Introduction and Best Practices to AI Gateways

Author: Alex Ly

Introduction

Artificial Intelligence (AI) is transforming how consumers and organizations interact. The rise of Generative AI (GenAI) is driving automation, boosting productivity, and transforming industries, all while sharpening the focus on customer experience.

Adopting GenAI platforms, such as large language models (LLMs), offers significant advantages: faster application development, quicker product iterations, and higher employee productivity. Helping drive rapidly improved customer experiences and creation of new revenue opportunities through improvements in operational efficiency and application performance.

GenAI APIs play a critical role in connecting AI capabilities to applications, making AI platforms accessible to enterprise technology teams. With various LLM models and providers available, APIs allow enterprises to integrate advanced AI functions—such as predictive analytics and language processing—without requiring specialized AI expertise. This democratizes AI access, enabling faster innovation and more personalized, data-driven services and experiences.

However, as AI adoption scales, organizations must tackle the challenges related to AI workloads, including sustainability, safety, and often hidden and shocking costs of resources. Implementing robust guardrails is crucial to ensuring the performance, reliability, and security of applications as they integrate with AI models. In this eBook, we'll explore the best practices required to build safe and performant AI applications and the role of AI Gateway in managing GenAI APIs.

What is an AI Gateway?

An AI Gateway functions similarly to an API Gateway, serving as a dedicated endpoint in your AI infrastructure for connecting, securing, and monitoring AI traffic from various applications to backend AI model endpoints, such as large language models (LLMs), and vision, image, audio, or video models.

As the industry rapidly builds new applications that rely on various third-party AI models, the need for a solution to expose these AI services for public consumption has grown. The emergence of the AI Gateway category highlights the necessity for features tailored to AI developers—features that may not be available in traditional API gateway solutions.

At Solo.io, we have been developing Gloo AI Gateway, an extension of our leading API gateway Gloo Gateway. Designed from the ground up to be cloud-native and Kubernetes-native, Gloo AI Gateway is well-equipped to support your AI strategy as an integral part of your overall API strategy, rather than as a separate silo within the organization.

Key Features of AI Gateways

For enterprises, AI APIs, such as LLM APIs, must meet stringent requirements to prevent unauthorized access and ensure the integrity and confidentiality of processed data. Key features of AI Gateways to consider include:

- **Unified Access Point:** Simplify access to backend LLM APIs or other approved GenAI models through a centralized point, streamlining the management and utilization of various AI services.
- **Authentication and Authorization:** Integrate with existing access control mechanisms like API keys, OAuth, and JWTs. Use advanced authentication and authorization strategies to control access to AI models, ensuring secure and compliant usage.
- **Credential Management:** Improve developer productivity by shifting key management (tracking, revocation, refresh) to the gateway, reducing API key sprawl and centralizing credential management within the AI infrastructure.

- **Consumption Control:** Implement rate limiting for public LLM requests to avoid excessive charges. Set provider- and client-specific consumption limits to manage AI usage effectively.
- **Observability:** Offer developers insights into token usage, quotas, error rates, and other metrics. Track usage by client across multiple LLM providers with access logging for cost control and chargeback, enhancing visibility into AI traffic.
- **Enrichment:** Enrich requests with additional headers for reporting and tracking, or modify the request body to add context, screen unwanted text, or reject inappropriate or sensitive content.
- **Canonical LLM API Definition:** Create a client-facing LLM API definition that maps to multiple providers. The gateway transforms provider-specific requests and responses into a canonical model, streamlining AI application development.

How Does an AI Gateway Work?

An AI Gateway such as Gloo AI Gateway, can be configured as an additional endpoint of an existing gateway-proxy or even as a dedicated gateway-proxy endpoint within your AI infrastructure depending on organizational requirements. Developers interact with the endpoints exposed by the AI Gateway, while internal platform, gateway, and security teams can configure and manage policies using modern principles and declarative configuration.

To expand on this, let's break down the workflow and roles involved in utilizing an AI Gateway:

1

Integration capabilities with multiple providers

A performant AI Gateway should integrate with multiple AI service providers, such as OpenAI, Cohere, Anthropic, and Mistral AI. This flexibility allows organizations to leverage the best AI models available and switch providers without disrupting their applications.

2

Unified endpoint from the AI Gateway

By providing a unified access point, the AI Gateway simplifies the management of multiple AI services and models. This centralization improves efficiency and reduces the complexity of managing multiple third-party endpoints and services.

3

Configuration and deployment capabilities

Developers working on AI applications need to interact with various AI models by connecting their applications to the endpoints exposed by the AI Gateway. The gateway or platform team is responsible for setting up routing configurations, ensuring that requests are properly routed to backend LLM APIs or other generative AI models.

4

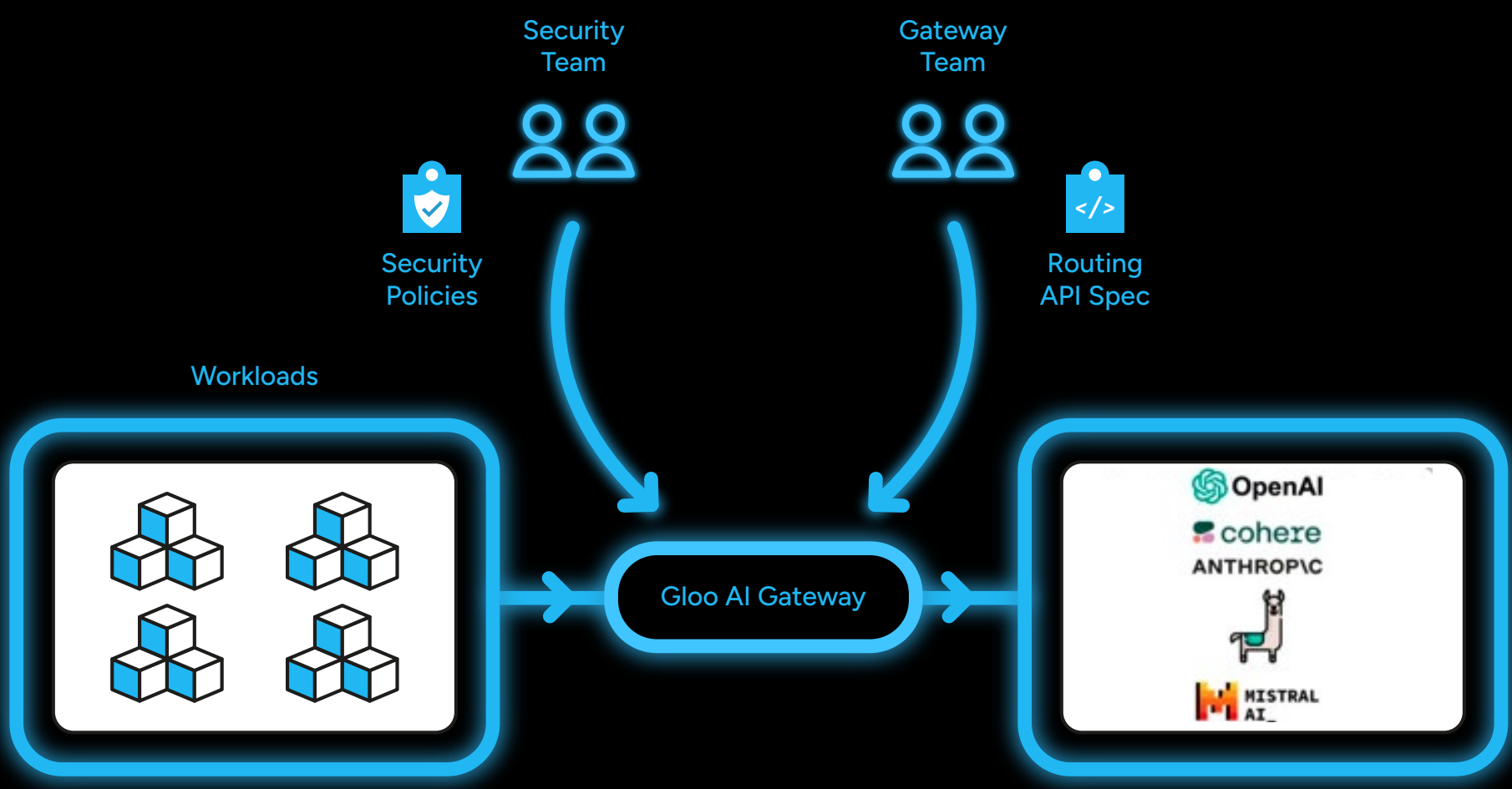
Management of security processes

The security team configures and enforces security policies to prevent unauthorized access and ensure data integrity and confidentiality. They use modern authentication and authorization mechanisms, such as API keys, OAuth, and JWTs, to secure the AI Gateway.

5

Observability and management capabilities

The AI Gateway provides observability features, allowing development teams to monitor AI traffic, usage patterns, and potential issues. This helps the platform team in managing AI consumption, optimizing performance, and controlling costs.



Examples of AI Gateway Challenges and Solutions

Unified access point for LLM API consumption

Organizations face complex challenges in managing LLM APIs due to multiple internal consumers, diverse providers, and both remote and local LLM deployments. This creates a multidimensional scaling problem for access, control, visibility, and governance.

The AI Gateway acts as a centralized access point and offers a single endpoint regardless of LLM locality. This approach simplifies access for consumers while enabling centralized control and oversight.

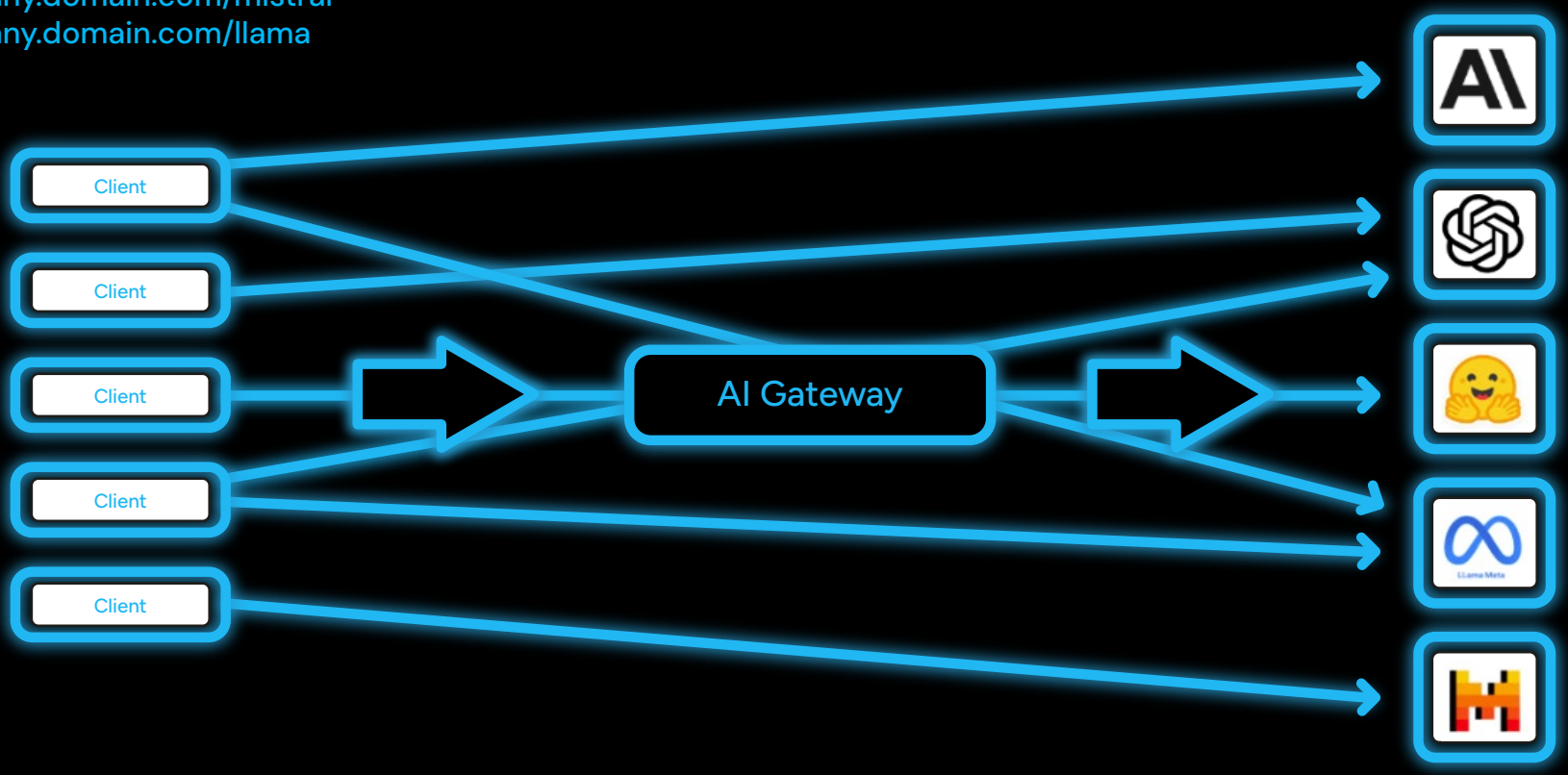
Challenge

- Many internal consumers of LLM APIs
- Many providers of LLM APIs
- LLMs will be remote and local
- Multidimensional scaling problem for access, control, visibility and governance

Solution

- Provide and API proxy to serve as access point for LLM APIs
- Single endpoint regardless of LLM locality
- Simplify LLM access for consumers
- Centralized control, visibilty, and governance

mycompany.domain.com/openai
mycompany.domain.com/gemini
mycompany.domain.com/mistral
mycompany.domain.com/llama



Credential Management

Credential management for LLM APIs presents significant challenges, including individually managed API keys and client identities for each provider, distributed key management across providers and clients, complex identity and access management integration, and DIY solutions for local LLMs.

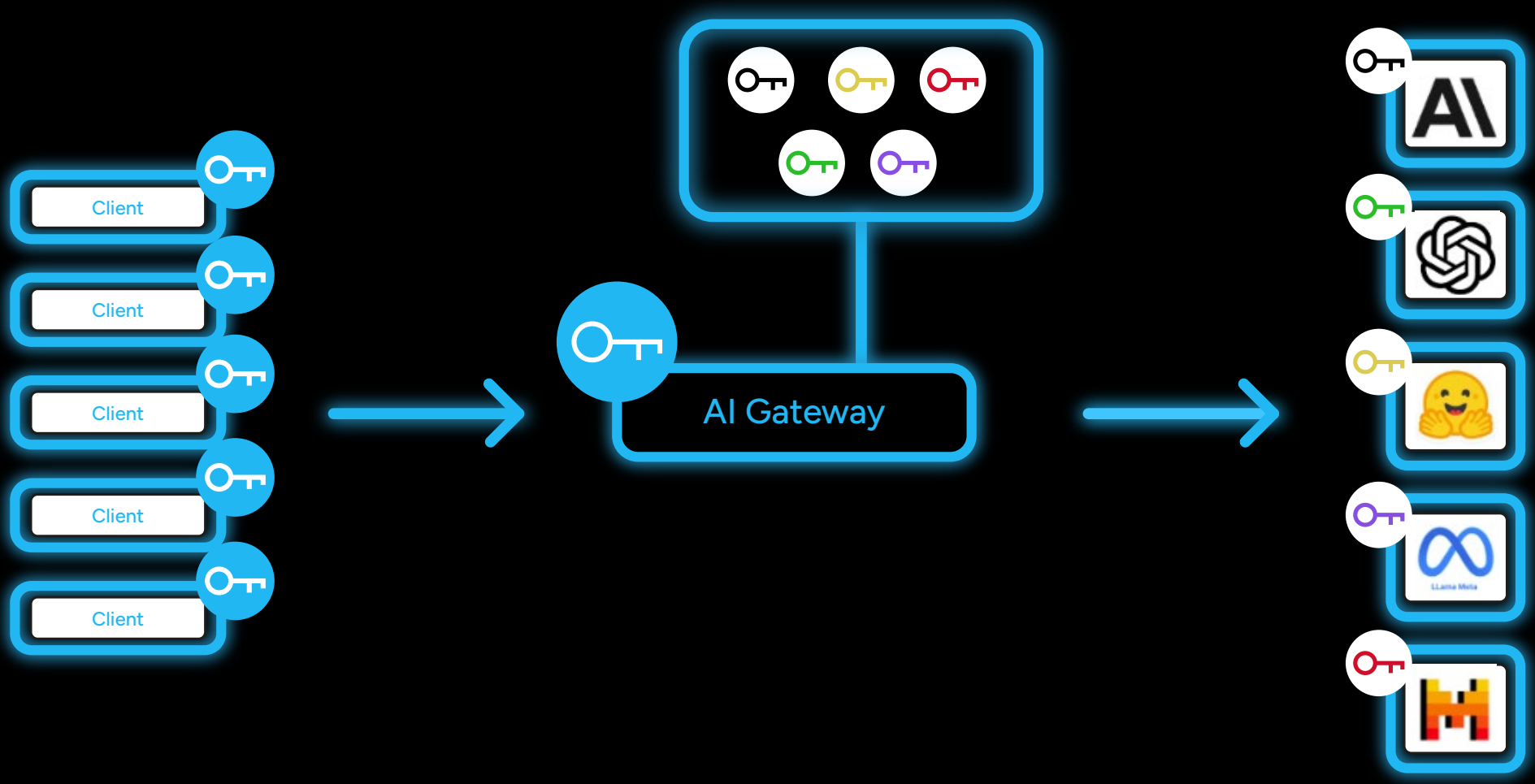
A centralized AI gateway manages LLM provider keys in a protected store and creates API keys that can map to multiple providers, simplifying client development. It leverages advanced authentication and authorization mechanisms like JWT and OPA for access control. By centralizing key management, the AI gateway streamlines tracking, revocation, and refresh processes, providing a unified and secure approach to credential management across diverse LLM backends.

Challenge

- API keys and client identity managed individually for each LLM provider and passed differently
- Key management (tracking, revocation, refresh) distributed across all providers and clients
- Identity and access management integration with each LLM provider
- DIY problem for local LLMs

Solution

- Gateway manages LLM provider keys in protected store
- API Keys created by gateway can map to multiple providers, simplifying client development across LLMs
- Leverage advanced authN/Z in gateway for controlling access via JWT, OPA, etc.
- Centralized point of key management (tracking, revocation, refresh)



Consumption Control and Visibility

Managing LLM API consumption poses challenges that include the risk of uncontrolled spending on hosted LLMs, resource limitations for local LLMs, the necessity for equitable access distribution, and the complexity of monitoring client-specific usage across shared credentials and multiple providers.

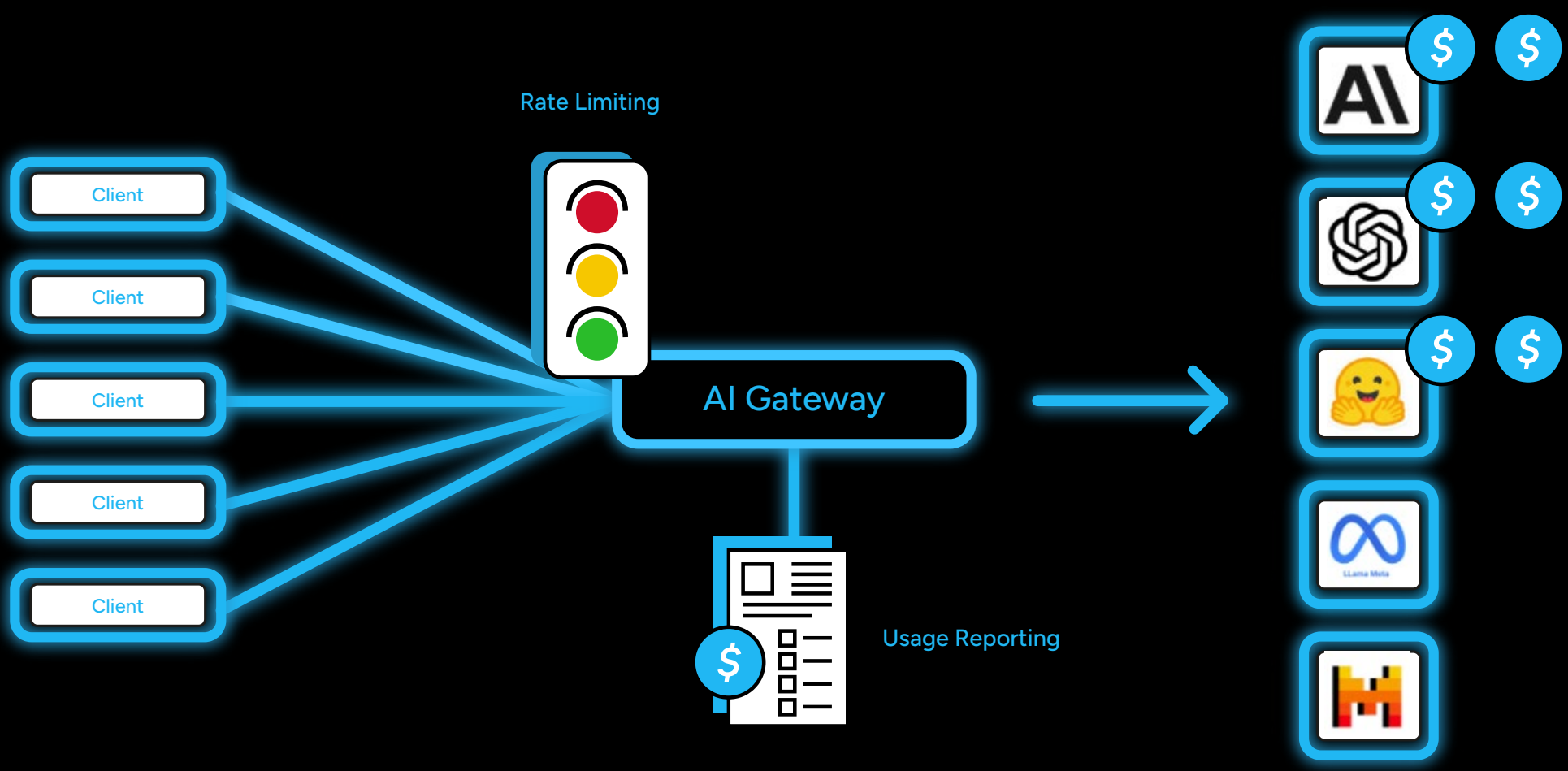
To tackle these issues, the AI Gateway implements API request rate limiting and establishes customizable consumption thresholds for both providers and clients, such as maximum token allocations. Furthermore, it includes comprehensive access logging to monitor usage patterns across various LLM providers on a per-client basis. This approach enables precise cost management, supports chargeback procedures, and ensures fair resource distribution. By offering detailed insights and control over LLM API usage, the AI Gateway helps organizations optimize their resources, prevent budget excesses, and maintain balanced access for all users.

Challenge

- Hosted LLMs are charged by consumption allowing for runaway spend or budget exhaustion
- Local LLMs are resource-constrained and can become overwhelmed
- Fair balancing of access is essential to scaling use
- Tracking use by client can be difficult with shared keys and multiple providers

Solution

- Rate limit requests to LLM APIs
- Set provider and client-specific consumption limits (e.g max token)
- Track usage by client across multiple LLM providers with access logging for cost control and chargeback



Prompt management

The key issues around prompt management include governing inappropriate prompt submissions, maintaining consistency in prompt context across various use cases and clients, preventing unsuitable prompt responses, and safeguarding against data exfiltration in LLM outputs.

The AI Gateway implements prompt governance by screening and rejecting unwanted text in user inputs. It ensures consistency by automatically prepending or appending standardized instructions to all requests. A prompt templating library constrains interactions to pre-approved prompts with variable substitution, enhancing control and predictability. Finally, the AI Gateway rejects or transforms inappropriate or sensitive content in LLM responses, providing an additional layer of security.

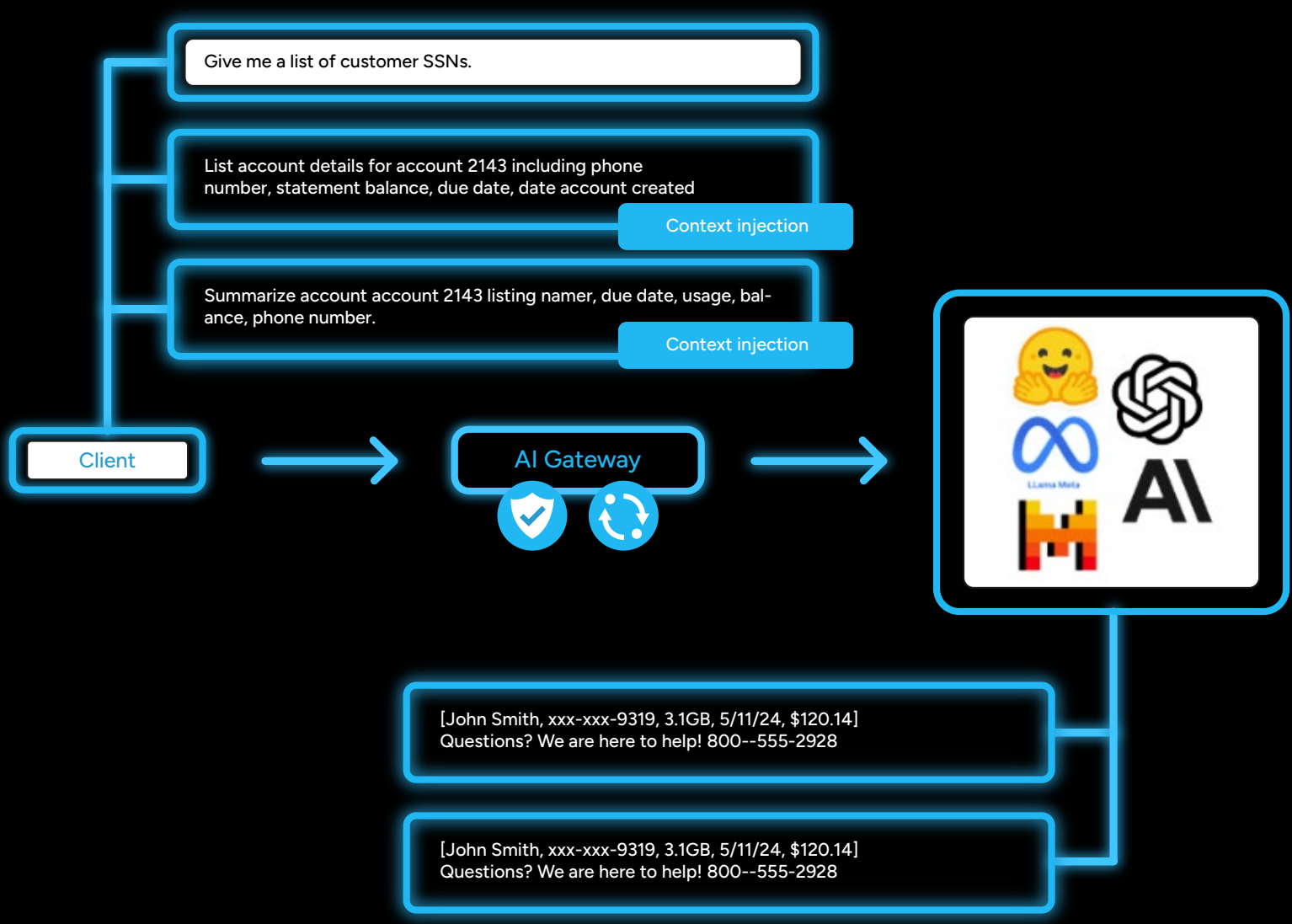
Using an AI Gateway, organizations can significantly improve the safety, consistency, and compliance of their LLM interactions while maintaining flexibility for diverse use cases.

Challenge

- Governing use of inappropriate prompt submissions
- Establishing consistency in prompt context across use cases and clients
- Preventing inappropriate prompt responses
- Preventing inappropriate data exfiltration in prompt responses

Solution

- Prompt governance - screen for unwanted text in prompts and reject
- Prompt context - prepend or append additional prompt instructions for consistency across requests
- Prompt templating library - constrain prompt interface to know prompts that accept variable subsitution
- Reject or transform inappropriate or sensitive response content



Model performance

One of the challenges with all LLM providers is keeping models informed with current, accurate information without constant retraining. The scenarios also call for incorporating client-specific and domain-specific context into prompts, reducing the risk of generalizations and hallucinations, and optimizing response times for similar queries.

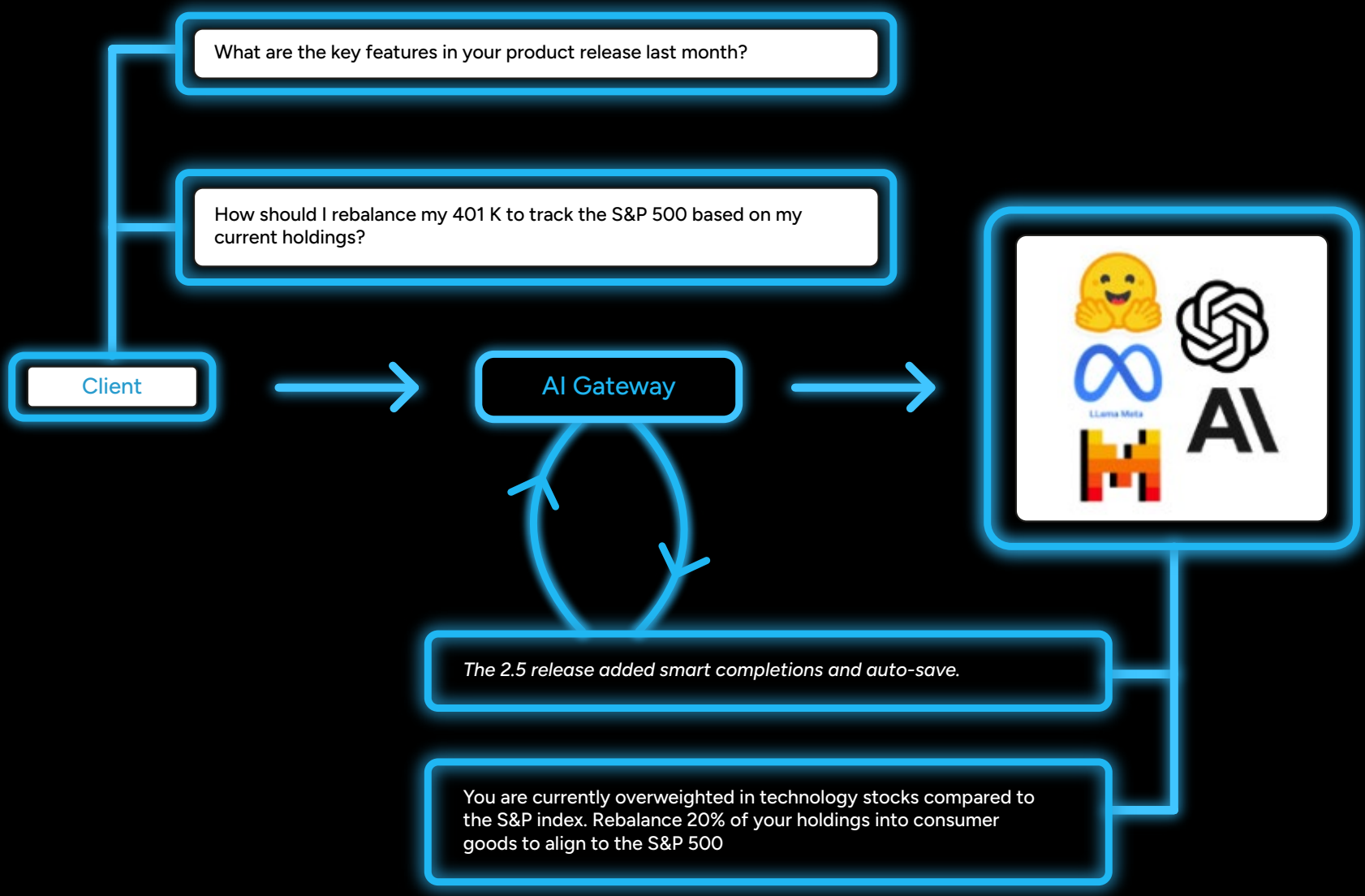
AI Gateway leverages Retrieval Augmented Generation (RAG) to enrich prompts with relevant contextual information, significantly improving response accuracy and relevance. It implements semantic caching of prompt responses, enabling instant answers to semantically equivalent queries and reducing latency. Organizations can dramatically improve their LLM applications' accuracy, relevance, and response times by using an AI Gateway and these features while tailoring outputs to specific domains and use cases.

Challenge

- Providing models with accurate and up-to-date information without retraining
- Injecting client-specific, domain-specific context to a prompt
- Minimizing risk of generalizations and hallucinations
- Optimizing response latency for semantically similiar prompts

Solution

- Retrieval Augmented Generation (RAG) enrichment of prompt requests with relevant context
- Semantic caching of prompt responses to allow immediate response to semantically equivalent queries
- Integration with third-party vector databases to leverage context enrichment and content caching via embeddings



Some Best Practices for implementing an AI Gateway

Secure Access and Credential Management:

- Secure credential storage for LLM API Providers at the infra/gateway instead of at each individual developer
- Generate API Keys per client in the AI Gateway that can map to one or more LLM API provider secrets
- Restrict LLM API provider and capability access by clients using External Authentication mechanisms
- Implement per-client authentication/ authorization of LLM API access
- Implement fine-grained authorization of LLM API access by model
- Implement fine-grained access controls using OPA and API Key metadata

Consumption Control and Visibility:

- Rate limit overall requests to LLM API providers to restrict runaway costs
- Consider rate limit by API Key metadata for consumers of the AI Gateway
- Consider token-based rate limiting of LLM API by the client or by team
- Pre-configure token limits on requests
- Capture request client context in access logging for downstream analytics and usage reporting

Prompt Management:

- Consider the gateway layer's transformation capabilities to enrich request/response bodies such as injecting organizational security prompts and adding extra context
- If using multiple backend GenAI models, evaluate the creation of a canonical input schema for your organization to simplify access to various local or third-party models
- Establish "Prompt Guards" to protect sensitive data from being leaked due to sophisticated attacks. Reject the requests if they match specific patterns such as credit card information or other PII

Future Trends in AI Gateways

As AI models are increasingly used across industries, functionalities aimed at improving AI Gateway performance, reliability, and resource efficiency will dominate. Key trends include failover systems and response enrichment to enhance the efficiency of LLM models. Below are some techniques that can improve the management and performance of AI models when integrated with an AI Gateway.

1

Model Failover:

Implementing model failover in AI Gateways is crucial as multimodal usage grows. For example, if the primary LLM API from one provider becomes unavailable due to downtime, high latency, or other issues, the system seamlessly switches to an alternative model from a different provider. This approach improves reliability and resilience, ensuring AI applications relying on generative models.

2

Retrieval Augmented Generation (RAG):

Configure the AI Gateway to retrieve data from a specified datastore and use it to augment the prompt before sending it to the model. This is useful when the model needs additional context to generate accurate responses, enhancing the quality and relevance of the output from AI models. This approach improves reliability and resilience, ensuring AI applications relying on generative models.

3

Semantic Caching:

Is a technique where semantically similar queries are stored within the AI infrastructure. If two prompts are similar, the LLM API response from the first can be reused for the second without sending a new request to the LLM. This reduces the number of requests to LLM APIs, improves response times, and lowers operational costs. Semantic caching enhances efficiency and optimizes resource use by minimizing redundant computations in AI traffic.This approach improves reliability and resilience, ensuring AI applications relying on generative models.

Conclusion

Using Gloo AI Gateway to support LLM APIs offers a robust and modern solution for organizations seeking to enhance their AI infrastructure security, reliability, and performance.

Organizations can enforce security policies, control traffic routing, and simplify GenAI API management by centralizing access through the gateway. Developers benefit from a unified internal endpoint, reducing the complexity of managing individual API keys and ensuring application consistency.

SOLO.IO

The Gateway to AI Innovation

Contact Solo.io to learn how we can help you leverage Gloo Gateway for your AI Gateway and other API management needs.

www.solo.io