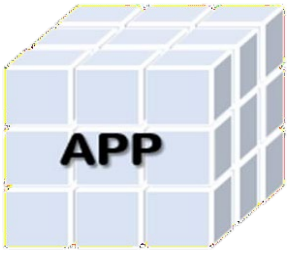
WEBFRONT-K API Security White Paper

OWASP API Security Top10 Response & API Security White Paper

Modern Application, MSA and API inside of it

In today's fast-changing world, businesses strive to develop innovative applications that save time and money while improving reliability. This application is called a modern application ("modern app").



[Figure 1] Architecture of modern apps

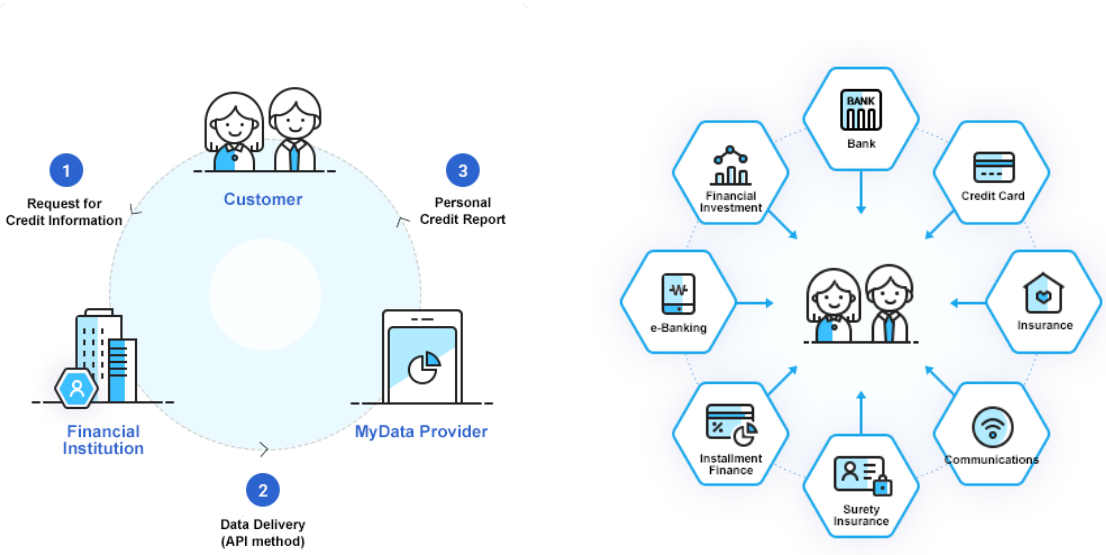
MSA (MicroService Architecture) is at the heart of what made this modern app development.

MSA allows applications to be developed by defining and creating systems through the use of small independent and self-contained services that align closely with business activities. This is similar to division of labor in the manufacturing industry where it improved productivity in the past. Most web/mobile applications and IoT devices are developed based on MSA. Just as MSA is at the center of the modern apps, API (Application Programming Interface) is the important technical concept to MSA. API is the interface which connects all the different applications, and it plays an important role of binding each service into one. API, therefore, is commonly used in application development.

Open Banking (MyData) and Standard API

API has proven its reliability because it has not only provided an innovative concept for modern app, MSA, but also it has been used and verified in websites since the 2000s. In addition, it also has a high level of security as it allows SSL communication. As of January 5th, 2022, API is used as a mandatory standard in Open Banking (MyData) in Korea.

When providing services related to personal information, the financial industry should use API, not web scraping.



[Figure 2] Concept of financial MyData (Source: MyData Portal)

When using web scraping, there is a high risk of leaking customer account information by bringing all paged data. Another risk of Web Scraping is its vulnerability of collecting excessive personal information.
These risks are the main reason behind the transition from Web Scraping to API Does using API as a standard ensure safety without vulnerabilities? The answer is no. The increased use of API generated many cases of security vulnerability. For instance, a total of 2,942 CVE vulnerabilities were registered by 2021. Now, let's take a look at API security.

WAAP and OWASP API Security Top 10

The following explains how global institutions and companies organize and respond to the concept of API security. This will be very helpful for those in charge of security who have not yet established an API security policy.

First, let's take a look at the proposal and analysis offered by Gartner, a global IT market research firm. In 2019, Gartner proposed the concept called Web Application and API Protection (WAAP), the model evolved from Web Application Firewall (WAF) with 4 key functions (such as application protection, DDoS defense, bot management and API protection). Gartner expressed the importance of API applications. In 2021, a market analysis report called 'Magic Quadrant for WAAP 2021' was released, and WAAP was officially introduced to the market as an independent product line. Gartner defined API security as an extended concept of WAF, which should be controlled with application protection, DDoS protection and bot management.

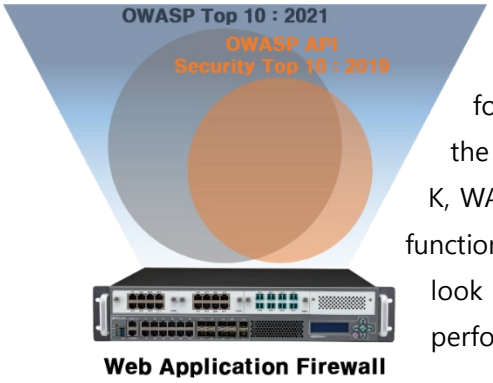
In the same year that Gartner presented the WAAP concept, the International Web Security Standards Organization (OWASP) also published OWASP API Security Top10:2019. OWASP is the institution that provides vulnerability reports every 3 to 4 years by analyzing security vulnerabilities of web applications. It published this separate API vulnerability report as the use of API increased in the application environment.

However, many of its vulnerabilities overlapped with existing OWASP Top 10. In addition, as the trend of application development moved to Clouds, many of the vulnerabilities included in OWSAP API Security Top 10: 2019 also became on the list of OWASP TOP 10: 2021. This is a natural result because the operating environment of the API exists in the category of Web. In particular, REST API, a most frequently used one, is operated based on HTTP protocol and it uses JSON or XML, which is commonly used in web application, for its data transfer. Therefore, the vulnerabilities of web applications can also be those of API.

OWASP TOP 10 : 2021	OWASP API Security Top 10 : 2019
A01 - Broken Access Control	A01 - Broken Object Level Authorization
A02 - Cryptographic Failures	A02 - Broken User Authorization
A03 - Injection	A03 - Excessive Data Exposure
A04 - Insecure Design	A04 - Lack of Resource & Rate Limiting
A05 - Security Misconfiguration	A05 - Broken Function Level Authorization
A06 - Vulnerable and Outdated Components	A06 - Mass Assignment
A07 - Identification and Authentication Failures	A07 - Security Misconfiguration
A08 - Software and Data Integrity Failures	A08 - Injection
A09 - Security Logging and Monitoring Failures	A09 - Improper Assets Management
A10 - Server Side Request Forgery	A10 - Insufficient Logging & Monitoring

[Figure 3] OWASP Top 10: 2021 vs. OWASP API Security Top 10: 2019

It can be concluded that the key to API security is the security of web applications. A lot of institutions and companies have already well-managed the security of web applications with WAF, a security device.



[Figure 4]
Concept diagram for OWASP Top 10: 2021 and OWASP API Security Top 10: 2019

Of course, conventional WAF functions may not be enough for API security. However, WAF has already been evolved into the one with API security functions like WAAP. The WEBFRONT-K, WAF of PIOLINK, has already been equipped with API security functions to keep pace with global market trends. Now, let's take a look at what core technologies WEBFRONT-K is based on to perform API security.

6 Key technologies for API security

As we mentioned, API security is based on the security of web applications. Therefore, a certain level of API security is possible with functions like a response to attacks (e.g. Buffer overflow, Injection and XSS) on web applications, prevention of sensitive information leakage, a permission list, a block list, access log control and blocking credential stuffing. However, in order to respond to OWASP vulnerabilities, which is the most important in application security, the following technologies must be equipped.

1) Mutual TLS (mTLS)

Since TLS has been used for security connection between clients and servers for a long time, those who work in the information security and IT sectors would know its concept very well. The existing servers with TLS applied can be accessed by any clients. However, in this Zero-Trust Security era, allowing all clients to access can be seen as weak in security. For a server that mainly provides sensitive information through API, it may be the best solution to allow access only to specific clients. This led to the introduction of mutual TLS (mTLS). mTLS is a technology in which the client verifies the server and vice versa, the server verifies the client. That is, after storing the certificates for client and server together in the web firewall, authentication is performed as a server when communicating with a client and as a client when communicating with a server. mTLS is a very important technology for API security, as it is included in the 'Open Banking(MyData) Technology Guidelines' issued by the Financial Security Institute.

2) Cloaking identification information

When some identification information is needed at the API endpoint, the client requests it with the identification information. However, if the request field is exposed to the attacker, the attack on Broken Object Level Authorization (BOLA) can be made (OWASP API Security Top10: 2019 A1.)

For example, if shopName mentioned below is assumed as the information with which the name of a shop can be identified, sensitive data including revenue can be accessed by changing the information into other names.

e.g.) /shops/{shopName}/revenue_data.json

As it can be abused easily, WAF should respond to such threats with encryption or masking.

3) API token authentication and integrity test

Vulnerability related to authentication is ranked as one of the most fatal vulnerabilities in all kinds of application environments including API and is included in OWASP API Security Top10: 2019 A2 and OWASP Top10: 2021 A7. Then, how is API different from other existing authentication relevant vulnerabilities?

API that often uses JSON controls the authentication and authority of users by using JSON Web Token (JWT.) As JWT is composed of header (types and algorithms of a token), payload (client information) and signature (hash values made of a secret-key of header, payload, and a server), the forged/falsified JWT without the secret-key used for signature cannot match up with the original JWT. WAF, therefore, can verify the integrity of JWT given by a client by using a secret-key of a server.

4) Limiting the thresholds of each API and methods

A server needs resources like network bandwidth, CPU, memory, storage, etc. to properly process service requests. If such limited resources are not well-managed with appropriate policies, vulnerabilities caused by lack of resource and rate limiting can lead to attacks like DoS/DDoS (OWASP API Security Top 10: 2019 A4.) To protect resources from vulnerabilities, the policy that sets the thresholds of requests is needed. In case of API, in particular, it is needed to limit the thresholds of each API transaction and HTTP methods according to usage purpose.

5) JSON response cloaking

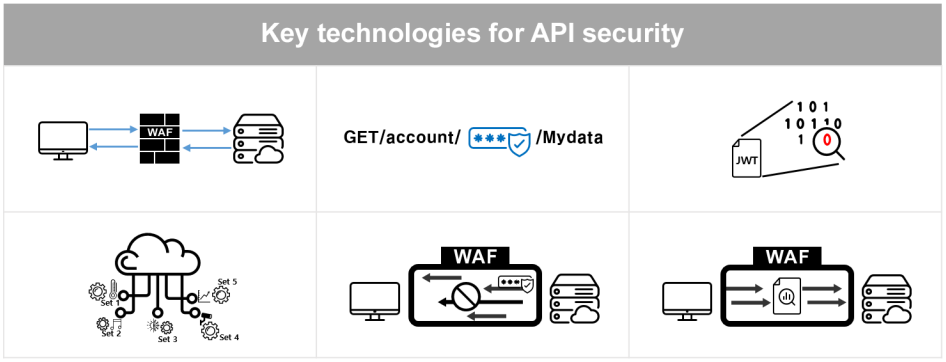
API, depending on its implementation types, can be vulnerable in a way that sensitive information can be exposed by providing excessive amounts of data (OWASP API Security Top 10: 2019 A3, A6). Through cloaking, such excessive data exposure needs to be blocked. The key point of API security is the capability of applying cloaking to JSON content type, which is a common API transmission.

6) JSON request field test

No matter how secure the system is, if the operation is sketchy, it can lead to serious vulnerabilities. This point is well-revealed in OWASP API Security Top 10: 2019 A7 and OWASP Top 10: 2021 A5 under the same name of Security Misconfiguration. Then, what functions does WAF use to respond to such vulnerability? First, it uses the function of examining the request field sent from a client and deciding whether it is justifiable. Also, as specified in JSON response cloaking above, the key point here is whether the technologies of analyzing and responding to JSON content type are supported for API security.

WEBFRONT-K equipped with key technologies of API security

Until now, we have looked through the key technologies that WEBFRONT-K uses for API security. Checking once more, if it has the detection and response technologies of existing WAF as well as the 6 key technologies (mTLS, cloaking identification information, API token authentication and integrity test, limiting the thresholds of each API and methods, JSON response cloaking and JSON request field test), it can respond to API-related general vulnerabilities including OWASP API Security Top 10: 2019.



[Figure 5] 6 key technologies of WEBFRONT-K for API security

API Security – It is a must, not an option

API security is in fact, one part of web application security that most institutions and companies have well-managed so far. The reason why it seems unfamiliar is simply, putting not enough attention on it. As explained in this White Paper, if we respond to OWASP vulnerabilities one by one, there will be no big issue.

API security is now a must, not an option in web application security. As using API in Open Banking becomes mandatory in 2022, it is a good time for investing more resources into API security.

We hope that this White Paper is helpful for those preparing for API security. If you want more information, please consult with API security experts of PIOLINK. (Email: waf@piolink.com)