SRC CYBER SOLUTIONS LLP

THREATX

A Security Practitioner's Introduction to API Protection

Five Requirements for Protecting APIs Against Attacks

INTRODUCTION

The acceleration of digital transformation initiatives is fueling API growth.

In fact, APIs have become the de facto standard for distributed computing as developers increasingly rely on them to quickly deliver new digital services and capabilities. Unfortunately, as often happens in IT, the rapid proliferation of APIs has surpassed security's ability to protect these assets.

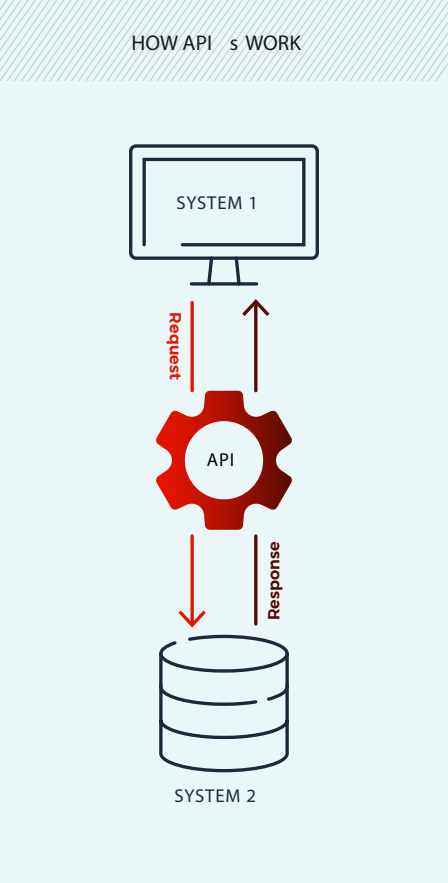
According to Gartner, by 2023, over 50% of B2B transactions will be performed through real-time APIs.

As a result, APIs represent a rapidly growing attack surface. Gartner believes that APIs will be the majority of the attack surface for 90% of web-enabled apps. This is an attack surface over which most security teams have little or no visibility and control. By their very nature, APIs expose application logic and sensitive data, and nothing is stopping attackers from exploiting them. Many high-profile security breaches — think Peloton, Venmo, Facebook, and the U.S. Postal Service, to name a few — were the result of unprotected APIs.

Protecting APIs is critical for protecting valuable business assets and preserving the organization's ability to rapidly innovate. But the task requires more than an API gateway or web application firewall (WAF). API-based attacks are highly sophisticated, and API protection demands an equally sophisticated defense. This white paper introduces the security risks associated with APIs and the five requirements for protecting APIs against attacks.

The Role of APIs in Modern Development

Nearly every modern software application uses – or is – an API. So, what exactly is an API?



Simply put, an application programming interface (API) is an application that enables two or more software components to communicate with each other using a set of definitions and protocols. Most use HTTP for transport. The API documentation contains information on how developers are to structure the requests and responses. Some APIs are employed in server-to-server connections. More commonly, APIs are used as the backend to mobile apps or JavaScript-powered web user interfaces.

APIs have become central not only to the way applications are developed, but also to scaling business strategy. APIs provide developers with a fast, highly modular way to add capabilities to their applications. The reuse of APIs helps facilitate rapid development by eliminating the need to build out common functionality from scratch. For example, instead of spending time developing their own mapping service, developers can integrate an existing service using an API.

APIs also facilitate systems integration. Developers use APIs to connect services and transfer data, allowing users to access the same data across multiple solutions. APIs can also automate repeatable tasks for better efficiency and accuracy.

PROBABLY MOST IMPORTANTLY

APIs provide an adaptive way to work with mobile devices and cloud applications.

The State of API Security

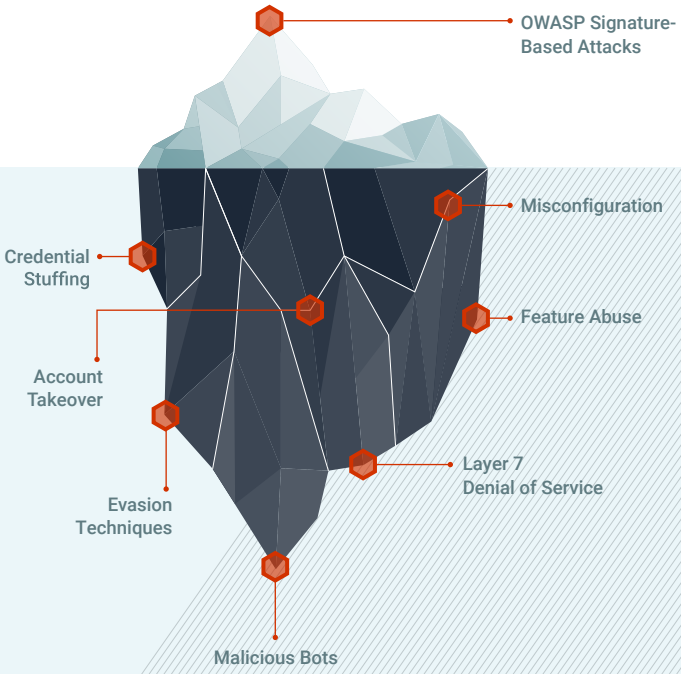
Despite all their benefits, APIs also introduce risk.

So much so, in fact, that the Open Web Application Security Project (OWASP) Foundation publishes a dedicated list of API-specific vulnerabilities. First published in 2019, the OWASP API Top 10 serves as an addendum to its list of common web application vulnerabilities. But while the OWASP API Top 10 is a start in the right direction, it is not sufficient to protect APIs in today's complex threat environment.

Whereas traditional forms-based web applications were discrete applications, public-based APIs, by their very nature, expose application logic and sensitive data such as personally identifiable information (PII) — making them a prime attack target. For example, an attacker can experiment by submitting invalid data to the APIs until something breaks. An attacker can catalog system objects, often by their JSON representation, and gain access to objects that they shouldn't be allowed to.

Many of the most common API vulnerabilities are related to authentication and authorization flaws. For instance, developers may loosely configure an API's authorization checks based on assumptions made about the clients interacting with it. This leaves application functions or API calls available to any attacker who steals the API keys or — worse — who discovers an API that doesn't require keys.

Even when developers configure APIs with strong authentication requirements, they may violate the principle of least privilege by granting API clients access to more data and application functions than necessary. This practice stems from a well-meaning intention to present information as completely as possible and to future-proof an API, making it applicable for applications that were not originally envisioned at the time of development. However, this practice presents a fair amount of risk. The data returned via the API can represent an information leak or otherwise function as an attack vector.



Unfortunately, traditional WAFs and many API security solutions are not designed to stop sophisticated API attacks.

Even addressing the OWASP API Top 10 still leaves APIs vulnerable. Attackers exploit legitimate application functionality and leverage the flaws buried deep in the application logic. They combine multiple evasion techniques with various attack types, making it nearly impossible for a signature-based solution to identify every combination. They can use thousands of bots to distribute the attack and vary user signatures while mimicking legitimate user behavior. A traditional application firewall can't protect against this activity. If the malicious activity looks like normal activity to the firewall, the request won't be blocked.

SRC CYBER SOLUTIONS LLP

THREATX

Doesn't my API gateway/SAST/DAST solution do that?

THE SHORT ANSWER IS NO.

API gateways and static application security testing (SAST) and dynamic application security testing (DAST) scanners do not provide API protection.

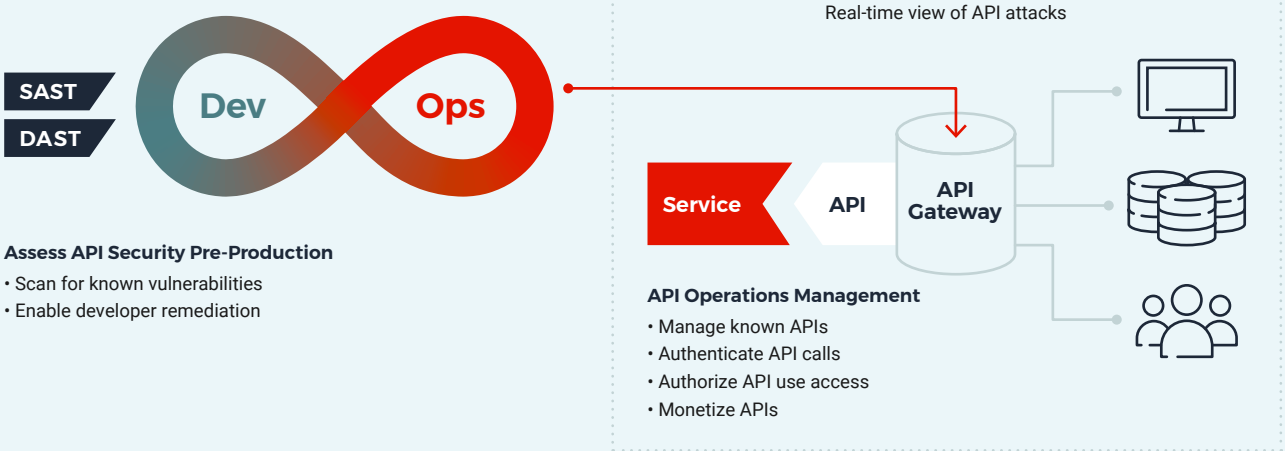
SAST and DAST scanners are an important part of a security program, but they are primarily used to identify known vulnerabilities pre-production. They can't detect what appears to be legitimate user behavior executed by attackers in production code.

While API gateways play an important role in controlling and monitoring authentication, their focus is on API operations. API gateways cover a variety of operational tasks such as controlling access, translating a RESTful request to SOAP, or elastically scaling resources if there is a spike of traffic hitting an API.

That said, it's worth noting that API and cloud infrastructure providers can't afford not to offer some form of API security. However, security as an add-on as offered by these providers lacks the depth and breadth needed to provide adequate protection. API gateways excel at the operational aspects that keep APIs running properly. They are not designed to provide API protection capabilities.

In addition to providing sub-optimal protection, gateways create siloed views and increase operational overhead. As organizations expand into hybrid and multi-cloud deployments, the provider-specific point solutions accumulate. An Amazon gateway has different capabilities than the Google gateway, and neither has the other's context. This can be to an attacker's advantage as they string together vulnerabilities and attacks to create a complex kill chain.

API threat protection solutions support all API infrastructures as well as the many traditional web assets with the sole purpose of monitoring and identifying risky behavior and attacks. These solutions focus on understanding the API attack surface, detecting and blocking the ever-growing spectrum of modern API threats, and keeping up with the changing techniques attackers use to evade detection. An API threat protection solution provides an integrated view of risk, which is essential for repelling modern attacks and demands considerable expertise and focus to keep pace with evolving threats.



SRC CYBER SOLUTIONS LLP

THREATX

Five Requirements for Protecting APIs

Understanding the security challenges of APIs, let's look at the five requirements for protecting them.

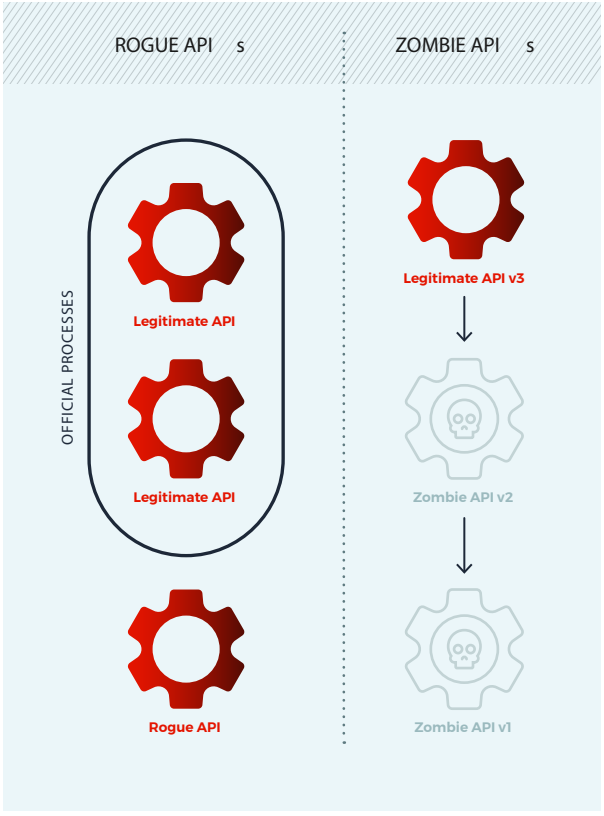
1 Discover Known and Unknown APIs

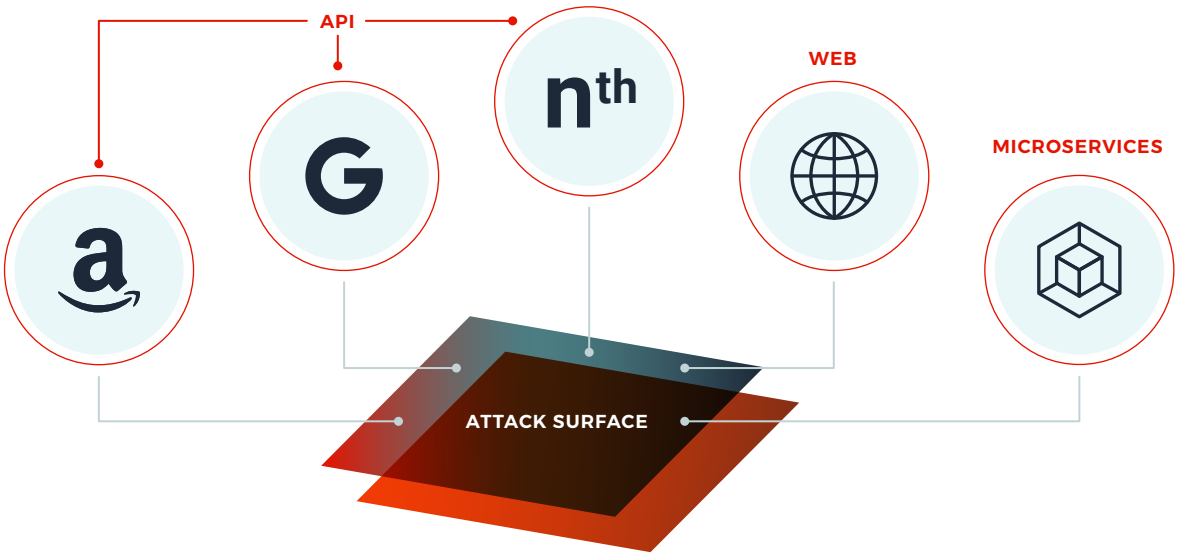
The attack surface created by APIs is significantly larger than most organizations realize.

The proliferation of APIs across a distributed infrastructure inevitably leads to API sprawl, and organizations often don't know how many APIs they have or how they're utilized. Some of those APIs may be known, but organizations often have a host of unknown APIs as well.

Like shadow IT, rogue or shadow APIs are APIs that exist outside of an organization's official security and operational maintenance processes. Countless scenarios can lead to a rogue API. A developer may need to quickly stand up an API to resolve an immediate problem or to develop a proof of concept for a larger project. Regardless of the motivation, the lure of quickly deploying a new API endpoint makes it easy for developers to get their work done in the moment and worry about security and maintenance later.

Similarly, building the next piece of code or revenue-generating application tends to take priority over sunsetting or deprecating old APIs. Zombie APIs are APIs that were previously valid and approved but were eventually abandoned or replaced by newer versions. Sometimes these old APIs are intentionally left in production to support legacy versions of an application. But if these APIs aren't properly removed, they can leave organizations with outdated, vulnerable points that attackers can exploit.





The highly distributed nature of today's IT environments also contributes to API sprawl. APIs are often deployed across multiple environments and tools. Developers stand up APIs behind multiple gateways (and sometimes no gateways) due to different deployments across multiple cloud and on-premises environments.

Ideally, an API's schema provides insights on API utilization. An API schema provides definitions of acceptable use for that API, including what can be exposed, what methods can be used, what are the parameters, keys that enforce utilization, etc. But schemas frequently don't exist.

DISCOVERY MODE

The best mode of discovery is real-time analysis of the traffic hitting those endpoints. Analytics can identify which endpoints are no longer used, and which clients are still actively using old endpoints.

ACCORDING TO OUR RESEARCH

70%–80%

of APIs in the wild have no formal schema. The schema definitions are often neglected, not created, and/or not maintained well.

2 Visualize Risk to Your API Attack Surface

Understanding and protecting any attack surface requires visibility.

API discovery capabilities are needed to allow security teams to visualize the entirety of their API attack surface and observe the utilization of APIs in both real time and over time. This is key to understanding how APIs are being used and to identifying rogue and zombie APIs. As part of the discovery, key metrics should be gathered such as request counts, methods, parameters, and error codes, plus how APIs are utilized and how they're being invoked.

REAL-TIME TRAFFIC ANALYSIS

Multi-step attacks can be much harder to detect as attackers build on earlier reconnaissance and information leaks. Visibility into real-time traffic analysis is critical for detecting key indicators of early-stage reconnaissance.

ATTACKER-CENTRIC VIEW

Attackers use bots and automation to blend in with normal user traffic and abuse application features while also obscuring the true nature of the attack. Attacks will often develop over time as attackers steadily look for weak spots in defenses while staying below the thresholds of rate-based rules.

Continuously monitoring API behavior and correlating behavior over time can produce a unified risk score that increases based on attacker behavior. This allows security teams to see coordinated attacks and to identify and stop low and slow attacks that would otherwise fly under the radar.

ATTACK DASHBOARD

Real-time traffic analysis can generate an overwhelming amount of data. Security teams need an attack dashboard that provides a centralized view of how and where APIs may be deployed.

The dashboard should highlight the most important data, such as endpoints with potential vulnerabilities, abnormal traffic and utilization, malicious attackers, techniques utilized and what they targeted, and prioritize issues with visualizations that efficiently and effectively convey risk.

3 Detect and Block Real-Time Attacks

Real-time scanning of all inbound API traffic is critical to effectively identifying and blocking attacks.

BLOCK AUTOMATED, MULTI-STEP ATTACKS

Multi-step attacks involving bots that mimic human behavior can be very difficult to detect. Traffic must be analyzed well enough to separate legitimate from illegitimate traffic without impacting application performance. Real-time monitoring should include advanced threat engagement techniques, such as IP fingerprinting, interrogation, and tar-pitting. These capabilities enable a platform to identify and stop the most complex attacks, including large-scale bots and DDoS-level attacks.

STOP ATTACKS IN REAL TIME

Offline solutions can develop sophisticated correlation rules to detect complex attacks, but they have a couple of disadvantages. Offline solutions can't take advantage of real-time, interactive techniques like IP interrogation and fingerprinting, active deception, or user-level tar pitting/rate limiting. Attack identification often occurs well after the fact, increasing the time to detection and the risk of data loss.

TRACK THE ATTACKER

Detection of complex, multi-step attacks can be difficult to model in legacy static signature firewall solutions and API gateways. Most legacy WAFs look at individual ingress transactions and make simple decisions: good traffic or bad traffic. They look at each inbound request and try to match it to a known signature. For instance, most OWASP types of signatures look for an exact expression match within a request, and it either matches or it doesn't. But with a multi-step, automated attack, it's difficult to detect and block the right ones without causing false positives or writing hundreds, sometimes thousands, of rules.

Attacker-centric behavioral analytics is key to minimizing false positives while effectively stopping sophisticated, multi-step API-based attacks. The system identifies and correlates activity to more precisely identify threats to APIs. The platform responds to multi-step attack patterns over time, adjusting to the attacker's behavior and blocking suspicious entities and IPs when behaviors surpass an acceptable risk threshold.

EXPERT HELP

A good API protection platform is supported 24/7 by trained expert security professionals who use attacker behavior analytics, application profile and traffic, and known attacks and patterns to analyze all HTTP traffic to APIs and applications. This allows the SOC to apply risk scores to potentially harmful traffic. As potential attackers progress through the kill chain, they are blocked and reported for further investigation.

4

Expose Insights to Enable Attack Forensics

Most blocking solutions are black boxes.

Not only is it difficult to determine the general efficacy of these solutions, but they don't provide enough information during major attacks to respond quickly. And without adequate information, teams can't build policies or rules to respond.

Security teams need insight into whether malicious traffic should be a concern and dictates additional security measures or hardening. Security teams need to understand the level of sophistication and characteristics of attackers. Are they executing high volume (spray and pray) or low and slow attacks leveraging highly targeted techniques? How persistent and perseverant are the attackers? It's also important to know what assets are being targeted and why. Are certain APIs more vulnerable due to external interest, poor security hygiene in development, or highly vulnerable technologies? Are peer organizations seeing a spike in traffic, too?

Visibility drives understanding, which leads to appropriate responses. Through advanced risk analysis, a platform can identify key attributes of an attack, such as attack patterns over time (e.g., low and slow), the use of advanced evasion techniques, and details of the attack target. These insights allow security teams to understand the goals and nature of a threat, in turn enabling a more comprehensive security strategy that reduces future risk.

5

Assess and Enforce API Schema Compliance

The last requirement for a modern API protection strategy is the ability to ensure that API functionality is aligned with the organization's stated goals and objectives.

While not all API protocols have easily definable specifications, the vast majority of APIs currently deployed are RESTful APIs and can be described by an OpenAPI Specification (OAS) and defined in a JSON Schema. The Schema can be used to map an API's endpoints, input/output parameters, methods, and authentication methods. GraphQL, gRPC, Kafka, and others have different protocols, but the strategy is the same in terms of specifying how the API should be used.

Not only is a specification a best practice as part of API design and implementation, it's also a great way to evaluate API utilization and ensure it conforms to the specification. One of the best indicators of malicious activity is attempted misuse of an API. Attackers are constantly probing API endpoints to identify any number of vulnerabilities that can be exploited. Comparing actual utilization against what's expected is one of best ways to identify malicious users and the techniques they are using. Given that many sophisticated attacks are distributed across hundreds, if not thousands, of IPs, detection often requires complex behavior analysis and tight coupling with blocking capabilities to mitigate.

Unfortunately, many companies struggle to consistently manage and maintain schemas. Multiple API gateways, different technology stacks, and plain old lack of hygiene can impact the efficacy of schema compliance.

SOLUTIONS

The best solutions include API discovery based on real traffic and the ability to compare traffic metrics against a "Security Schema" based on integration of legitimate schema files or to build a customized version on the fly based on actual traffic.

CONCLUSION

As critical building blocks of modern applications, APIs represent one of the fastest-growing facets of an organization's attack surface.

Security teams need a holistic solution that helps them get in *front* of the API security problem. ThreatX offers the fastest and simplest way to discover, catalog, and protect all your API endpoints. ThreatX's API protection platform provides instant API protection for all APIs, regardless of their status or which gateway they sit behind — no schema required. You're protected without any separate discovery, integration, instrumenting of your code base, or talks with your development teams. Plus, ThreatX backs up automated detection with a security operations center (SOC) that can intervene to block new attack patterns 24/7, at no extra charge.

Learn more about ThreatX's API Attack Protection →

SRC CYBER SOLUTIONS LLP

THREATX

www.srccybersolutions.com
sales@srccybersolutions.com
+91 120 232 0960 /1

ABOUT SRC CYBER SOLUTIONS LLP

At SRC Cyber Solutions LLP, we provide Next Generation, Automated and User-Friendly solutions in partnership with AUTOMOX for Patch and Endpoint Management, IRONSCALES for Comprehensive Email Security and Anti-Phishing Protection, THREATX for WAAP (WAF++) with an Attack-Centric approach for Web Application and API protection and Project Ares for Online Gamified Simulation-based Cyber Security Training.