API7 Technical White Paper

(Version: 2021-06)

1. Overall Introduction

Shenzhen Zhiliu Technology's API Gateway product (hereinafter referred to as API7) is built based on Apache APISIX, a top-level project of the Apache Software Foundation. API7 consists of 3 components: API Gateway, ManagerAPI and Dashboard Control Panel.

As an important component in microservice architecture, the API gateway is the core entry and exit point for traffic, it is used to process business-related requests, which can effectively solve the problems of massive requests and malicious access to ensure business security and stability.
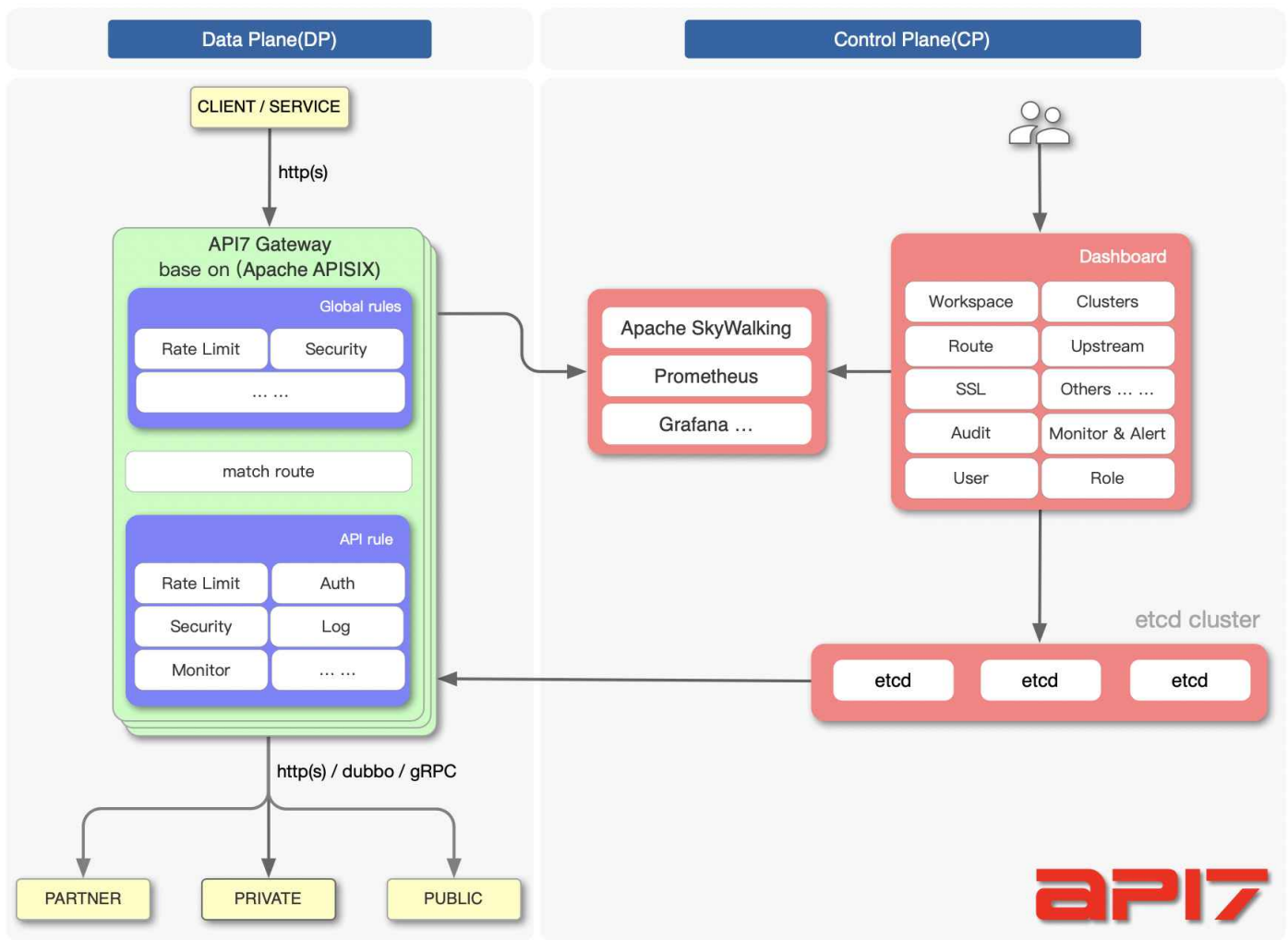
Figure 1-1 API7 Architecture

API7 consists of 3 components: API Gateway, ManagerAPI, and Dashboard Control Panel.

1. **API Gateway**

API Gateway is used to carry and process business traffic. After administrators configure routing rules, the gateway will forward requests to upstream services according to the rules. In addition, with more than 50 built-in plug-ins, API7 is able to meet most business demands, such as authentication, security protection, traffic control, analysis and monitoring, request/response conversion, etc. If the built-in plug-ins cannot meet the demands, API7 also support custom plug-ins written in Lua, which can be used at all stages of request entry and upstream response.

2. **ManagerAPI**

ManagerAPI is used to manage API gateways by accessing their exposed RESTful API interfaces to manage resources such as routes, upstreams, certificates, global plug-ins, consumers, etc.

3. **Dashboard Control Panel**

Dashboard control panel is a user interface used to simplify API gateway management. It supports monitoring and analysis, log auditing, multi-tenant management, multi-cluster switching, multiple work partitions, and other capabilities. Administrators can operate the API gateway through the Dashboard control panel.

1.1 Architecture

1. **Data Plane**

The data plane is used to receive and process caller requests, using Lua and Nginx to dynamically control request traffic. When a request comes in, it is matched based on predefined routing rules, and the matched request is forwarded by API7 gateway to the corresponding upstream service. During this process, API7 gateway has the ability to use a series of plug-ins to operate on the request from entry to exit, depending on the configuration of the different plug-ins in the preset rules. For example, the request may go through several steps such as authentication (to avoid replay attacks, parameter tampering, etc.), request audit (request source information, upstream processing time, etc.), route processing (to obtain the final upstream service address according to the preset rules), request forwarding (the gateway forwards the request to the upstream target node), and request response (after the upstream processing is completed, the gateway returns the result to the caller).

2. **Control Plane**

The control plane contains the ManagerAPI and the default configuration center ETCD. when the administrator accesses and operates the console, the console will call the ManagerAPI to send the configuration to the ETCD, and with the ETCD Watch mechanism, the configuration will take effect in real time in the gateway. For example, an administrator can add a route and configure a rate limit plug-in, and when the rate limit threshold is triggered, the gateway will temporarily block access to subsequent requests matching that route. With ETCD's Watch mechanism, API7 will notify each gateway node within milliseconds when the administrator updates the configuration in the control panel.

3. **Others**

As shown in Figure 1-1, API7 adopts the architecture of separating the data plane and the control plane, and the configuration center receives and sends down the configuration so that the data plane will not be affected by the control plane. The configuration center is ETCD by default, but it also supports Consul, Nacos, Eureka, etc., so you can choose according to your actual situation. In addition, enterprise users only need to focus on the business itself, and most functions not related to the business can be implemented by the built-in plug-ins of API7, such as authentication, performance analysis, etc.

1.2 Highlights

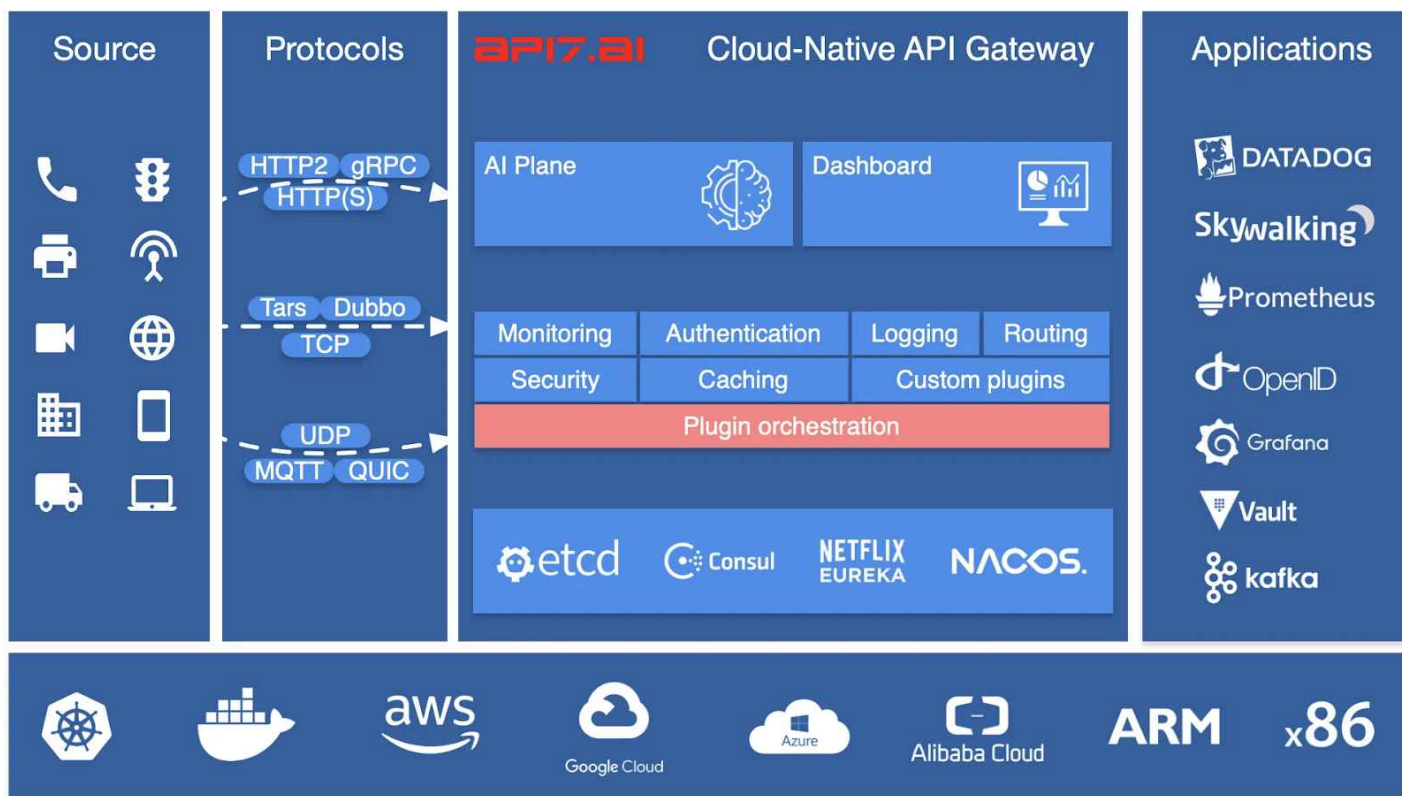**Figure 1-2 API7 Highlights**

1. **Cloud Native**

API7 is a cloud-native API gateway. It is neither platform-related nor vendor-locked. API7 supports bare metal, virtual machines, Kubernetes, OpenShift, ARM64, and other platforms. In addition, API7 can easily interface with other components such as SkyWalking, Prometheus, Kafka, Zipkin, etc., to empower the enterprise.

2. **High Availability**

By default, API7 uses ETCD as the configuration center. ETCD supports distributed and high availability, and has a lot of practical experience in K8s and other fields, which makes API7 easily support millisecond configuration updates and thousands of gateways; the gateways are stateless and can be expanded or reduced at will.

3. Protocol conversion

API7 supports many protocol types, such as TCP/UDP, Dubbo, MQTT, gRPC, SOAP, WebSocket, etc.

4. Security Protection

API7 has a variety of built-in authentication and security capabilities, such as Basic Auth, JSON Web Token, IP blacklist and whitelist, OAuth, etc.

5. High Performance

API7 uses Radixtree algorithm for high-performance, flexible routing with QPS of about 140K and latency of about 0.2 ms in AWS 8-core servers.

6. Full Dynamic Capability

API7 supports modifying the gateway configuration, adding or modifying plug-ins, etc., which can take effect in real time without restarting or reloading the gateway service. API7 also supports dynamic loading of SSL certificates.

7. High Scalability

With the flexible plug-in mechanism, you can customize the functions for internal services. API7 supports customized load balancing algorithms and routing algorithms. It is not limited to API gateway implementation. You can implement serverless by dynamically executing user-defined functions at runtime, making the gateway edge nodes more flexible.

8. Rich Governance Capacity

Such as fault isolation, service meltdown, service downgrade, flow limit and rate limit, etc. After enabling active health check, API7 will support the ability to intelligently track unhealthy upstream nodes and automatically filter unhealthy nodes to improve overall service stability.

1.3 Function Modules

API7 mainly contains the following functional modules:

- **User System:** With the help of user system, the administrator will assign certain accesses and resources for each user in the system, and the user cannot override the accesses to the resources. API7 supports account password login and SSO login.

- **Permission System:** API7 has a built-in Role-Based AccessControl system (RBAC), which allows administrators to create different roles with the help of the console, and by binding users to roles, fine-grained permission control can be achieved.

- **Multi-tenancy (multiple working partitions):** API7 supports multi-tenancy based on working partition isolation, where administrators can create different working partitions and specify which users have access to which resources on the working partition.

- **Multi-environment:** API7 supports multiple ETCD clusters, with no data sharing among clusters.

- **Authentication:** API7 includes a variety of authentication plug-ins, such as basic-auth, jwt-auth, key-auth, wolf-rbac, and so on. In addition, with the built-in HMAC plug-in, request parameters can be signed and verified using AK/SK to achieve tamper-proof requests and replay-proof requests, and to achieve the purpose of authentication.

- **Service Routing:** API7 is based on Radixtree for efficient route matching, and is currently the fastest API gateway for matching routes. It supports full path matching, prefix matching, and also supports using Nginx built-in variables as matching conditions to achieve fine-grained routing. In addition, API7 supports traffic mirroring and advanced route matching for fine-grained route management features such as grayscale publishing. It also supports service discovery and multiple registries, and has the ability to triage requests based on parameters such as Header, Query, and Cookie.

- **Protocol Conversion:** API7 supports many communication protocols, such as TCP/UDP, Dubbo, MQTT, gRPC, WebSocket, etc. API7 is able to convert HTTP protocol to other protocols of back-end services. API7 exposes a unified HTTP portal to the outside world, and

administrators can complete the protocol conversion settings through the console interface, and support the parameter mapping of requests and back-end services. APIs can be configured through the console interface.

· **Service Governance:** API7 supports service meltdown, flow limiting, rate limiting, IP blacklist and white list, fault isolation, etc., which can be easily and clearly set through the dashboard control panel.

· **Custom Plug-ins:** API7 has more than 50 built-in plug-ins, covering various categories such as security protection, traffic control, logging, etc., which can meet the needs of most enterprises. For specific business, API7 now supports custom plug-ins written in Lua, and the plug-ins can be applied to all stages of traffic in and out. Thanks to the fully dynamic capability, new and modified plug-ins can take effect in real time without downtime and restart, avoiding interruption of business.

· **Analysis and Monitoring:** API7 has built-in analysis and monitoring functions such as request auditing, monitoring and alerting, statistical reports, etc. API7 can record information of each request of all nodes and conduct statistics of successful requests and abnormal requests. You can view the number of successful requests, failed requests, error codes, request delays and other metrics in the console. In addition, with the ability of Grafana, API7 can meet the demand for more multi-dimensional analysis and monitoring.

· **Full lifecycle Management:** API7 supports API version management, API grouping, API release, API abolishment, online debugging and other functions, and is compatible with OpenAPI 3.0 standard, enabling API document generation, API import/export and other features to execute users' data migration operations.

1.4 Feature List

Categories	Feature Modules	Features	API7	Kong	Zuul2
		HTTP/1.1 and HTTP 2	✅	✅	✅
		HTTP/3	✅	✅	❌

API and Service Governance	Protocols	TLS / HTTPS	✔	✔	✔
		MQTT	✔	✖	✖
		TCP	✔	✔	✖
		UDP	✔	✔	✖
		HTTP to gRPC/Dubbo conversion	✔	✔	✖
		Websocket	✔	✔	✔
		Dubbo	✔	✖	✖
		Customized Layer 4 and Layer 7 protocols	✔	✖	✖
	Platforms	Bare Metal	✔	✔	✔
		Virtual Machines	✔	✔	✔
		Kubernetes	✔	✔	✔
		ARM64	✔	✔	✔
		Kunpeng (certified by Huawei Cloud)	✔	✖	✖
		AWS, GCP, Ali Cloud, Tencent Cloud and other public clouds	✔	✔	✔
	Fine-grained Routing	URI Parameter Matching	✔	✔	✔
		HTTP Header Matching	✔	✔	✖
		HTTP Request Method Matching	✔	✔	✖
		Support for all Nginx variables matching	✔	✖	✖
		Support for Lua snippets to implement custom matches	✔	✖	✖
		Support for conditional expressions	✔	✖	✖
		Support IPv6	✔	✔	✔
		GeoIP Geological Location Matching	✔	✔	✖

	Routing Time To Live (TTL)	✅	❌	❌
	Priority Matching	✅	✅	✅
Load-Balance	Round Robin	✅	✅	✅
	Weighted Round Robin	✅	✅	✅
	Consistent Hash (Chash)	✅	✅	❌
	Sticky Session	✅	✅	❌
	Least Connections	✅	✅	❌
	EWMA	✅	❌	❌
	Support for custom load balancing algorithms	✅	❌	✅
Rewrite Request	URI Rewrite	✅	✅	✅
	Add, modify and delete HTTP request headers	✅	✅	✅
	Support 301 and 302 Redirection	✅	✅	✅
	Force a jump to HTTPS	✅	✅	❌
Rewrite Response	Add, modify and delete HTTP response headers	✅	✅	✅
	Modify HTTP response code	✅	✅	✅
	Modify response body	✅	✅	✅
Service Discovery and Registration	Default ETCD and support for ETCD clustering	✅	❌	❌
	Consul	✅	❌	❌
	Eureka	✅	❌	✅
	Nacos	✅	❌	❌
	Redis	✅	❌	❌
	Traffic Control/ Cluster Traffic Control	✅	✅	❌

	Fault tolerance and downgrading	Rate Limit	✔	✔	✖
		Concurrency Limit	✔	✔	✔
		Upstream Active Health Check	✔	✔	✖
		Upstream Passive Health Check	✔	✔	✖
		Service Meltdown	✔	✔	✔
		Service Downgrade	✔	✔	✔
		API Meltdown	✔	✔	✖
		Timeout	✔	✔	✔
	Traffic Control	Grayscale Release	✔	✔	✖
		Blue-Green Release	✔	✔	✖
		Traffic Mirroring	✔	✖	✖
		Fault Injection	✔	✖	✖
	API Management	Multi API Aggregation	✔	✖	✖
		Version Management	✔	✖	✖
		Release and Abolish API	✔	✖	✖
		Swagger and OpenAPI	✔	✖	✖
		Generate SDK and documentation	✔	✖	✖
	Plug-ins Management	Dynamically add, modify and delete plug-ins	✔	✖	✔
		Plug-in orchestration (low code)	✔	✖	✖
		Support for writing custom plug-ins in Lua, Java and Go	✔	✔	✖
		RBAC	✔	✖	✖
		Multi-tenant	✔	✖	✖
		Multi-working partition	✔	✔	✖

Category	Subcategory	Feature			
Security	**User Management**	SSL Certificate Management	✅	✅	❌
		Control access with Admin API Key and IP restrictions	✅	❌	❌
	Communication Encryption	mTLS	✅	✅	✅
		Automatic rotation of SSL certificates	✅	✅	❌
		Supports GmSSL	✅	❌	❌
	Attack Prevention	IP Blacklist and Whitelist	✅	✅	❌
		URI Blacklist and Whitelist	✅	✅	❌
		Anti-ReDOS attacks	✅	❌	❌
		Anti-Replay Attack	✅	❌	❌
	Authentication	key-auth	✅	✅	❌
		basic-auth	✅	✅	❌
		JWT	✅	✅	❌
		API Signature Verification (HMAC)	✅	✅	❌
		OAuth2	✅	✅	❌
		SSO	✅	✅	❌
		Auth0, Okta, etc.	✅	✅	❌
Observability	**Metrics**	Prometheus	✅	✅	❌
	Tracing	SkyWalking	✅	❌	✅
		Zipkin	✅	✅	✅
		OpenTracing	✅	✅	✅
	Logs	Kakfa	✅	✅	❌
		HTTP Logger	✅	❌	❌
		TCP Logger	✅	❌	❌

		UDP Logger	✅	❌	❌
Performance and High Availability	**QPS**	Single Core Performance	Extremely High	High	Low
	Latency	Minimum latency per request	Excellent	Moderate	Low
	Deployment	Data plane stateless	✅	✅	✅
		Supports Cluster as Configuration Center	✅	❌	❌
	Cluster Management	Supports configuration and management of multiple clusters	✅	❌	❌
		Automatic cluster selection under multi-tier networks	✅	❌	❌
	Dynamic and hot updates	All changes are hot updated and take effect in real time	✅	❌	❌
		Plug-in hot updates	✅	❌	✅
		Hot update of the program itself	✅	✅	❌
Operations and Maintenance	**CLI**	Command Line Tools	✅	✅	❌
	Admin API	Use REST API for control and easy integration	✅	✅	❌
	Single Node	Use yaml file to define all rules	✅	✅	✅
	Rollback	Supports unlimited rollback of operations	✅	❌	❌
	Helm charts	Easier O&M under k8s	✅	❌	❌
	Global Plug-ins	Simplify operations	✅	✅	✅
	Health Check	Versioning and operational monitoring of data plane nodes	✅	✅	✅
		Provides configuration center status	✅	✅	❌

		and version information			
		Node load status monitoring	✔	✔	✖
	Service Observability	Service Invocation Topology	✔	✔	✖
		Data Throughput	✔	✔	✔
		Response time statistics	✔	✔	✔
		Upstream response time statistics	✔	✔	✔
		Status Code Statistics	✔	✔	✔
		API call statistics	✔	✔	✖

Chart 1-1 API7 Features

1.5 Feature Highlights

1. **API Full Lifecycle Management**

It covers API design, creation, testing, deployment, management, operation and maintenance, and offline phases, which can further help enterprises optimize API management process and increase enterprise value. With OpenAPI 3.0 standard, you can easily import and export APIs and generate documents to make more use of API capabilities.

2. **Multi-tenant Capability (multiple workspaces)**

API7 supports project isolation through work partitions to support multi-tenant capability. Combined with user system and permission management, different users have different permissions for resources under different work partitions, allowing for fine-grained permission control of resources.

3. **Multi-Protocol Conversion**

Since API7 supports communication protocols such as Dubbo, gRPC, and MQTT, API7 supports unified exposure of RESTful APIs to the outside world, reducing internal service protocol transformation.

4. **Full Dynamic Capability**

With the fully dynamic capability of API gateway, from gateway configuration to plug-in modification, it can take effect in real time without restarting the service, avoiding service interruptions that affect business traffic and produce unpredictable results. In addition, API7 also supports dynamic loading of SSL certificates.

5. Custom Plug-ins

API7 has more than 50 plug-ins built in, which can be used in combination to meet most gateway requirements.

With the unique low-code capability of the API7, you can combine plug-ins by drawing flowcharts to achieve a more advanced way of using plug-ins. If existing plug-ins do not meet your specific needs, customized plug-ins written in Lua is also supported by API7. Customized plug-ins can be used at all stages from request entry to response return, such as init, rewrite, access, balancer, header filter, body filter, log, etc.

6. Analysis and Monitoring

API7 integrates with Prometheus to obtain detailed API call data, including but not limited to access sources, success rates, top-95 values, top-99 values, success/failure response code distributions, QPS, and other metrics.

7. Dashboard

API7 has built-in dashboard control panel and ManagerAPI. Dashboard control panel facilitates users to configure rules through visual panel. ManagerAPI facilitates users to use automation tools or integrate into internal business to control gateway nodes.

2 . Feature Introduction

2.1 Plug-ins

API7 has more than 50 built-in common plug-ins, covering authentication, security protection, traffic control, analysis and monitoring, request/response conversion and many other categories. Some popular plug-ins are listed in the chart below.

Categories	Plug-in Name	Function	Scenario
		Authentication plug-in can effectively protect Route, Service from illegal, unprivileged access.	
	authz-keycloak	When the plug-in is enabled, API7 will support working with KeyCloak authentication service to complete identity authentication.	Identity Verification
	basic-auth	When the plug-in is enabled, the client needs to use the correct account and password when accessing.	
	hmac-auth	When the plug-in is enabled, in addition to verifying the validity of the client's identity, its end request parameters will also be signed and verified to avoid parameter tampering or secondary access (replay attack).	
Authentication	jwt-auth	When the plug-in is enabled, JSON Web Token will be used for validity verification, and the client needs to add the correct Token content to the HTTP request header when accessing.	
	key-auth	When the plug-in is enabled, the client needs to carry the correct key in the request header or query string when accessing the resource.	
	wolf-rbac	With this plug-in enabled, the gateway will support wolf-based authentication and authorization features.	
	openid-connect	When the plug-in is enabled, the gateway will support authentication and token introspection.	
	api-breaker	After enabling this plug-in, the gateway will determine whether the upstream is abnormal according to the configuration, and if it is abnormal, it will directly return the preset error code and no longer access the upstream for a certain period of time.	Meltdown
Safety	consumer-restriction	When this plug-in is enabled, if whitelist is set, consumers outside the whitelist will be denied requests by the gateway; if blacklist is set, consumers	

		inside the blacklist will be denied requests by the gateway.	
	cors	Support browser requests to the service by enabling the CORS plug-in.	
	fault-injection	When the fault injection plug-in is enabled, it will return the specified HTTP status code and response content directly to the incoming request for service maintenance purposes.	Fault Injection
	ip-restriction	Restrict access to services by whitelisting or blacklisting IP addresses, setting single or multiple addresses, or setting IP ranges in a CIDR fashion.	
	referer-restriction	When the plug-in is enabled, the Referer information in the request header is used to determine if the request needs to be restricted.	
	request-validation	When the gateway forwards the request upstream, the plug-in uses JSONSchema to validate the request header against the request body, and requests that fail the validation are rejected.	
	uri-blocker	When the plug-in is enabled, the gateway will return the specified status code when the request path matches the preset rule.	
Traffic Control	limit-conn	Enabling this plug-in will limit the number of concurrent requests.	Flow and Rate limit
	limit-count	With this plug-in enabled, requests that exceed a preset value within a fixed time window will be rejected.	
	limit-req	When enabled, the plug-in will use the Leaky Bucket algorithm to limit the request rate.	
	traffic-split	The plug-in allows us to dynamically control the ratio of traffic directed to different upstream services.	
Serverless	The Serverless plug-in dynamically executes Lua code in the gateway access phase to enable the execution of FaaS functions in a serviceless environment.		

	serverless-post-function	The functions configured in this plug-in will be run before other plug-ins.	
	serverless-pre-function	The functions configured in this plug-in will be run after other plug-ins.	
Observability	**error-log-logger**	The plug-in will push the contents of the error.log file generated by the gateway to the specified server using TCP protocol.	
	http-logger	This plug-in is used to send request data, response data, and contextual information to the HTTP server.	
	kafka-logger	The plug-in will send the log data to Kafka.	
	prometheus	The plug-in will expose the relevant metrics of the gateway in Prometheus' data format.	
	request-id	The plug-in will add a request-id request header to each request processed by the gateway to identify the API request.	
	skywalking	SkyWalking is an observability analysis platform, and this plug-in will proactively report data to SkyWalking so that we can easily view the status of the gateway through SkyWalking.	
	sls-logger	This plug-in is used to send request data, response data and contextual information to AliCloud SLS logging service.	
	syslog	This plug-in is used to send request data, response data, and contextual information to Syslog.	
	tcp-logger	This plug-in is used to send access-log data as TCP to the specified server	
	udp-logger	This plug-in is used to send access-log data to the specified server as UDP. Since UDP does not require three handshakes, it is efficient and has good real-time transmission.	
	zipkin	The plug-in will report gateway timing and trace data to Zipkin, including but not limited to TraceID, node	

		information, request information, latency, etc. This can help us locate problems encountered with the gateway through Zipkin.	
Others	**batch-requests**	The plug-in will support the use of Pipeline form to receive multiple requests and send them to the corresponding upstream service, whose response content is a combination of the response content of multiple requests. This is useful when a client wants to access multiple APIs.	
	grpc-transcode	The plug-in will support sending RESTful API requests to the gRPC upstream service.	
	proxy-cache	The plug-in will support caching upstream service response content, when the content requested by the client already exists in the cache, the content will be returned directly from the cache, without the need to request the upstream service again. This will effectively reduce the pressure on the upstream service. In addition, when the upstream node fails, it can also temporarily return the cached content without returning the error page to improve the user experience.	
	proxy-mirror	The plug-in supports mirrored replication of requests for better bypassed request analysis.	
	proxy-rewrite	Before the request sent by the client reaches the upstream service, the plug-in will modify the request according to the specified rules, including but not limited to the request body, request header, request path and other parameters.	
	response-rewrite	Before the response from the upstream service reaches the client, the plug-in will modify the response content according to the specified rules, including but not limited to the response body, response header and other parameters.	

Chart 2-1 API7 Plug-ins

2.2 Authentication

API7 has built-in authentication authentication plug-ins such as key-auth, basic-auth, jwt-auth, etc. Taking HMAC plug-in as an example, API7 can work with AK/SK to encrypt the request parameters to ensure that the request has not been tampered with.

Request parameters such as Request Path, Request Query String, timestamp, and signature algorithm, are included in Request Header to avoid request tampering and replay attacks.

2.3 Grayscale Release

Routing is the core function of the API gateway, which is used to route and match requests passing through the API gateway and forward them to the corresponding upstream service. When the upstream service finishes processing, the result is returned to the client. If a request does not match a route, the gateway will return a 404 status code because the route has not been published to the gateway or the route is not configured.

With API7's powerful routing capabilities, the need for grayscale publishing and blue-green deployment can also be realized so that enterprises can smoothly upgrade their services in a stable manner. In addition, when the API gateway forwards requests to upstream services, it will carry some HTTP request headers to mark that the traffic came from the gateway.

Take a grayscale release as an example: after starting a grayscale release, first start the new version of the service (application) and give it to testers to test the new version. If the test is OK, then a small amount of traffic can be switched to the new version, followed by a running status check of the new version and the collection of various data. When the new version is confirmed to be working well, then gradually switch more traffic to the new version. Until 100% of the traffic is switched to the new version, then the old version is shut down and the grayscale release is completed. If you find any problem with the new version during the grayscale release, you can immediately switch the traffic back to the old version, so that the negative impact will be kept to a minimum.
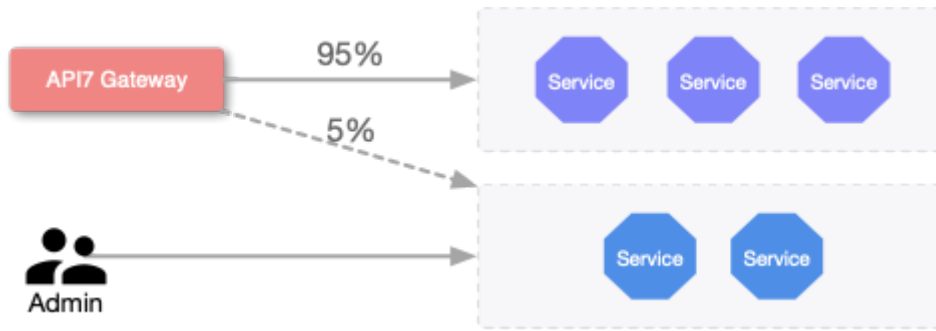
Chart 2-1 Grayscale Release

2.4 Service Governance

API7 has built-in service governance features such as flow and rate limiting, service meltdown, IP blacklist and whitelist, and fault isolation.

1. **Flow Limit and Rate Limit**

API7 limits the flow and rate based on the Leaky Bucket algorithm, with three built-in limit-count, limit-req, and limit-conn plug-ins to limit the flow and rate:

Name	Description
limit-count	rate limit based on fixed window
limit-req	Request rate limit based on the leaky bucket principle
limit-conn	Limit concurrent requests

Chart 2-2 Flow Limit and Rate Limit Plug-ins

Take limit-req for example, which contains the following parameters:

Parameter	Type	Required	Range	Description
rate	integer	Yes	> 0	The maximum allowed request rate. Requests greater than rate but less than rate + burst will be delayed. The unit is seconds.
burst	integer	Yes	>= 0	The rate of requests that are allowed to be delayed.

key	string	Yes	remote_addr, server_addr,http_x_real_ip, http_x_forwarded_for,consumer_name	The keyword used to limit the request rate (request count basis).
rejected_code	integer	Yes	200 ~ 599	This status code will be returned when the request rate exceeds rate + burst. The default status code is 503.

Chart 2-3 limit-req Plug-in Parameters

After creating the route through the control panel, bind the limit-req plug-in for it, assuming that the rate is set to 1, burst to 2, and rejected_code to 503, which means that the rate per second is 1. When the rate exceeds 1 but is less than 3, the request will be delayed; when the rate exceeds 3, the request will be rejected and the rejected_code will be returned, such as 503.

2. IP blacklist and whitelist

API7 has a built-in IP blacklist and whitelist plug-in that allows administrators to set it through the control panel. By setting up a blacklist IP list or a whitelist IP list, you can control the access to resources of routes and services.

3. Meltdown

When a request arrives at the API gateway, there are 3 sorts of possibile outcomes:

· A normal request and a normal response.

· The request is normal and the response is abnormal.

· The request is abnormal.

When a large number of requests arrive, if the upstream service cannot respond to the requests in time and is in a blocking state, there will be a situation where the upstream service is defeated. As an API gateway, it should be able to detect and handle abnormal problems in time to avoid more serious problems. In this case, API gateway service degradation will come into play.

2.5 Log Auditing

API7 has a built-in log auditing module, which collects system security events, administrator operation records, system operation logs, system operation status and other kinds of information in the information system centrally, and then stores and manages them centrally in the form of logs in a unified format after normalization, filtering and consolidation, combining with rich log statistical summary and correlation analysis functions to realize comprehensive auditing of information system logs. Through post-event analysis and reporting system, administrators can easily and efficiently conduct targeted security audits on information systems; when encountering special security events or configuration failures, the log auditing system can help administrators conduct rapid configuration positioning and rollback. Only administrators with authority can perform operation rollback.

2.6 Refined Routing

API7 will triage the matched requests according to the preset weights and parameters.

1. Triage by weight

The administrator creates each upstream object through the control panel, and during the configuration process, allows to set the weight value for each upstream service instance, if the value is 0, it means that no traffic is assigned to the example. In addition, upstream supports algorithms such as round robin polling with weight, chash, and exponentially weighted moving average (EWMA).

2. Streaming by parameters

The API gateway supports triage based on each parameter of the request and its value, for example:

Parameters	Value

Request Header	{
	
	    "vars": [["http_user_agent", "~*", "android"]], // Whether request header User-Agent matches android
	
	    "uri": "/hello",
	
	    "upstream_id": "1"
	
	}

| Request Host | ```json
{
    "hosts": ["www.my.com"], // Whether host matches "www.my.com"
    "uri": "/hello",
    "upstream_id": "1"
}
``` |
| Request Path | ```json
{
    "uri": "/hello", // Whether path matches "/hello"
    "upstream_id": "1"
}
``` |
| Request Query String | ```json
{
    "vars": [["arg_theme", "==", "light"]], // Whether theme in query string matches light
    "uri": "/hello",
    "upstream_id": "1"
}
``` |
| Request Cookie | ```json
{
    "vars": [["cookie_token", "==", "1234"]], // Whether token field in cookie matches "1234"
    "uri": "/hello",
    "upstream_id": "1"
}
``` |

Chart 2-4 Streaming Parameters

When the request is matched with each parameter, the traffic will be assigned to the corresponding upstream service.

2.7 Monitoring and Alerting

API7 records the basic information and status of each request. With the help of the statistical report page in the dashboard control panel, administrators can see the status of each service call, status code distribution, number of successes, number of failures, top 95 values, top 99 values and other information. It is convenient for administrators to understand the health of the

system. In addition, the data plane will regularly report the traffic processing situation, and the administrator can view the gateway operation status and other indicators, such as error rate, number of requests, status code distribution, etc., within a certain time period through the control panel. When the administrator presets the alarm rules through the control panel, if the traffic reported by the gateway matches the rules, it will trigger the preset policies, such as sending station letters, email alerts, SMS and Webhook notifications, etc.

2.8 Protocol Conversion

API7 exposes RESTful APIs uniformly to the outside world, which can be set by administrators in the control panel. These APIs correspond to microservices/upstream services in the enterprise and support proxies for protocols such as Dubbo, gRPC, WebServices, MQTT, etc., in addition to common HTTP services.
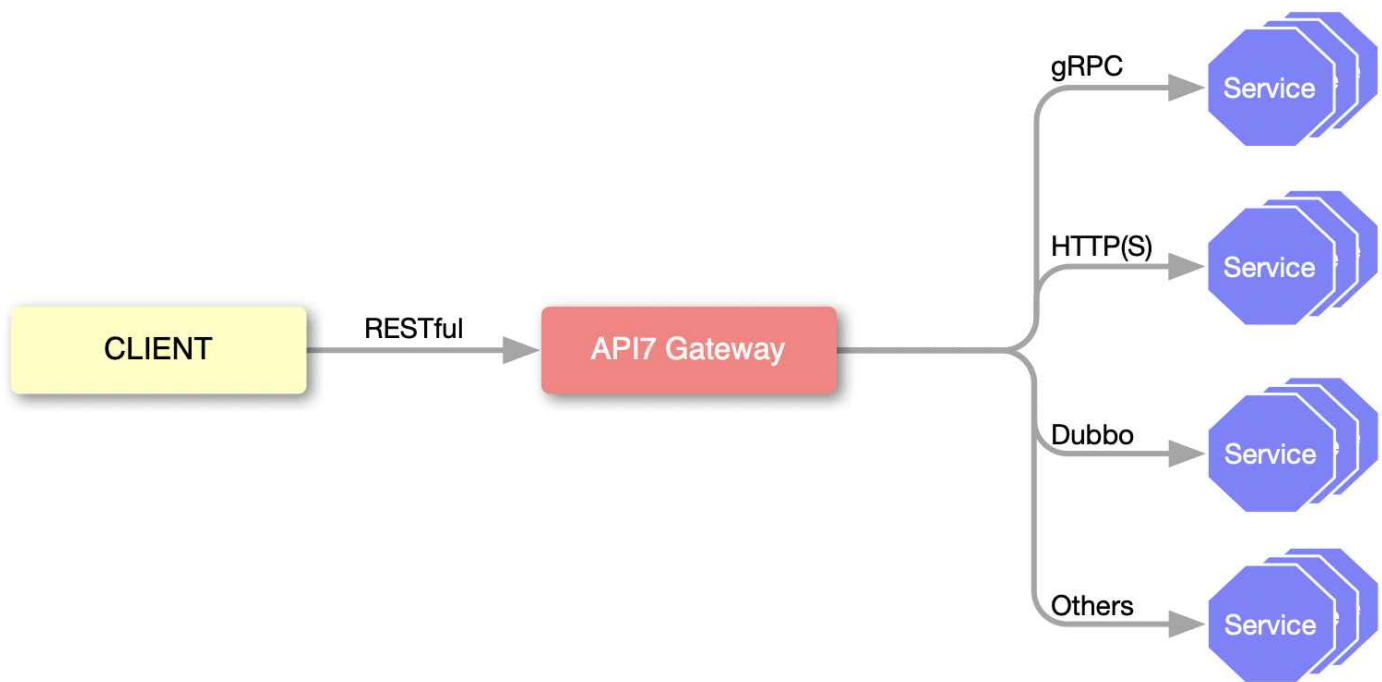

Figure 2-2 Protocol Conversion

The control panel allows administrators to create upstream services of different protocols and supports creating route objects to bind to upstream services, which are the APIs to be used by the caller. in the process of configuring routes, administrators need to set the HTTP methods (such as GET, POST, PATCH, etc.) listened to by the route, HTTP host name, and other parameters to match requests as rules.

After the route is configured and published, when the request is processed by the API gateway, the API gateway will match the corresponding request according to each routing rule, construct the request content of different protocols and forward it to the upstream service.

2.9 Multi-tenant and Multi-working Partition

API7 has a built-in workspace module, super administrators need to create multiple workspaces, then create ordinary users and assign different permissions (in the configuration of permissions, you can bind workspace and resource permissions), so that the combination of the user system and permission management can achieve different users in different workspaces, different permissions for different resources, in order to achieve fine-grained control of resources permissions.

2.10 Performance

API7 adopts excellent performance solutions in all aspects from route matching, JSONSchema validation, and plug-in operation.

Take route matching as an example, API7 uses the self-developed radixtree (open-sourced by Shenzhen Zhiliu Technology) algorithm for routing, which does not reduce efficiency when the number of routes is very large because its time complexity is O(K) (K is the length of the route string, independent of the number of routes).

The following figure shows the latency comparison between API7 and Kong Enterprise Edition (Kong EE) at 10000 rps.
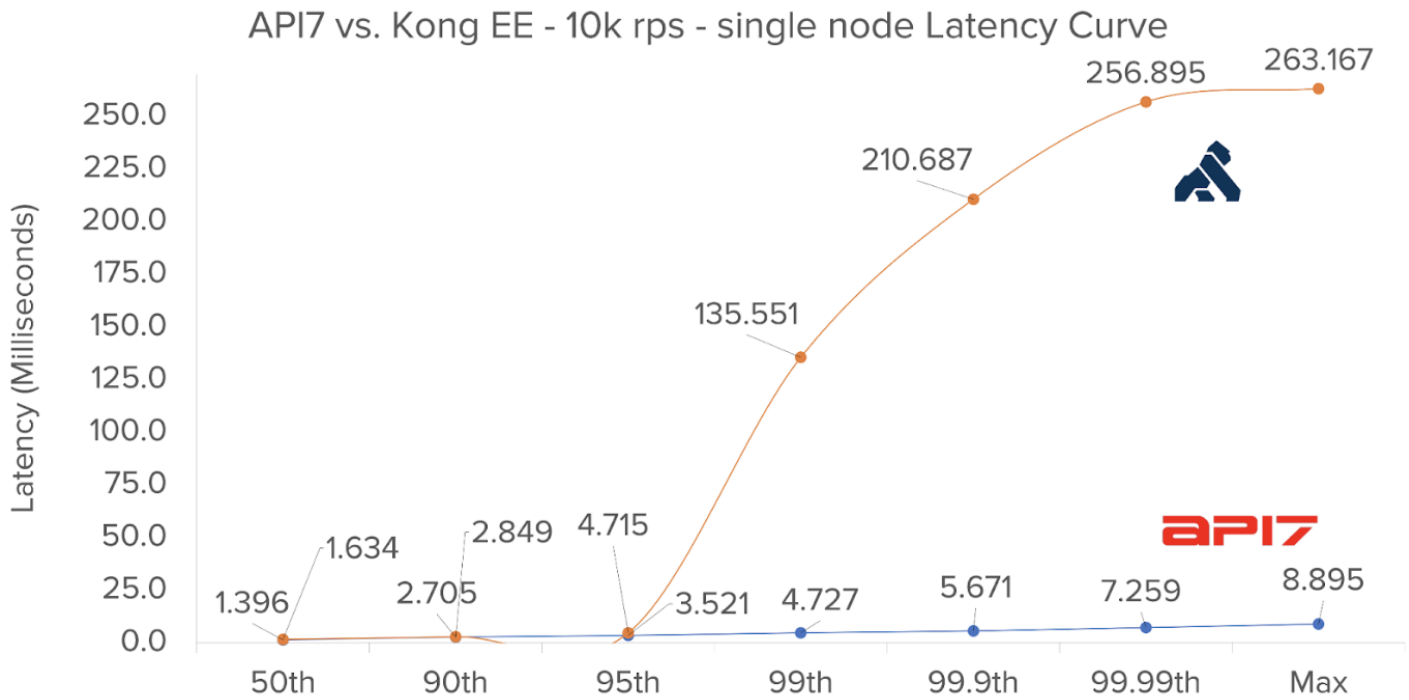
Figure 2-3 API7 v.s. Kong EE with 10krps and single node

As shown in Figure 2-3, the latency performance of API7 is very stable, 99.9% of requests are processed within 6 milliseconds; while Kong EE's latency is 10 times higher than API7.

The following figure shows the latency performance for the same 10,000 rps with the JWT authentication plug-in enabled: