The State of the SameSite: Studying the Usage, Effectiveness, and Adequacy of SameSite Cookies

Soheil Khodayari and Giancarlo Pellegrino

CISPA Helmholtz Center for Information Security

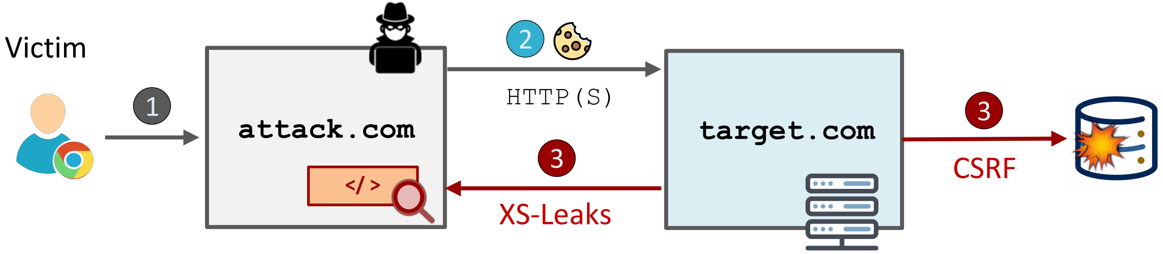
43rd IEEE Symposium on Security and Privacy
May 23-26, 2022

SameSite Policies

- Browsers include cookies in HTTP requests regardless of their context

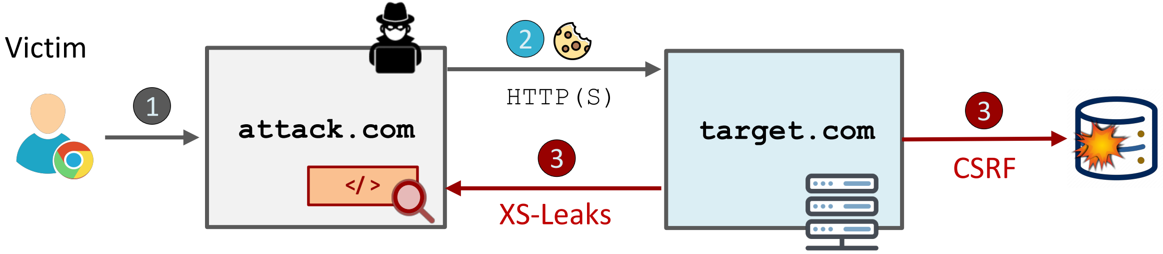
SameSite Policies

- Browsers include cookies in HTTP requests regardless of their context
- Abused by Cross-Site (XS) attacks



SameSite Policies

- Browsers include cookies in HTTP requests regardless of their context
- Abused by Cross-Site (XS) attacks



- **Solution:** limit the cookies' scope to a same-site context

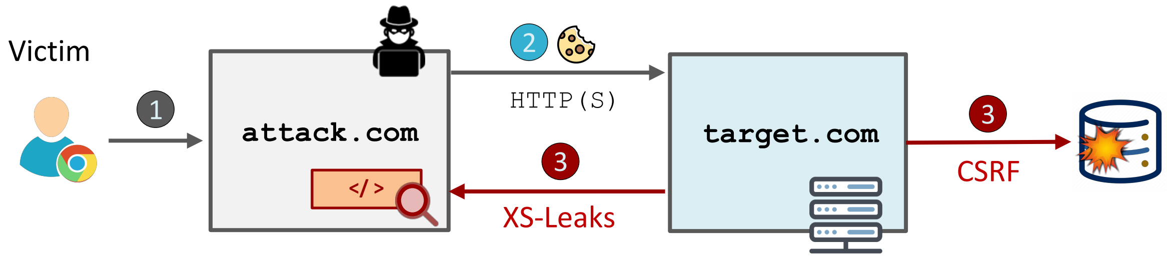
[1] *Ryck et. al., ESORICS'11.*

[2] *Czeskis et. al., WWW'13.*

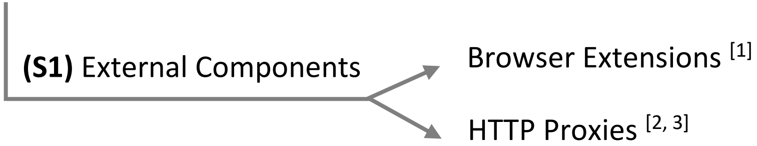
[3] *Johns et. al., AppSec'06.*

SameSite Policies

- Browsers include cookies in HTTP requests regardless of their context
- Abused by Cross-Site (XS) attacks



- **Solution:** limit the cookies' scope to a same-site context



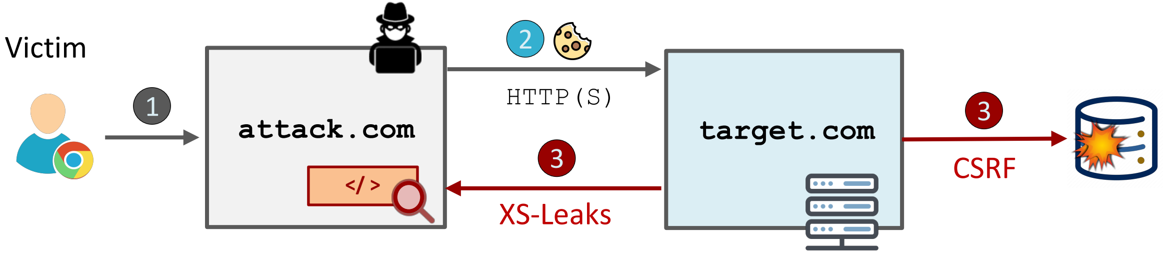
[1] *Ryck et. al., ESORICS'11.*

[2] *Czeskis et. al., WWW'13.*

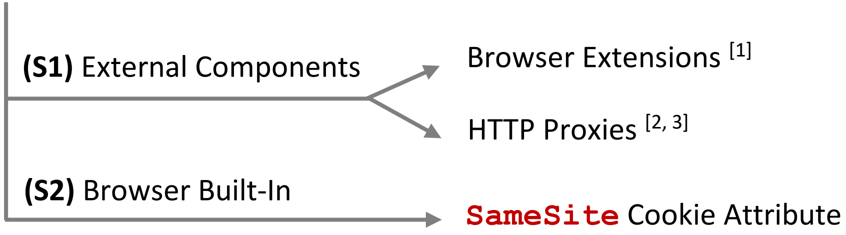
[3] *Johns et. al., AppSec'06.*

SameSite Policies

- Browsers include cookies in HTTP requests regardless of their context
- Abused by Cross-Site (XS) attacks



- **Solution:** limit the cookies' scope to a same-site context



(S1) External Components → Browser Extensions [1], HTTP Proxies [2, 3]

(S2) Browser Built-In → **SameSite** Cookie Attribute

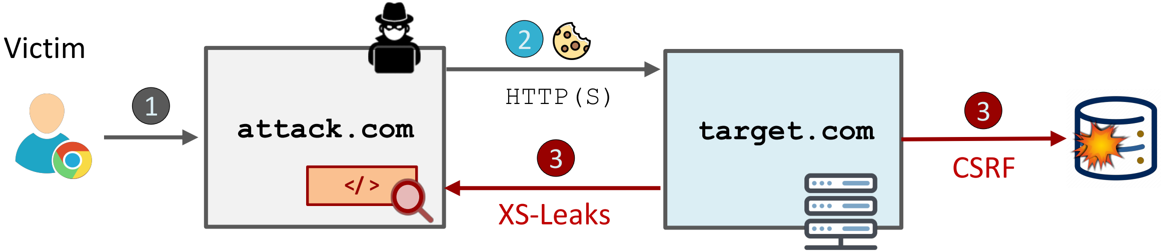
[1] *Ryck et. al., ESORICS'11.*

[2] *Czeskis et. al., WWW'13.*

[3] *Johns et. al., AppSec'06.*

SameSite Cookies

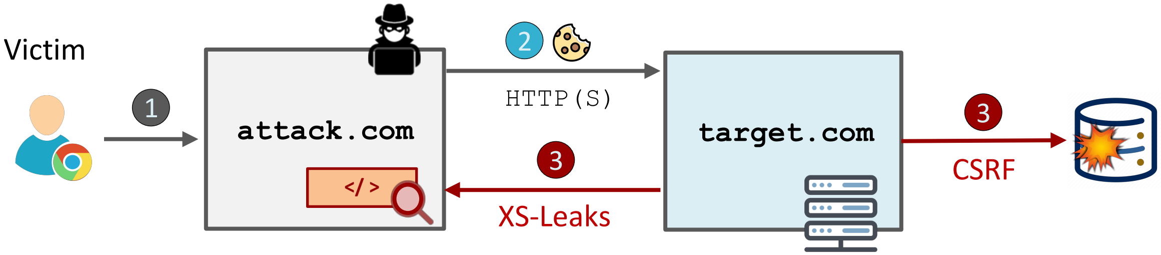
- The `SameSite` attribute introduces three pre-defined policies:



SameSite Cookies

- The `SameSite` attribute introduces three pre-defined policies:

None 🍪 in **all** XS requests in step ②

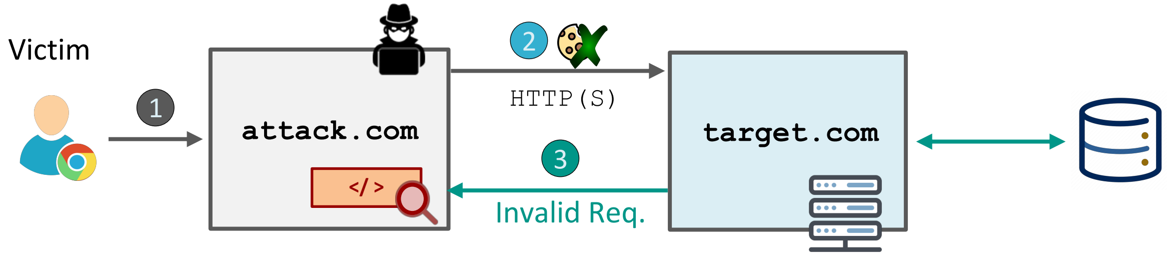


SameSite Cookies

- The `SameSite` attribute introduces three pre-defined policies:

None 🍪 in **all** XS requests in step 2

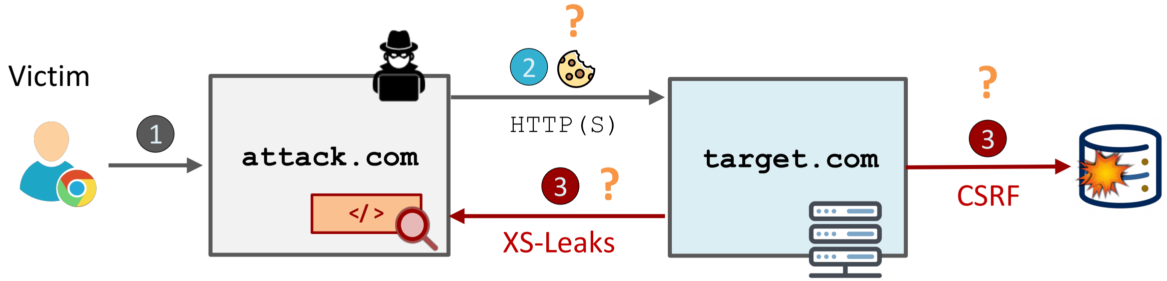
Strict 🍪 in **No** XS requests in step 2



SameSite Cookies

- The `SameSite` attribute introduces three pre-defined policies:

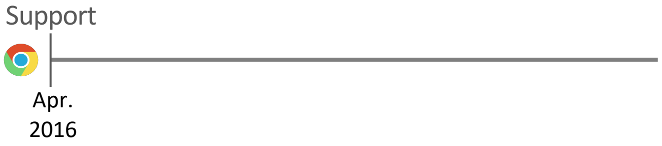
None	🍪 in **all** XS requests in step ②
Lax	🍪 in **some** XS requests in step ② (e.g., navigations)
Strict	🍪 in **No** XS requests in step ②

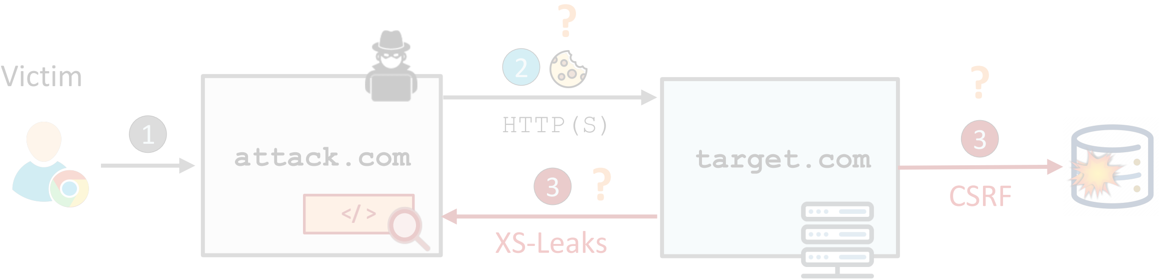


SameSite Cookies

- The `SameSite` attribute introduces three pre-defined policies:

None	🍪 in all XS requests in step 2
Lax	🍪 in some XS requests in step 2 (e.g., navigations)
Strict	🍪 in No XS requests in step 2

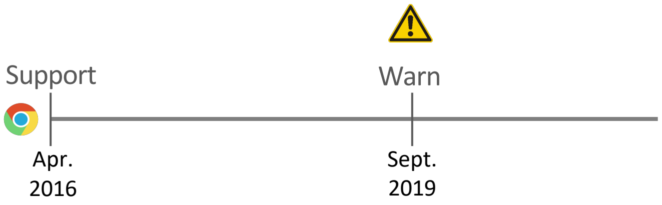


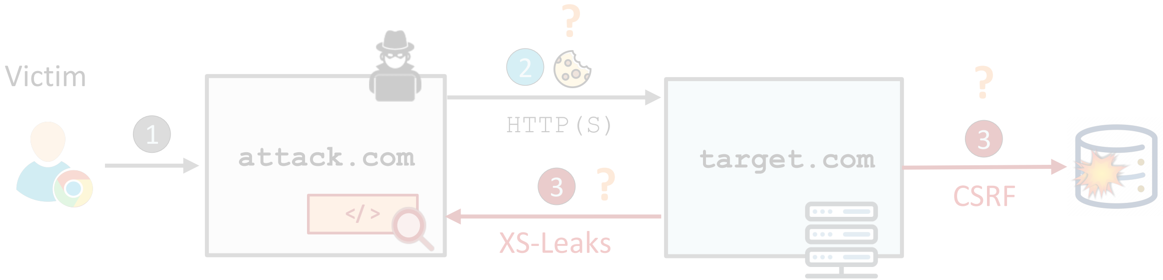


SameSite Cookies

- The `SameSite` attribute introduces three pre-defined policies:

None	🍪 in **all** XS requests in step 2
Lax	🍪 in **some** XS requests in step 2 (e.g., navigations)
Strict	🍪 in **No** XS requests in step 2

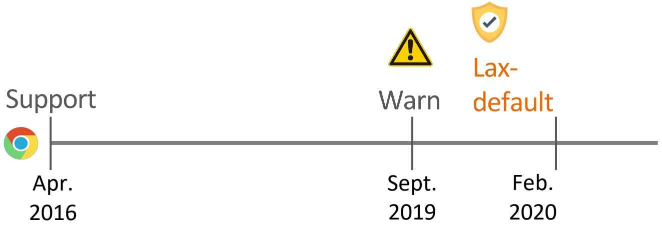


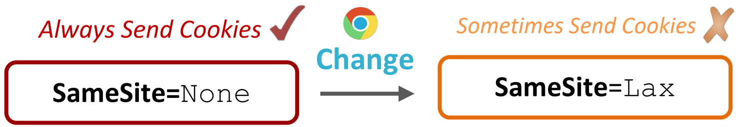


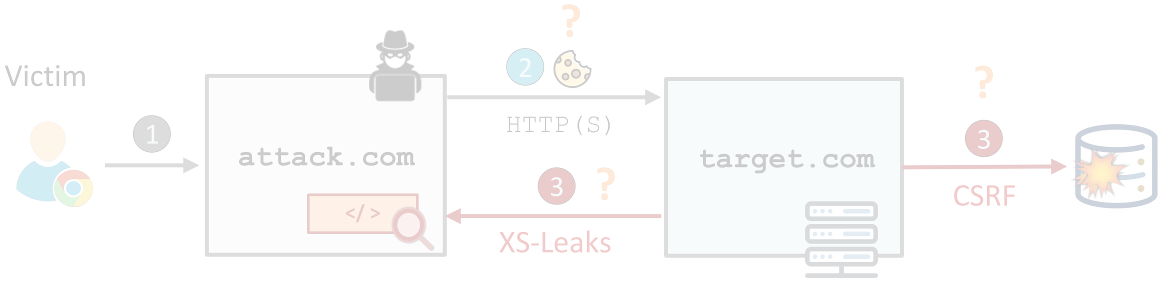
SameSite Cookies

- The SameSite attribute introduces three pre-defined policies:

None	🍪 in all XS requests in step 2
Lax	🍪 in some XS requests in step 2 (e.g., navigations)
Strict	🍪 in No XS requests in step 2







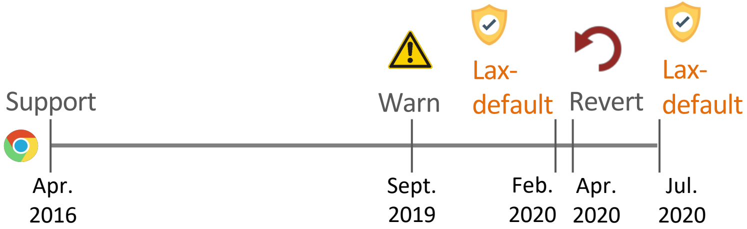
SameSite Cookies

- The `SameSite` attribute introduces three pre-defined policies:

None 🍪 in **all** XS requests in step 2

Lax 🍪 in **some** XS requests in step 2 (e.g., navigations)

Strict 🍪 in **No** XS requests in step 2



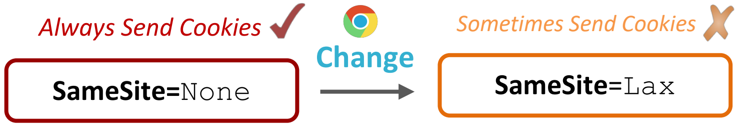
Support — Apr. 2016

Warn — Sept. 2019

Lax-default — Feb. 2020

Revert — Apr. 2020

Lax-default — Jul. 2020



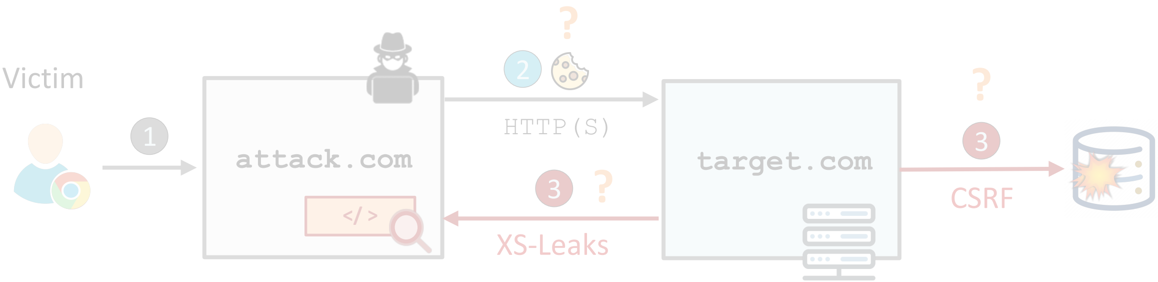
Always Send Cookies ✓

SameSite=`None`

Change

Sometimes Send Cookies ✗

SameSite=`Lax`



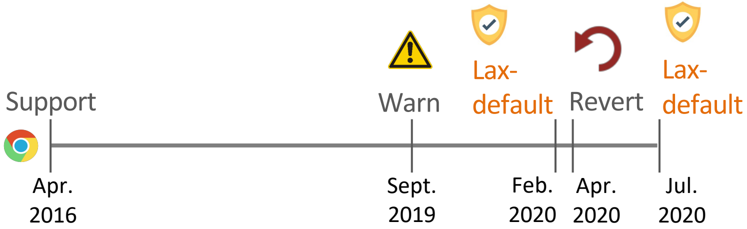
SameSite Cookies

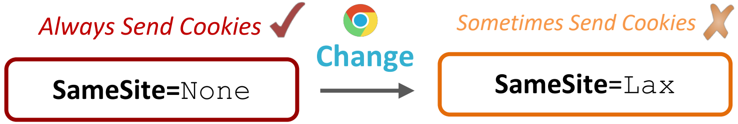
- The `SameSite` attribute introduces three pre-defined policies:

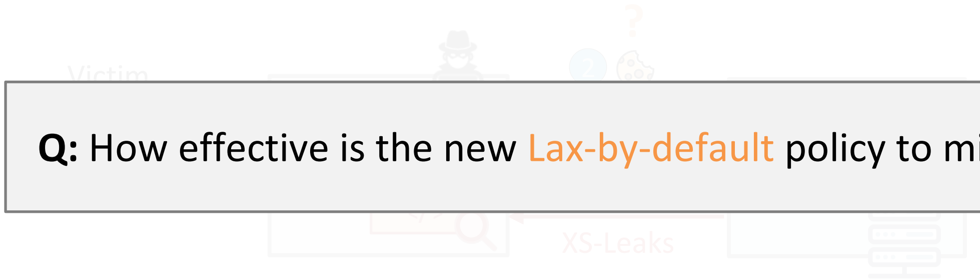
None — in **all** XS requests in step 2

Lax — in **some** XS requests in step 2 (e.g., navigations)

Strict — in **No** XS requests in step 2







Q: How effective is the new Lax-by-default policy to mitigate XS attacks?

Problem Statement

(RQ1) Trend Analysis of Adoption of SameSite Policies

Problem Statement

(RQ1) Trend Analysis of Adoption of SameSite Policies

(RQ2) Functionality Breakage by the New Lax-by-Default

Problem Statement

(RQ1) Trend Analysis of Adoption of SameSite Policies

(RQ2) Functionality Breakage by the New Lax-by-Default:

(RQ3) Lax Adequacy and Threats to its Effectiveness

Problem Statement

(RQ1) Trend Analysis of Adoption of SameSite Policies

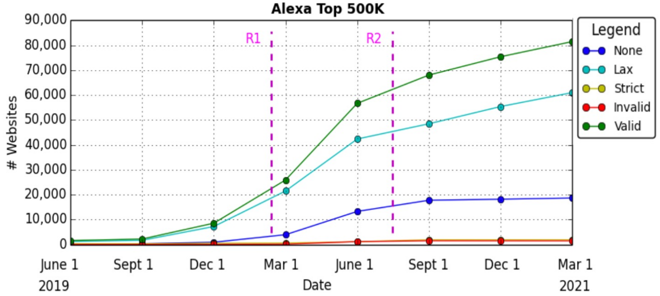
(RQ2) Functionality Breakage by the New Lax-by-Default:

(RQ3) Lax Adequacy and Threats to its Effectiveness

(RQ4) Browser Policy Inconsistencies and Web Frameworks' Defaults

RQ1: Adoption of SameSite Policies

Longitudinal Analysis



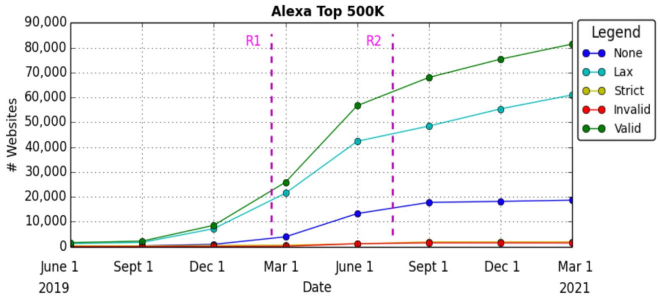
RQ1: Adoption of SameSite Policies

Longitudinal Analysis

Valid Policies

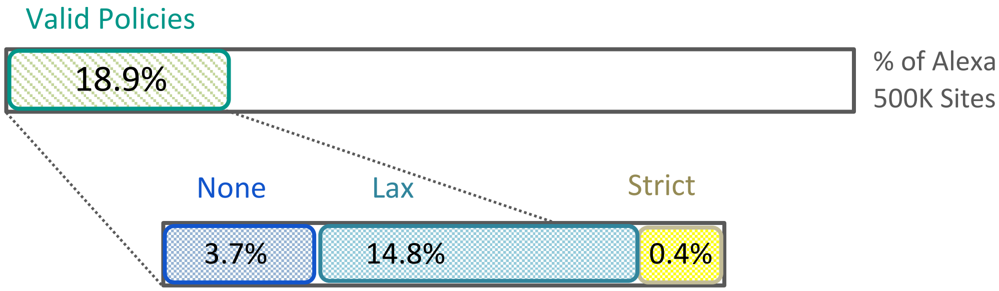
18.9%

% of Alexa 500K Sites



RQ1: Adoption of SameSite Policies

Longitudinal Analysis



Valid Policies

18.9%

% of Alexa 500K Sites

None

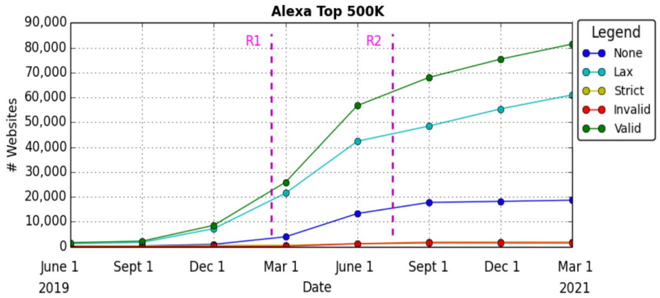
Lax

Strict

3.7%

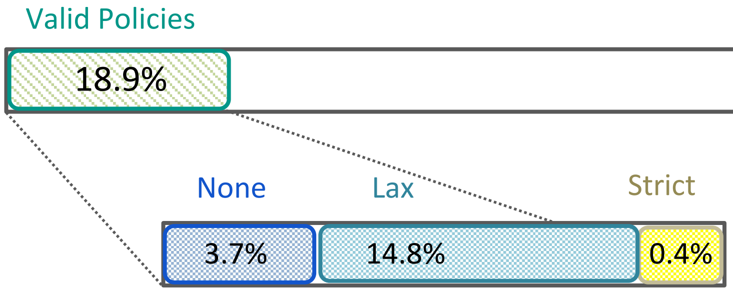
14.8%

0.4%



RQ1: Adoption of SameSite Policies

Longitudinal Analysis



Valid Policies

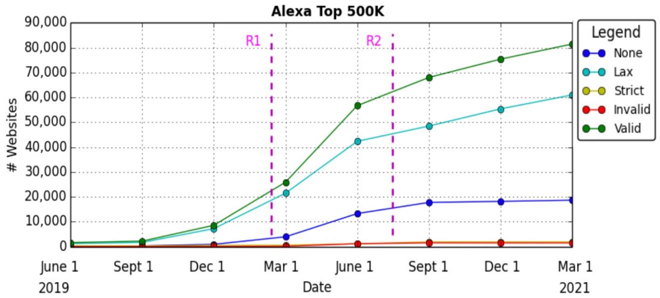
18.9%

% of Alexa 500K Sites

None 3.7%

Lax 14.8%

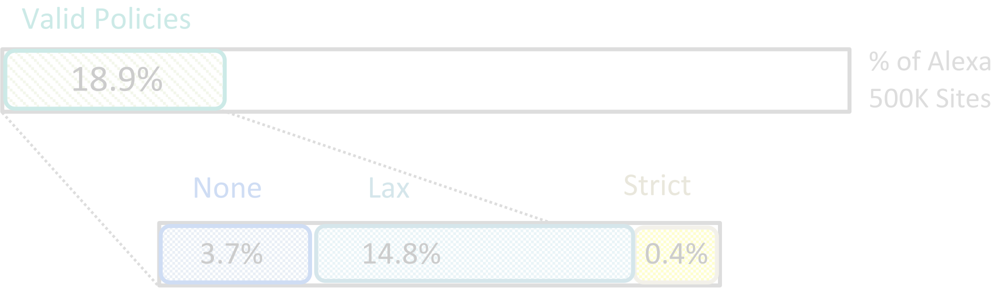
Strict 0.4%



- `None` usage increases by site popularity (i.e., 8.1% of top 10K and 18% of 1K)

RQ1: Adoption of SameSite Policies

Longitudinal Analysis



Valid Policies

18.9%

% of Alexa 500K Sites

None

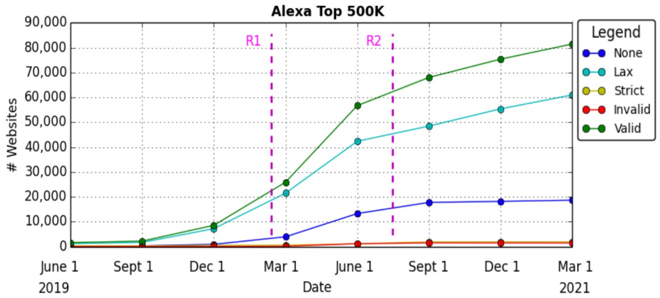
Lax

Strict

3.7%

14.8%

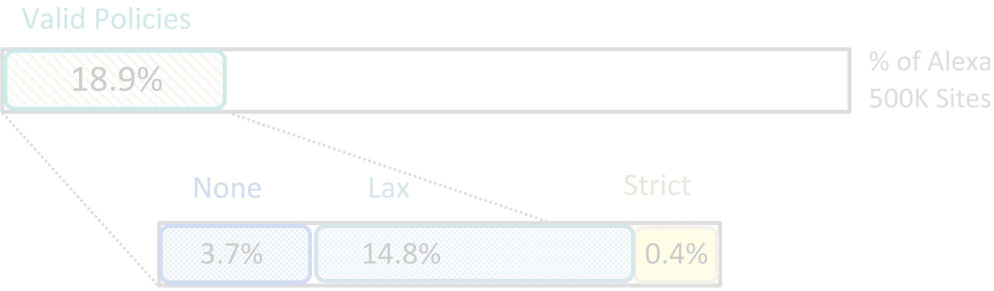
0.4%



- None usage increases by site popularity (i.e., 8.1% of top 10K and 18% of 1K)
- Rollout dates R1 and R2:
  - Steep increase of SameSite usage after Lax-by-default

RQ1: Adoption of SameSite Policies

Longitudinal Analysis



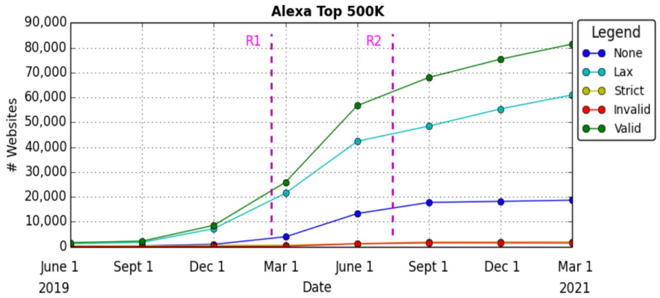
Valid Policies

18.9%

% of Alexa 500K Sites

None Lax Strict

3.7% 14.8% 0.4%

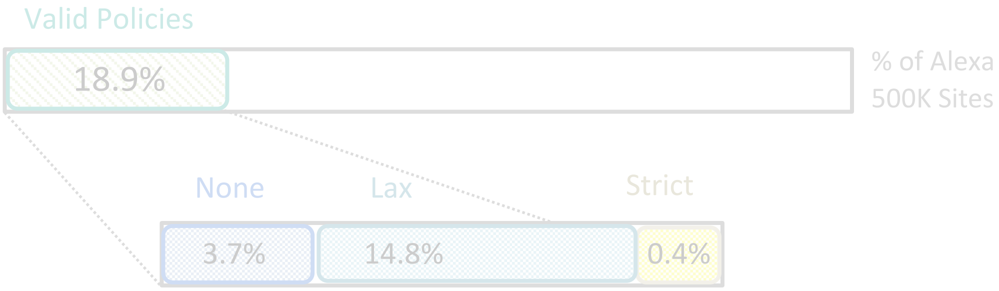


- None usage increases by site popularity (i.e., 8.1% of top 10K and 18% of 1K)
- Rollout dates R1 and R2:
  - Steep increase of SameSite usage after Lax-by-default

Stricter policies: ~ 7x growth in Lax, ~ 4x growth in Strict

RQ1: Adoption of SameSite Policies

Longitudinal Analysis



Valid Policies

18.9%

% of Alexa 500K Sites

None

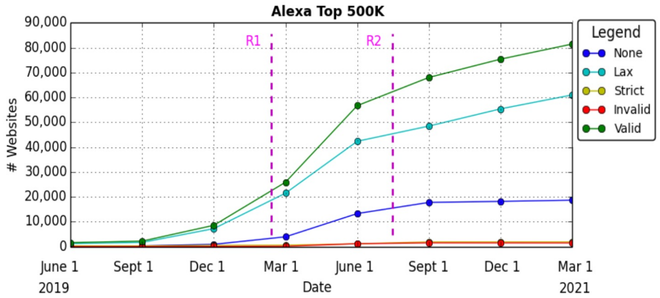
Lax

Strict

3.7%

14.8%

0.4%



- None usage increases by site popularity (i.e., 8.1% of top 10K and 18% of 1K)
- Rollout dates R1 and R2:
  - Steep increase of SameSite usage after Lax-by-default

Stricter policies: ~ 7x growth in Lax, ~ 4x growth in Strict

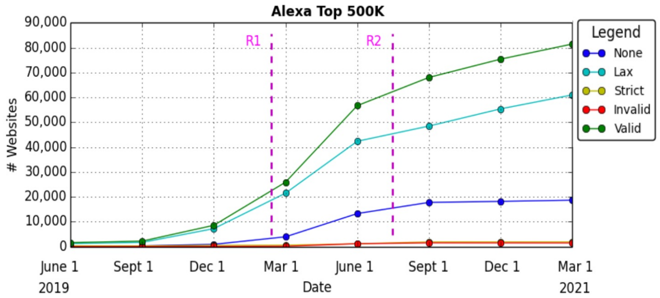
None policy: at least 20x growth, with even more increase in more popular sites

RQ1: Adoption of SameSite Policies

Valid Policies

18.9%

% of Alexa 500K Sites



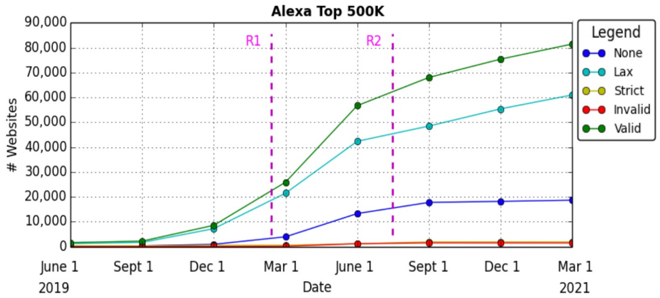
RQ1: Adoption of SameSite Policies

Longitudinal Analysis

Valid Policies | Invalid

18.9% | 0.3%

% of Alexa 500K Sites



- Example:
  - `SameSite=1`

Should be treated as None [RFC 6265bis]

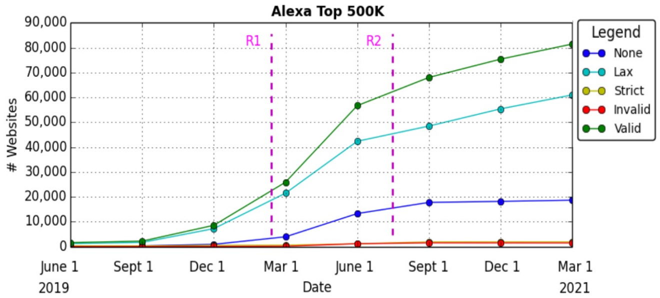
1,430 sites (0.3%) set an invalid policy due to developers' mistakes

RQ1: Adoption of SameSite Policies

CISPA
HELMHOLTZ CENTER FOR
INFORMATION SECURITY

Longitudinal Analysis

Valid Policies	Invalid	Missing Policy	
18.9%	0.3%	80.8%	% of Alexa 500K Sites



- Example:
  - `SameSite=1`

Should be treated as None [RFC 6265bis]

1,430 sites (0.3%) set an invalid policy due to developers' mistakes

The remaining 80.8%:

- No explicit policy found, rely on the default browser behaviour

RQ2: Functionality Breakage

- Websites use XS requests for various functionalities
  - Social media share buttons, advertising, etc

RQ2: Functionality Breakage

- Websites use XS requests for various functionalities
  - Social media share buttons, advertising, etc

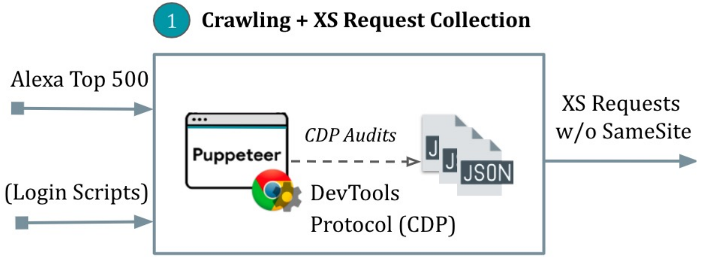
Q: What functionalities are affected by the new default policy?

RQ2: Functionality Breakage

- Websites use XS requests for various functionalities
  - Social media share buttons, advertising, etc

Q: What functionalities are affected by the new default policy?

- **Methodology**

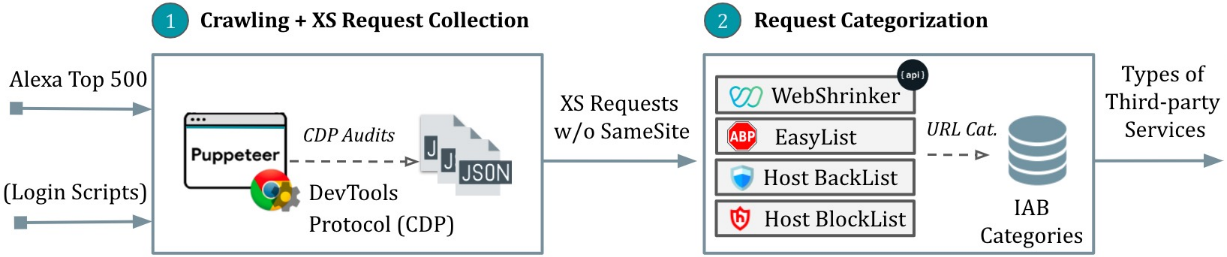


RQ2: Functionality Breakage

- Websites use XS requests for various functionalities
  - Social media share buttons, advertising, etc

Q: What functionalities are affected by the new default policy?

- **Methodology**

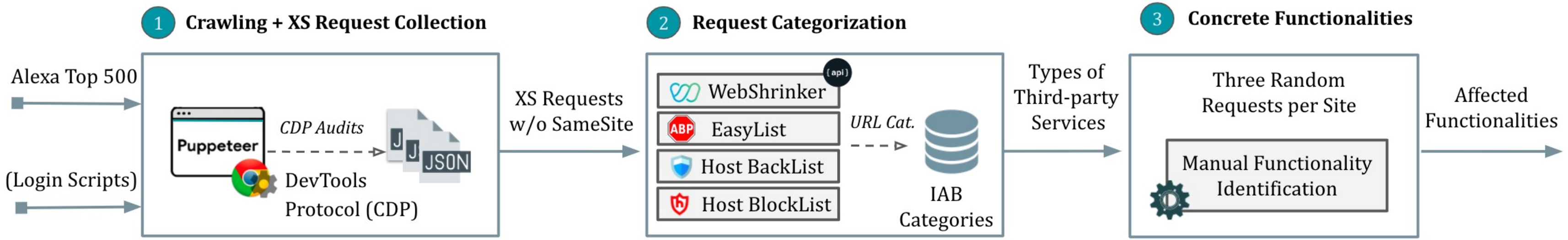


RQ2: Functionality Breakage

- Websites use XS requests for various functionalities
  - Social media share buttons, advertising, etc

Q: What functionalities are affected by the new default policy?

- **Methodology**



1 **Crawling + XS Request Collection**

Alexa Top 500

(Login Scripts)

Puppeteer

CDP Audits

DevTools Protocol (CDP)

JSON

XS Requests w/o SameSite

2 **Request Categorization**

WebShrinker

EasyList

Host BackList

Host BlockList

URL Cat.

IAB Categories

Types of Third-party Services

3 **Concrete Functionalities**

Three Random Requests per Site

Manual Functionality Identification

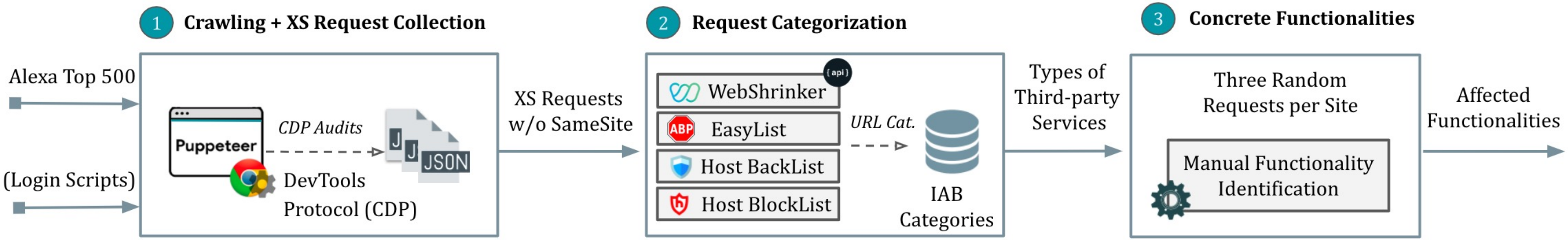
Affected Functionalities

RQ2: Functionality Breakage

- Websites use XS requests for various functionalities
  - Social media share buttons, advertising, etc

Q: What functionalities are affected by the new default policy?

- **Methodology**



1 Crawling + XS Request Collection

Alexa Top 500

(Login Scripts)

Puppeteer

CDP Audits

DevTools Protocol (CDP)

JSON

XS Requests w/o SameSite

2 Request Categorization

WebShrinker

EasyList

Host BackList

Host BlockList

URL Cat.

IAB Categories

Types of Third-party Services

3 Concrete Functionalities

Three Random Requests per Site

Manual Functionality Identification

Affected Functionalities

- Identify potentially affected functionality before Lax-by-default rollout (R2), confirm breakage afterwards

RQ2: Functionality Breakage

Data Collection

- 211 sites, and 9,073 unique URLs
- 22,992 XS requests without `SameSite`

RQ2: Functionality Breakage

Data Collection

- 211 sites, and 9,073 unique URLs
- 22,992 XS requests without `SameSite`

Affected Functionalities

- 32 different types of affected third-party functionalities
- E.g., file sharing, live chat, advertising, or analytics

RQ2: Functionality Breakage

Data Collection

- 211 sites, and 9,073 unique URLs
- 22,992 XS requests without `SameSite`

Affected Functionalities

- 32 different types of affected third-party functionalities
- E.g., file sharing, live chat, advertising, or analytics

Breakage

- Examined three random requests per site

		After R2	
Functionality	**# Requests**	**# Broken**	**# Patched**
Advertising / Tracking	374	93	281
Single-Sign On	81	1	80
Social Media Like / Share	76	11	65
Live Chat Frames	62	8	54
PDF Embed APIs	13	4	9
(Re-)CAPTCHA	12	2	10
Content Servers / CDNs	9	0	9
Survey/Rating Services	6	1	5
Total	**633**	**120**	**513**

RQ2: Functionality Breakage

Data Collection

- 211 sites, and 9,073 unique URLs
- 22,992 XS requests without `SameSite`

Affected Functionalities

- 32 different types of affected third-party functionalities
- E.g., file sharing, live chat, advertising, or analytics

Breakage

- Examined three random requests per site

		After R2	
Functionality	**# Requests**	**# Broken**	**# Patched**
Advertising / Tracking	374	93	281
Single-Sign On	81	1	80
Social Media Like / Share	76	11	65
Live Chat Frames	62	8	54
PDF Embed APIs	13	4	9
(Re-)CAPTCHA	12	2	10
Content Servers / CDNs	9	0	9
Survey/Rating Services	6	1	5
Total	**633**	**120**	**513**

⚠ Functionalities implemented by 19% of the affected requests are broken, affecting 17.5% of sites

RQ2: Functionality Breakage

Data Collection

- 211 sites, and 9,073 unique URLs
- 22,992 XS requests without `SameSite`

Affected Functionalities

- 32 different types of affected third-party functionalities
- E.g., file sharing, live chat, advertising, or analytics

Breakage

- Examined three random requests per site

		After R2	
Functionality	**# Requests**	**# Broken**	**# Patched**
Advertising / Tracking	374	93	281
Single-Sign On	81	1	80
Social Media Like / Share	76	11	65
Live Chat Frames	62	8	54
PDF Embed APIs	13	4	9
(Re-)CAPTCHA	12	2	10
Content Servers / CDNs	9	0	9
Survey/Rating Services	6	1	5
Total	**633**	**120**	**513**

Functionalities implemented by 19% of the affected requests are broken, affecting 17.5% of sites

The majority of broken requests (i.e., 77.5%) are for online ads & user tracking

RQ3: Lax Adequacy and Threats to Effectiveness

Benefit
Lax

***Q:** How adequate are SameSite cookies to prevent XS attacks?*

RQ3: Lax Adequacy and Threats to Effectiveness

Benefit
Lax

***Q:** How adequate are SameSite cookies to prevent XS attacks?*

i. Systematically reviewed existing literature for threats enabling XS attacks

ii. Determined the threats’ severity by quantifying their prevalence

RQ3: Lax Adequacy and Threats to Effectiveness

Benefit
Lax

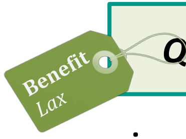
***Q:** How adequate are SameSite cookies to prevent XS attacks?*

i. Systematically reviewed existing literature for threats enabling XS attacks

ii. Determined the threats’ severity by quantifying their prevalence

Threats → 10 Threats Including Three New Ones

RQ3: Lax Adequacy and Threats to Effectiveness



***Q**: How adequate are SameSite cookies to prevent XS attacks?*

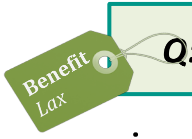
i. Systematically reviewed existing literature for threats enabling XS attacks

ii. Determined the threats’ severity by quantifying their prevalence

Threats → 10 Threats Including Three New Ones

- → Not Protected by Lax?
- → Can Bypass Lax Protection?

RQ3: Lax Adequacy and Threats to Effectiveness



***Q**: How adequate are SameSite cookies to prevent XS attacks?*

i. Systematically reviewed existing literature for threats enabling XS attacks

ii. Determined the threats’ severity by quantifying their prevalence

Threats → 10 Threats Including Three New Ones

→ Not Protected by Lax?

- *Replaying State-changing GET*
- *Window Properties Leak*
- …

→ Can Bypass Lax Protection?

RQ3: Lax Adequacy and Threats to Effectiveness

Benefit
Lax

***Q**: How adequate are SameSite cookies to prevent XS attacks?*

i. Systematically reviewed existing literature for threats enabling XS attacks

ii. Determined the threats’ severity by quantifying their prevalence

Threats → 10 Threats Including Three New Ones

→ Not Protected by Lax?

- *Replaying State-changing GET*
- *Window Properties Leak*
- …

→ Can Bypass Lax Protection?

- *Forging State-changing POST*
- *SameSite Intra/Inter Page Inconsistency*
- *SameSite User-Agent Inconsistency*
- …

RQ3: Lax Adequacy and Threats to Effectiveness

Benefit
Lax

***Q**: How adequate are SameSite cookies to prevent XS attacks?*

i. Systematically reviewed existing literature for threats enabling XS attacks

ii. Determined the threats' severity by quantifying their prevalence

Threats → 10 Threats Including Three New Ones

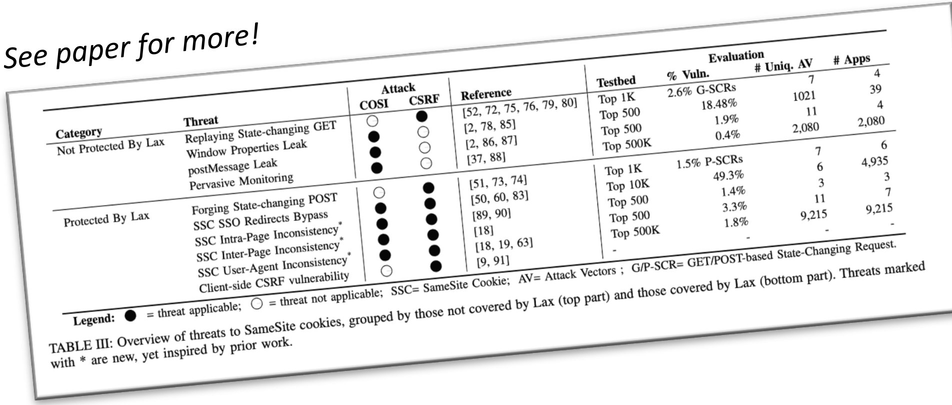
→ Not Protected by Lax?

- *Replaying State-changing GET*
- *Window Properties Leak*
- …

→ Can Bypass Lax Protection?

- *Forging State-changing POST*
- *SameSite Intra/Inter Page Inconsistency*
- *SameSite User-Agent Inconsistency*
- …

See paper for more!



Category	Threat	Attack		Reference	Evaluation			
		COSI	CSRF		Testbed	% Vuln.	# Uniq. AV	# Apps
Not Protected By Lax	Replaying State-changing GET	○	●	[52, 72, 75, 76, 79, 80]	Top 1K	2.6% G-SCRs	7	4
	Window Properties Leak	●	○	[2, 78, 85]	Top 500	18.48%	1021	39
	postMessage Leak	●	○	[2, 86, 87]	Top 500	1.9%	11	4
	Pervasive Monitoring	●	○	[37, 88]	Top 500K	0.4%	2,080	2,080
Protected By Lax	Forging State-changing POST	○	●	[51, 73, 74]	Top 1K	1.5% P-SCRs	7	6
	SSC SSO Redirects Bypass	●	●	[50, 60, 83]	Top 10K	49.3%	6	4,935
	SSC Intra-Page Inconsistency*	●	●	[89, 90]	Top 500	1.4%	3	3
	SSC Inter-Page Inconsistency*	●	●	[18]	Top 500	3.3%	11	7
	SSC User-Agent Inconsistency*	●	●	[18, 19, 63]	Top 500K	1.8%	9,215	9,215
	Client-side CSRF vulnerability	○	●	[9, 91]	-	-	-	-

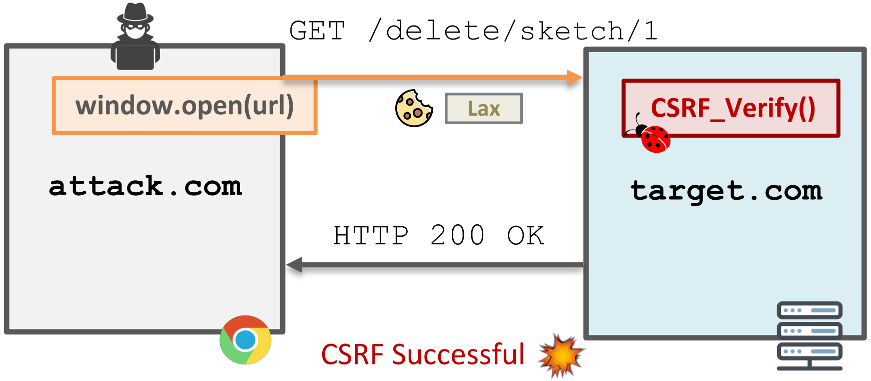
Legend: ● = threat applicable; ○ = threat not applicable; SSC= SameSite Cookie; AV= Attack Vectors ; G/P-SCR= GET/POST-based State-Changing Request.

TABLE III: Overview of threats to SameSite cookies, grouped by those not covered by Lax (top part) and those covered by Lax (bottom part). Threats marked with * are new, yet inspired by prior work.

Threat: CSRF by Replaying State-changing GET

Threat

- Top-level GET requests not covered by Lax
- Developers may misuse GET requests for state-changing operations



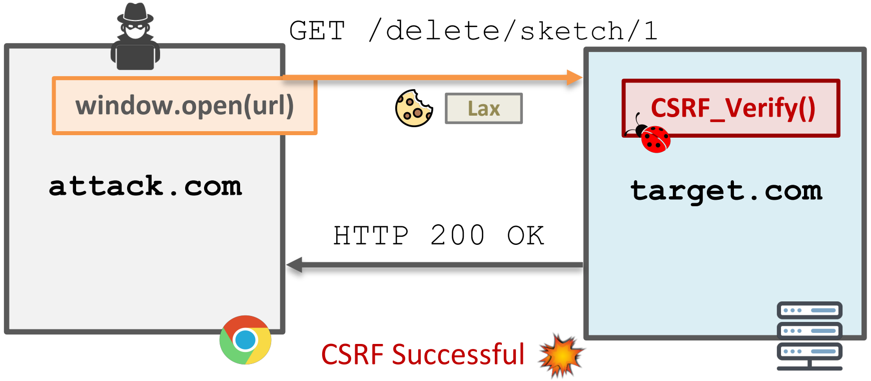
Threat: CSRF by Replaying State-changing GET

Threat

- Top-level GET requests not covered by Lax
- Developers may misuse GET requests for state-changing operations

Methodology

- Alexa top 1K, 42.5K URLs
- Located HTML forms with a CSRF token
- Quantified GET-based state-changes (lower-bound)
- Manually checked if CSRF verification is correct



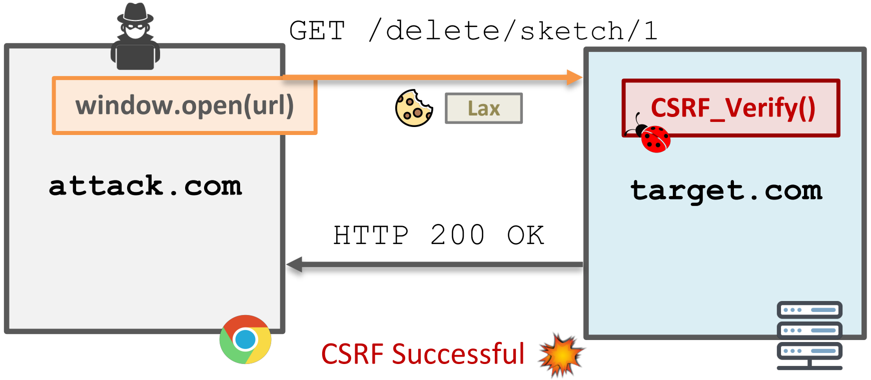
Threat: CSRF by Replaying State-changing GET

Threat

- Top-level GET requests not covered by Lax
- Developers may misuse GET requests for state-changing operations

Methodology

- Alexa top 1K, 42.5K URLs
- Located HTML forms with a CSRF token
- Quantified GET-based state-changes (lower-bound)
- Manually checked if CSRF verification is correct



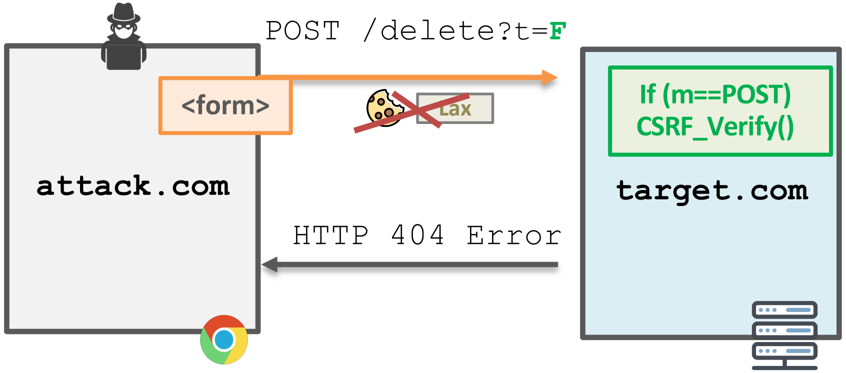
Results

- 6.9K state-changing requests, 10.3% are GET-based (in 88 webapps)
- 2.6% of the GET-based requests are forgeable due to faulty CSRF token verification ⚠
  - E.g., delete user sketches in Pixiv, or change user settings in Mailchimp

Threat: CSRF by Forging State-changing POST

Threat

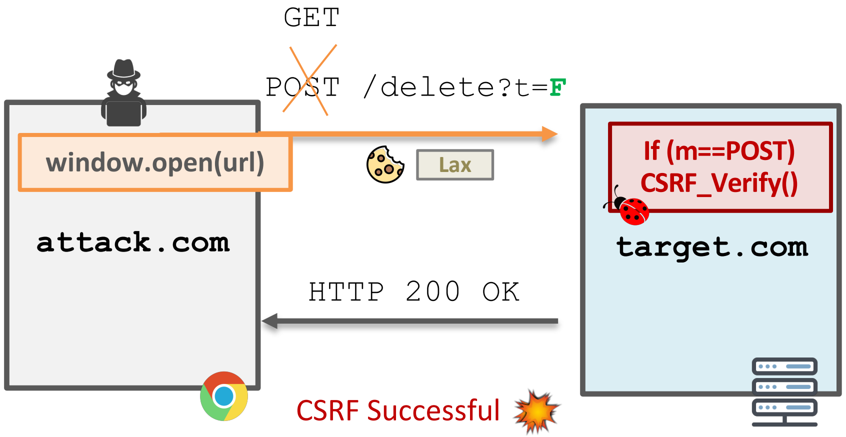
- Forge POST requests with GET to bypass Lax protection



Threat: CSRF by Forging State-changing POST

Threat

- Forge POST requests with GET to bypass Lax protection



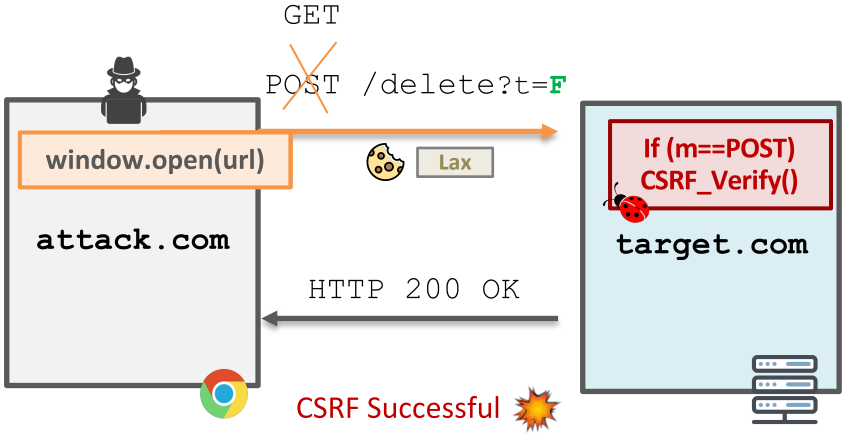
Threat: CSRF by Forging State-changing POST

Threat

- Forge POST requests with GET to bypass Lax protection

Methodology

- 6.2K state-changing POST in webapps of Alexa top 1K
- Selected one random request per webapp
- Checked CSRF by replaying the request with GET



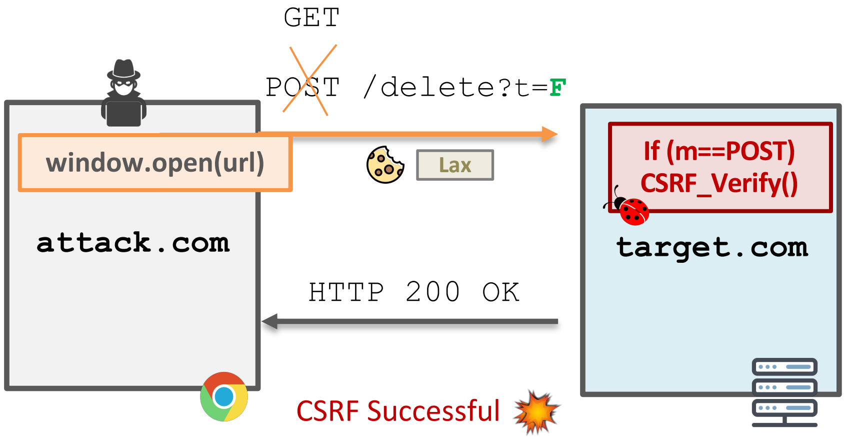
Threat: CSRF by Forging State-changing POST

Threat

- Forge POST requests with GET to bypass Lax protection

Methodology

- 6.2K state-changing POST in webapps of Alexa top 1K
- Selected one random request per webapp
- Checked CSRF by replaying the request with GET



Results

- 1.5% of state-changing POST requests are forgeable with GET ⚠
- Affected popular sites:
  - Add or remove movies from user watchlist in IMDB
  - Remove notification alerts in Meetup

New Threats: Policy Downgrades

Intra-page Inconsistency

- Webapps may set redundant cookies to support incompatible clients
- **Bypass:** cookies with no `SameSite` and `Strict`, or `Lax` and `None`

Vuln: 1.4% of top 500 sites, e.g., GitHub, CNN, and Yahoo

```
// for incompatible clients
Set-cookie: 3pc-legacy=value;
// for newer clients
Set-cookie: 3pc=value; SameSite=Strict;
```

New Threats: Policy Downgrades

Intra-page Inconsistency

- Webapps may set redundant cookies to support incompatible clients
- **Bypass:** cookies with no `SameSite` and `Strict`, or `Lax` and `None`

Vuln: 1.4% of top 500 sites, e.g., GitHub, CNN, and Yahoo

```
// for incompatible clients
Set-cookie: 3pc-legacy=value;
// for newer clients
Set-cookie: 3pc=value; SameSite=Strict;
```

Inter-page Inconsistency

- Different policies for the same cookie across two webpages

Vuln: 3.3% of top 500 sites, e.g., AliExpress and Vimeo

GET /account.php \r\n

```
Set-cookie: 3pc=value; SameSite=Strict; Path=/
```

GET /index.php \r\n

```
Set-cookie: 3pc=value; SameSite=None; Path=/
```

New Threats: Policy Downgrades

Intra-page Inconsistency

- Webapps may set redundant cookies to support incompatible clients
- **Bypass:** cookies with no `SameSite` and `Strict`, or `Lax` and `None`

Vuln: 1.4% of top 500 sites, e.g., GitHub, CNN, and Yahoo

```
// for incompatible clients
Set-cookie: 3pc-legacy=value;
// for newer clients
Set-cookie: 3pc=value; SameSite=Strict;
```

Inter-page Inconsistency

- Different policies for the same cookie across two webpages

Vuln: 3.3% of top 500 sites, e.g., AliExpress and Vimeo

```
GET /account.php \r\n
Set-cookie: 3pc=value; SameSite=Strict; Path=/
GET /index.php \r\n
Set-cookie: 3pc=value; SameSite=None; Path=/
```

User-Agent Inconsistency

- Different SameSite policy based on the User-Agent (e.g., mobile vs desktop)

Vuln: 1.8% of the top 500K sites, 138 webapps among the top 1K sites

RQ3: Lax Adequacy and Threats to Effectiveness

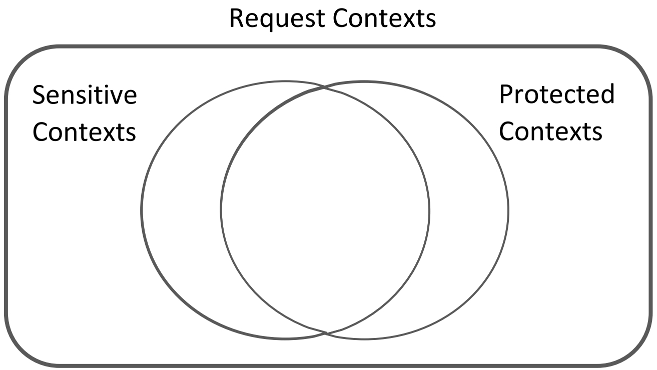
Benefit
Lax

Q: *How adequate are SameSite cookies to prevent XS attacks?*

RQ3: Lax Adequacy and Threats to Effectiveness

Benefit
Lax

***Q**: How adequate are SameSite cookies to prevent XS attacks?*



Request Contexts

Sensitive Contexts

Protected Contexts

RQ3: Lax Adequacy and Threats to Effectiveness

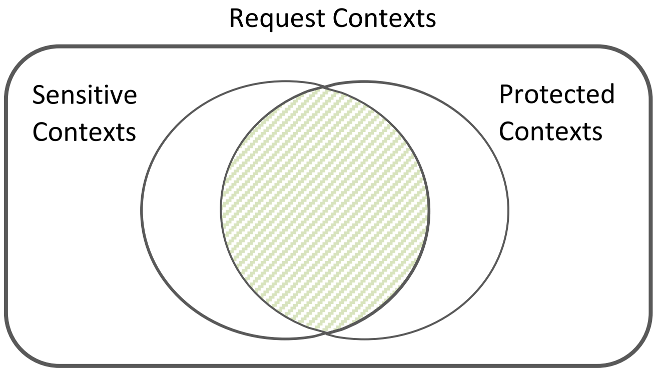
Benefit
Lax

***Q**: How adequate are SameSite cookies to prevent XS attacks?*

Lax

The Good

✓ Lax-by-default cookies significantly reduce the attack surface for XS attacks



RQ3: Lax Adequacy and Threats to Effectiveness

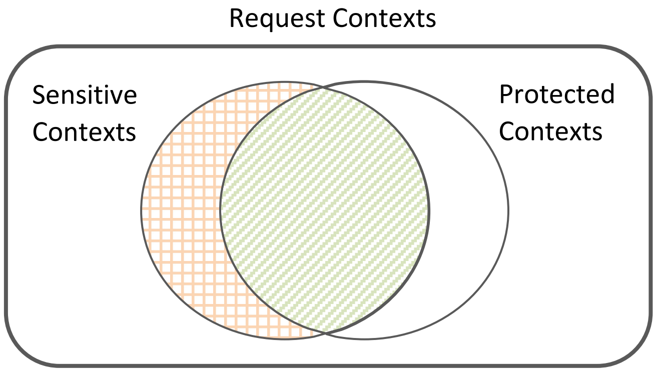
Benefit
Lax

***Q:** How adequate are SameSite cookies to prevent XS attacks?*

Lax

The Good ✓ Lax-by-default cookies significantly reduce the attack surface for XS attacks

The Ugly ⚠ Mismatch between XS request contexts protected by Lax and the ones used by websites



RQ3: Lax Adequacy and Threats to Effectiveness

Benefit
Lax

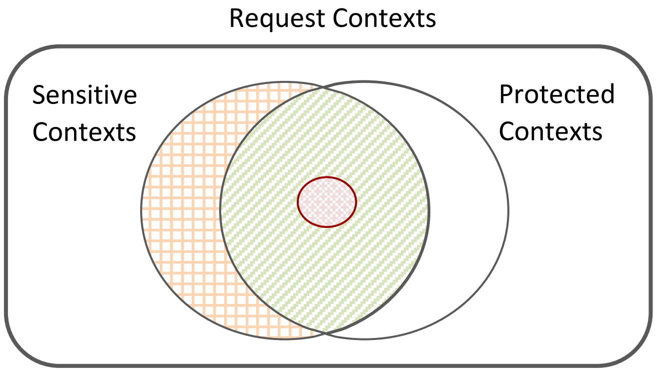
Q: *How adequate are SameSite cookies to prevent XS attacks?*

Lax

The Good ✓ Lax-by-default cookies significantly reduce the attack surface for XS attacks

The Ugly ⚠ Mismatch between XS request contexts protected by Lax and the ones used by websites

The Bad ☠ Implementation mistakes can lead to SameSite policy bypass



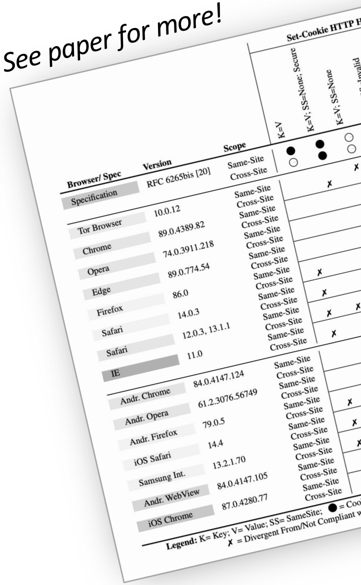
RQ4: Browser Inconsistencies and Web Frameworks

- Web browsers exhibit seven divergent behaviours wrt. SameSite cookie policy
  - `SameSite=Invalid`
  - `SameSite=None` w/o `Secure`

RQ4: Browser Inconsistencies and Web Frameworks

- Web browsers exhibit seven divergent behaviours wrt. SameSite cookie policy
  - `SameSite=Invalid`
  - `SameSite=None` w/o `Secure`

- Correct and secure use of SameSite requires developer's awareness



See paper for more!

			Set-Cookie HTTP H			
Browser/ Spec	**Version**	**Scope**	K=V	K=V; SS=None; Secure	K=V; SS=None	S=Invalid
Specification	RFC 6265bis [20]	Same-Site	●	●	○	
		Cross-Site	○	●	○	
Tor Browser	10.0.12	Same-Site			✗	
		Cross-Site		✗		
Chrome	89.0.4389.82	Same-Site				
		Cross-Site				
Opera	74.0.3911.218	Same-Site				
		Cross-Site				
Edge	89.0.774.54	Same-Site				
		Cross-Site				
Firefox	86.0	Same-Site				
		Cross-Site	✗			
Safari	14.0.3	Same-Site				
		Cross-Site	✗			
Safari	12.0.3, 13.1.1	Same-Site				
		Cross-Site	✗	✗		
IE	11.0	Same-Site				
		Cross-Site	✗			
Andr. Chrome	84.0.4147.124	Same-Site				
		Cross-Site				
Andr. Opera	61.2.3076.56749	Same-Site				
		Cross-Site				
Andr. Firefox	79.0.5	Same-Site				
		Cross-Site	✗			
iOS Safari	14.4	Same-Site				
		Cross-Site	✗			
Samsung Int.	13.2.1.70	Same-Site				
		Cross-Site	✗			
Andr. WebView	84.0.4147.105	Same-Site				
		Cross-Site				
iOS Chrome	87.0.4280.77	Same-Site				
		Cross-Site				

Legend: K= Key; V= Value; SS= SameSite; ● = Coo
✗ = Divergent From/Not Compliant w

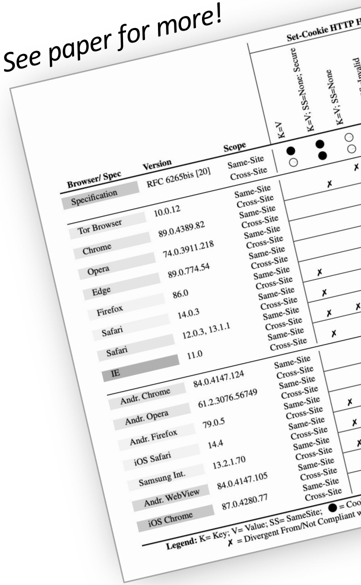
RQ4: Browser Inconsistencies and Web Frameworks

- Web browsers exhibit seven divergent behaviours wrt. SameSite cookie policy
  - `SameSite=Invalid`
  - `SameSite=None` w/o `Secure`

- Correct and secure use of SameSite requires developer's awareness

- Even when browsers enforce a default Lax policy, web frameworks' built-in APIs can downgrade it to None by default

See paper for more!



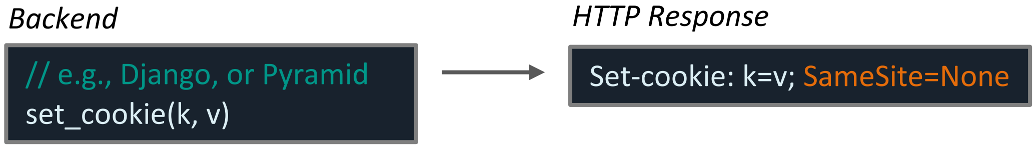
Browser/ Spec	Version	Scope	K=V	K=V; SS=None; Secure	K=V; SS=None
			Set-Cookie HTTP H		
Specification	RFC 6265bis [20]	Same-Site	●	●	○
		Cross-Site	○	●	○
Tor Browser	10.0.12	Same-Site			✗
		Cross-Site		✗	
Chrome	89.0.4389.82	Same-Site			
		Cross-Site			
Opera	74.0.3911.218	Same-Site			
		Cross-Site			
Edge	89.0.774.54	Same-Site			
		Cross-Site			
Firefox	86.0	Same-Site			
		Cross-Site	✗		
Safari	14.0.3	Same-Site			
		Cross-Site	✗		
Safari	12.0.3, 13.1.1	Same-Site			
		Cross-Site	✗	✗	
IE	11.0	Same-Site			
		Cross-Site	✗		
Andr. Chrome	84.0.4147.124	Same-Site			
		Cross-Site			
Andr. Opera	61.2.3076.56749	Same-Site			
		Cross-Site			
Andr. Firefox	79.0.5	Same-Site			
		Cross-Site	✗		
iOS Safari	14.4	Same-Site			
		Cross-Site	✗		
Samsung Int.	13.2.1.70	Same-Site			
		Cross-Site	✗		
Andr. WebView	84.0.4147.105	Same-Site			
		Cross-Site			
iOS Chrome	87.0.4280.77	Same-Site			
		Cross-Site			

Legend: K= Key; V= Value; SS= SameSite; ● = Coo
✗ = Divergent From/Not Compliant w

RQ4: Browser Inconsistencies and Web Frameworks

- Web browsers exhibit seven divergent behaviours wrt. SameSite cookie policy
  - `SameSite=Invalid`
  - `SameSite=None` w/o `Secure`
- Correct and secure use of SameSite requires developer’s awareness
- Even when browsers enforce a default Lax policy, web frameworks’ built-in APIs can downgrade it to None by default

⚠ Affects 24% of the top five frameworks of top five programming languages

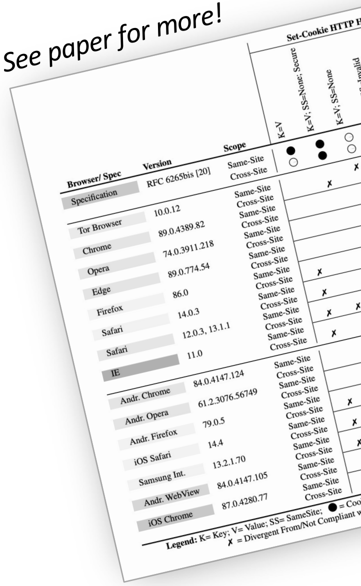


Backend

```
// e.g., Django, or Pyramid
set_cookie(k, v)
```

HTTP Response

```
Set-cookie: k=v; SameSite=None
```



See paper for more!

			Set-Cookie HTTP H			
Browser/ Spec	Version	Scope	K=V	K=V; SS=None; Secure	K=V; SS=None	=Invalid
Specification	RFC 6265bis [20]	Same-Site	●	●	○	
		Cross-Site	○	●	○	
Tor Browser	10.0.12	Same-Site		✗	✗	
		Cross-Site				
Chrome	89.0.4389.82	Same-Site				
		Cross-Site				
Opera	74.0.3911.218	Same-Site				
		Cross-Site				
Edge	89.0.774.54	Same-Site				
		Cross-Site				
Firefox	86.0	Same-Site	✗			
		Cross-Site				
Safari	14.0.3	Same-Site	✗			
		Cross-Site				
Safari	12.0.3, 13.1.1	Same-Site	✗	✗		
		Cross-Site				
Safari	11.0	Same-Site	✗			
		Cross-Site				
IE						
Andr. Chrome	84.0.4147.124	Same-Site				
		Cross-Site				
Andr. Opera	61.2.3076.56749	Same-Site				
		Cross-Site				
Andr. Firefox	79.0.5	Same-Site	✗			
		Cross-Site				
iOS Safari	14.4	Same-Site	✗			
		Cross-Site				
Samsung Int.	13.2.1.70	Same-Site	✗			
		Cross-Site				
Andr. WebView	84.0.4147.105	Same-Site				
		Cross-Site				
iOS Chrome	87.0.4280.77	Same-Site				
		Cross-Site				

Legend: K= Key; V= Value; SS= SameSite; ● = Coo
✗ = Divergent From/Not Compliant w

RQ4: Browser Inconsistencies and Web Frameworks

CISPA
HELMHOLTZ CENTER FOR
INFORMATION SECURITY

- Web browsers exhibit seven divergent behaviours wrt. SameSite cookie policy
  - `SameSite=Invalid`
  - `SameSite=None` w/o `Secure`
- Correct and secure use of SameSite requires developer's awareness
- Even when browsers enforce a default Lax policy, web frameworks' built-in APIs can downgrade it to None by default

⚠ Affects 24% of the top five frameworks of top five programming languages

Backend

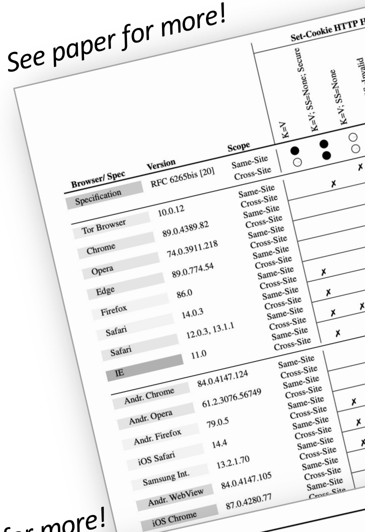
```
// e.g., Django, or Pyramid
set_cookie(k, v)
```

→

HTTP Response

```
Set-cookie: k=v; SameSite=None
```

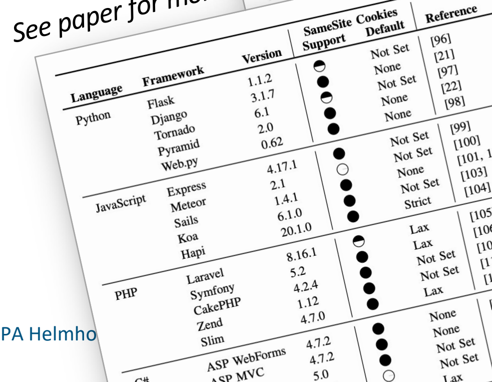
See paper for more!



Browser/ Spec	Version	Scope	K=V	K=V; SS=None; Secure	K=V; SS=None	…=Invalid
Specification	RFC 6265bis [20]	Same-Site	●	●	○	
		Cross-Site	○	●	○	✗
Tor Browser	10.0.12	Same-Site Cross-Site				
Chrome	89.0.4389.82	Same-Site Cross-Site				
Opera	74.0.3911.218	Same-Site Cross-Site				
Edge	89.0.774.54	Same-Site Cross-Site				
Firefox	86.0	Same-Site Cross-Site				
Safari	14.0.3	Same-Site Cross-Site				
Safari	12.0.3, 13.1.1	Same-Site Cross-Site				
IE	11.0	Same-Site Cross-Site				
Andr. Chrome	84.0.4147.124	Same-Site Cross-Site				
Andr. Opera	61.2.3076.56749	Same-Site Cross-Site				
Andr. Firefox	79.0.5	Same-Site Cross-Site				
iOS Safari	14.4	Same-Site Cross-Site				
Samsung Int.	13.2.1.70	Same-Site Cross-Site				
Andr. WebView	84.0.4147.105	Same-Site Cross-Site				
iOS Chrome	87.0.4280.77					

Set-Cookie HTTP H

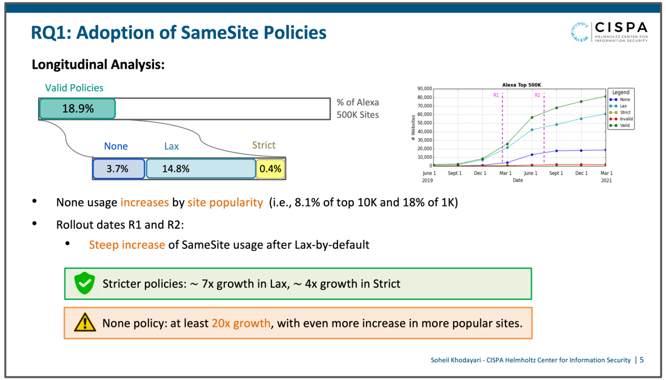
See paper for more!



Language	Framework	Version	SameSite Cookies Support	SameSite Cookies Default	Reference
Python	Flask	1.1.2	◓	Not Set	[96]
	Django	3.1.7	●	None	[21]
	Tornado	6.1	◓	Not Set	[97]
	Pyramid	2.0	●	None	[22]
	Web.py	0.62	●	None	[98]
JavaScript	Express	4.17.1	●	Not Set	[99]
	Meteor	2.1	○	Not Set	[100]
	Sails	1.4.1	●	None	[101, 1
	Koa	6.1.0	●	Not Set	[103]
	Hapi	20.1.0	●	Strict	[104]
PHP	Laravel	8.16.1	◓	Lax	[105
	Symfony	5.2	●	Lax	[106
	CakePHP	4.2.4	●	Not Set	[10
	Zend	1.12	●	Not Set	[1
	Slim	4.7.0	●	Lax	[1
C#	ASP WebForms	4.7.2	●	None	
	ASP MVC	4.7.2	●	None	
		5.0	●	Not Set	
			○	Not Set	
				Lax	

Conclusion

Conclusion



RQ1: Adoption of SameSite Policies

Longitudinal Analysis:

Valid Policies

18.9%

% of Alexa 500K Sites

None Lax Strict

3.7% 14.8% 0.4%

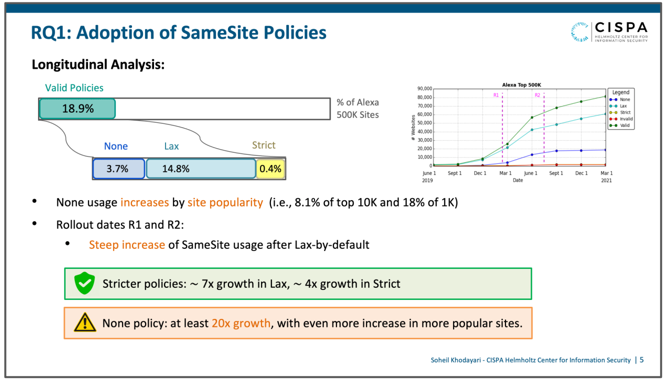
- None usage increases by site popularity (i.e., 8.1% of top 10K and 18% of 1K)
- Rollout dates R1 and R2:
  - Steep increase of SameSite usage after Lax-by-default

✓ Stricter policies: ~ 7x growth in Lax, ~ 4x growth in Strict

⚠ None policy: at least 20x growth, with even more increase in more popular sites.

Soheil Khodayari - CISPA Helmholtz Center for Information Security | 5

Conclusion



RQ1: Adoption of SameSite Policies

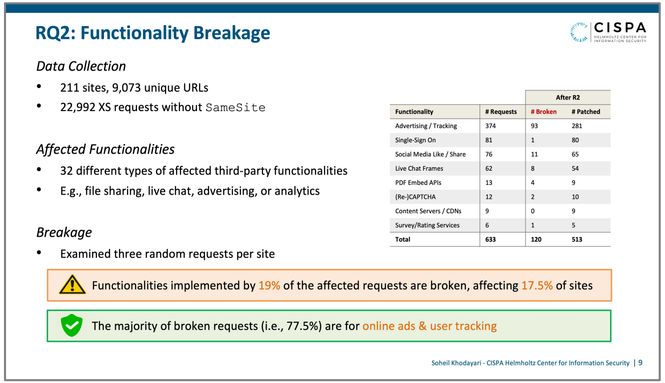
Longitudinal Analysis:

- None usage increases by site popularity (i.e., 8.1% of top 10K and 18% of 1K)
- Rollout dates R1 and R2:
  - Steep increase of SameSite usage after Lax-by-default

✓ Stricter policies: ~ 7x growth in Lax, ~ 4x growth in Strict

⚠ None policy: at least 20x growth, with even more increase in more popular sites.

Soheil Khodayari - CISPA Helmholtz Center for Information Security | 5



RQ2: Functionality Breakage

Data Collection

- 211 sites, 9,073 unique URLs
- 22,992 XS requests without `SameSite`

Affected Functionalities

- 32 different types of affected third-party functionalities
- E.g., file sharing, live chat, advertising, or analytics

Breakage

- Examined three random requests per site

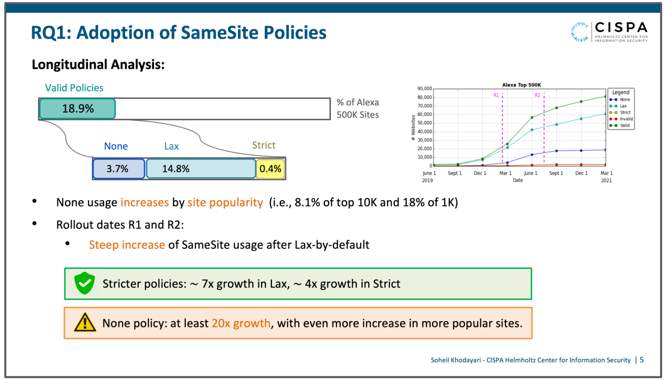
		After R2	
Functionality	**# Requests**	**# Broken**	**# Patched**
Advertising / Tracking	374	93	281
Single-Sign On	81	1	80
Social Media Like / Share	76	11	65
Live Chat Frames	62	8	54
PDF Embed APIs	13	4	9
(Re-)CAPTCHA	12	2	10
Content Servers / CDNs	9	0	9
Survey/Rating Services	6	1	5
Total	**633**	**120**	**513**

⚠ Functionalities implemented by 19% of the affected requests are broken, affecting 17.5% of sites

✓ The majority of broken requests (i.e., 77.5%) are for online ads & user tracking

Soheil Khodayari - CISPA Helmholtz Center for Information Security | 9

Conclusion



RQ1: Adoption of SameSite Policies

Longitudinal Analysis:

Valid Policies

18.9% — % of Alexa 500K Sites

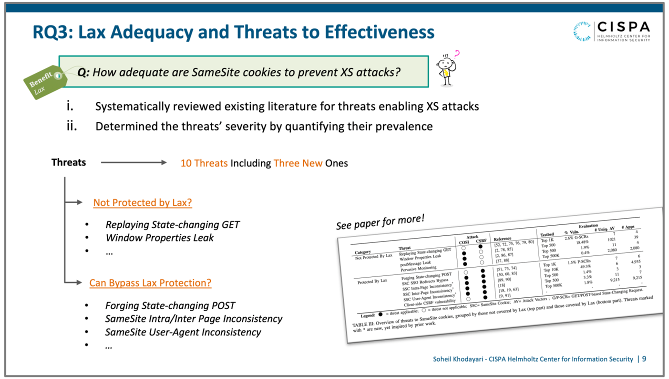
None	Lax	Strict
3.7%	14.8%	0.4%

- None usage increases by site popularity (i.e., 8.1% of top 10K and 18% of 1K)
- Rollout dates R1 and R2:
  - Steep increase of SameSite usage after Lax-by-default

✓ Stricter policies: ~ 7x growth in Lax, ~ 4x growth in Strict

⚠ None policy: at least 20x growth, with even more increase in more popular sites.

Soheil Khodayari - CISPA Helmholtz Center for Information Security | 5



RQ3: Lax Adequacy and Threats to Effectiveness

Benefit Lax

Q: *How adequate are SameSite cookies to prevent XS attacks?*

i. Systematically reviewed existing literature for threats enabling XS attacks

ii. Determined the threats' severity by quantifying their prevalence

Threats → 10 Threats Including Three New Ones

Not Protected by Lax?

- *Replaying State-changing GET*
- *Window Properties Leak*
- …

Can Bypass Lax Protection?

- *Forging State-changing POST*
- *SameSite Intra/Inter Page Inconsistency*
- *SameSite User-Agent Inconsistency*
- …

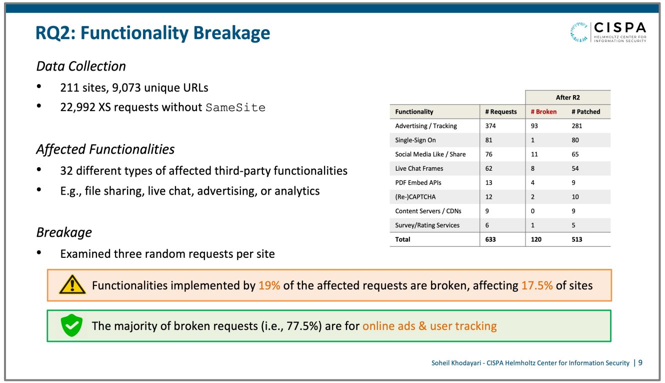
See paper for more!

Category	Threat	Attack COSI	Attack CSRF	Reference	Evaluation Testbed	Evaluation % Vuln.	Evaluation # Uniq. AV	Evaluation # Apps
Not Protected By Lax	Replaying State-changing GET	○	●	[52, 72, 75, 76, 79, 80]	Top 1K	2.6% G-SCRs	7	4
	Window Properties Leak	●	○	[2, 78, 85]	Top 500	18.48%	1021	39
	postMessage Leak	●	○	[2, 86, 87]	Top 500	1.9%	11	4
	Pervasive Monitoring	●	○	[37, 88]	Top 500K	0.4%	2,080	2,080
Protected By Lax	Forging State-changing POST	○	●	[51, 73, 74]	Top 1K	1.5% P-SCRs	7	6
	SSC SSO Redirects Bypass	●	●	[50, 60, 83]	Top 10K	49.3%	6	4,935
	SSC Intra-Page Inconsistency*	●	●	[89, 90]	Top 500	1.4%	3	3
	SSC Inter-Page Inconsistency*	●	●	[18]	Top 500	3.3%	11	7
	SSC User-Agent Inconsistency*	●	●	[18, 19, 63]	Top 500K	1.8%	9,215	9,215
	Client-side CSRF vulnerability	○	●	[9, 91]	-	-	-	-

Legend: ● = threat applicable; ○ = threat not applicable; SSC= SameSite Cookie; AV= Attack Vectors ; G/P-SCR= GET/POST-based State-Changing Request.

TABLE III: Overview of threats to SameSite cookies, grouped by those not covered by Lax (top part) and those covered by Lax (bottom part). Threats marked with * are new, yet inspired by prior work.

Soheil Khodayari - CISPA Helmholtz Center for Information Security | 9



RQ2: Functionality Breakage

Data Collection

- 211 sites, 9,073 unique URLs
- 22,992 XS requests without `SameSite`

Affected Functionalities

- 32 different types of affected third-party functionalities
- E.g., file sharing, live chat, advertising, or analytics

Breakage

- Examined three random requests per site

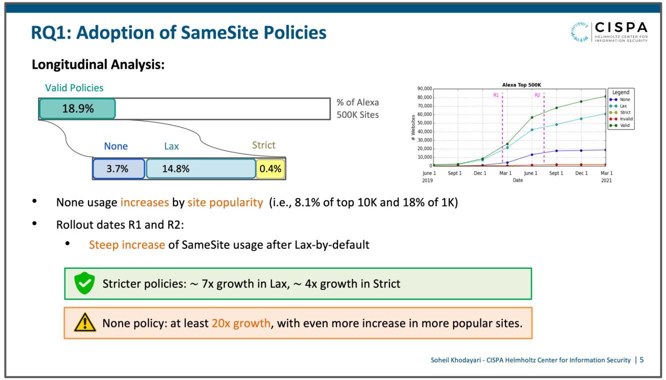
		After R2	
Functionality	**# Requests**	**# Broken**	**# Patched**
Advertising / Tracking	374	93	281
Single-Sign On	81	1	80
Social Media Like / Share	76	11	65
Live Chat Frames	62	8	54
PDF Embed APIs	13	4	9
(Re-)CAPTCHA	12	2	10
Content Servers / CDNs	9	0	9
Survey/Rating Services	6	1	5
Total	**633**	**120**	**513**

⚠ Functionalities implemented by 19% of the affected requests are broken, affecting 17.5% of sites

✓ The majority of broken requests (i.e., 77.5%) are for online ads & user tracking

Soheil Khodayari - CISPA Helmholtz Center for Information Security | 9

Conclusion



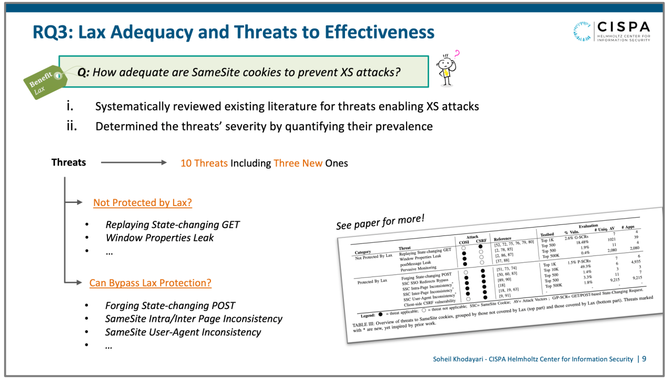
RQ1: Adoption of SameSite Policies

Longitudinal Analysis:

- None usage increases by site popularity (i.e., 8.1% of top 10K and 18% of 1K)
- Rollout dates R1 and R2:
  - Steep increase of SameSite usage after Lax-by-default

Stricter policies: ~ 7x growth in Lax, ~ 4x growth in Strict

None policy: at least 20x growth, with even more increase in more popular sites.



RQ3: Lax Adequacy and Threats to Effectiveness

Q: *How adequate are SameSite cookies to prevent XS attacks?*

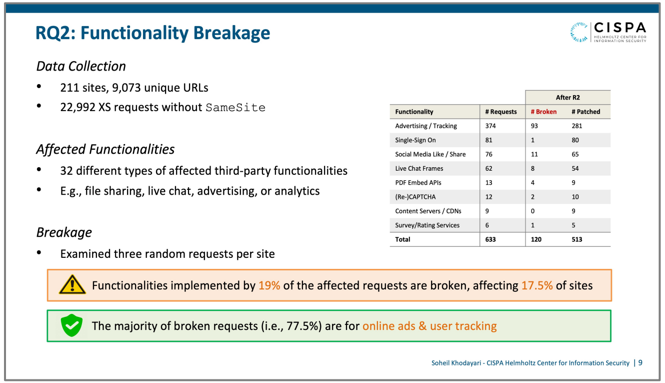
i. Systematically reviewed existing literature for threats enabling XS attacks

ii. Determined the threats' severity by quantifying their prevalence

Threats → 10 Threats Including Three New Ones

- Not Protected by Lax?
  - *Replaying State-changing GET*
  - *Window Properties Leak*
  - …
- Can Bypass Lax Protection?
  - *Forging State-changing POST*
  - *SameSite Intra/Inter Page Inconsistency*
  - *SameSite User-Agent Inconsistency*
  - …

See paper for more!



RQ2: Functionality Breakage

Data Collection

- 211 sites, 9,073 unique URLs
- 22,992 XS requests without `SameSite`

Affected Functionalities

- 32 different types of affected third-party functionalities
- E.g., file sharing, live chat, advertising, or analytics

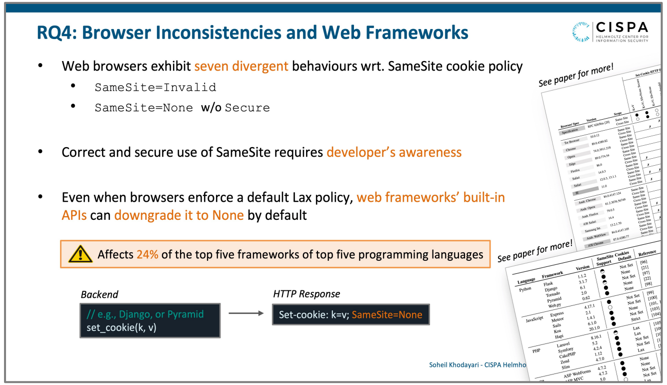
Breakage

- Examined three random requests per site

		After R2	
Functionality	**# Requests**	**# Broken**	**# Patched**
Advertising / Tracking	374	93	281
Single-Sign On	81	1	80
Social Media Like / Share	76	11	65
Live Chat Frames	62	8	54
PDF Embed APIs	13	4	9
(Re-)CAPTCHA	12	2	10
Content Servers / CDNs	9	0	9
Survey/Rating Services	6	1	5
Total	**633**	**120**	**513**

Functionalities implemented by 19% of the affected requests are broken, affecting 17.5% of sites

The majority of broken requests (i.e., 77.5%) are for online ads & user tracking



RQ4: Browser Inconsistencies and Web Frameworks

- Web browsers exhibit seven divergent behaviours wrt. SameSite cookie policy
  - `SameSite=Invalid`
  - `SameSite=None` w/o `Secure`
- Correct and secure use of SameSite requires developer's awareness
- Even when browsers enforce a default Lax policy, web frameworks' built-in APIs can downgrade it to None by default

Affects 24% of the top five frameworks of top five programming languages

See paper for more!

See paper for more!

Backend

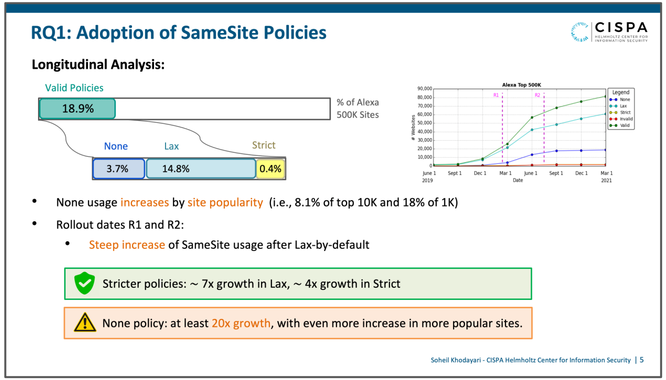
```
// e.g., Django, or Pyramid
set_cookie(k, v)
```

→ *HTTP Response*

```
Set-cookie: k=v; SameSite=None
```

Conclusion

Thank You!



RQ1: Adoption of SameSite Policies

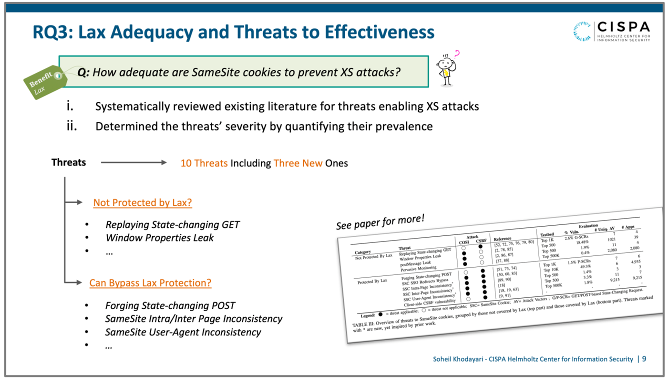
Longitudinal Analysis:

- None usage increases by site popularity (i.e., 8.1% of top 10K and 18% of 1K)
- Rollout dates R1 and R2:
  - Steep increase of SameSite usage after Lax-by-default

Stricter policies: ~ 7x growth in Lax, ~ 4x growth in Strict

⚠ None policy: at least 20x growth, with even more increase in more popular sites.

Soheil Khodayari - CISPA Helmholtz Center for Information Security | 5



RQ3: Lax Adequacy and Threats to Effectiveness

Q: *How adequate are SameSite cookies to prevent XS attacks?*

i. Systematically reviewed existing literature for threats enabling XS attacks

ii. Determined the threats' severity by quantifying their prevalence

Threats → 10 Threats Including Three New Ones

Not Protected by Lax?

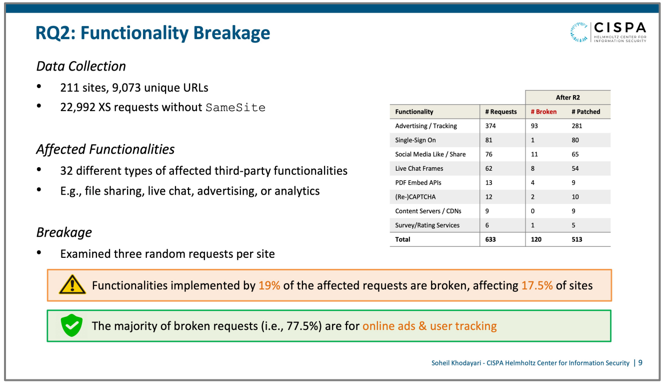
- *Replaying State-changing GET*
- *Window Properties Leak*
- …

Can Bypass Lax Protection?

- *Forging State-changing POST*
- *SameSite Intra/Inter Page Inconsistency*
- *SameSite User-Agent Inconsistency*
- …

See paper for more!

Soheil Khodayari - CISPA Helmholtz Center for Information Security | 9



RQ2: Functionality Breakage

Data Collection

- 211 sites, 9,073 unique URLs
- 22,992 XS requests without `SameSite`

Affected Functionalities

- 32 different types of affected third-party functionalities
- E.g., file sharing, live chat, advertising, or analytics

Breakage

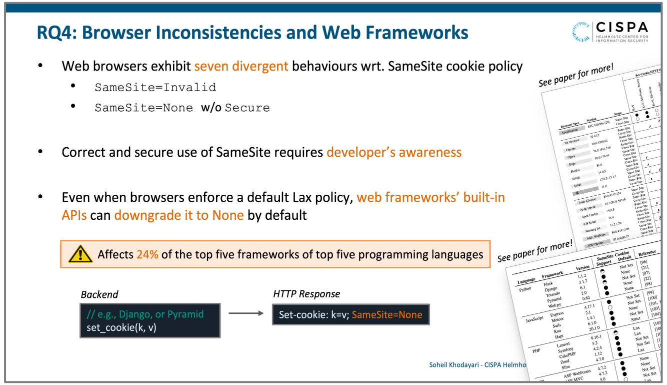
- Examined three random requests per site

		After R2	
Functionality	# Requests	# Broken	# Patched
Advertising / Tracking	374	93	281
Single-Sign On	81	1	80
Social Media Like / Share	76	11	65
Live Chat Frames	62	8	54
PDF Embed APIs	13	4	9
(Re-)CAPTCHA	12	2	10
Content Servers / CDNs	9	0	9
Survey/Rating Services	6	1	5
Total	633	120	513

⚠ Functionalities implemented by 19% of the affected requests are broken, affecting 17.5% of sites

The majority of broken requests (i.e., 77.5%) are for online ads & user tracking

Soheil Khodayari - CISPA Helmholtz Center for Information Security | 9



RQ4: Browser Inconsistencies and Web Frameworks

- Web browsers exhibit seven divergent behaviours wrt. SameSite cookie policy
  - `SameSite=Invalid`
  - `SameSite=None` w/o `Secure`
- Correct and secure use of SameSite requires developer's awareness
- Even when browsers enforce a default Lax policy, web frameworks' built-in APIs can downgrade it to None by default

⚠ Affects 24% of the top five frameworks of top five programming languages

See paper for more!

See paper for more!

Backend → *HTTP Response*

```
// e.g., Django, or Pyramid
set_cookie(k, v)
```

```
Set-cookie: k=v; SameSite=None
```

Soheil Khodayari - CISPA Helmho

@Soheil__K