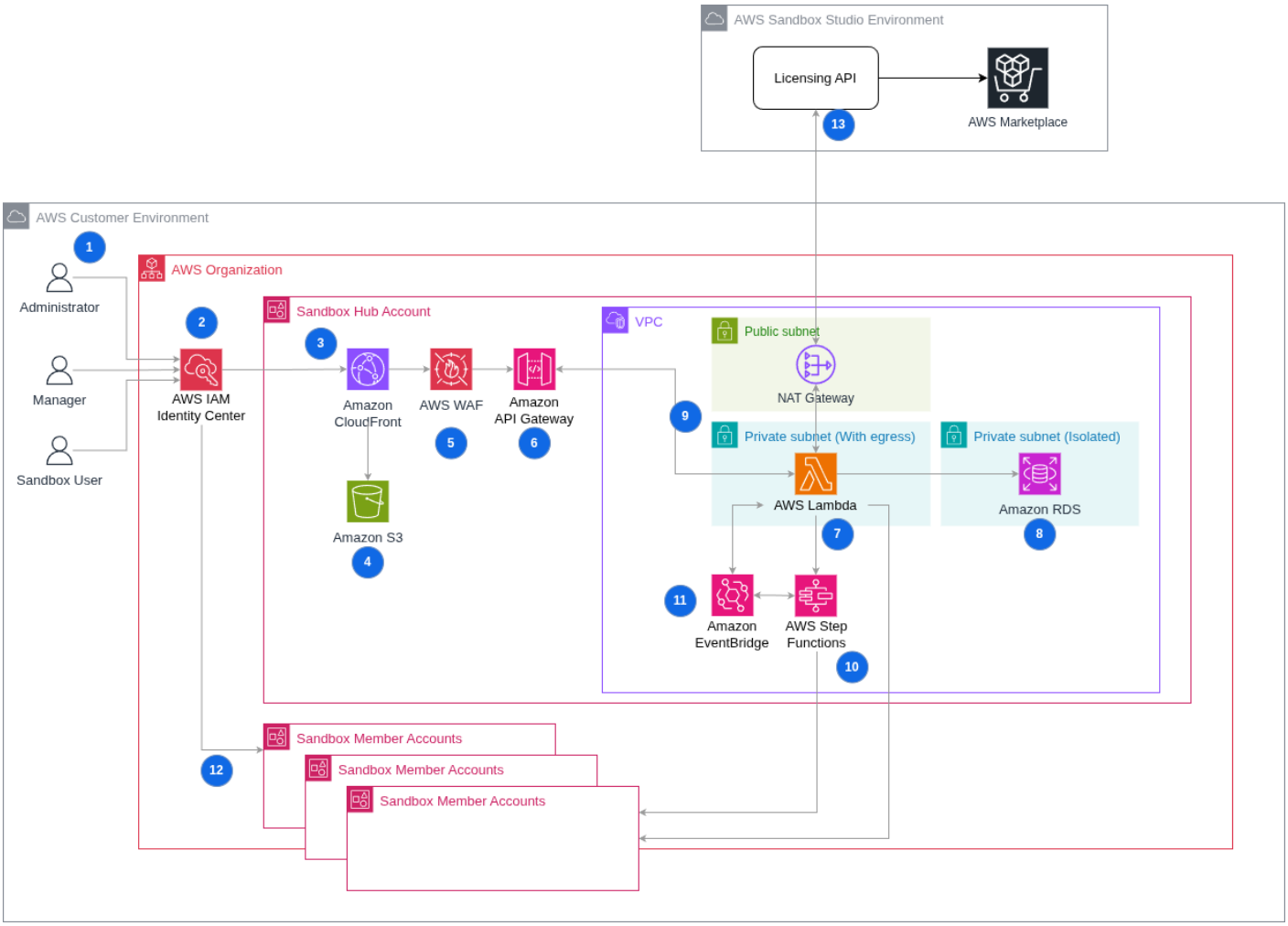
Solution Architecture

Sandbox Studio solution is built entirely on AWS services, with each component playing a specific role in delivering, securing, and managing sandbox environments. The architecture uses managed services to ensure scalability, security, and automation.

The diagram below shows the main components and how they interact. Follow the numbered sections in this guide to understand the purpose and function of each component in the solution.



1. User Roles & Responsibilities

Sandbox Studio supports **three types of users**, each with distinct responsibilities:

1. Administrators

Responsible for configuring and maintaining Sandbox Studio for their organisation.
Key responsibilities include:

- Setting **global policies** such as maximum budget thresholds and cleanup rules.

- Managing **AWS integration**, including permissions and guardrails.
- Provisioning new sandbox accounts when needed.
- Overseeing **security and governance** settings.

2. Managers

Oversee day-to-day sandbox usage within a team or department.
Key responsibilities include:

- Approving or rejecting sandbox requests.
- Assigning account templates to users.
- Tracking **spending** and **activity** for accounts under their supervision.

3. Sandbox Users

Request and use sandbox accounts for **development, testing, training, or experimentation**.
They must operate within:

- Guardrails
- Permissions
- Budget limits

---

2. Authentication and Access

- All users access Sandbox Studio via a **SAML 2.0 application** using **AWS IAM Identity Center**.
- IAM Identity Center can:
  - Use its **own internal user store**, or
  - Integrate with **external identity providers** (e.g., Okta, Microsoft Entra ID).
- Most organisations with an existing **AWS Organisation** use an external provider for centralised identity management.

---

3. Application Entry Point

- The **web UI** is accessed through **Amazon CloudFront**, which serves as a single entry point for:
  - The static web UI (hosted in Amazon S3).
  - API endpoints (via Amazon API Gateway).

---

4. UI Hosting

- **Amazon S3** hosts static assets such as **HTML, CSS, and JavaScript** files.

---

5. API Protection

- **AWS WAF** protects API Gateway from common exploits, bots, and resource abuse.

---

6. API Gateway

- The web UI communicates with **Amazon API Gateway REST APIs** to:
  - Fetch data
  - Update configuration and status information
- **AWS Lambda functions** authorize requests using **role-based access control** based on IAM Identity Center groups.

---

7. Backend

AWS Lambda is used throughout Sandbox Studio to run backend logic, including:

- **Authorizing API requests** based on group memberships.
- **Reading and writing data** to a database.
- **Monitoring account leases** for budget or duration threshold breaches.
- **Invoking lifecycle actions** such as account cleanup, OU movement, and permission updates.

---

8. Database

- **AWS Lambda** functions read and write configuration and status data to a **PostgreSQL** database deployed using **Amazon Relational Database Service (RDS)**.
- The RDS database runs **inside a VPC** in the **sandbox hub account**.

---

9. Networking

The Amazon **Virtual Private Cloud (VPC)** hosts the PostgreSQL RDS database used by Sandbox Studio.
Key characteristics include:

- **Private subnets** for hosting the RDS database securely.
- **VPC-enabled Lambda functions** to allow direct database access.
- **Network isolation** from other AWS resources to protect sensitive configuration and status data.

---

10. Account Lifecycle Management

- AWS Step Functions coordinate the lifecycle of sandbox accounts, including:
  - Onboarding new accounts
  - Terminating leases
  - Cleaning up accounts for reuse

- Step Functions move accounts between **Organizational Units (OUs)** based on their current status.
- Onboarding or termination events trigger dedicated cleanup workflows. These workflows can invoke other AWS services, such as **AWS CodeBuild**, to run resource deletion tools like **AWS Nuke**, ensuring all user-created resources are removed before the account is returned to the available pool.

---

11. Event-Driven Automation

- **Amazon EventBridge** routes lifecycle events such as
  - **Lease budget breaches**
  - **Lease duration breaches**

- When triggered, these events can:
  - Send email notifications
  - Invoke Lambda functions and Step Functions to manage lifecycle actions

---

12. Sandbox Account Access

- Users can access assigned AWS sandbox accounts via:
  - **AWS IAM Identity Center Access Portal** (console access)
  - **Programmatic access** using generated credentials
- The Sandbox Studio web UI provides **SSO links** for direct AWS console login.

---

13. Licensing Server

- Sandbox Studio regularly queries the Sandbox Studio Software Licensing Service to confirm the customer's entitlement. That service will also query the AWS Marketplace.

Note: a number of other supporting AWS services are used by Sandbox Studio. Please see AWS services in this solution for the full list.

Revision #17
Created 2025-07-14 21:13:04 UTC
Updated 2025-10-21 06:58:48 UTC by Paul