Aembit Server Workload Cookbooks: *AI Edition*

How to Securely Configure Workload Access for OpenAI, Claude, and Gemini

Gartner COOL VENDOR 2022

CSO Cybersecurity startups to watch for in 2023

2024 WINNER CYBER SECURITY EXCELLENCE AWARDS

RSAC Innovation Sandbox 2024 FINALIST

2024 SINET16 INNOVATOR AWARD

aembit

Secure Workload Access to Large Language Models (LLMs): Implementation Best Practices

The explosion in AI adoption is reshaping how software operates — not just how it's built. 78% of organizations now use AI in at least one business function, up from 55% in 2023, according to recent research, with generative AI driving automation, predictive intelligence, and operational efficiency across industries. Developers are increasingly embedding foundation models into backend logic, CI/CD pipelines, internal tools, and orchestration frameworks — transforming not only customer-facing applications but also core infrastructure.

Non-human identities – not users – now drive most digital interactions. CI/CD pipelines, service meshes, backend services, and AI systems all require secure machine-to-machine authentication for tasks like deploying software, managing microservices, accessing data, and training models.

That creates a different kind of access challenge.

These non-human actors need to authenticate to powerful external APIs like OpenAI, Claude, and Gemini. But the dominant method remains static and long-lived API keys — stored in plaintext files, injected via environment variables, or passed around from team to team. There's rarely visibility into which workloads are using which keys. Rotation is manual. Enforcement of least privilege is virtually impossible.

As AI adoption deepens – and the number of services calling AI endpoints accelerates – these practices won't scale. They increase the attack surface, introduce operational drag, and create brittle security boundaries across environments.

That's where our series of Aembit Server Workload Cookbooks come in.

Each "recipe" guide offers a step-by-step, production-ready configuration for securely connecting workloads to a specific service. They were built with the Aembit Workload IAM Platform in mind, but their value extends well beyond it. They are useful even if you're managing access another way — with cloud-native IAM, your own secrets engine, or custom scripts.

They cover a wide range of credential types, based on what each AI service supports. While some providers have adopted more advanced mechanisms like personal access tokens or OIDC, others – including those in this set – still rely on API keys. That's not a recommendation, but a reflection of reality. In those cases, the focus shifts to managing what's available as securely as possible: limiting exposure, enforcing policy, and maintaining visibility into which workload is using which key.

Even if you're not using Aembit, here's what you'll still walk away with:

- **Reference architectures for workload-to-AI authentication** — including how to pass credentials via headers or bearer tokens, and how to avoid storing them in plaintext.
- **Configuration patterns you can adapt** — with clear guidance on API endpoint structure, supported authentication schemes, and how to apply least privilege.
- **Real-world caveats** — such as how OpenAI, Claude, and Gemini SDKs behave when credentials are injected at runtime, and what to watch for in production.
- **Security principles that scale** — including how to shift from long-lived credentials to workload identity, how to separate access by environment, and how to apply just-in-time credential injection.
- **Design patterns for any platform** — whether you're building your own credential lifecycle tooling or evaluating options like Vault, AWS IAM, or service meshes.

These guides are relevant for DevSecOps teams, platform engineers, and security architects tasked with securing access between internal workloads and external AI services. The focus here is not on abstract principles or product marketing, but on practical implementation: how to apply access policies, manage credentials securely, and integrate AI without introducing unnecessary risk.

As more workloads interact with AI models directly — often in real time and without human intervention — access control becomes a foundational part of your infrastructure. These guides offer a repeatable model for getting that control right.

This AI edition is just the beginning. In the coming months, this series will expand to cover CI/CD pipelines, cloud platforms and services (PaaS), data and analytics tools, databases, financial services APIs, productivity and collaboration suites, and more — each with clear guidance and Aembit-specific configurations where relevant.

Now, let's get cooking!

1. OpenAI

Overview

OpenAI is an artificial intelligence platform that allows developers to integrate advanced language models into their applications. It supports diverse tasks such as text completion, summarization, and sentiment analysis, enhancing software functionality and user experience.

Below you can find the Aembit configuration required to work with the OpenAI service as a Server Workload using the OpenAI API.

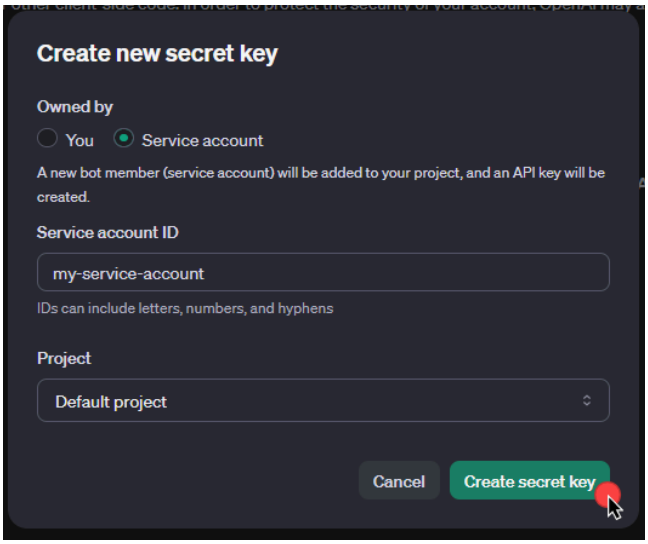
Prerequisites

Before proceeding with the configuration, ensure you have an OpenAI account and API key. If you have not already generated a key, follow the instructions below. For more details on API key authentication, refer to the official OpenAI API documentation.

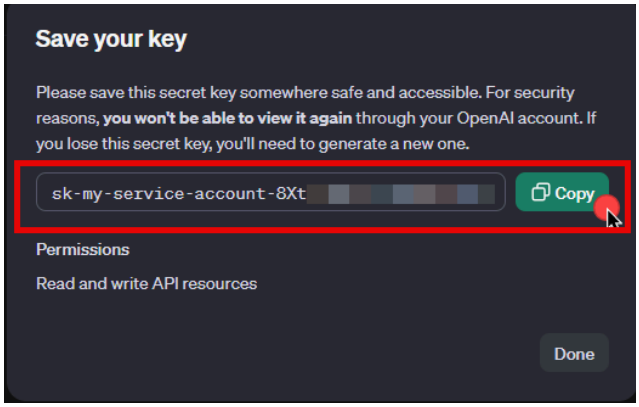
Create Project API Key

1. Sign in to your OpenAI account.
2. Navigate to the API Keys page from the left menu.
3. Click on Create new secret key button in the middle of the page.
4. A pop-up window will appear. Choose the owner and project (if not multiple, the default project is selected). Then, fill in either the optional name field or service account ID, depending on the owner selection.
    - If the owner is selected as You, under the Permissions section, select the permissions (scopes) for your application according to your needs.

- Click on Create secret key to proceed.



5. Click Copy and securely store the key for later use in the configuration on the tenant.



Note

The following configuration steps will also work with user API keys; however, project API keys are recommended as they offer more granular control over the resources.

Server Workload Configuration

1. Create a new Server Workload.
    - Name - Choose a user-friendly name.
2. Configure the service endpoint:
    - Host - api.openai.com
    - Application Protocol - HTTP
    - Port - 443 with TLS
    - Forward to Port - 443 with TLS
    - Authentication method - HTTP Authentication
    - Authentication scheme - Bearer

Credential Provider Configuration

1. Create a new Credential Provider.
    - Name - Choose a user-friendly name.
    - Credential Type - API Key
    - API Key - Paste the key copied from OpenAI Platform.

Client Workload Configuration

Aembit now handles the credentials required to access the Server Workload, eliminating the need for you to manage them directly. You can safely remove any previously used credentials from the Client Workload.

If you access the Server Workload through an SDK or library, it is possible that the SDK/library may still require credentials to be present for initialization purposes. In this scenario, you can provide placeholder credentials. Aembit will overwrite these placeholder credentials with the appropriate ones during the access process.

Access Policy

Create an Access Policy for a Client Workload to access the OpenAI Server Workload. Assign the newly created Credential Provider to this Access Policy.

Required Features

You will need to configure the TLS Decrypt feature to work with the OpenAI API Server Workload.

2. Claude

Overview

Claude is an artificial intelligence platform from Anthropic that allows developers to embed advanced language models into their applications. It supports tasks like natural language understanding and conversation generation, enhancing software functionality and user experience.

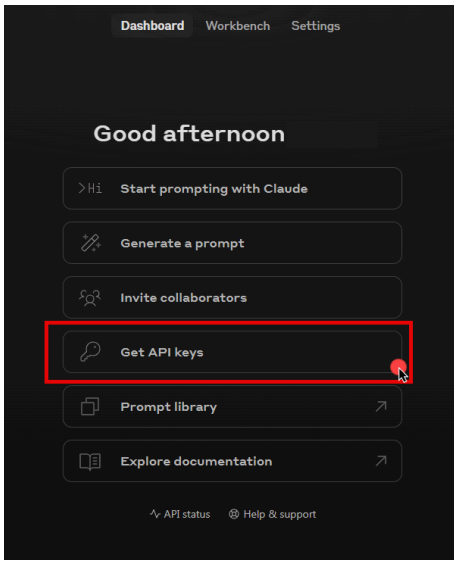
Below you can find the Aembit configuration required to work with the Claude service as a Server Workload using the Claude API and Anthropic's Client SDKs.

Prerequisites

Before proceeding with the configuration, ensure you have an Anthropic account and API key. If you have not already generated a key, follow the instructions below. For more details about Claude API, refer to the official Claude API documentation.

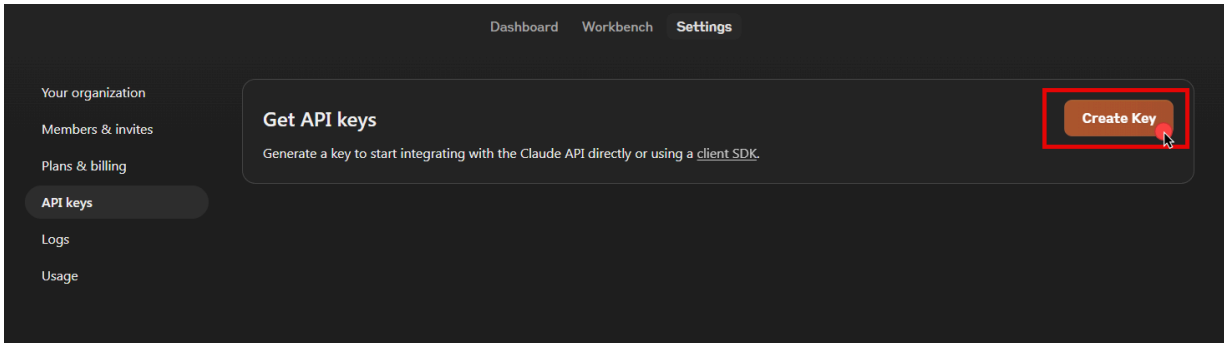
Create Project API Key

1. Sign in to your Anthropic account.
2. Navigate to the API Keys page by clicking the Get API Keys button from the dashboard menu.

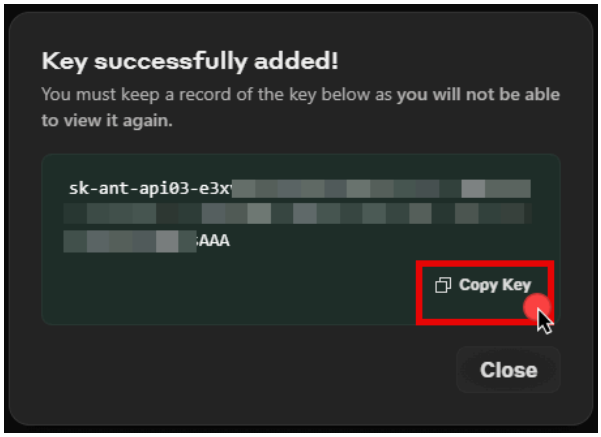


3. Click the Create key button in the top right corner of the page.

4. A pop-up window will appear. Fill in the name field, then click Create Key to proceed.



5. Click Copy and securely store the key for later use in the configuration on the tenant.



Server Workload Configuration

1. Create a new Server Workload.
    - Name - Choose a user-friendly name.
2. Configure the service endpoint:
    - Host - api.anthropic.com
    - Application Protocol - HTTP
    - Port - 443 with TLS
    - Forward to Port - 443 with TLS
    - Authentication method - HTTP Authentication
    - Authentication scheme - Header
    - Header - x-api-key

Credential Provider Configuration

1. Create a new Credential Provider.
    - Name - Choose a user-friendly name.
    - Credential Type - API Key
    - API Key - Paste the key copied from Anthropic Console.

Client Workload Configuration

Aembit now handles the credentials required to access the Server Workload, eliminating the need for you to manage them directly. You can safely remove any previously used credentials from the Client Workload.

If you access the Server Workload through an SDK or library, it is possible that the SDK/library may still require credentials to be present for initialization purposes. In this scenario, you can provide placeholder credentials. Aembit will overwrite these placeholder credentials with the appropriate ones during the access process.

Access Policy

Create an Access Policy for a Client Workload to access the Claude API Server Workload. Assign the newly created Credential Provider to this Access Policy.

Required Features

You will need to configure the TLS Decrypt feature to work with the Claude API Server Workload.

Note

If you are using the SDK, you will need to configure the SSL_CERT_FILE environment variable and point it to a file containing the tenant root CA. The specific commands may vary depending on how your application is launched. Below command lines are examples for the Python SDK:

```
Unset

wget https://<your_tenant_ID>.aembit.io/api/v1/root-ca -O
tenant.crt
SSL_CERT_FILE=./tenant.crt python3 ./your_app.py
```

3. Gemini

Overview

Gemini is an AI platform that allows developers to integrate multimodal capabilities into their applications, including text, images, audio, and video processing. It supports tasks such as natural language processing, content generation, and data analysis.

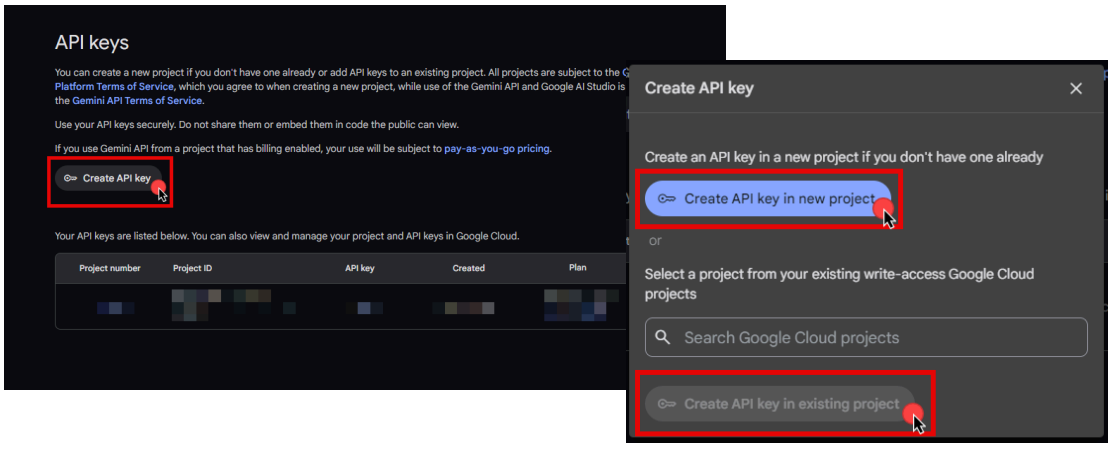
Below you can find the Aembit configuration required to work with the Google Gemini service as a Server Workload using the REST API.

Prerequisites

Before proceeding with the configuration, ensure you have a Google account and an API key. If you have not already created a key, follow the instructions below. For more details about the Gemini API, refer to the official Gemini API documentation.

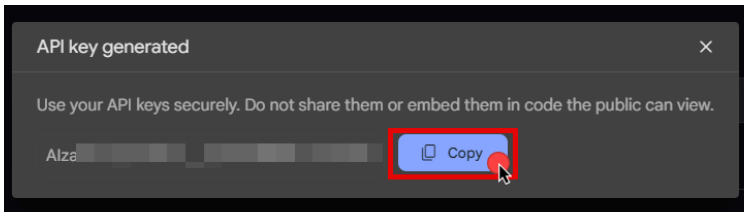
Create API Key

1. Navigate to the API Keys page and sign in to your Google account.
2. Click the Create API key button in the middle of the page.



3. Click the Got it button on the Safety Setting Reminder pop-up window.

4. If you do not already have a project in Google Cloud, click Create API key in new project. Otherwise, select from your projects and click Create API key in existing project.
5. Click Copy and securely store the key for later use in your tenant configuration.



Server Workload Configuration

1. Create a new Server Workload.
   - Name - Choose a user-friendly name.
2. Configure the service endpoint:
   - Host - generativelanguage.googleapis.com
   - Application Protocol - HTTP
   - Port - 443 with TLS
   - Forward to Port - 443 with TLS
   - Authentication method - HTTP Authentication
   - Authentication scheme - Header
   - Header - x-goog-api-key

Credential Provider Configuration

1. Create a new Credential Provider.
   - Name - Choose a user-friendly name.
   - Credential Type - API Key
   - API Key - Paste the key copied from Google AI Studio.

Client Workload Configuration

Aembit now handles the credentials required to access the Server Workload, eliminating the need for you to manage them directly. You can safely remove any previously used credentials from the Client Workload.

If you access the Server Workload through an SDK or library, it is possible that the SDK/library may still require credentials to be present for initialization purposes. In this scenario, you can provide placeholder credentials. Aembit will overwrite these placeholder credentials with the appropriate ones during the access process.

Access Policy

Create an Access Policy for a Client Workload to access the Gemini Server Workload. Assign the newly created Credential Provider to this Access Policy.

Required Features

You will need to configure the TLS Decrypt feature to work with the Gemini Server Workload.

Conclusion

With AI services becoming embedded into the core of application workflows, the need for secure, scalable workload access is only increasing. Whether you're just starting to formalize how machine identities authenticate to external APIs or already managing workload identity at scale, these cookbooks are meant to help you take the next step – with actionable patterns and reusable configurations.

If you'd like to see how Aembit can help automate and enforce these practices across your environment, you can request a demo here. We're happy to show how it works in practice and how you can integrate it with your existing infrastructure.

aembit

Gartner COOL VENDOR 2022 · CSO Cybersecurity startups to watch for in 2023 · 2024 WINNER CYBER SECURITY EXCELLENCE AWARDS · RSAC Innovation Sandbox 2024 FINALIST · 2024 SINET16 INNOVATOR AWARD