PIOLINK

보안위협분석

JSON을 이용한 WAF 우회

2023년 3월 20일

㈜파이오링크 사이버위협분석팀

'22 12월 해킹 연구팀 TEAM82에서 유명 WAF(Web Application Firewall)에서 JSON을 이용하여 Injection이 가능하다는 것을 확인하였다. WAF는 SQL Injection 공격을 쉽게 탐지하지만 SQL 구문에 JSON을 포함하게 되면 우회할 수 있게 된다.

Team82 연구에 따르면 Palo Alto Networks, Amazon Web Services, Cloudflare, F5 및 Imperva의 5개 주요 글로벌 공급업체에서 판매한 WAF에서 JSON Injection공격에 취약한 것으로 알려졌다. 해당 취약점은 현재는 패치 된 상태이다.

공격자들은 JSON Injection을 통해 WAF를 우회하고 추가 취약점을 이용해 데이터를 유출할 수 있다. 우리는 작년에 해당 기사를 접하고 나서 자사 WAF 대상으로 테스트한 결과를 공개하기로 하였다.

DB에 JSON 기능이?

주요 데이터 베이스에서는 10년이 넘는 기간 동안 JSON을 지원했다. JSON은 데이터 저장 및 전송의 주요 형식 중 하나이다. SQL에서도 다른 어플리케이션에서 데이터와 상호 작용을 하기위해서는 JSON 지원이 필요하다. 현재 모든 주요 관계형 데이터 베이스에는 JSON 구문을 지원한다. 관계형 데이터 베이스에는 MSSQL, PostgreSQL, MySQL 등이 포함된다.

모든 버전의 데이터 베이스에서 사용가능한 것은 아니고 아래의 표에 표기된 버전 이상의 데이터 베이스에서 JSON기능이 지원된다.

[표] JSON 지원 가능한 데이터 베이스 엔진

	JSON Support	Enabled by Default	Year JSON Added	JSON Parser Used
PostgreSQL	YES	YES	v9.2 (2012)	Proprietary
MySQL	YES	YES	v5.7.8 (2015)	Proprietary
SQLite	YES	YES	v3.38.0 (2022)	Proprietary
SQL Server	YES	YES	SQL Server (2016)	Proprietary

SQL에서 JSON을 사용하면 어플리케이션이 SQL API 내에서 데이터를 가져오고, 데이터 베이스 내에서 여러 소스를 결합하는 등 다양한 작업을 수행하고 JSON 형식으로 변환이 가능하다.

이처럼 데이터 베이스에서는 JSON에 대한 지원이 되는 반면 WAF 등 보안장비에서는 JSON에 대한 지원이 모두 가능한 것은 아니다.

아래는 우리가 테스트 했던 내용이다. 데이터 베이스에서 JSON을 사용하기 위해 먼저, 테이블을 생성할 때 JSON을 선언하여 생성하였다.

```
CREATE TABLE users(
    id integer  AUTO_INCREMENT primary key,
    user VARCHAR(200),
password VARCHAR(200),
```

```
    info JSON
);
```

〈그림〉 테이블 생성 쿼리문

그리고 테이블을 생성할 때 info의 형식을 JSON으로 지정하였다.

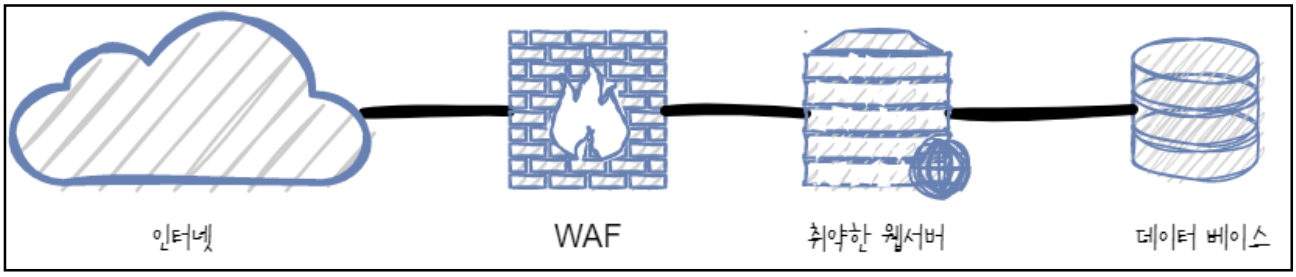
```
Insert into users(user, password, info) values('admin', 'admin', '{"age":44,"name":"aaa"}');
```

〈그림〉 JSON 데이터 삽입문

데이터 또한 JSON형식으로 name : value 형식으로 입력하였다.

공격 방법

해당 취약점을 테스트 하기 위해서는 취약한 환경을 구축해야 한다. 취약한 환경은 아래 그림과 같다.



〈그림〉 취약 구성도

해당 취약점을 간단하게 설명하면 데이터 베이스에서 지원하는 JSON을 사용하여 WAF에 JSON Injection 구문을 생성 및 쿼리 전달을 할 수 있다.

WAF에서 많은 Injection 구문을 탐지하지만 JSON의 구문에 대해서는 잘 탐지 하지 못하는 점을 이용하여 각 데이터 베이스에 맞는 구문으로 JSON을 이용하여 TRUE 명령어를 만들어 쿼리를 전송하는 것이다. 아래의 구문처럼 말이다.

```
PostgreSQL: '{"b":2}'::jsonb 〈@ '{"a":1, "b":2}'::jsonb Is the left JSON contained in the right one? True.

SQLite: '{"a":2,"c":[4,5,{"f":7}]}' -〉 '$.c[2].f' = 7 Does the extracted value of this JSON equals 7? True.

MySQL: JSON_EXTRACT('{"id": 14, "name": "Aztalan"}', '$.name') = 'Aztalan' Does the extracted value of this JSON equals to ‘Aztalan’? True.
```

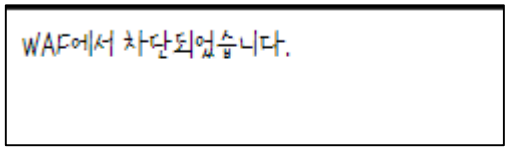
〈그림〉 JSON기반 True 구문

그럼 각 데이터 베이스에 해당 공격 쿼리를 전송하면 공격이 가능한가?

그렇지 않다. 해당 공격을 성공하기 위해서는 WAF가 탐지하지 못하는 SQL 구문을 찾아야한다.

즉, WAF에서 탐지 못하는 SQL 구문이 존재할 경우 취약 구문에 JSON 구문을 붙여 연달아 구문을 전송하여 WAF를 우회하여 명령을 전달하는 방식인 것이다.

좀 더 쉽게 예를 들어 설명해 보겠다.
우리가 SQL Injection으로 알고 사용하고 있는 공격 구문인 1' or '1'='1 --을 입력했을 때 일반적인 WAF에서는 아래 그림과 같이 차단한다.



〈그림〉 WAF 차단 화면

차단되는 SQL인젝션 구문 대신 ' or data @〉 '{"a" : "b"}' --와 같이 JSON 구문을 포함해서 쿼리를 보내게 되면 정상적으로 쿼리가 전송되어 WAF를 우회할 수 있는 것이다. 흥미롭지 않은가? 물론 예를 든 JSON Injection 구문은 일반적인 WAF에서 차단한다. 이해를 돕기 위해 설명한 것으로 실제 공격 시 WAF를 우회할 수 있는 JSON 구문을 찾는 것이 핵심이다.

우리는 작년에 이러한 취약점을 인지하고, 자사 WAF 제품을 대상으로 공격을 수행하였다. 아래는 우리가 시도했던 공격 구문 예시이다.

```
' or JSON_EXTRACT('{"id": 1, "user": "admin"}', '$.user')
' or JSON_CONTAINS(info,'$.age')
' and JSON_ARRAY(name, age, email)
```

〈그림〉 JSON 인젝션 공격 구문 일부

해당 쿼리보다 더 전문적인 구문을 전송하였지만 오남용을 방지하기 위하여 간단한 구문을 나열하였으며, 실제 위험 쿼리 구문을 테스트를 해본 결과 다행히 자사 WAF 제품은 차단을 성실히 수행하였다.

결론

JSON을 통하여 WAF를 우회하는 방법을 간단하게 알아보았다. JSON 은 우리 주변에서 데이터 저장 및 전송 등에 많이 사용하고 있으나, 데이터 베이스에서 지원되는 기능이 있음을 인지하지 못하거나 WAF에서는 JSON Injection에 대한 탐지가 아직 많이 미흡하다는 것을 알 수 있다. 실제로 JSON을 이용한 WAF 우회에 관한 해외 보안기업 테스트에서 위에서 설명한 공격 쿼리를 전송할 경우 글로벌 WAF 제품들에서 정상적으로 WAF 우회가 가능하여 JSON 명령 실행이 가능하였음을 보여주었다. 파이오링크 사이버위협분석팀에서는 자사 제품을 대상으로 이러한 위협 테스트를 지속적으로 수행하여 고객의 보안인프라에 홀이 생기지 않도록 사전예방 활동을 수행하고 있다.

[참조] https://claroty.com/team82/research/js-on-security-off-abusing-json-based-sql-to-bypass-waf