AppSentinels.ai
Application Security Re-invented

OWASP Web Top-10

vs

OWASP API Top-10

Illusion of Security due to similarities?

In 2019, OWASP released first version of API Security Top 10. Like the omnipresent OWASP Top 10, the API Security Top 10 delivers a prioritized list of the most critical application security issues with a focus on the APIs. In this whitepaper, we would like to share an overview of the API top 10 with comparisons to the OWASP top 10 for web applications and break any false sense of security by seeing similarities in the list.

APIs – the Foundations of Applications

APIs are foundational element of innovation in today's app-driven world. Whether it is monolithic, micro- services, serverless or no-code frameworks, APIs are everywhere. From banks, retail, and transportation to IoT, autonomous vehicles and smart cities, APIs are a critical part of modern mobile, SaaS, and web applications. APIs can be found in customer-facing, partner-facing and internal applications. As per an Akamai report, in 2019 API's were contributing 83% of the overall internet traffic.

While API's have made development faster and applications more dynamic, they have presented new set of security challenges and possibilities for hackers. By nature, APIs carry sensitive information and are land directly on crown jewels of an organization. They are fast becoming preferred attack vectors for application attacks. Gartner predicts that by 2022, APIs will be the most frequent attack vector leading to breaches for web applications.

Enter the API Security Top 10

With the growing adoption of APIs, it is important API's get attention of the Security leaders when it comes to AppSec, and the API Security Top 10 is a great step in that direction.

Upon first examination, the API Top 10 shares a lot in common with the traditional Top 10. In fact, 6 out of the 10 categories in the API Top 10 are also in the traditional Top 10 or have a close resemblance. This is not surprising, as while APIs provide new avenues to hackers, they also share many of the challenges and weaknesses of web applications.

The table below provides a side-by-side comparison and highlights the areas in the API Top 10 that are also found in the 2017 OWASP Top 10.

OWASP API Security Top 10	OWASP Security Top 10 (2017)
API1: Broken Object Level Authorization	A1: Injection
API2: Broken User Authentication	A2: Broken Authentication
API3: Excessive Data Exposure	A3: Sensitive Data Exposure
API4: Lack of Resources & Rate Limiting	A4: XML External Entities (XXE)
API5: Broken Function Level Authorization	A5: Broken Access Control
API6: Mass Assignment	A6: Security Misconfiguration
API7: Security Misconfiguration	A7: Cross-Site Scripting (XSS)

API8: Injection	A8: Insecure Serialization
API9: Improper Assets Management	A9: Using Components with known vulnerabilities
API10: Insufficient Logging & Monitoring	A10: Insufficient Logging & Monitoring

However, AppSec teams should not be trapped into a false sense of security because of these similarities. APIs introduces many new issues and challenges that organizations need to account for. API breaches are zero-days business logic attacks that are blind-spots of current traditional tools like WAF, RASP or SAST.

Let us briefly look at the categories where the OWASP Top 10 and API Top 10 are similar, and then where the API Top 10 introduces new concepts that deserve closer look by AppSec teams to plan defences.

Similarities Between the API Top 10 and OWASP Top 10

API2: Broken User Authentication

API2: Broken User Authentication is directly comparable to A2: Broken Authentication. They both address issues such as the susceptibility of the application to Brute Force and credential stuffing attacks, weak passwords, and the exposure or weak management of session IDs. However, the API2 also calls out the broad challenge of monitoring all the many flows for authenticating to the API. It also calls out treating API's responsible for authentication differently and with stricter controls following principles seen in privilege access management tools. Additionally, teams need to be aware of properly securing JSON Web Tokens (JWT) and proper authentication hygiene such as not using API keys for authentication.

API3: Excessive Data Exposure

At a high level, API3 looks related to A3: Sensitive Data Exposure. However, they do have a vastly different set of issues. The traditional OWASP A3 takes approach of encryption and retention of data, ensuring data is encrypted at rest and in transit and secured while in use. The API3 conversely focuses heavily on the issue that APIs often rely on the client to filter data in API responses. This could allow an attacker to easily sniff API response traffic to see sensitive data. This vulnerability requires that developers take the added time to specifically pick and choose the data that needs to be returned from the API rather than simply relying on the client to filter sensitive data.

API5: Broken Function Level Authorization

API5 is like A5: Broken Access Control. Both controls call out the risks where a user could access functions that they should not have access to through simple modifications of URL, such as changing a GET method to a PUT, or guessing URL parameters for sensitive functions. In terms of complexity of executing such attacks, the two vary vastly. In API's these attacks are extremely simple to execute and can cause much bigger harm.

API7: Security Misconfiguration

API7 maps directly to A6: Security Misconfiguration. Both controls focus on the identification of unpatched flaws, exposed accounts, assets, and other information that could lead to unauthorized access or knowledge of the application like returning stack trace in an error response that can provide unintended details about the application architecture to a hacker.

API8: Injection

API8 maps directly to A1 of the OWASP Top 10. In addition to the need to validate input for a variety of malicious inputs, security teams will need to account for additional challenges of APIs. This includes the ability to sanitize all inputs– even from internal sources such as internal modules or services.

API10: Insufficient Logging and Monitoring

Both the API Top 10 and OWASP Top 10 end with 'Insufficient Logging and Monitoring'. These issues highlight an all-too-common problem that allows threats to go unnoticed for extended periods of time simply because organizations lack the alerting and visibility into security events in their applications. While this is a common issue in traditional web applications, it is greatly exacerbated in APIs which often receive far less threat-based monitoring and alerting as compared to the web frontend.

New Categories in the API Top 10

API1: Broken Object Level Authorization

One of the most significant differences in the API Top 10 is the importance placed on object-level authorization. This is an incredibly common problem in APIs. As the API Top 10 states, *"APIs tend to expose endpoints that handle object identifiers, creating a wide attack surface Level Access Control issue."* This can lead to problems when user-based access controls discussed previously (API2) are not checked and enforced at the object level. Additionally, developers should ensure they use "random and unpredictable values as GUIDs for records' IDs" to prevent attackers from easily manipulating or guessing commands.

API4: Lack of Resources & Rate Limiting

It is important for security teams to understand how users are consuming resources to avoid denial-of- service attacks. And while rate limiting is discussed in other sections of the OWASP Top 10, it gets called out specifically in the API Top 10 due to the frequency of attack in the wild. In addition to tracking standard aspects of an application such as the number of overall requests, security teams also need to track resource-intensive requests that could consume large amounts of memory or compute on the application.

API6: Mass Assignment

Applications and their APIs often need users to update information such as their contact information or other personal data. However, problems can occur if the application extends

this capability to other internal properties such as a state flag or other setting that should only be set by an administrator or the application itself. For example, an attacker might find that in addition to updating the user's mailing address, he can also update the amount of funds in his account, allowing him to effectively steal from the application. As a result, applications should "avoid using functions that automatically bind a client's input into code variables or internal objects" and use whitelists and blacklists to further control user access to specific variables.

API9: Improper Assets Management

This issue relates to the prevalence of old or unpatched APIs that can accumulate in an application unnecessarily. These may be APIs that were used in the development and testing process or for older versions of the application but were never carefully removed or retired. These APIs are often vulnerable or poorly secured and can allow an attacker to easily abuse the application without the need for a more sophisticated exploit. This issue underscores the importance of enumerating all API hosts in an environment, understanding their purpose, age, and susceptibility to attack. Likewise, organizations should be able to apply monitoring and protection against all API hosts to protect them from abuse.

Conclusion

Here we have briefly introduced some of the high points of the OWASP API Top 10 and how it both aligns with and differs from the OWASP Top 10. While this introduction provides a basic roadmap for some of the more common API-related security issues, there remains a wealth of additional details and considerations when teams need to implement a robust API-ready security program. AppSentinels provides a unique approach to application security that provides effective security both to traditional web applications and APIs. Additionally, the AppSentinels platform detects both known and unknown attacks using a combination of advanced AI/ML models that build complete context of an application, track every user activity, map data flow including sensitive and PII and more. Further, AppSentinels support dev-ops friendly insertion and requires no agent or instrumentation. This allows organizations to secure their applications in any state from a full spectrum of threats.