The State of the Cross-domain Nation

Sebastian Lekies, Martin Johns and Walter Tighzert

W2SP / May 26th 2011

Executive summary

We did an exhaustive survey on the current practice of permitting client-side cross-domain HTTP requests

- Flash, Silverlight (and CORS)

Result: A considerable fraction of sites utilize potentially insecure policies

Agenda

Technical background

Methodology

Results

Conclusion

Technical Background

The how and why of client-side cross-domain requests

Client-side cross-domain requests

- Active code in the browser can initiate cross-domain HTTP requests and receive the corresponding HTTP response
- Generally forbidden by the same origin policy
- However, can be conducted with Flash, Silverlight, or CORS under certain circumstances

The need for this mechanism is not immediately obvious

Alternative: Server-side proxies

- Capable of cross-domain data retrieval
- Compliant with the same-origin policy
  - Requests are routed through the script's original host

Advantage of client-side cross-domain requests:

The HTTP requests are created in the user's current authentication context

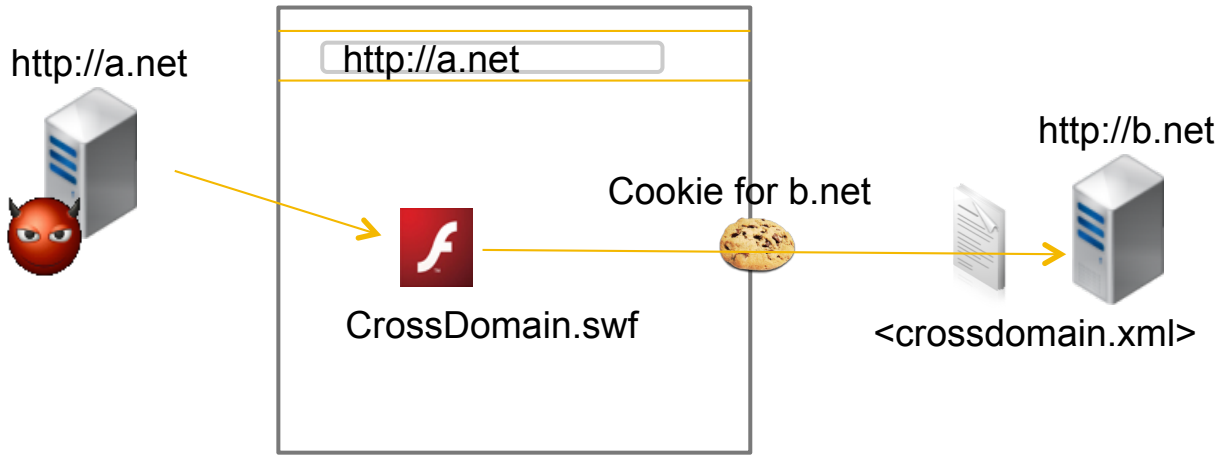
- Cookies
- Creation within the current intranet

This allows application scenarios which are impossible with server-side proxies

Security implications

Scenario

1. An adversary controlled client-side script is permitted to create cross-domain HTTP requests and receive the corresponding HTTP responses
2. These requests are created in the user's current authentication context
   - I.e., the requests carry the user's session cookies



Potential attack vectors

Leakage of sensitive information

- The adversary can request sensitive web resources

Circumvention of CSRF protection

- Token-based CSRF protection relies on the fact, that the adversary cannot read cross-domain data

Session hijacking

- Chaining requests & reading responses
  - Capabilities equal to XSS session hijacking

Allowing client-side cross-domain requests

To avoid the outlined security implications cross-domain HTTP requests have to be allowed by the receiving site

Flash

crossdomain.xml policy files

- List of trusted sites which are allowed to create requests
- Before issuing a request, the flash-plugin first retrieves the policy and verifies that the origin of the requesting script is listed in the policy

Silverlight

clientaccesspolicy.xml policy files

- Similar mechanism as the one pioneered by Flash, with subtle differences
- Fallback to subset of crossdomain.xml policy files possible

CORS

- HTTP response header
- Allows fine grained control based on incoming origin-headers

Insecure conditions

```
<cross-domain-policy>
  <site-control
    permitted-cross-domain-policies="all" />
  <allow-access-from domain="*" />
  <allow-http-request-headers-from domain="*"
        headers="*" />
</cross-domain-policy>
```

Wildcard policies

- “*”
- Whitelists all existing domains
- Results in conditions that roughly match a XSS flaw

Transitivity of insecurity

- If a site is compromised or allows invalidated file uploads, all sites that whitelist this site are exposed to the described attacks

The Survey: Methodology

Research questions

(R1) Penetration

- How prevalent are cross-domain policies?
- Which technologies are used for this purpose?
- Can a trend towards CORS be observed?
- What kind of sites issue cross-domain policies?

(R2) Security

- How high is the ratio of potentially insecure policies?
- How is the relationship between (in)security and site category?
- Is there a correlation between (in)security and site popularity?
- Which are the sites that are most often whitelisted?

Identifying insecure policies

Observation: A wildcard alone does not cause insecurities

A necessary condition is that the permissive site indeed conducts authentication tracking

Our approach

- Check for evidence that indicates that a authentication state can be provided by the site
  - Password fields
  - Login dialogues
  - Session identifiers (HTTPonly cookies, naming conventions)
- If authentication forms pointed to different (sub)-domains, we also checked the policy file for the form's target domain

Classification of sites

Correlation between potential insecurity and purpose of the site

- Hence, site classification needed
- Alexa categories did not provide reliable quality

Our approach: Utilize delicious.com top tags

- Downside: Limited set of sufficiently tagged sites (approx. 17.000)

Alexa

video (25972), youtube (18248), videos (16876), entertainment (9200), media (7544), web2.0 (6743), social (4646), fun (4624), music (3378), community (3141)

Delicious top tags

Probing for CORS adoption

Looking for CORS is not straight forward

- No central policy file
- The CORS response headers may only be set for
  - specific origin domains and
  - certain target URLs

Our approach

- If a crossdomain.xml or clientaccesspolicy.xml file is present, set the origin header to one of the whitelisted domains
- If no or a wildcard policy was found, use an arbitrary origin

This is obviously incomplete

- No deep crawl of the sites
- Not obvious which domains to set in the origin header, if no further evidence is present

Data collection

Shallow crawl of the top 1.000.000 sites in the Alexa index

- Collect crossdomain.xml, clientaccesspolicy.xml files, and CORS headers
- If authentication forms are encountered, get the policy-files for the target domain

Resulting data

- 1.093.127 sites examined
  - Alexa top 1.000.000 plus subdomains which receive authentication info
- 5 days for the crawl using a distributed crawling infrastructure

Results

Results
Penetration

1.093.127 domains scanned

	Total	Percentage
Flash	82.052	8%
Silverlight	995	0,09%
Cors	215	0,02%

67.974 unique consumers

The actual number might be much higher, as we can't identify consumers of wildcard policies

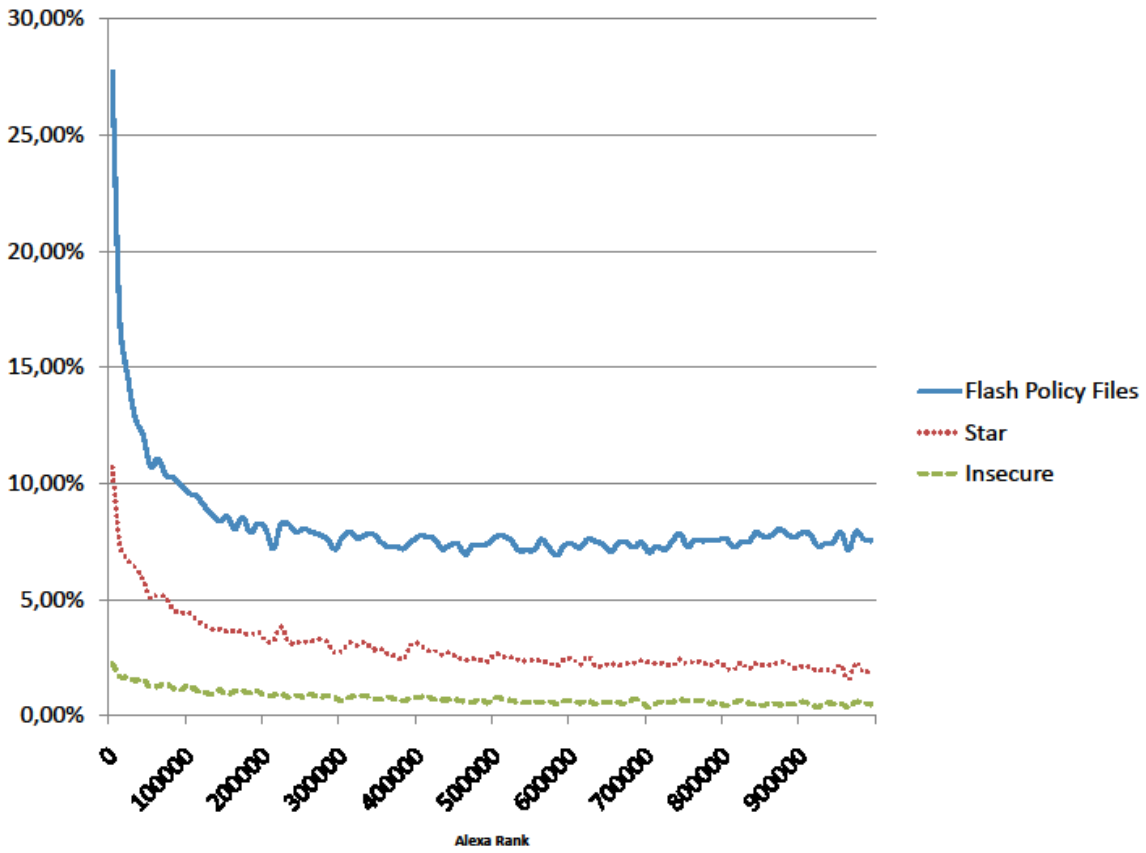
Results
Penetration / Security - Flash

Wildcard-policy

- 31.011 files (37,7% of all crossdomain.xml) resulting in 2,8% potentially insecure sites

When checking for authentication

- 15.060 sites (1,3% of all analyzed sites)



Collected crossdomain.xml files

Results
Penetration / Comparison to 2008

Grossman study in 2008

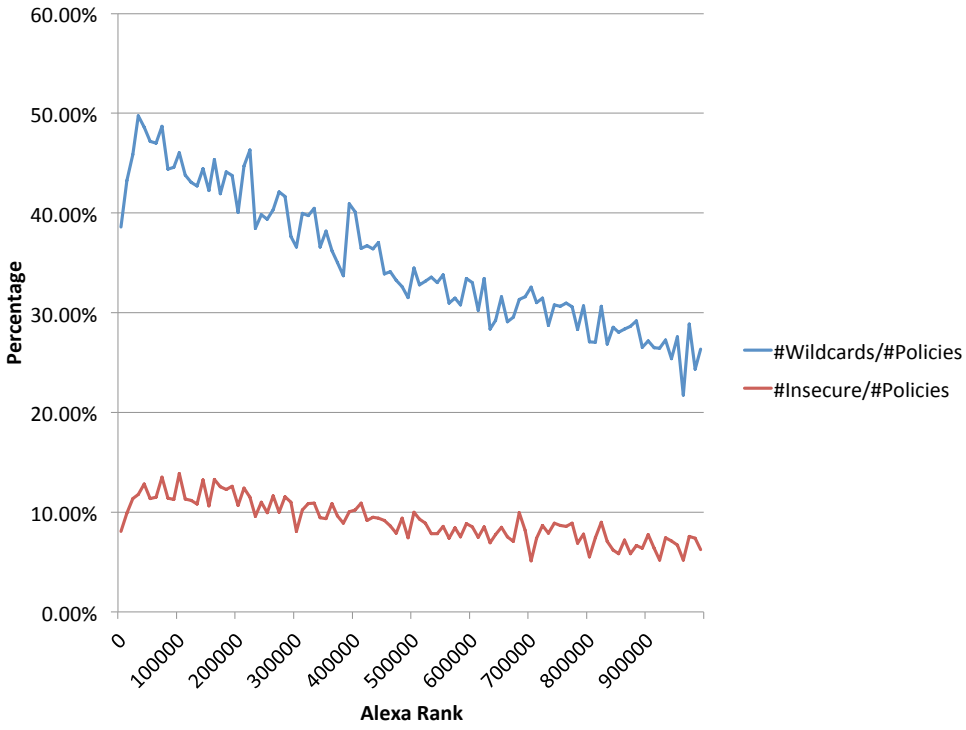
- Alexa Top 500 and Fortune 500
  - 28% providing a crossdomain.xml policy
  - 7% with a wildcard-policy

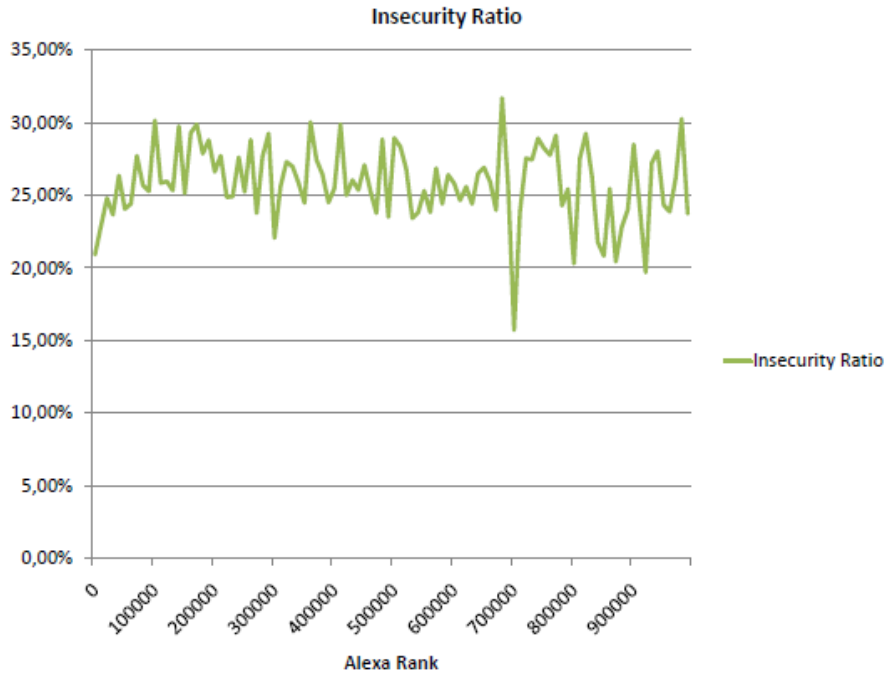
Our results (2011)

- Alexa top 1000
  - 48% provide a crossdomain.xml policy
  - 12% with a wildcard policy

→Indicator that adoption of the technology is increasing

Results
Relative security - Flash

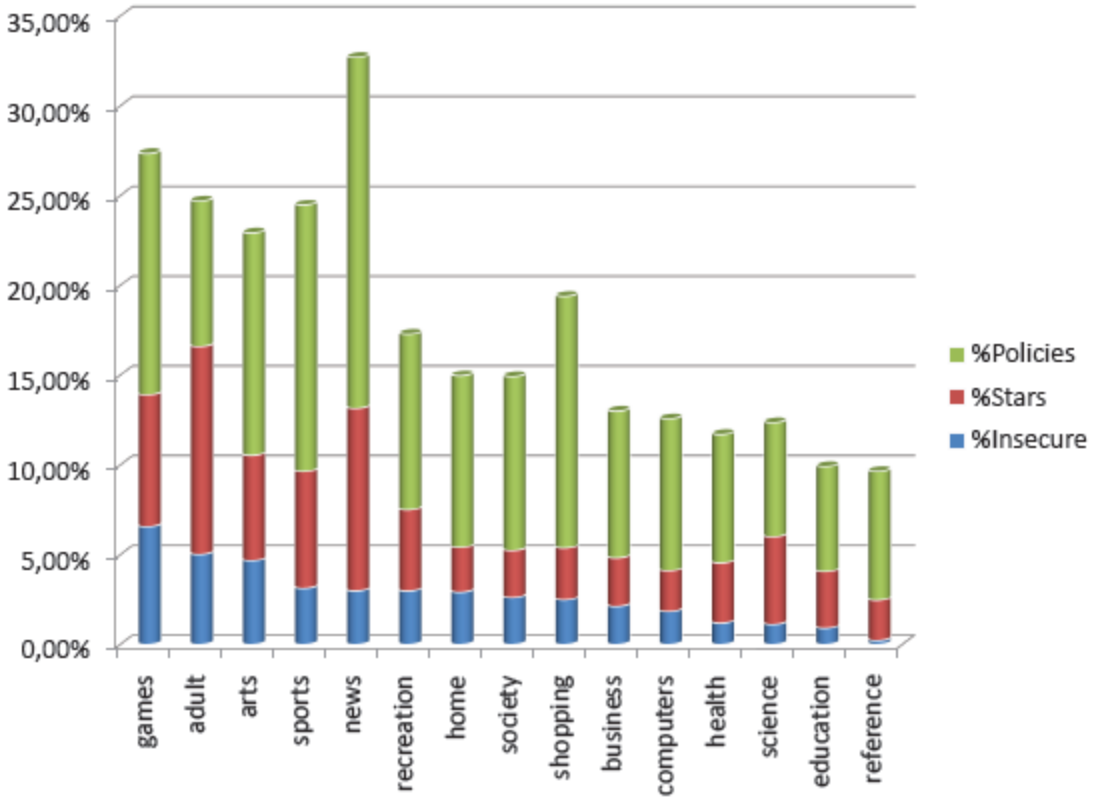




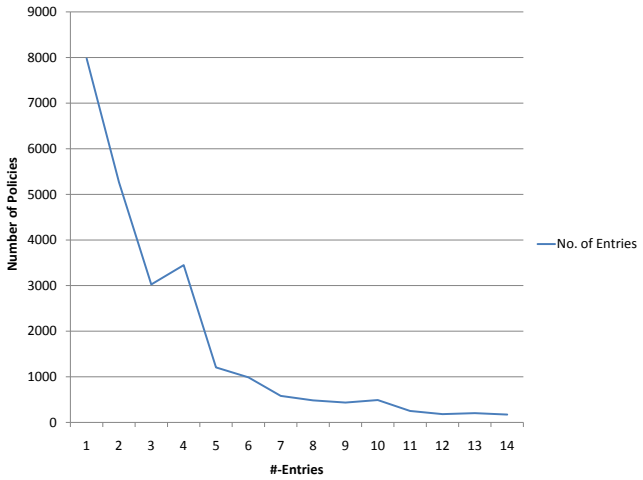
Conclusion: No apparent „long-tail“ effect.

Results
Security - Flash

Mapping policy files to the top-categories



Results
Transitivity of vulnerability



#	Domain	Alexa	#Ref.	Category	Subcategory
1	ning.com	249	1188	Society	Social
2	cooliris.com	2231	1076	Computers	Software
3	mochiads.com	23560	726	Business	Advertisement
4	brightcove.com	5125	718	Arts	Video
5	mochimedia.com	2822	405	Games	Video Games
6	2mdn.net	1892	394	Business	Advertisement
7	facebook.com	2	347	Society	Social
8	amazonaws.com	157	305	Computers	internet
9	weebly.com	462	267	Computers	internet
10	userplane.com	24671	255	Society	Social

Observations:

- The majority of sites whitelist 7 or less domains
- Only few domains are whitelisted by more than 300 policies

Conclusion

Conclusion

The number and percentage of insecure sites is considerable

This (in connection with many partially incorrect policies) suggests that the general knowledge on how to use this technique securely is still weak

- One third of all policies are wildcard policies
- Out of these 15.060 are insecure sites according to our criteria

No apparent signs for adoption of CORS

- However, as noted our methodology is insufficient for a full assessment

No long tail effect

What did we not examine?

- Flash subpolicies
- Consumer behavior

Thanks for listening

Martin Johns
SAP Research Karlsruhe
martin.johns@sap.com