API Security in the Cloud Native Era

The Role of APIs in Cloud Native Architecture

Modern cloud native applications are composed of dozens of loosely coupled microservices, enabling developers to create complex applications with great ease and speed. This type of architecture constantly changes based on customer needs, and the decoupled nature of microservices enables developers to push new code and functionality very frequently. The connectivity and communications among microservices are via application programming interfaces (APIs) such as REST, gRPC and GraphQL.

In cloud native applications, a single client's web request (i.e., north-south traffic) that hits your Kubernetes® cluster can spawn tens or even hundreds of API calls between internal microservices (i.e., east-west traffic). It is never enough to only secure the front-end web interface of your cloud native application—you must also apply rigorous application layer protection for your cloud native APIs.

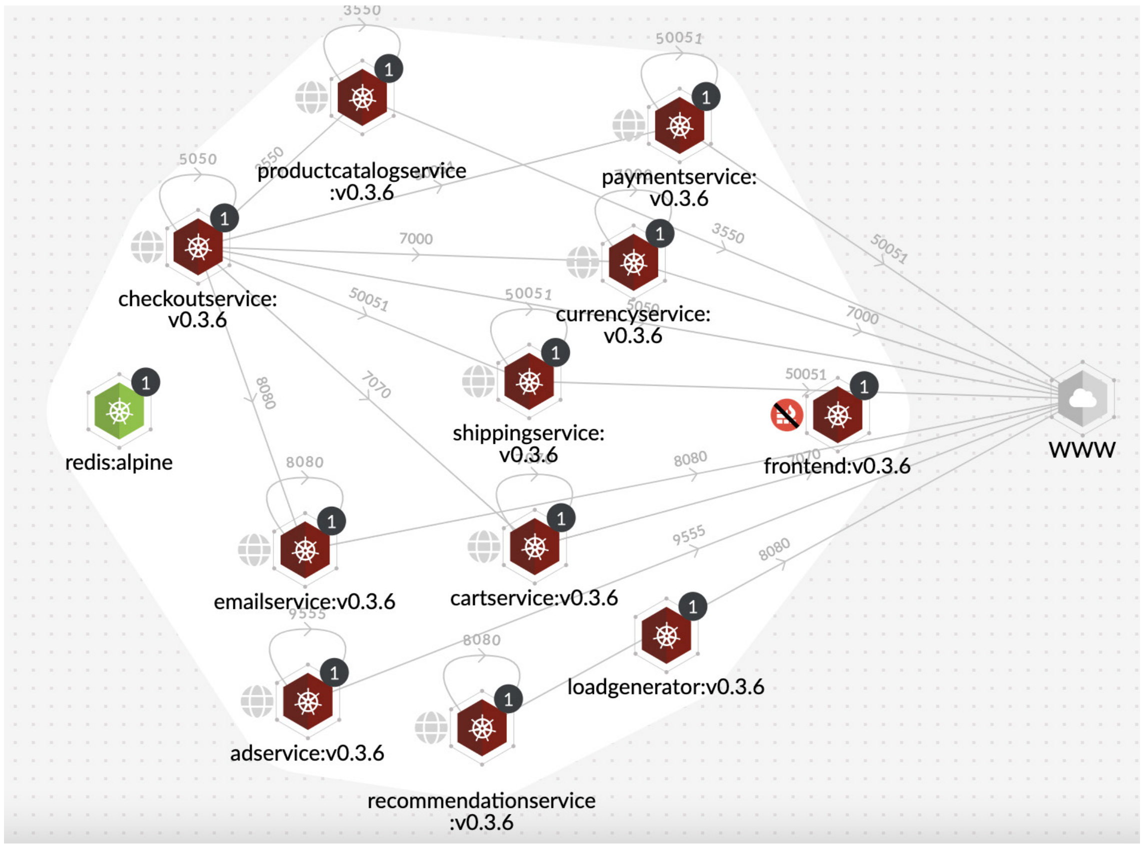


Figure 1: A typical cloud native application

Traditional Approach to Secure Web Apps

The common approach to securing web applications in the monolith application world was to deploy a traditional web application firewall (WAF) at the perimeter, so it would be able to intercept and inspect HTTP traffic sent by web clients. This approach made total sense when the potential risk to the application was mostly from malicious user input embedded in standard HTTP web form submissions or browser requests. However, when dealing with highly distributed, cloud native microservices architecture, this approach is no longer suitable for the following reasons:

- Modern applications often consume input from a much wider range of sources. These include standard web requests, mobile device API calls, cloud events, IoT device telemetry communication, cloud storage, etc. Inspecting input at the (web) perimeter does not provide full security coverage and may miss potentially hazardous payloads.
- Client inbound HTTP requests (i.e., north-south traffic) are often the first step in a long sequence of communication flows. In many cases, a single inbound request will generate dozens of internal API calls (i.e., east-west traffic). If those internal API calls are not properly inspected and validated, API endpoints are left unprotected.

- Internal API endpoints are often misconfigured and may allow unauthorized direct access to individual microservices, essentially exposing application logic to malicious actions.

For the reasons above, it is critical that all API endpoints, both external and internal, are continuously monitored and protected rigorously.

You Cannot Protect What You Don't Know

In order to protect your APIs, you must first discover and catalog all API endpoints. In a perfect world, this would only entail getting an inventory of all API endpoints from your developers—perhaps by exporting the list as OpenAPI specification files or by statically scanning the code and extracting all possible API endpoints. In the real world, these approaches have limitations:

- As applications rapidly evolve and change over time, new APIs are created, and older versions of APIs are slowly deprecated. Your development team might not be fully aware of the inventory, which means your OpenAPI specification files can easily become incomplete or inaccurate.
- Microservices architectures enable developers to develop, build and release new APIs frequently, sometimes several times a day. This means that by the time you receive your API inventory list, it's already outdated.
- It is common to expose temporary API routes, for debugging and testing purposes during the development phase. Although such temporary ("shadow") endpoints shouldn't be left exposed in production applications, the reality is that they often are.

For the reasons described above, it is absolutely essential that you continuously monitor all of your API endpoints and make sure that all of them are properly secured. When a new API is added or an existing API changes, you must make sure that you receive a notification or alert to avoid security gaps.

Continuous API Discovery in Cloud Native Applications

API discovery techniques can be roughly categorized into three main methods:

1. **Manually or automatically exporting OpenAPI specification files.** These files include a list of API endpoints and the contract by which they operate, such as message structure, parameter types and expected values, and HTTP methods. This method has several drawbacks. As previously explained, such files might not be up to date. In addition, this method only tackles the inventory aspect of protecting APIs—you will still need to monitor traffic in order to detect unauthorized actions on the discovered APIs.
2. **Real-time observation of all API traffic.** This is usually achieved by monitoring all relevant network interfaces and ports. The benefit of using this method is that it enables real-time discovery, inspection, and rapid reaction to detected unauthorized actions.
3. **Collecting traffic logs from various sources.** This includes cloud logs, HTTP daemon logs, full transaction logs from gateways, VPC traffic mirroring logs, etc. Once traffic logs are collected, they need to be filtered to include only API traffic, which is necessary for flagging API endpoints and potential unauthorized actions. The main benefit of this approach is that it is mostly non-intrusive. However, it also poses challenges:
    - **You will be required to apply numerous different traffic collectors in each and every environment you own.** This may include: saving full HTTP transaction logs from API gateways, enabling traffic mirroring and sending this traffic to a dedicated collector node, deploying traffic sensors within orchestrated container environments (e.g., Kubernetes), and so on. The task of applying traffic collectors requires careful attention since you must be aware of all the relevant log sources in order to gain complete API coverage and visibility.
    - **Delayed response time.** Traffic and log collection is a process that will usually take time—often several minutes. Add the time required by the log analysis process, and you end up with a slow response time. This can mean that by the time your security devices are made aware of an attack, it is too late.
    - **Log data incompleteness.** Not all log sources include the full HTTP transaction log data, such as HTTP headers or HTTP request bodies. This data is crucial, both for profiling API endpoints and their normal behavior, as well as for enforcing security actively.

    It should be noted that some niche API security vendors whose products rely solely on log traffic collection tout themselves as "agentless." However, in many scenarios and environments, customers end up having to deploy software agents for log collection—whether a dedicated log collector VM image, a Kubernetes DaemonSet with traffic-collecting container pods, or a web server and proxy plugins, and so forth.

Since each of these approaches has both benefits and shortcomings, it is clear that a mix of multiple methods will yield the best results and will allow each method's benefits to compensate for other methods' shortcomings.

Protecting APIs vs. Protecting Applications

As previously explained, APIs play a key role in cloud native applications—they serve as the glue between microservices, or more precisely, they are the main method by which the different components of your application communicate with each other. However, applications are composed of many different building blocks besides APIs. These may include:

- Microservices often deployed as containers
- Serverless functions
- Databases
- Cloud storage buckets
- Hosts/VMs
- Message queues
- Container orchestration platforms

Given the complexity and the wide array of components in your applications, even the most rigorous API security solution will only provide security coverage for API endpoints. However, it will be completely oblivious to all other components, which may contain vulnerabilities and misconfigurations that can enable attackers to perform unauthorized actions.

Moreover, only applying API security controls means that you are bolting on security at the end of your development process, which is far too late. Most security gaps can already be mitigated by scanning for vulnerabilities (e.g., scanning code repositories, container and VM images or serverless functions). On top of that, you can reduce your attack surface by ensuring that infrastructure as code templates are properly configured as strictly as possible.

In addition to API security, code scanning and configuration analysis, it is also crucial to apply workload runtime protections. These protections serve as an additional layer of defense and make sure that your workloads are behaving as expected. Not all hazardous application payloads arrive through web APIs. For example, malicious files could be uploaded to cloud storage buckets and consumed by the application. All traffic inspection methods (whether based on live traffic inspection or traffic log analysis) are prone to false negatives, which is why runtime protection is a critical capability in an end-to-end application-layer protection solution.

While API discovery and inventory should be your first step in this process, it cannot be the only step. Once the discovery step is completed, you will need to actively protect your applications and APIs against abuse. Using a short-term Band-Aid solution, i.e., just discovering API endpoints, is not enough, and you must apply multilayered, defense-in-depth security to protect against API abuse.

API Security with Prisma Cloud

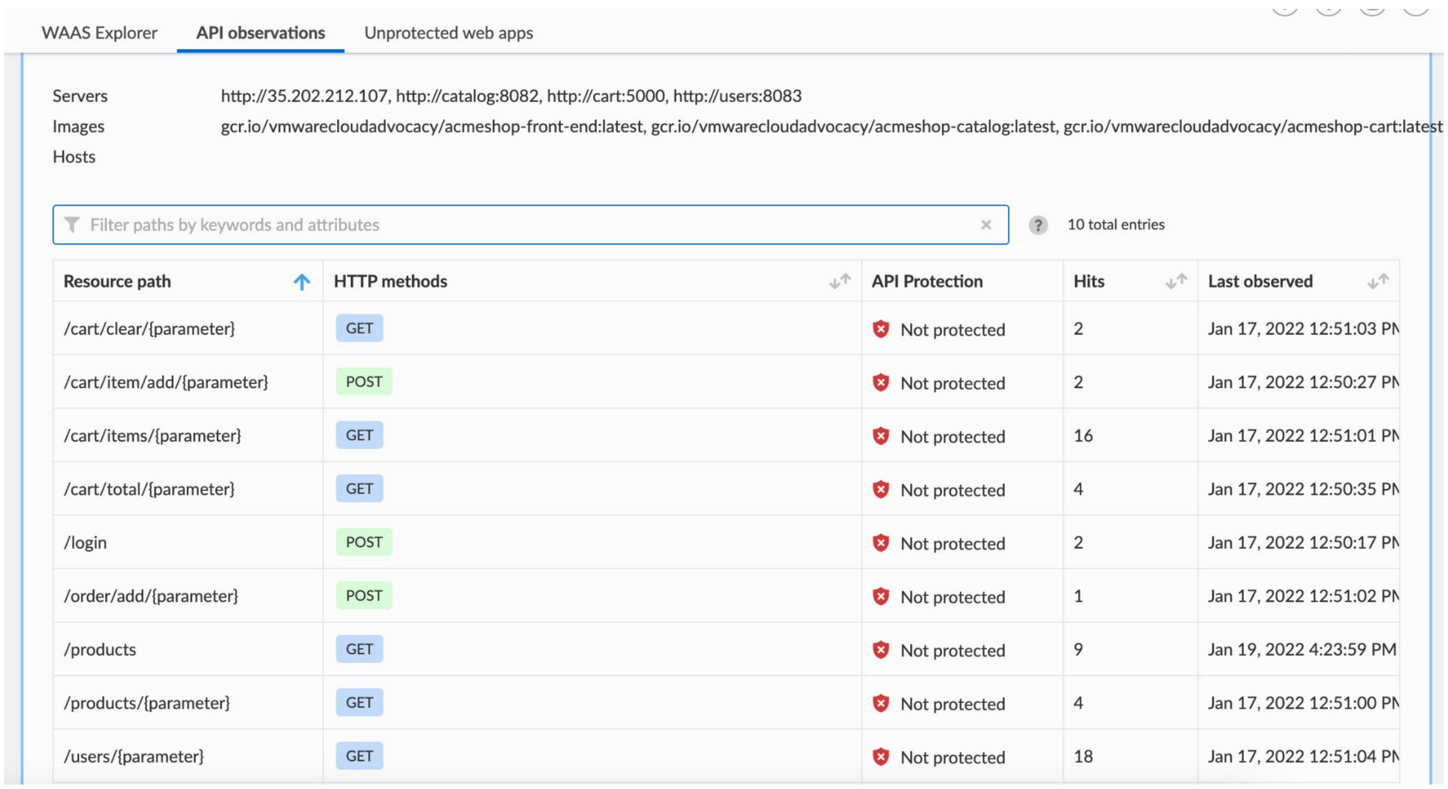
Prisma® Cloud by Palo Alto Networks takes a holistic approach to cloud native application security by providing users with the most complete and comprehensive end-to-end security solution. At Palo Alto Networks, we understand that modern cloud native applications are complex and require an end-to-end, defense-in-depth approach. Prisma Cloud's Web App & API Security (WAAS) module provides best-of-breed API security capabilities as part of our broader Cloud Native Application Protection Platform (CNAPP) solution that includes:

- Cloud Security Posture Management
- Cloud code security
- Cloud Workload Protection
- Cloud data security
- Network Microsegmentation

These are among many other critical capabilities that provide full security coverage for your entire application development lifecycle.

The Prisma Cloud WAAS provides users with the following benefits:

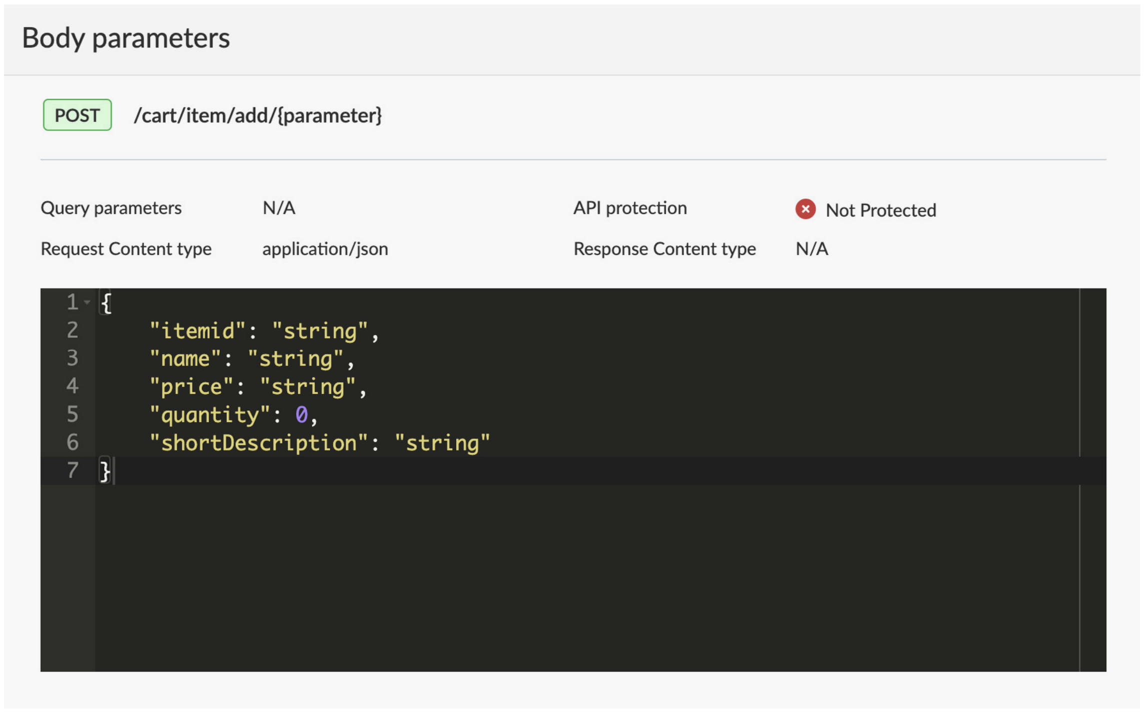
- Easy deployment process, suited for different types of cloud workloads such as hosts, VMs, containers, orchestrated container clusters and serverless functions
- Ability to protect applications wherever deployed, whether running on-premises, in the private cloud, public cloud, multicloud or hybrid deployments
- Ability to easily monitor and secure all applications from a single management console for all your applications
- Prisma Cloud's multilayered web & API defense includes:
  - » A state-of-the-art web application firewall
  - » Real-time automated API discovery in all environments and API traffic profiling via machine learning message structure and input format to create a normal baseline
  - » Real-time automated discovery of unprotected web applications and APIs
  - » API protection against OWASP API Top 10 risks
  - » API message validation using OpenAPI specification file enforcement and machine learning profiling of API calls
  - » DoS attack protection through rate controls
  - » Bot risk management, including passive and dynamic detection through client challenges
  - » Robust real-time network access controls to prevent unauthorized access to APIs and web applications
  - » Virtual patching for newly discovered zero days, exploits or other emerging threats
  - » Custom rules for an additional mechanism to protect your APIs using a specific set of criteria
  - » Dashboard reporting, analytics, and alert integrations with leading industry service providers



Servers	http://35.202.212.107, http://catalog:8082, http://cart:5000, http://users:8083
Images	gcr.io/vmwarecloudadvocacy/acmeshop-front-end:latest, gcr.io/vmwarecloudadvocacy/acmeshop-catalog:latest, gcr.io/vmwarecloudadvocacy/acmeshop-cart:latest
Hosts	

Resource path	HTTP methods	API Protection	Hits	Last observed
/cart/clear/{parameter}	GET	Not protected	2	Jan 17, 2022 12:51:03 PM
/cart/item/add/{parameter}	POST	Not protected	2	Jan 17, 2022 12:50:27 PM
/cart/items/{parameter}	GET	Not protected	16	Jan 17, 2022 12:51:01 PM
/cart/total/{parameter}	GET	Not protected	4	Jan 17, 2022 12:50:35 PM
/login	POST	Not protected	2	Jan 17, 2022 12:50:17 PM
/order/add/{parameter}	POST	Not protected	1	Jan 17, 2022 12:51:02 PM
/products	GET	Not protected	9	Jan 19, 2022 4:23:59 PM
/products/{parameter}	GET	Not protected	4	Jan 17, 2022 12:51:00 PM
/users/{parameter}	GET	Not protected	18	Jan 17, 2022 12:51:04 PM

Figure 2: Automated API discovery in Prisma Cloud WAAS


Body parameters

POST /cart/item/add/{parameter}

Query parameters	N/A	API protection	Not Protected
Request Content type	application/json	Response Content type	N/A

```
{
    "itemid": "string",
    "name": "string",
    "price": "string",
    "quantity": 0,
    "shortDescription": "string"
}
```

Figure 3: Automated API message profiling in Prisma Cloud WAAS

Summary

As cloud native application development becomes the de facto method for building and delivering modern applications, organizations must adopt a modern cloud native application security solution that provides end-to-end security and defense in depth.

Application security is best approached as a continuous process closely coupled to your CI/CD pipelines and covering the many different facets involved:

- Detect and prevent code-level vulnerabilities from reaching production systems.
- Ensure that VM and container images do not contain known vulnerable packages.
- Verify that cloud configuration and IAM policies are as strict as possible.
- Enforce security best practices on infrastructure as code templates.
- Apply workload runtime protection.
- Detect and prevent abuse of web applications and APIs in real time.

Prisma Cloud is at the forefront of cloud native security and provides customers with the most comprehensive security capabilities necessary for protecting their entire cloud native application stack.

To test out all the great functionality and more of Prisma Cloud's WAAS module, request a hands-on demo.

Written by By Ory Segal, Sr. Director Product Management, Prisma Cloud.

paloalto NETWORKS

3000 Tannery Way
Santa Clara, CA 95054

Main: +1.408.753.4000
Sales: +1.866.320.4788
Support: +1.866.898.9087

www.paloaltonetworks.com