Cross-Origin State Inference (COSI) Attacks: Your Browser is Leaking Your Secrets

Avinash Sudhodanan
Soheil Khodayari
Juan Caballero

24/02/2020, NDSS 2020

COSI Attack

A malicious web site infers the state of a user (the victim) at another web site

Alice
(victim)

COSI Attack

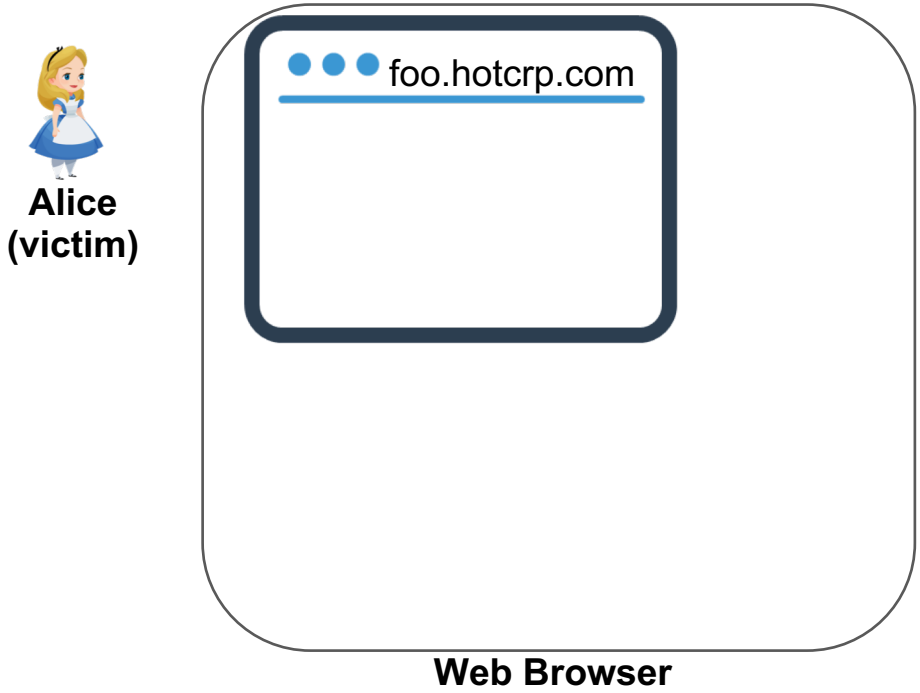
A malicious web site infers the state of a user (the victim) at another web site

Alice (victim)

Web Browser

COSI Attack

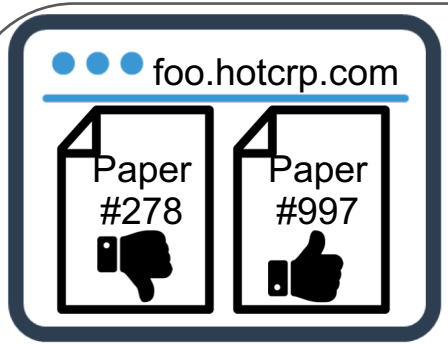
A malicious web site infers the state of a user (the victim) at another web site



COSI Attack

A malicious web site infers the state of a user (the victim) at another web site

Alice (victim)

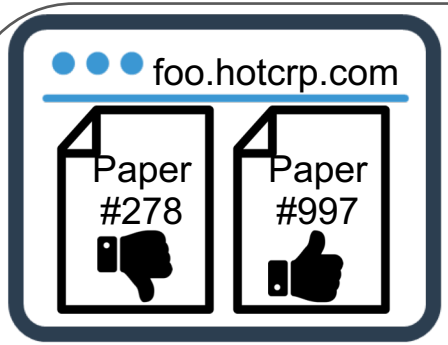


Web Browser

COSI Attack

A malicious web site infers the state of a user (the victim) at another web site

Alice (victim)

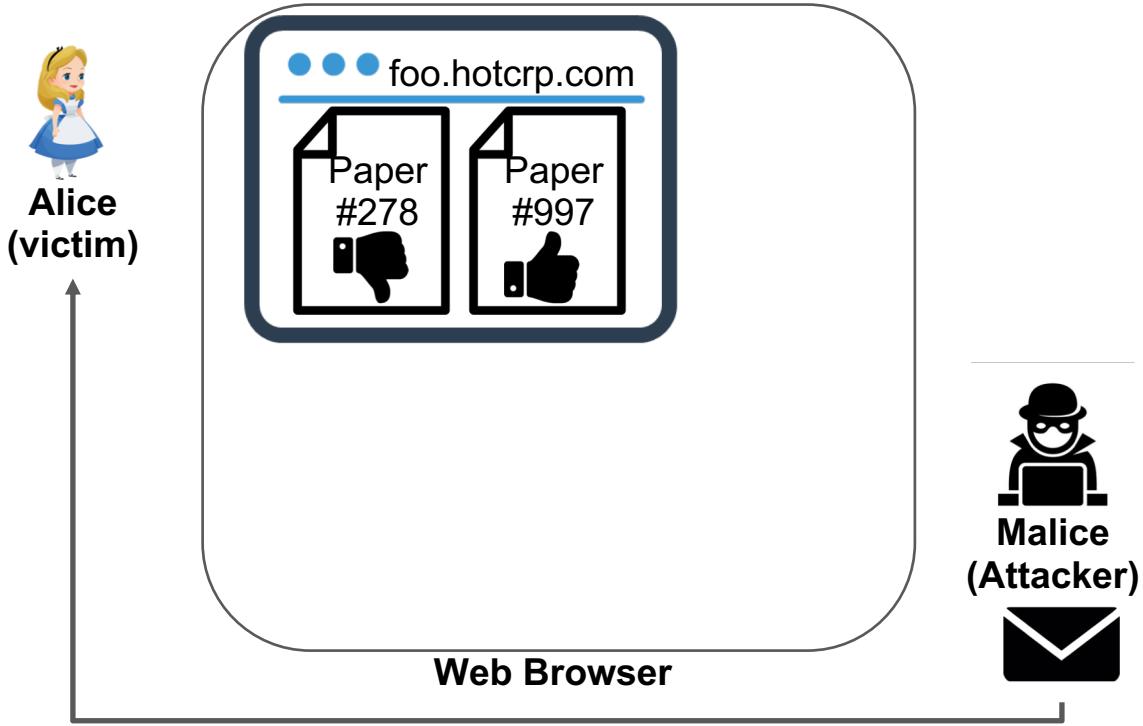


Malice (Attacker)

Web Browser

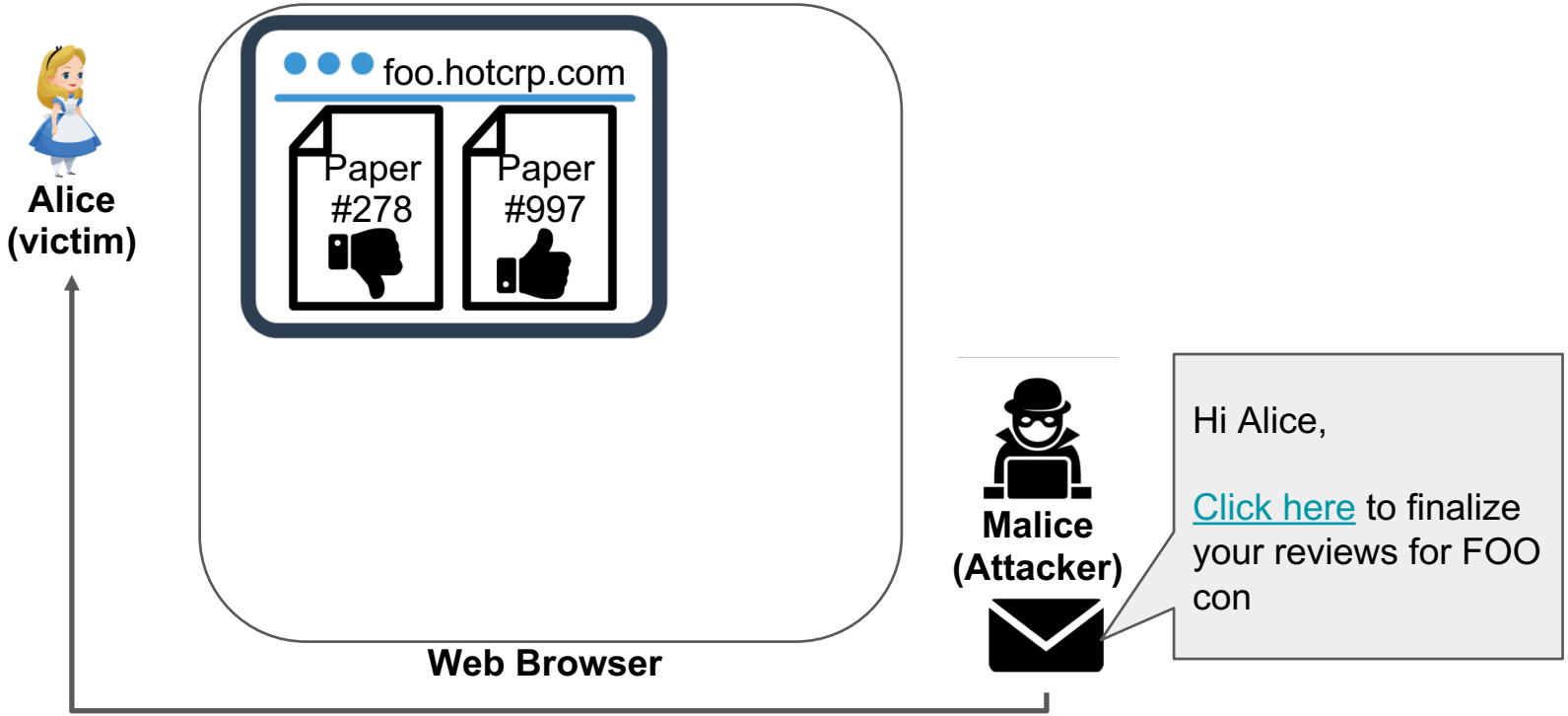
COSI Attack

A malicious web site infers the state of a user (the victim) at another web site



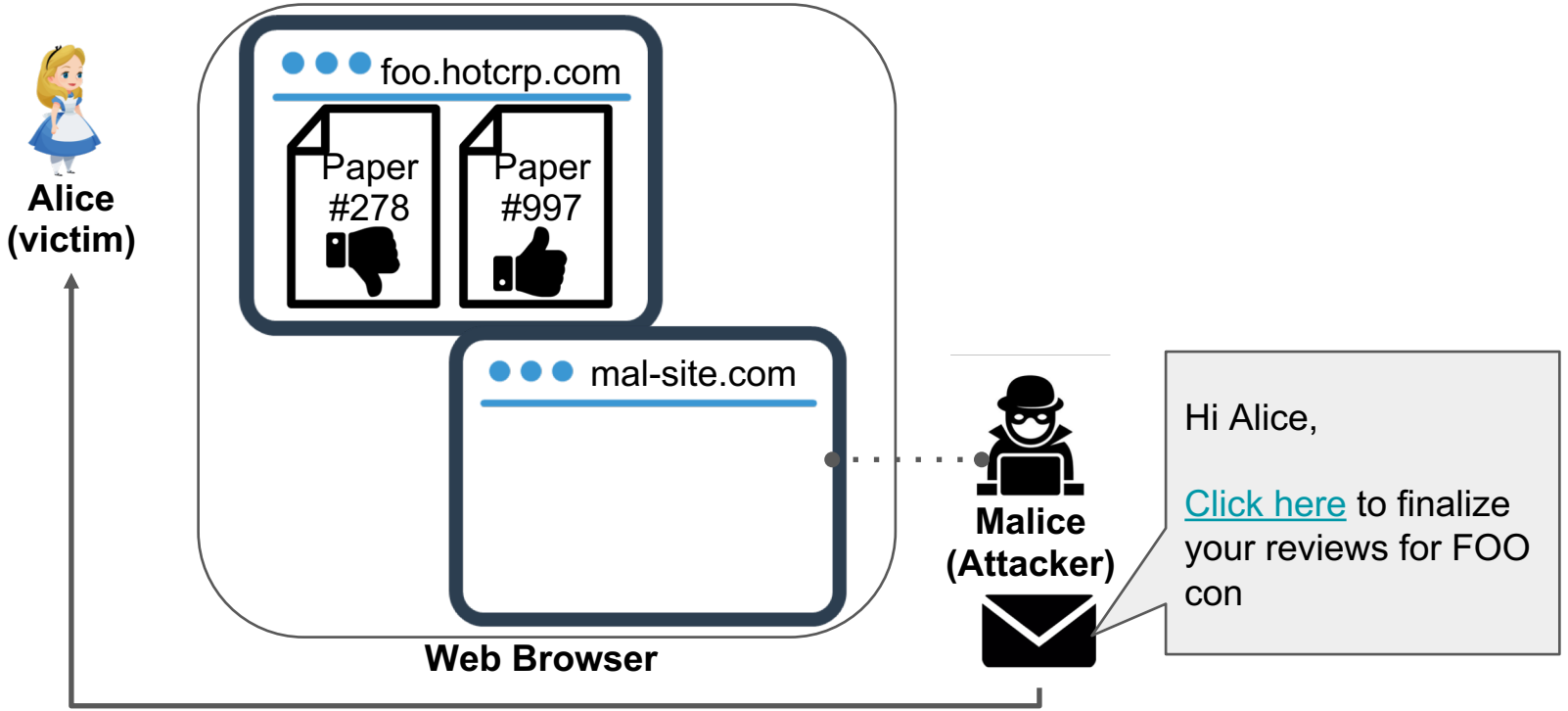
COSI Attack

A malicious web site infers the state of a user (the victim) at another web site



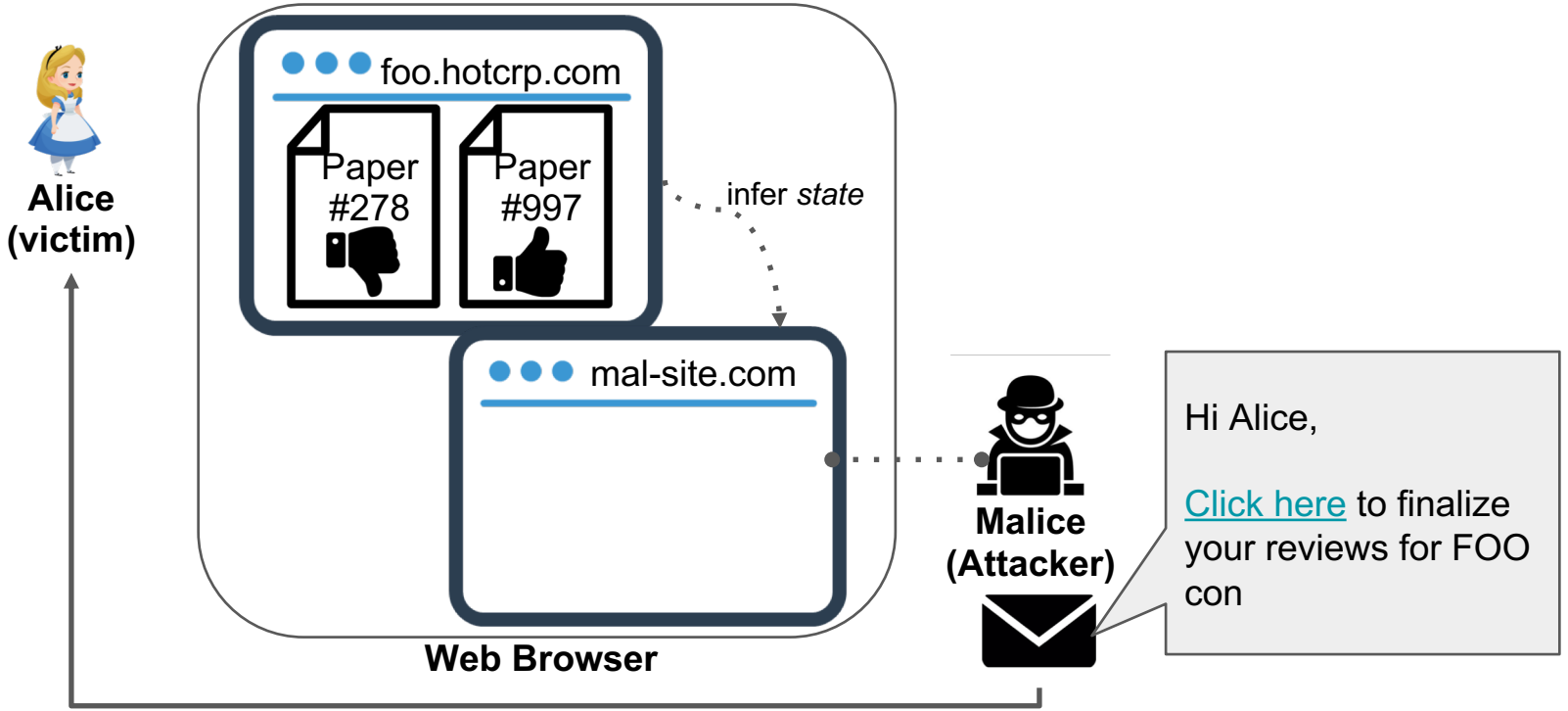
COSI Attack

A malicious web site infers the state of a user (the victim) at another web site



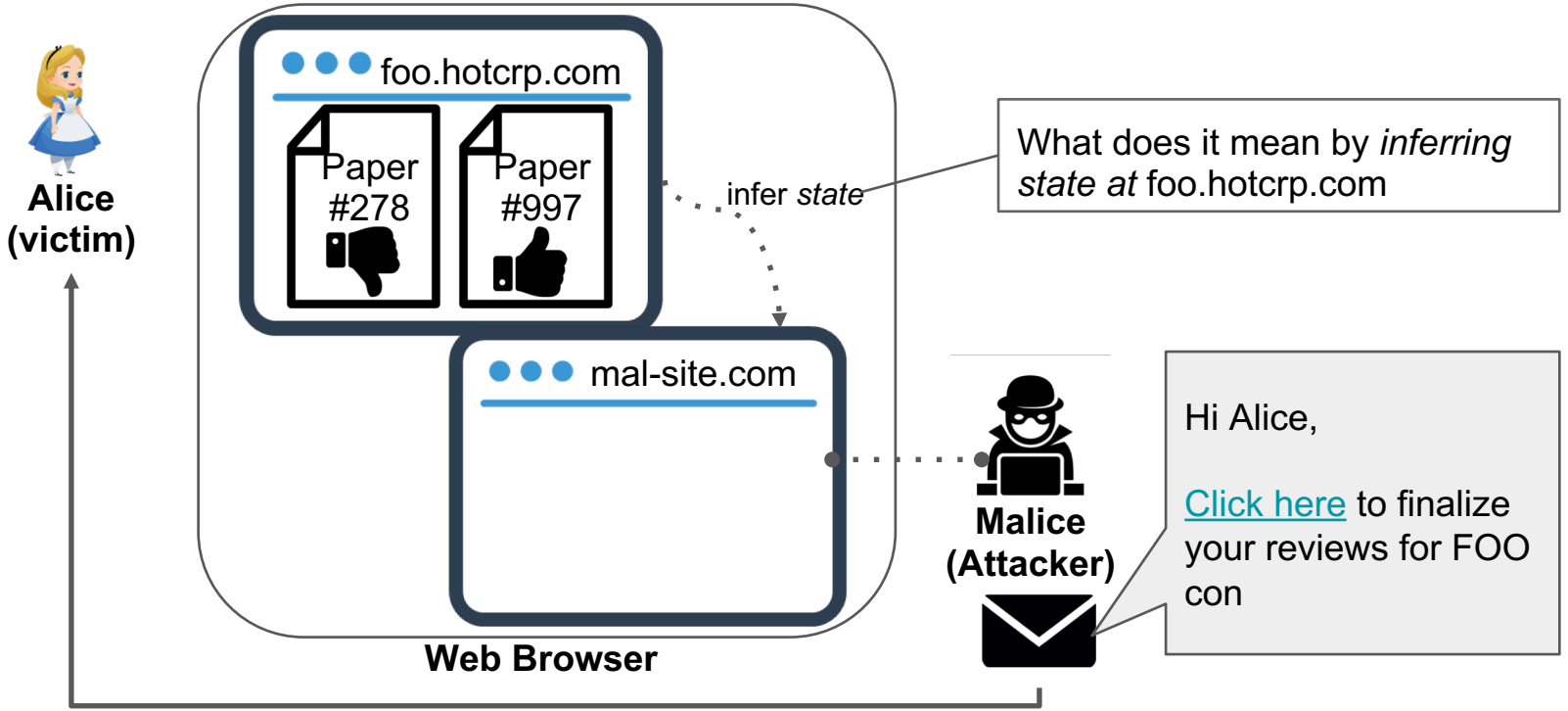
COSI Attack

A malicious web site infers the state of a user (the victim) at another web site



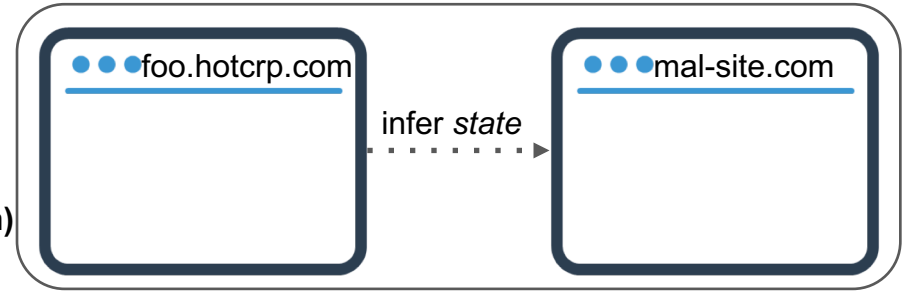
COSI Attack

A malicious web site infers the state of a user (the victim) at another web site



States

Alice (victim)



Web Browser

COSI Attack:
- Attacker's goal: infer states
- Known by different names

Login	Logged In / Logged Out	Login Detection, Login Oracle
Account Type	Reviewer / Author / Admin	
Content Ownership	Owns a review of paper #278 / Does not own a review of paper #278	
Account Ownership	Owns the account *user217* / Does not own the account *user217*	Deanonymization

State-dependent URLs (SD-URLs)

URLs returning different responses depending on the requesting **browser's state**

SD-URL: *https://foo.hotcrp.com/api.php/review?p=278*

State-dependent URLs (SD-URLs)

URLs returning different responses depending on the requesting **browser's state**

SD-URL: *https://foo.hotcrp.com/api.php/review?p=278*	
State	**Response**

State-dependent URLs (SD-URLs)

URLs returning different responses depending on the requesting **browser's state**

SD-URL: *https://foo.hotcrp.com/api.php/review?p=278*	
State	**Response**
Logged In · Reviewer · Reviews paper #278	

State-dependent URLs (SD-URLs)

URLs returning different responses depending on the requesting **browser's state**

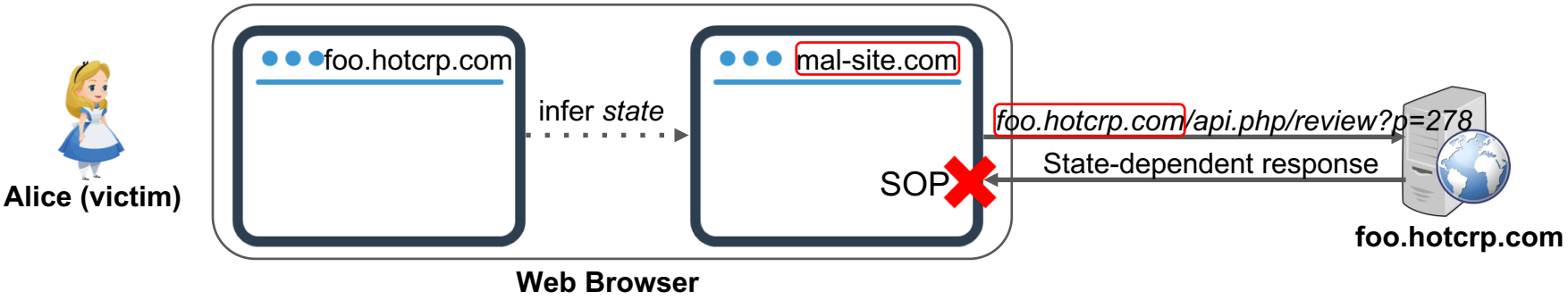
SD-URL: *https://foo.hotcrp.com/api.php/review?p=278*	
State	**Response**
Logged In · Reviewer · Reviews paper #278	code = 200

State-dependent URLs (SD-URLs)

URLs returning different responses depending on the requesting **browser's state**

SD-URL: *https://foo.hotcrp.com/api.php/review?p=278*	
State	**Response**
Logged In · Reviewer · Reviews paper #278	code = 200
Logged In · Reviewer · Not review paper #278	code = 403

State-dependent URLs (SD-URLs)



SD-URL: *https://foo.hotcrp.com/api.php/review?p=278*

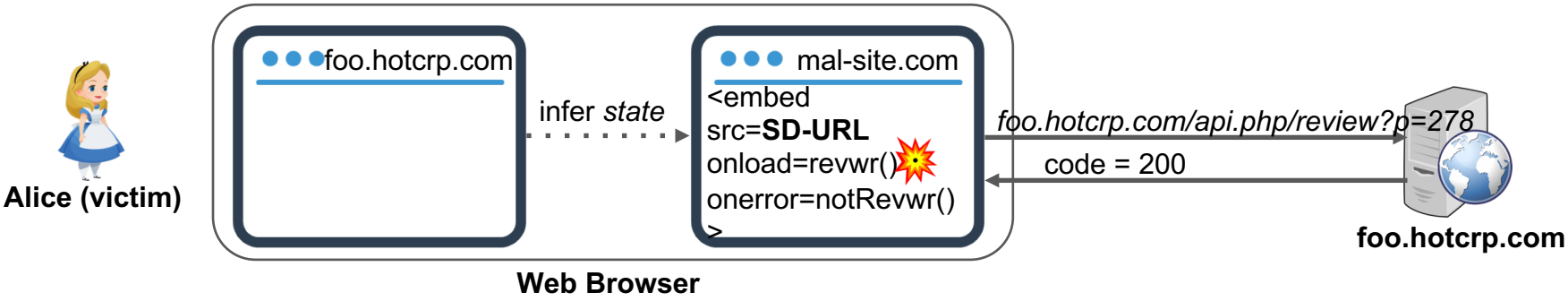
State	**Response**
Logged In, Reviewer, Reviews paper #278	code = 200
Logged In, Reviewer, Not review paper #278	code = 403

XS-Leaks

Browser side-channels for inferring the response of **cross-origin** requests

Leak Type	**References**
Events-Fired	[Grossman2006Blog, Goethem2015CCS, Cardwell2011Blog, ..]
Object-Properties	[Grossman2012Blog, Schwenk2017USENIX, Masas2018Blog..]
JS-Error	[Grossman2006Blog, Shiflett2006Blog]
CSS-Properties	[Evans2008Blog]
CSP-Violation	[Homakov2013Blog, Gulyas2018WPES]
Timing	[Bortz2007WWW, Evans2009Blog, Goethem2015CCS, ..]
AppCache	[Lee2015NDSS]

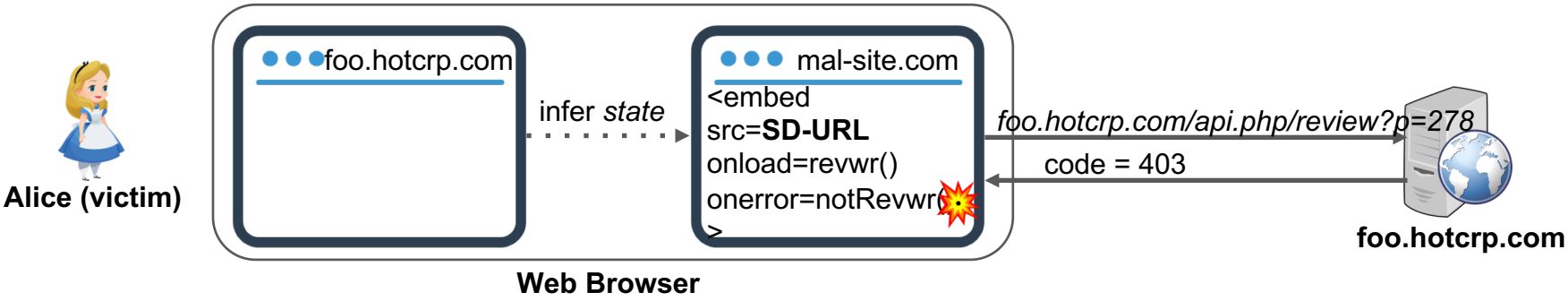
Events Fired XS-Leak



SD-URL: *https://foo.hotcrp.com/api.php/review?p=278*

State			Response
Logged In	Reviewer	Reviews paper #278	code = 200
Logged In	Reviewer	Not review paper #278	code = 403

Events Fired XS-Leak



SD-URL: *https://foo.hotcrp.com/api.php/review?p=278*

State			Response
Logged In	Reviewer	Reviews paper #278	code = 200
Logged In	Reviewer	Not review paper #278	code = 403

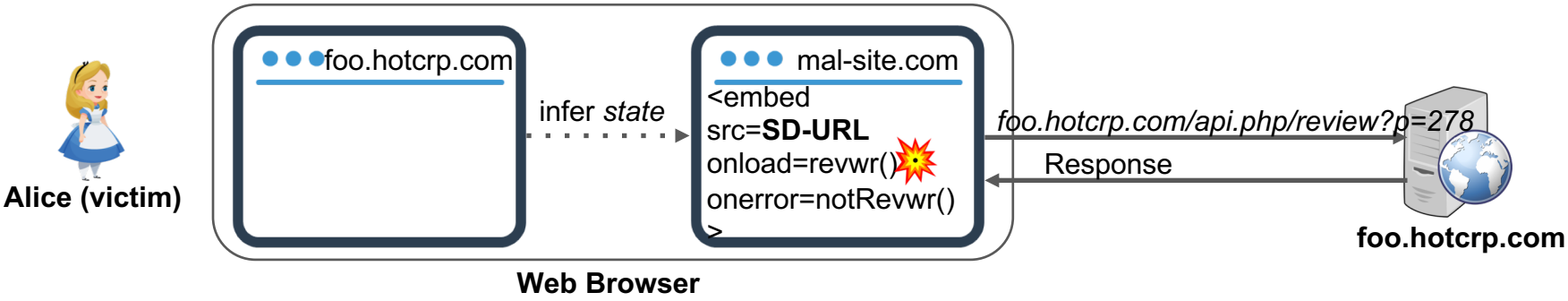
Multiple States, Same Response

SD-URL: *https://foo.hotcrp.com/api.php/review?p=278*	
State	**Response**
Logged In · Reviewer · Reviews paper #278	code = 200
Logged In · Reviewer · Not review paper #278	code = 403

Multiple States, Same Response

SD-URL: *https://foo.hotcrp.com/api.php/review?p=278*	
State	**Response**
Logged In · Reviewer · Reviews paper #278	code = 200
Logged In · Reviewer · Not review paper #278	code = 403
Logged Out	code = 200

Multiple States, Same Response

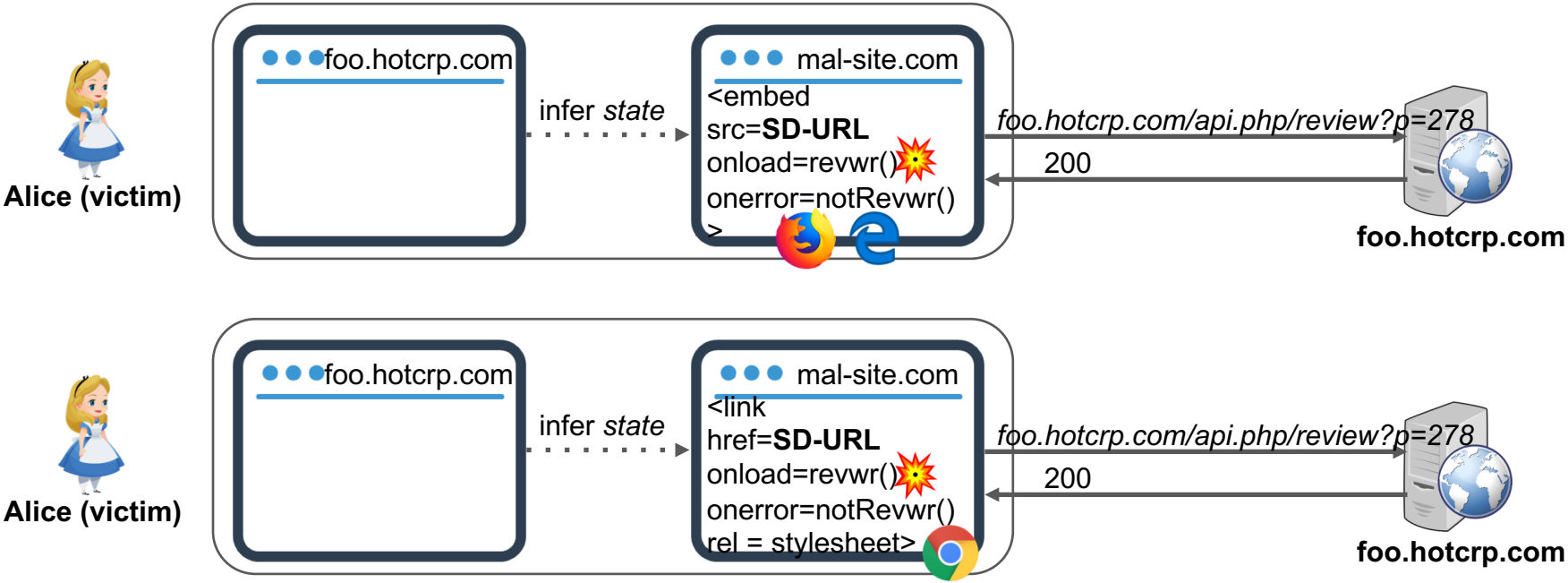


SD-URL: *https://foo.hotcrp.com/api.php/review?p=278*

State			Response
Logged In	Reviewer	Reviews paper #278	code = 200
Logged In	Reviewer	Not review paper #278	code = 403
Logged Out			code = 200

Same Attack Payload, Browser-specific Behavior

The same XS-Leak payload may work differently on **different browsers**



Attack Classes

Attack Classes

Name

:

Attack Classes

Name	SD-URL Responses	
	Response A	Response B

:

Attack Classes

Name	SD-URL Responses		XS-Leak	
	Response A	Response B	Inclusion	Manifest.

:

Attack Classes

Name	SD-URL Responses		XS-Leak		Browser Support		
	Response A	Response B	Inclusion	Manifest.	Chrome	Firefox	Edge

:

Attack Classes

Name	SD-URL Responses		XS-Leak		Browser Support		
	Response A	**Response B**	**Inclusion**	**Manifest.**	**Chrome**	**Firefox**	**Edge**
EF-StatusErrorScript	*code* = 200 *content-type* = text/javascript	*code* = 4xx || 5xx	<script>	onload / onerror	✓	✓	✓

:

Attack Classes

Name	SD-URL Responses		XS-Leak		Browser Support		
	Response A	Response B	Inclusion	Manifest.	Chrome	Firefox	Edge
EF-StatusErrorScript	*code* = 200 *content-type* = text/javascript	*code* = 4xx || 5xx	<script>	onload / onerror	✓	✓	✓

Class	SD-URL Responses		Attack Page's Logic		Browsers		
	Response A	*Response B*	*Inclusion Methods*	*Leak Method*	*Firefox*	*Chrome*	*Edge*
EF-StatusErrorScript	sc = 200, ct = text/javascript	sc = (4xx OR 5xx)	`script src=URL`	[onload] / [onerror]	✓	✓	✓
EF-StatusErrorObject	sc = 200, ct ≠ (audio OR video)	sc ≠ (200 OR 3xx)	`object data=URL`	[onload] / [onerror]	✓	✗	✗
EF-StatusErrorEmbed	sc = 401, ct = (text/html)	sc ≠ 401, ct = (text/html)	`embed src=URL`	[] / [onload]	✗	✗	✓
EF-StatusErrorLink	sc = (200 OR 3xx), ct ≠ text/html	sc ≠ (200 OR 3xx)	`link href=URL rel=prefetch`	[onload] / [onerror]	✗	✓	✗
EF-StatusErrorLinkCss	sc = (200 OR 3xx), ct = text/css	sc ≠ (200 OR 3xx), ct ≠ text/css	`link href=URL rel=stylesheet`	[onload] / [onerror]	✓	✓	✗
EF-RedirStatLink	sc = 3xx	sc ≠ 3xx, cto = nosniff, ct ≠ (text/css OR text/html)	`link href=URL rel=stylesheet`	[onload] / [onerror]	✗	✓	✗
EF-StatusErrorIFrame	sc = (200 OR 3xx OR 4xx or 5xx), ct= (text/javascript OR text/css)	sc = (200 OR 3xx OR 4xx or 5xx), ct ≠ (text/javascript OR text/css)	`iframe src=URL`	[] / [onload]	✗	✗	✓

:

Attack Classes

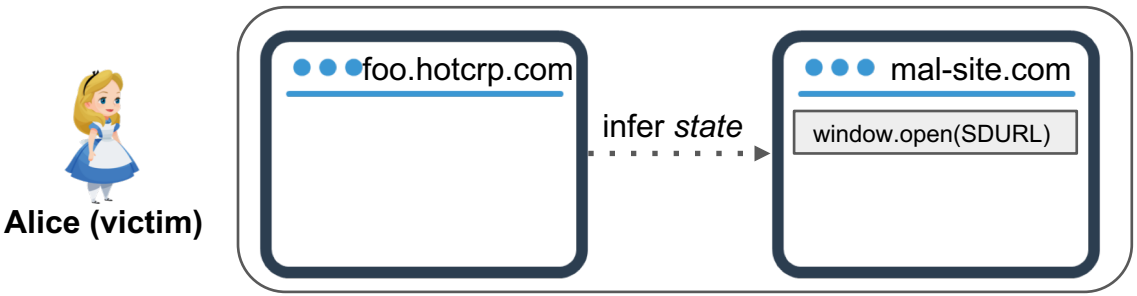
Name	SD-URL Responses		XS-Leak		Browser Support		
	Response A	**Response B**	**Inclusion**	**Manifest.**	**Chrome**	**Firefox**	**Edge**
EF-StatusErrorScript	*code* = 200 *content-type* = text/javascript	*code* = 4xx || 5xx	<script>	onload / onerror	✓	✓	✓

- **40** attack classes
- **21** new attack classes
- **1** completely novel XS-Leak (based on **postMessage** API)

New XS-Leak: postMessage broadcasts

- SD-URL property

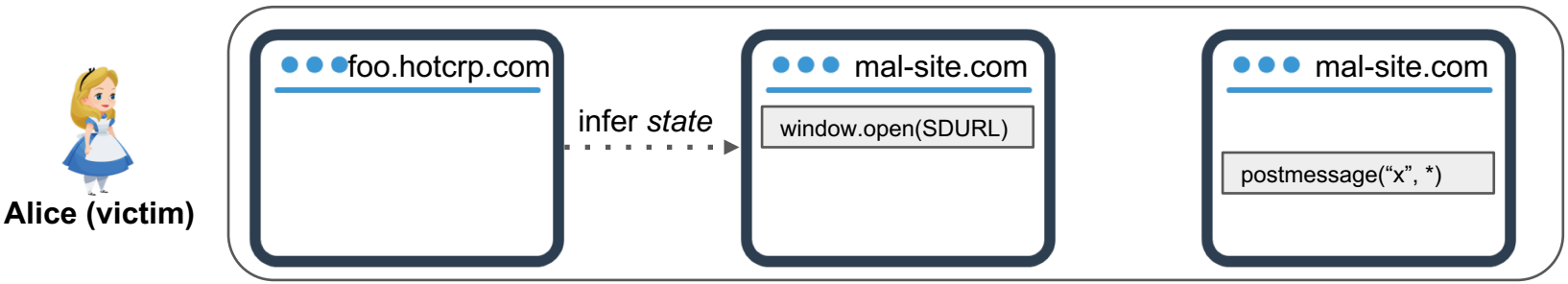
State	Response
A	Broadcasts message “x”
B	Broadcasts message “y”



New XS-Leak: postMessage broadcasts

- SD-URL property

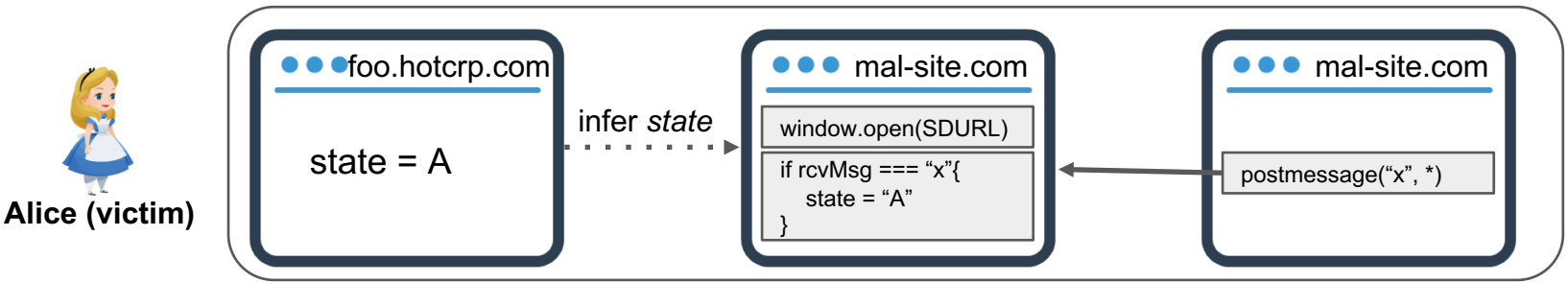
State	Response
A	Broadcasts message “x”
B	Broadcasts message “y”



New XS-Leak: postMessage broadcasts

- SD-URL property

State	Response
A	Broadcasts message “x”
B	Broadcasts message “y”



Related Work

Attack	References
Events-Fired	[Grossman2006Blog, Goethem2015CCS, Cardwell2011Blog, ..]
Object-Properties	[Grossman2012Blog, Schwenk2017USENIX, Masas2018Blog..]
JS-Error	[Grossman2006Blog, Shiflett2006Blog]
CSS-Properties	[Evans2008Blog]
CSP-Violation	[Homakov2013Blog, Gulyas2018WPES]
Timing	[Bortz2007WWW, Evans2009Blog, Goethem2015CCS, ..]
AppCache	[Lee2015NDSS]
:	:

- Given **different names** login detection, login oracle, URL status identification, etc.
- Not much discussion on
  - need to handle multiple states
  - support for multiple browsers
  - automatic detection of COSI attacks and automatic creation of attack pages

Contributions

Present COSI attacks as a **comprehensive category**

Introduce the concept of **attack classes**

Identify a new XS-Leak (based on postMessage API)

Present **Basta-COSI**, a tool to automatically identify COSI attacks and build complex attack pages

Test **4** stand-alone web apps and **58** top web sites, discovering COSI attacks on all of them

Outline

Introduction

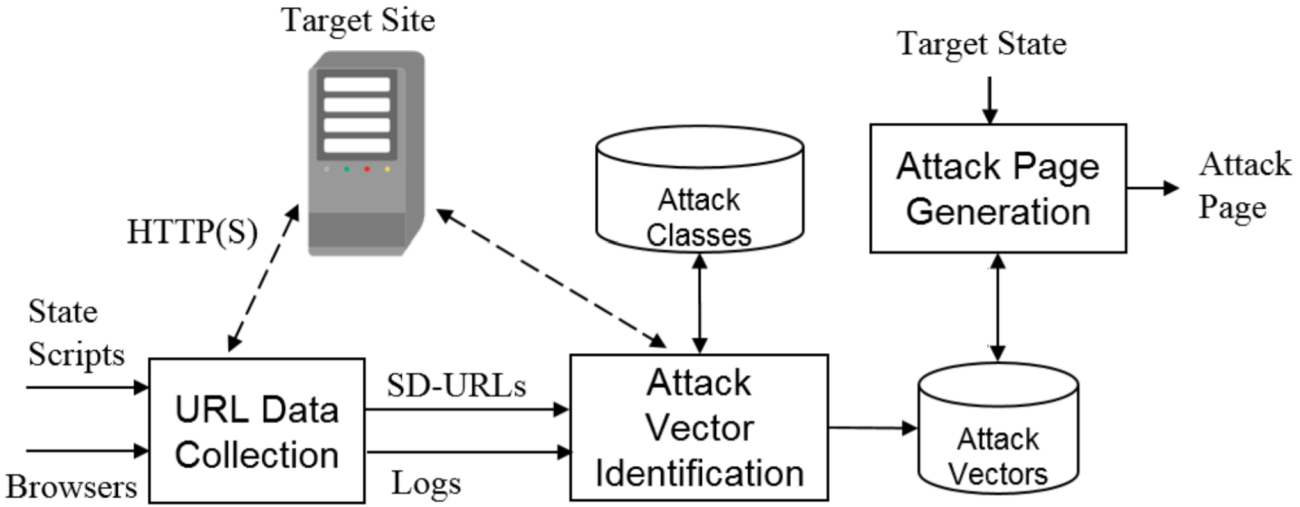
➔ **Approach**

Evaluation

Conclusion

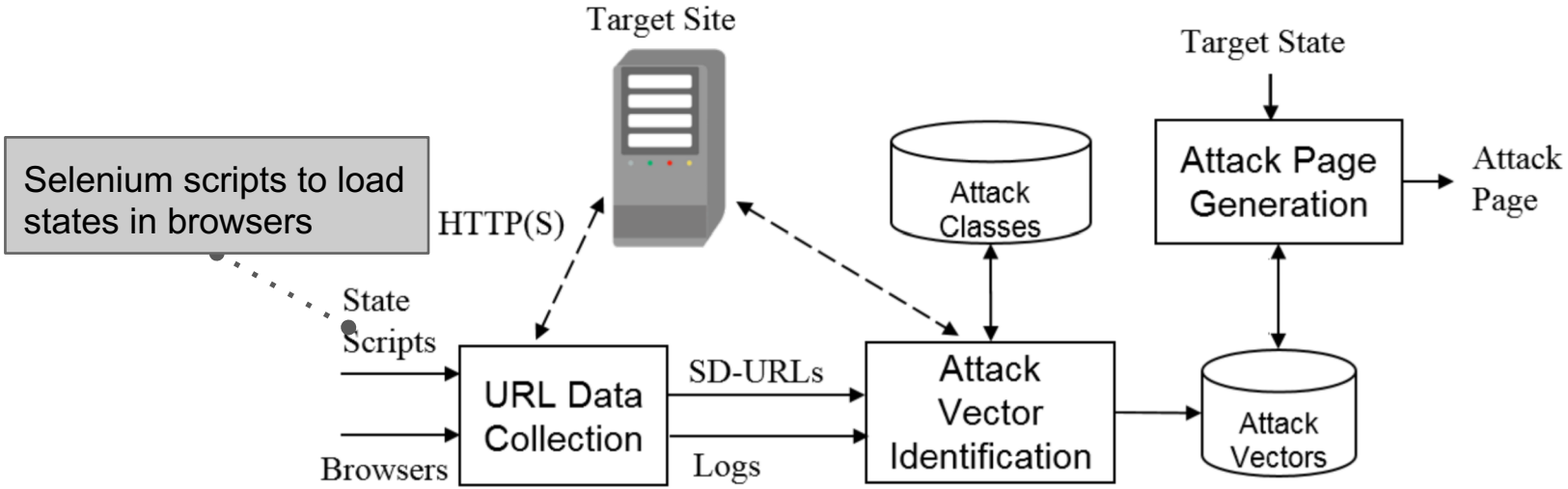
Basta-COSI: Architecture

Tool to automatically identify COSI attacks on web apps



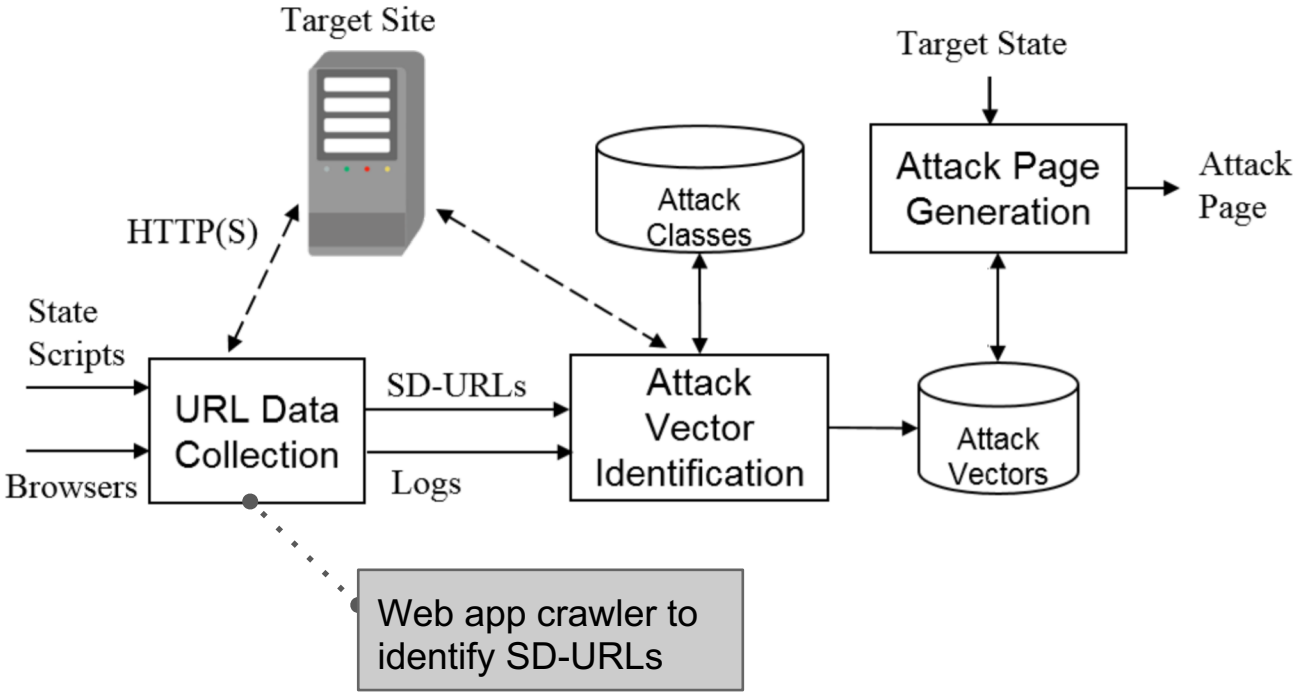
Basta-COSI: Architecture

Tool to automatically identify COSI attacks on web apps



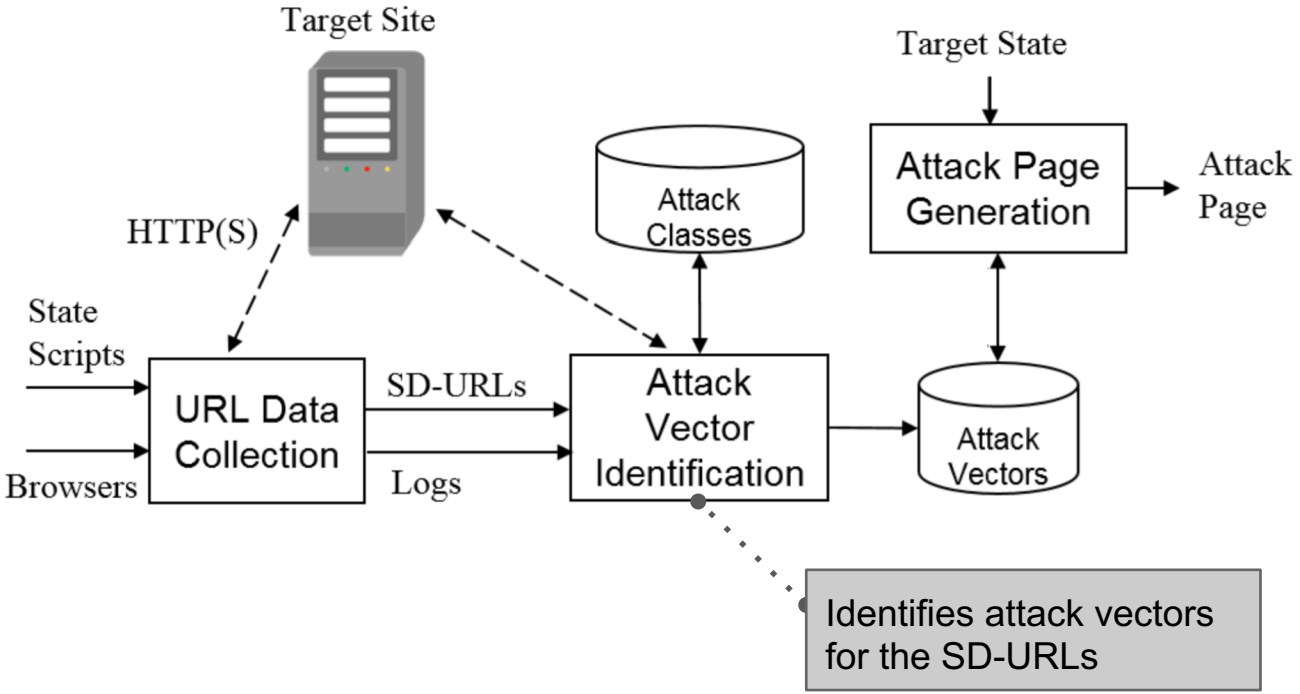
Basta-COSI: Architecture

Tool to automatically identify COSI attacks on web apps



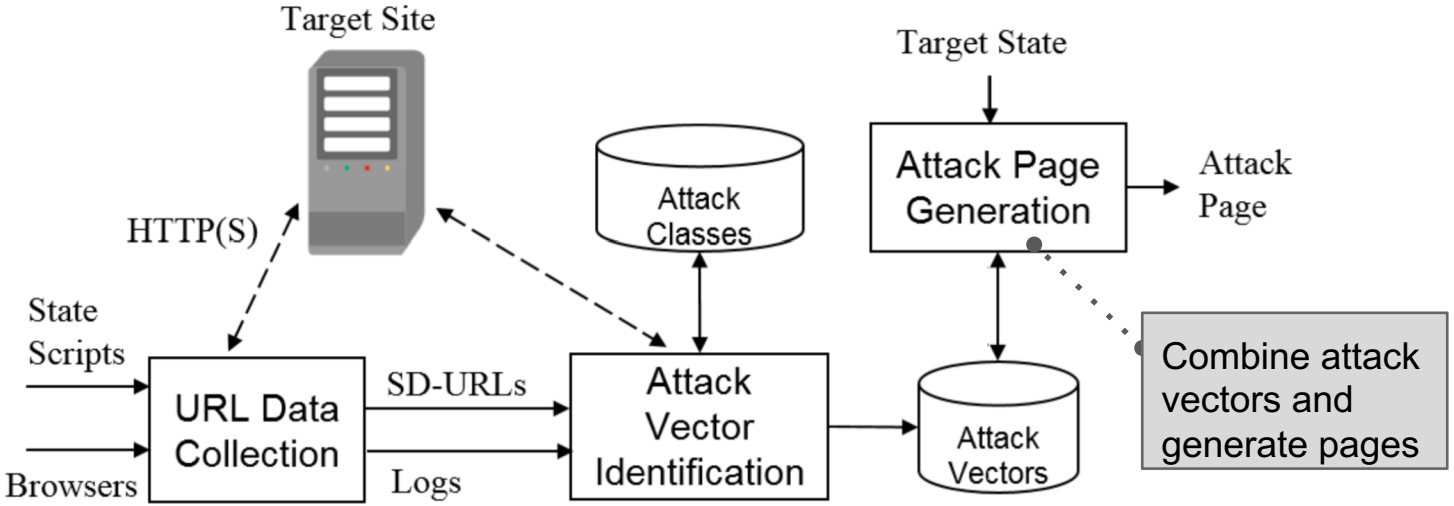
Basta-COSI: Architecture

Tool to automatically identify COSI attacks on web apps



Basta-COSI: Architecture

Tool to automatically identify COSI attacks on web apps



Outline

Introduction

Approach

➔ **Evaluation**

Conclusion

Experiments

Tested:

- **4** stand-alone web applications (HotCRP, GitLab, GitHub Enterprise, OpenCart)
- **58** web sites from the Alexa Top 150 (having account creation)

Results:

- Found at least one attack on **all sites**
- Responsibly disclosed, vulnerabilities confirmed, some already fixed, some bug bounties paid

Results (Excerpt)

Web Site Vendor	Top Vulnerabilities
HotCRP.com	Deanonymize reviewer

Results (Excerpt)

Web Site Vendor	Top Vulnerabilities
HotCRP.com	Deanonymize reviewer
Blogger 	Deanonymize blog/file owner
Pornhub	Deanonymize channel owner

Results (Excerpt)

Web Site Vendor	Top Vulnerabilities
HotCRP.com	Deanonymize reviewer
Blogger	Deanonymize blog/file owner
Pornhub	Deanonymize channel owner
IMDb	Deanonymize user
imgur	
LinkedIn	

Results (Excerpt)

Web Site Vendor	Top Vulnerabilities
HotCRP.com	Deanonymize reviewer
Blogger	Deanonymize blog/file owner
Pornhub	Deanonymize channel owner
IMDb	Deanonymize user
imgur	
LinkedIn	
GitHub GitLab	Deanonymize users and role

Ethics: All tests were performed on the accounts we controlled

Defenses

Web site-based:

- SameSite Cookies (control automatic sending of cookies in **3rd party context**)
- Cross-Origin Resource Policy
- Fetch Metadata
- Cross-Origin Opener Policy..

Browser-based:

- Enforce default SameSite policy
- Tor browser behavior SameSite=Lax (not send cookies in 3rd party context)
  - does not prevent window based attacks (postMessage, frame count)
  - reported and Tor is planning to fix this

Conclusions

Present COSI attacks as a **comprehensive category**

Introduce the concept of **COSI attack classes**

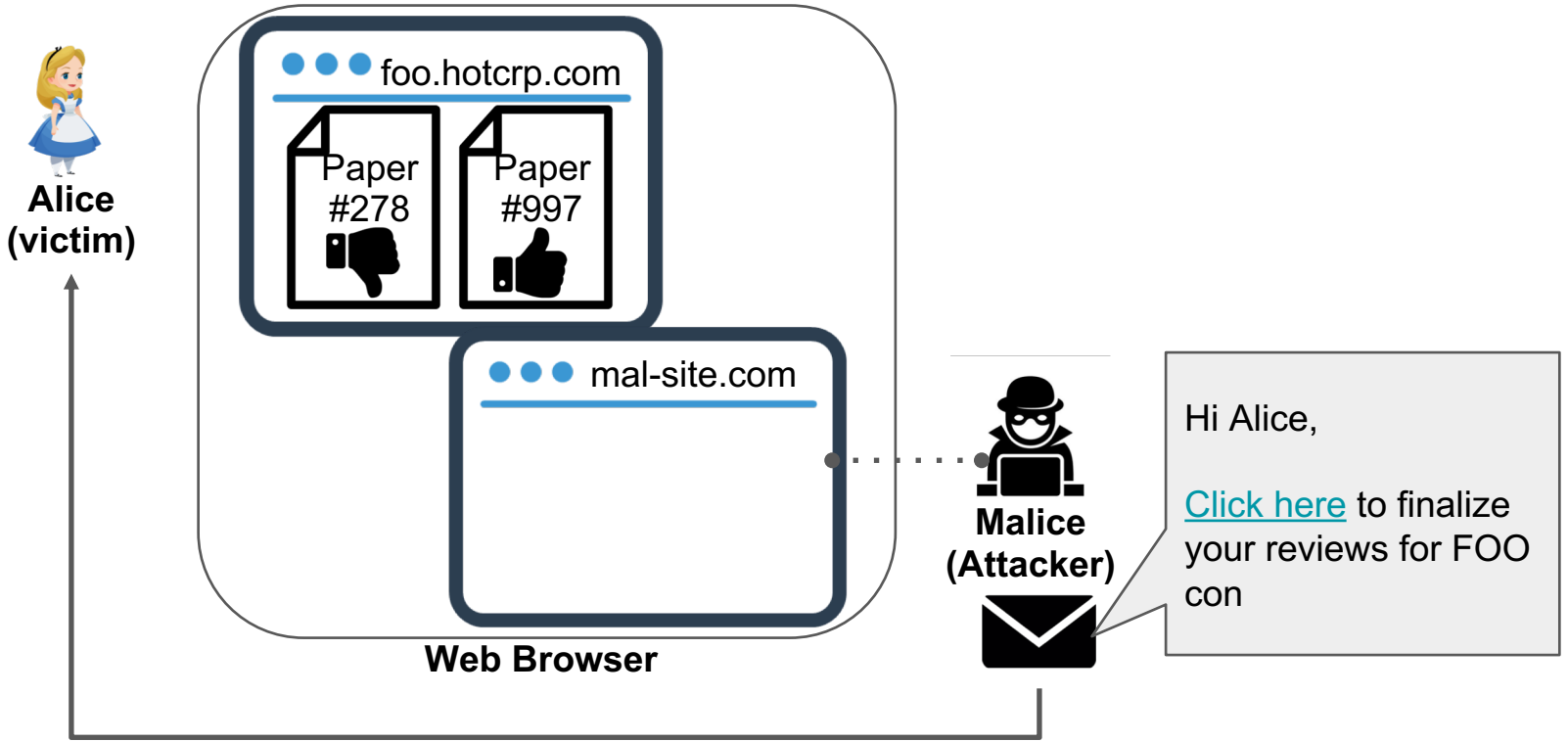
Identify a **new XS-Leak** (based on postMessage API)

Present Basta-COSI, a tool to **automatically** identify COSI attacks and build **complex** attack pages

Test **4** stand-alone web apps and **58** top web sites, finding COSI attacks on each of them

Thank You

Questions?


Alice
(victim)
foo.hotcrp.com
Paper
#278
Paper
#997
mal-site.com
Web Browser
Malice
(Attacker)
Hi Alice,
Click here to finalize your reviews for FOO con