apigee

Google Cloud

1

2

Inside the API Product Mindset

PART 3

Building and Managing Secure APIs

4

- Field-tested best practices
- Real-world use cases
- API security checklist

Table of contents

Inside the API product mindset

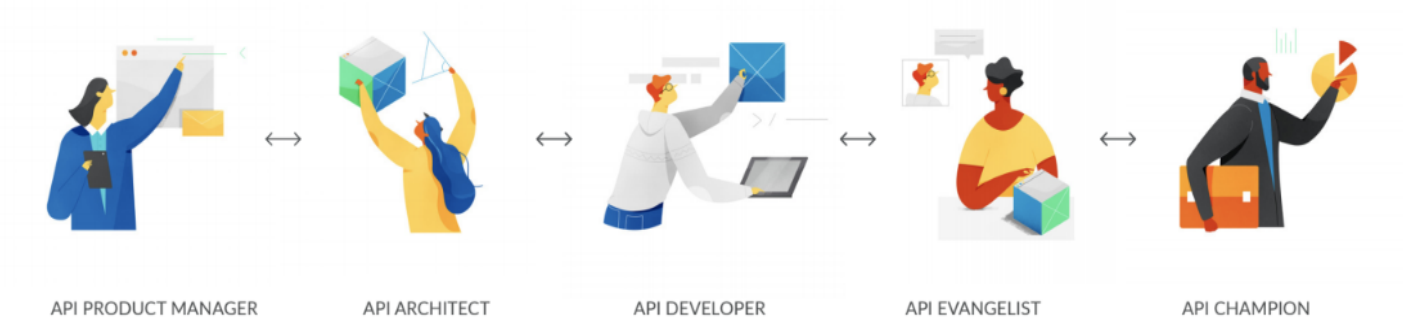
APIs (or application programming interfaces) are the de facto standard for building and connecting modern applications. With APIs, a business can securely share its data and services with developers, both inside and outside the enterprise, to foster new operational efficiencies, unlock new business models, and enable business transformation.

APIs are often characterized as products for developers who build the connected experiences that power the digital economy. All businesses have valuable digital assets—functionality, data, etc.—but many of the systems that contain this value were never meant to easily connect. APIs abstract this complexity into an interface that enables developers to leverage systems in which they have no expertise, and to combine digital assets in new ways.

In this way, APIs are not only expressions of a business's capabilities and points of differentiation but also the mechanisms that make those capabilities and differentiation leverageable for strategic purposes.

With this in mind, many successful organizations manage APIs like products, with full lifecycles, long-term roadmaps, a customer-centric approach, and constant iteration to meet business needs. In our experience in Google Cloud's Apigee team, organizations that treat APIs as products—as opposed to one-off *technology projects*—are more likely to realize the potential value of APIs as business accelerators.

As we explored in The API Product Mindset ebook, typical roles on an API product team include an **API Product Manager** who owns the processes and cross-functional coordination critical to the API's success; an **API Architect** responsible for designing and guiding the creation of APIs; an **API Developer** who builds APIs from the API Architect's designs and implements security policies and other protocols; an **API Evangelist** who serves as the voice of API consumers and owns partner and developer outreach; and an **API Champion** who works closely with internal executive sponsors to communicate the value of the API program to the rest of the organization.



The API product team typically carries several critical responsibilities:

- Design secure and easy-to-use APIs and bring them to market
- Deliver a world-class API developer experience
- Drive ongoing API improvements with monitoring and analytics
- Maximize the business value of APIs through API monetization, ecosystem participation, developer evangelism, etc.

This ebook dives deeper into one important aspect of this process: designing secure, easy-to-use APIs.

Balancing Protection and Ease of Use

APIs expose data or functionality for use by applications and developers, which means they are the doors and windows that allow access to a business's valuable digital assets—and thus to the heart of the business itself. Like all doors and windows that provide access to something valuable, APIs should be designed with security top of mind.

API developers generally understand the importance of adhering to API design principles because no one wants to design or implement a bad API. Even so, it may be tempting to look for shortcuts to meet aggressive sprint timelines, get to the finish line, and deploy an API. Though understandable, these shortcuts may pose a serious risk: unprotected APIs.

Vulnerable APIs can expose a business's core data and services to a variety of malicious attacks. Many large enterprises and organizations have suffered breaches, the consequences of which can range from embarrassing to catastrophic, as a result of holes in API defenses.

These holes are not always due to negligence. Developing secure APIs isn't as simple as it might seem, as the API provider must often strike a balance between ease of use and security.

One of the main goals of an API is ease of use; an API is of little value if developers aren't consuming it, and no one wants to work with an API that is so locked-down with security mechanisms that they get in the way of productivity. The challenge in API security isn't locking down the API; rather, it's managing APIs that are secure yet still flexible and easily-accessible enough to foster innovation.

Striking this balance means API providers should generally avoid the complex systems dependencies and heavy-handed governance models that typified previous generations of IT strategy, when connected digital markets hadn't yet risen to prominence and the corporate firewall was seen as a fortress to repel outsiders.

Indeed, legacy security practices in general may no longer be viable because they rely on limiting the rate at which enterprises update their software. Enterprises that persist in a slow, cumbersome update process run the risk of being unable to react to new vulnerabilities. For most organizations, the best path is to adopt technology strategies, such as API-first development, that support constant updates and provide a plane for implementing security controls.

As part of the API product mindset, enterprises should always consider the following API security best practices: data encryption, end user and application authorization, rate limiting, and bot detection.

Field-tested best practices

Don't mess with TLS

"Transport layer security," or TLS, is the foundation of API security. Protecting network traffic using an encrypted channel is the easiest way to ensure that sensitive information is not susceptible to attacks while in transit. Encrypting traffic over the network should be seen as an ironclad requirement; no API should go without it.

TLS also helps enterprises ensure that the client, such as a mobile app, is communicating with the correct server. For example, a free WiFi network in a coffee shop might be attached to a fake DNS (domain name system) server set up to route banking transactions to another site. Without TLS, a mobile app using this WiFi network could be open to attack. It's worth noting that though this feature of TLS is activated by default, a number of client libraries and tools make it easy to turn off or bypass this feature—don't do it.

API developers must also be cognizant of using TLS properly, as there are many different options. What cipher suite to use? What encryption algorithm to apply? These things change all the time—and it's important that companies stay on top of TLS changes, update their configurations by following the advice of internal security teams and external experts, and expect more changes in the future. If a team hard-codes particular TLS versions and cipher suites into servers, for example, they may make future updates more difficult. Many API teams test TLS configurations with services such as the SSL Server Test from Qualisys SSL Labs.

Beyond TLS, API developers must vigilantly keep up with the security world and should constantly assess and consider measures that go beyond basic encryption. For example, API providers might consider employing trace tools for debugging issues, data masking for trace/logging, and tokenization for PCI (payment card industry) and PII (personally identifiable information) data.

SECTION SUMMARY

TLS is the foundation

- Never turn off TLS.
- Keep up with TLS changes.
- Consider going beyond encryption with trace tools, data masking, and tokenization.

Ensure strong authentication for both end-users and applications

API teams should build APIs that provide authentication for both end users and applications.

OAuth is the de facto open standard for API security, enabling token-based authentication and authorization on the Internet. It provides a way for end users and applications to gain limited access to a protected resource without the need for the user to divulge their login credentials to the app. For APIs, it allows a client that makes an API call to exchange some credentials for a token, and that token gives the client access to the API.

Unlike a password, a token uniquely identifies a single application on a single device. A key best practice for API teams is to build authentication of all end users into critical applications, as it's the only way to keep security credentials outside the app. API teams building API products should familiarize themselves with the full capabilities of OAuth and current authentication best practices.

Security-minded API teams will additionally recognize that OAuth is not just about authenticating end users—authenticating applications is also a fundamental part of API security. For example, by authenticating applications, a provider can stop runaway applications that continue eating up resources when they should have stopped running. Some API teams have neglected application authentication, but fortunately, OAuth now natively includes this concept; in order to build an application that gets an OAuth token, a developer must supply not only user credentials but also application credentials.

Application authentication does not provide complete security against attacks, as anyone with application credentials can access and potentially abuse APIs. But it provides an important extra layer of defense. Developers should consider using different credentials for each version of an application to make it easier to pull a bad version.

Though OAuth is an incredibly useful security standard, it's made up of a complex family of specs, and there are numerous ways to use it. To make leveraging OAuth simpler, many API teams rely on API management platforms to generate OAuth tokens and apply granular control over what a token is allowed to do.

API platforms can also help enterprises improve security by managing access within the API team to sensitive resources, such as the developer portal. Features such as role-based access control (RBAC) help API teams to manage access according to roles that define user privileges. Such roles might include "organization administratrator" with full access to resources; "read-only organization administrator" with read-only access; and "business user" with access to tools to create and manage API products but read-only access to other resources.

SECTION SUMMARY

Don't neglect authentication

- Use OAuth to authenticate users.
- Authenticate both end users and applications.
- Consider RBAC to manage access.

Rely on rate limiting

API teams should always consider using rate limits for additional API security, as any API could be subject to a brute force attack. In a brute force attack, automated software is used to generate a large number of consecutive guesses as to the value of required data, such as a login password. If there is no rate limit, these attacks can continue indefinitely, with bad actors deploying a distributed password-cracking API that keeps running until it manages to infiltrate a system.

In the face of such threats, it is critical to apply basic rate limits to APIs. For example, an API team might establish a limit that forbids an application from calling an API more than 500 times per second or a certain number per day. To avoid against performance lags, downtime, and other backend degradation, it is also a good practice to enforce a spike arrest, which throttles the number of requests that can be made to an API during a traffic surge, or per-app quotas. As with OAuth usage, many organizations use API management platforms to help them apply rate limiting and spike arrest capabilities across their APIs.

SECTION SUMMARY

Keep brute force attacks at bay and manage traffic

- Use rate limits to protect against brute force attacks.
- Apply spike arrests to avoid performance lags or downtime during a traffic surge.

Beware of bad bots

As business-critical functions have shifted to connected devices, the automated connecting of software and systems has become an indispensable part of how business gets done. Unfortunately, automation has also enabled new forms of cybercrime—namely, bot attacks, in which bad actors deploy automated software programs over the Internet for malicious purposes, such as identity theft.

Many bots are useful. In fact, millions of bots play critical roles in enabling the API-powered connected experiences driving the digital economy. The key for enterprises is to enable beneficial automation without also enabling harmful bots.

Bad bots might arise when a hacker acquires a compromised API key, perhaps from a partner or mobile app, and then reverse-engineer how the app works in order to emulate necessary API call flows. From there, the hacker can run hundreds or thousands of bots at scale, producing scores of ostensible "users" who appear to be doing normal things, such as purchasing products or accessing loyalty accounts. Because the bots are actually working in concert, however, they can end up manipulating the prices of goods on auction sites, impacting how best-selling products appear on search or recommendation engines, or awarding a hacker millions of loyalty points they did not earn—just to name a few examples.

These risks point to the obvious importance of properly managing API keys—but that's only a start. API teams must also monitor not only API access, but how traffic behaves. They should look at the behavior of all the users coming in to identify who they are, where they come from, and most importantly, what they do, for example.

There are numerous approaches to stopping malicious behaviors, but technologies and approaches that work for the network or the web do not necessarily work for APIs. To thwart bot attacks on APIs, API teams' tactics should include sophisticated machine learning-based solutions that can analyze API request traffic, identify patterns that might represent unwanted requests, and learn as hackers rotate their attempts across a large set of bots.

SECTION SUMMARY

Use machine learning to put bad bots in their place

- Monitor not only API access but also traffic patterns in order to spot suspicious behaviors.
- Apply sophisticated algorithms and machine learning to spot bad bots, and note that approaches that discern network or web attacks may not be effective for APIs.

Real-world use cases

Urban Science: Protecting a Rapidly Expanding API Program

Urban Science serves customers in the automotive industry by making business recommendations and proposing solutions based on scientifically validated results. The company provides decision support services to most global automotive OEMs and retailers and also serves the healthcare and retail markets.

Building on its history of providing market intelligence to the automotive industry, Urban Science has been offering its Marketing Intelligence Cloud solution for several years to U.S. automotive marketers and their agencies to enable them to make more data-based decisions to improve the effectiveness and efficiency of their marketing efforts. The platform relies heavily on APIs created and managed via the Apigee API management platform to keep data flowing quickly, securely, and efficiently.

> *"Having real-time data at our fingertips, combined with our ability to take action quickly, is instrumental in keeping our product always-on."*
>
> Luke Mercier, Urban Science

Urban Science uses Apigee security features to help ensure that the 11.5 million monthly transactions that run through the Marketing Intelligence Cloud remain protected—and will remain so even with the platform's exponential growth.

Apigee offers an API management proxy frontend to the platform and is used to help secure all endpoints. There's no access point to the Urban Science Marketing Intelligence Cloud that doesn't go through the Apigee gateway and its security features.

With 650 published APIs available to third parties as well as to its internal development teams, this protection is particularly important. But API management security features do more than protect sensitive information.

Apigee dashboards run continuously on several screens, where the Urban Science API team monitors traffic down to the minute. This data is used to observe the Marketing Intelligence Cloud platform's growth, adoption rates by product, and adoption rates by customers.

Additionally, the data is used to help scale the platform and plan for more resources, CPU memory, and storage. At the same time, team members use alerting and notification features to enable them to take action on potential problems in real time, often resolving possible outages before they can affect customers.

"We've prevented several outages with Apigee alerts," says Luke Mercier, Urban Science's global system manager. "We've taken action to stop an issue from becoming a systemic failure. Having real-time data at our fingertips, combined with our ability to take action quickly, is instrumental in keeping our product always-on."

API security checklist

API security should be of paramount importance to any enterprise that is exposing digital assets. Here are some key aspects of security that business leaders and API teams should look for when evaluating API management platform providers:

- **Authentication and Authorization:**
  - ☐ Support authentication of users and applications with TLS, SAML, OAuth 2, two-factor authentication, API keys, and mechanisms to block or limit known bad actors or people who abuse terms of service
  - ☐ Manage user identity and RBAC by integrating with LDAP and active directory
- **Threat Protection:**
  - ☐ Protect against malicious activities such as XML poisoning, JSON and SQL injection, and DoS attacks and DDoS attacks
  - ☐ Detect and prevent bot attacks in real-time using machine learning techniques
  - ☐ Provide quota and spike arrest and IP blocking capabilities to act against API abuse
- **Privacy and Compliance:**
  - ☐ Provide the ability to log all actions, and the ability to audit logs
  - ☐ Compliance with SOC 2 (Service Organization Control), PCI DSS (Payment Card Industry Data Security Standard), and HIPAA (Health Insurance Portability and Accountability Act)
- **Scale and Compute:**
  - ☐ Handle a massive number of API keys and tokens and compile policies in runtime

About Apigee API Management

The Apigee API management platform delivers full lifecycle API management to help businesses unlock the value of data and securely deliver modern applications. Apigee offers a rich set of security capabilities to help enterprises protect their digital assets. To learn more about how Apigee's authentication, monitoring, rate limiting, and bot protection capabilities help API product teams throughout the world, visit secure APIs.

Now that you've finished reading, why stop learning?
Visit the Apigee website for more.

apigee

Share this eBook

on social

with a colleague

Google Cloud