An Empirical Study on the Security of Cross-Domain Policies in Rich Internet Applications

Georgios Kontaxis, Demetris Antoniades,
Iasonas Polakis and Evangelos P. Markatos
Institute of Computer Science
Foundation for Research and Technology, Hellas
{kondax,danton,polakis,markatos}@ics.forth.gr

ABSTRACT

Adobe Flash and Microsoft Silverlight are two widely adopted platforms for providing Rich Internet Applications (RIA) over the World Wide Web. The need for RIAs to retrieve content hosted on different domains, in order to enrich user experience, led to the use of cross-domain policies by content providers. Cross-domain policies define the list of RIA hosting domains that are allowed to retrieve content from the content provider's domain. Misinterpretation or misconfigurations of the policies may give the opportunity to malicious RIAs to access and handle users' private data.

In this paper we present an extensive study on the deployment and security issues of cross-domain policies in the web. Through the examination of a large set of popular and diverse (both geographically and content-wise) websites, we reveal that about 50% (more than 6.500 websites) of the websites that have adopted such policies are vulnerable to attacks. Furthermore, we find such policies in more than 50% of the top 500 websites, examined both globally and per-country. Additionally, we examine local sets of e-shopping websites and find that up to 83% implement weak policies. Interestingly, we observe that the less popular a website is, the higher the probability that it will have a weak policy. Compared to previous studies there is an obvious increasing trend in the adoption of RIA but, at the same time, a decreasing trend regarding secure implementations. Through a proof-of-concept attack implementation and a number of real-world examples, we highlight the security impacts of these policy misconfigurations.

1. INTRODUCTION

Adobe Flash [1] and Microsoft Silverlight [2] are two popular platforms for serving Rich Internet Applications [4] through the web. According to the latest statistics [1, 3], 85 out of the top 100 websites serve Flash objects (or Flash movies) to their visitors, 98% of which have the ability to render such objects (53.5% for Silverlight). Such applications combine a graphics library and a script-like API to add functionality, content or multimedia features to a website. These APIs enable the developer, among other things, to fetch content from remote web locations (e.g. XML data such as RSS feeds, or communicate with a database via a remote PHP page). Flash applications are packaged into objects, embedded in an HTML page and later downloaded and executed within the user's browser.

[1]http://www.adobe.com/flashplatform/
[2]http://www.silverlight.net

The ability of RIA platforms for remote content retrieval aims for service enrichment. However, this also reinstates the problem of cross-domain access, i.e., enabling a flash object hosted by `domainA` to access data residing in `domainB`. While defense mechanisms have been successfully deployed for traditional web technologies [7], these solutions cannot be applied in the case of RIA platforms. In lieu of these techniques, *cross-domain policies* were introduced to restrict Flash objects from accessing arbitrary network destinations. Cross-domain policies are XML files that reside on the server-side and allow per-domain access to Flash object requests on an opt-in basis. Due to the popularity of Adobe Flash for delivering rich content to web users, such policies are often deployed. However, those who implement them do not always fully understand them or are unaware of their security implications.

A website's cross-domain policy with weak security properties may permit Flash objects from arbitrary locations to access its contents. Consider an attack scenario where an attacker crafts a malicious Flash object and places it under `http://attacker.com/malicious.swf`. Once victims visit the malicious URL, the object is loaded in their web browsers and enables the attacker to place arbitrary HTTP requests towards the site with the weak security policy. Since the malicious Flash object is loaded inside the victim's browser, any HTTP requests it makes are placed in the network from the victim's computer. Furthermore, the victim's browser appends the victim's credentials (e.g. HTTP cookies) to those requests. So, for instance, an attacker can request `http://shopping.com/cart` and receive the contents of the victim's shopping cart, if she is logged in `shopping.com`. Even worse, an attacker can place a request purchasing an item, using the victim's credit card which is stored in her account.

This paper provides awareness on the use of Adobe Flash and Microsoft Silverlight cross-domain policies. We conduct an extensive study across popular websites and present our findings regarding the adoption of such policies and their security. We provide real-world examples of policy weaknesses that we came across during our study. We present a proof-of-concept implementation of an end-to-end attack platform that can exploit such policy vulnerabilities. Finally, we discuss possible approaches to mitigate the problem. The contributions of our work can be summarized as follows:

- We provide vulnerability awareness regarding the use of cross-domain Flash and Silverlight web policies.

```
1 <?xml version="1.0"?>
2 <!DOCTYPE cross-domain-policy SYSTEM "http://www.
3  macromedia.com/xml/dtds/cross-domain-policy.dtd">
4 <cross-domain-policy>
5   <allow-access-from domain="sub1.domainA.com"/>
6   <allow-access-from domain="domainC.com"/>
7 </cross-domain-policy>
```

Listing 1: A valid cross-domain policy

```
1   <allow-access-from domain="*.sub1.domainA.com"/>
2   <allow-access-from domain="*.domainC.com"/>
3   <allow-access-from domain="*"/>
```

Listing 2: Weak directives in a cross-domain policy

- We present an extensive study on the deployment and security of Flash and Silverlight cross-domain policy files. Our results reveal more than 6.500 websites with weak policies and, thus, vulnerable to attacks.
- We present proof-of-concept implementations of attack scenarios that target weak cross-domain policies for Flash and Silverlight enabled websites.

2. BACKGROUND

This section provides background information on the internals of the Adobe Flash cross-domain policy and describes in detail several types of weak implementations. Microsoft Silverlight employs an almost identical policy system.

2.1 Adobe Flash cross-domain policy

For security reasons, an Adobe Flash object rendered in a web browser, via the Adobe Flash Player plugin, is not allowed to access data that resides outside the web domain on which the object is hosted. The basis of domain comparison is the domain name, not an IP address. The two domains must be an exact match, so a comparison of `http://www.domainA.com` with `http://images.domainA.com`, or `https://www.domainA.com` will fail. The Flash player is responsible for enforcing this security measure. However, there are cases where access to data in a different domain is required. For that matter, the Flash player is able to process a white-list-oriented XML policy file, named `crossdomain.xml`, hosted on the remote domain from which data is requested.

In detail, once a Flash object, located at domainA, attempts to access remote domainB, the Flash player, which is responsible for rendering the object, contacts the host computer of domainB and requests the policy file. If the file is unavailable, the Flash player terminates the network access attempt. If a policy has been defined by the administrator of domainB, then the Flash player determines whether it explicitly allows access from the Flash object's own domain. If access has been authorized, the player places the network request in question, and returns the response to the calling Flash Object, running inside the user's web browser. Otherwise the network access attempt is terminated and data requests are never issued. Listing 1 shows an example of a well-defined cross-domain policy file from domainA. Only requests from sub1.domainA.com (line 4) and domainC.com (line 5) are allowed.

To extend the previous scenario, one can consider the case where domainA has several subdomains that host Flash objects requiring access to content hosted on domainB. To enable such access, the policy file under domainB must contain multiple "allow-access-from" directives, one for each subdomain. However, it is also possible to use the wildcard character to match, and therefore include, any subdomain of domainA or simply any domain. The latter practice may seem more convenient and with the least administrative overhead in terms of maintaining the list of cooperating domains. However, problems arise when a policy is not tailored to a specific domain's design and structure, allowing an adversary to take advantage of it.

2.2 Weak Policy Implementations

In this section we present the weak policy implementations and explain the potential attacks they can lead to.

Any sub-domain weakness. The use of a wildcard to match any subdomain might create a potential security weakness. If users are able to upload content to a specific subdomain, an attacker could upload a Flash object of his own and, by running arbitrary code, be able to place requests towards the target subdomain and receive the responses. Assuming the target subdomain requires some form of authentication in order to accept requests and not all users have such, a malicious user could lure an authenticated user (victim) to visit his malicious Flash object. Once the malicious Flash object is loaded in the victim's browser, any requests towards the target subdomain will carry the victim's credentials, contained in an HTTP cookie or a session variable, and will originate from the victim's IP address. Thus, the adversary is able to launch a Cross-Site Request Forgery (CSRF) attack as CSRF defense mechanisms [7] are rendered ineffective. Lines 1 and 2 from Listing 2 are examples of the sub-domain weakness.

Any domain weakness. In the case where multiple different domains require access to data on domainA, a policy will have to be defined with multiple "allow-access-from" directives, one for each distinct domain. However, a single directive can be defined, using the wildcard to match against domains. This means that not only the previous set of domains will gain access, but any domain, e.g. `http://attacker.com`. This is the most dangerous type of weakness because it allows the malicious Flash object to be hosted anywhere, from the attacker's home computer to an arbitrary free-hosting service on the web. While in the previous situation the attacker must be able to upload something to a white-listed subdomain of domainA, which may not always be possible, in this case she is not restricted in any way. Flickr, a major image and video hosting website, had such a weak policy implementation [3]. Line 3 from Listing 2 is an example of the any-domain weakness.

Throughout this paper, unless otherwise specified, we use the term "weak" to describe only policies with any-domain, unrestricted-access directives, as they are the most dangerous and always exploitable. This places a lower limit to the count of vulnerable sites as it does not include the exploitable any-subdomain policies, the exact number of which is unknown due to their case-specific nature.

3. DATA COLLECTION

To examine both the policy usage and security issues of cross-domain policies we created a number of diverse (both geographically and content-wise) lists of websites. Our first and larger list includes the 100K most popular websites, according to Alexa [4]. We believe this to represent a complete list of both popular and not so popular websites. Furthermore, we compiled a list with the websites of the Fortune 500 companies to fill our global view set. For our geographically diverse set, formed by popular sites in a more local scale, we used lists with the 500 most popular websites, again according to Alexa, in the U.S.A, Great Britain, Germany, France

[3]http://blog.monstuff.com/archives/000302.html

[4]http://www.alexa.com/

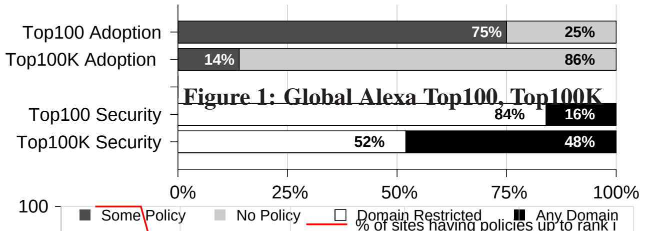

Figure 1: Global Alexa Top100, Top100K

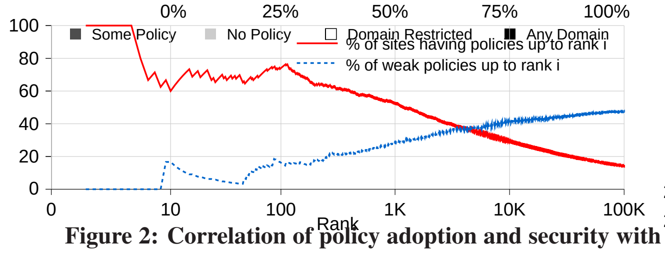

Figure 2: Correlation of policy adoption and security with site popularity rank.

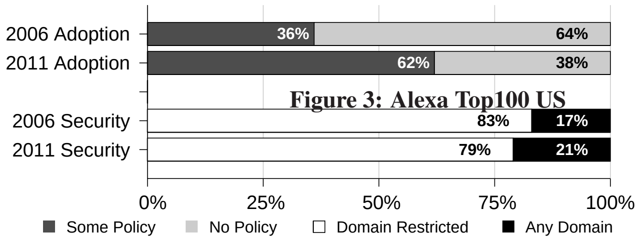

Figure 3: Alexa Top100 US

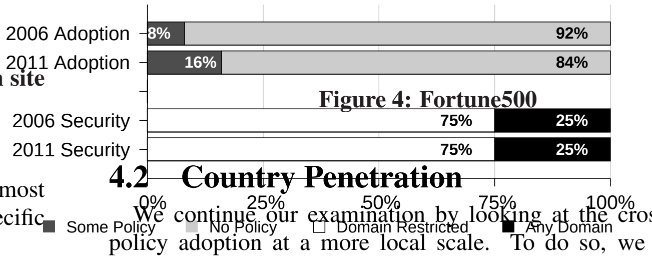

Figure 4: Fortune500

and Greece. Additionally, we have used a list with 500 of the most popular Greek e-shopping websites to form a set of content-specific sites in our country.

For each domain name in those lists we placed the following requests: `http[s]://domain/crossdomain.xml` and `http[s]://www.domain/crossdomain.xml` to download the policy file for Adobe Flash if one existed. A respective batch of requests was also sent to retrieve the policy file for Microsoft Silverlight. We subsequently examined those XML files and identified the domains that did return a valid policy. Furthermore, we evaluated the valid policies to automatically identify both types of weak implementations described in section 2.2. Our findings are presented in the following section. We focus on the findings from the `http://www.domain` requests as they represent the most common case and their differences with the results of the other requests are minor.

4. POLICY ADOPTION AND SECURITY

4.1 Global Penetration

We first examine the cross domain policy adoption at a global scale. Recall that the existence of a crossdomain policy file does not state whether a website supports RIA plugins or not, but whether a website allows RIA plugins to request and receive data from it. Figure 1 (top) presents our findings for the Top 100 and Top 100K websites, while Table 1 summarizes our findings for all sub-classes examined. We can see that the penetration ratio of domain policies diminishes as the total number of examined sites grows. For example, 75 out of the Top 100 sites serve a policy file, while this percentage drops to only 14% for the Top 100K sites. Furthermore, as we see in the bottom bars of Figure 1, security awareness also diminishes. That is, top popular sites seem to restrict their policies more carefully than not so popular sites – only 16% are weak cases for the Top 100 vs. almost 50% for the Top 100K sites.

To further understand the evolution of weak implementations as the site's rank moves away from the top, Figure 2 plots the cumulative percentage of weak policies in relation to the site's rank. We can observe a clear trend here; the less popular the site is, the higher the probability the website will have implemented a weak policy.

4.2 Country Penetration

We continue our examination by looking at the cross-domain policy adoption at a more local scale. To do so, we study the policy penetration for the U.S.A. and a number of major European countries. Table 2 summarizes our results. Countries like the U.S.A. with a high adoption rate (57%) of cross-domain policies have fewer weak policies (28% of deployed policies) than countries like Greece where the adoption rate is much lower (36%) but there are more weak policies (39%).This indicates that countries where such policies are more widely used are also more security-aware.

In a similar study, in 2006, Grossman [2] examined the cross-domain policy files for the top 100 sites in the U.S.A., according to Alexa, and the websites of the Fortune 500 list of companies. With a penetration of 36% and 8% respectively, Grossman found 6% and 2% to be wildcarded for any-domain. In January 2011, we re-examined the two lists. Figure 3 presents our findings along with the findings of the Grossman study. We see an increase in the penetration of cross-domain policies in 2011. More than 60% of the sites implement a cross-domain policy in the Alexa U.S.A top 100, an increase of almost 173% since 2006. Alarmingly, the number of weak policy deployments has also increased to 21%, exhibiting a growth of 124% since 2006. This indicates that, since 2006, more sites have adopted cross-domain policies, which is not surprising considering the capabilities of Flash technology. What is surprising though, is that the percentage of sites implementing weak policies has also increased and its increase is almost 70% of the growth rate of Flash technology adoption. In the case of the Fortune 500 sites, the penetration of such policies has doubled while the ratio of weak implementations has remained the same. The two groups of sites have a dice coefficient of 0.14, indicating two very different sets; the first one is more dynamic in terms of Flash technology adoption than the latter and appears to have more secure policies. At the same time, however, it also exhibits an increasing trend towards weak implementations.

4.3 Category Penetration

Besides examining local sites in each country, we examine specific site categories as provided by Alexa. In detail, we inspect the 500 most popular sites in a series of categories. Table 3 sum-

	Adobe Flash			Microsoft Silverlight		
	None	Some Policy		None	Some Policy	
Tier		Restricted	Unrestricted		Restricted	Unrestricted
Alexa Top 100	25%	84%	*16%*	97%	67%	*33%*
Alexa Top 1K	48%	71%	*29%*	98%	63%	*37%*
Alexa Top 10K	71%	58%	*42%*	99%	38%	*62%*
Alexa Top 100K	86%	52%	*48%*	99%	22%	*78%*
Fortune 500	84%	75%	*25%*	-	-	-

Table 1: Adobe Flash Cross-domain Policy and Microsoft Silverlight Client-access Policy adoption and security evaluation for the 100K most popular sites according to Alexa.

	Adobe Flash			Microsoft Silverlight		
	None	Some Policy		None	Some Policy	
Country		Restricted	Unrestricted		Restricted	Unrestricted
U.S.A.	43%	72%	*28%*	97%	69%	*31%*
Germany	55%	63%	*37%*	99%	50%	*50%*
Great Britain	58%	66%	*34%*	98%	89%	*11%*
France	63%	62%	*38%*	99%	75%	*25%*
Greece	64%	61%	*39%*	99%	50%	*50%*

Table 2: Adobe Flash Cross-domain Policy and Microsoft Silverlight Client-access Policy adoption and security evaluation for the 500 most popular sites in the U.S.A. and major European countries.

	Adobe Flash		
	None	Some Policy	
Category		Restricted	Unrestricted
Sports	54%	49%	*51%*
Health	85%	54%	*46%*
Society	75%	55%	*45%*
Adult	84%	60%	*40%*
Arts	52%	63%	*37%*
News	62%	64%	*36%*
Science	83%	66%	*34%*
Recreation	67%	68%	*32%*
Home	82%	69%	*31%*
Computers	68%	72%	*28%*
Shopping	68%	83%	*17%*

Table 3: Policy adoption and security evaluation for the top-500 sites in a series of content categories.

marizes our results. Cross-domain policies are adopted in 16% to 46% of the sites, depending on category. We can see that shopping sites are more aware of the security implications, with only 17% having unrestricted policies. All other categories have unrestricted policies in a larger percentage, ranging from 28% and up to 51%. We also examined the sites for Microsoft Silverlight, but omit the results due to the small adoption rate (less than 2% in all cases) and the lack of space.

Greek E-Shopping Sites: To examine both local and category specific websites we assembled a set with 500 popular Greek e-shopping sites by crawling the catalog of Skroutz [5], a popular Greek e-shopping search engine. Although, the adoption of cross-domain policy files is rather low (2.5%), the percentage of any-domain, unrestricted-access policies reaches an impressive 83.3%. Considering the nature of these sites (i.e., online shopping involving user accounts, personal and financial details), one can imagine the impact of an attack against their users.

4.4 Administrative Overhead

In the previous sections we have seen that an important number of cross-domain policies are vulnerable to attacks. In this section we examine whether a high administrative overhead is required in order to create and manage these policy files. To this extent, we measure the number of directives in each policy file found in the top 100K sites in the global set of Alexa. Without the use of wildcards for sub-domain or any-domain white-listing, each domain to be allowed access must be declared in a directive of its own. Figure 5 plots the Cumulative Distribution Function (CDF) of the length (number of directives) of the policy files. One may notice that 10% of those policies has more than 10 directives, 5% has more than 20 and there are sites that span over 100 or even 400 directives. Such long policies can prove hard to effectively maintain and could contain weaknesses that may go unnoticed.

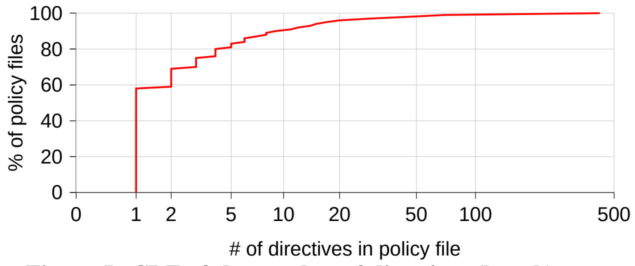

Figure 5: CDF of the number of directives (length) per cross-domain policy for top 100K global sites.

5. ATTACKS

In the previous section we showed that the percentages of cross-domain adoption and weaknesses are high among the web sites. In this section we present two attack scenarios that leverage a weak cross-domain policy and describe how they can be used for malicious purposes. Furthermore, we present a proof-of-concept implementation for exploiting such weaknesses and provide the technical details. Each scenario has been tested with at least one real case of a weak policy found during our experiments in section 4.

[5] http://www.skroutz.gr

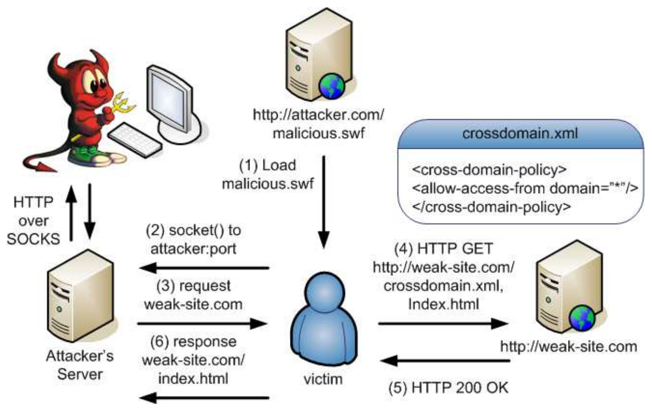

Figure 6: Attack Proxy Design

```
    <allow-access-from domain="attacker.com"
      to-ports="8080"/>
```

Listing 3: Cross-domain Policy enabling socket connections from the victim

5.1 Setting up the Attack

Prior to discussing the actual attack we first describe the steps needed by the attacker in order to use a Flash object as an attack proxy. The whole scenario is depicted in Figure 6. Initially, the attacker tricks her victim into loading `http://attacker.com/malicious.swf` which is a carefully crafted Adobe Flash object. This object can be masked as a movie, animation or game and thereby conceal its true purpose, or can be completely invisible (e.g., zero dimensions on an HTML page) and accessed as part of a perfectly legitimate site via an iframe in the form of an advertisement. As soon as `malicious.swf` loads, it can execute arbitrary code and open a standard network socket with a malicious server running on the attacker's home computer. Socket connections to network destinations are governed by the same rules as cross-domain site access but this is a server the attacker controls. For that matter, the Flash player inside the victim's browser issues a cross-domain access request towards the malicious server, which in return provides the necessary cross-domain policy (Listing 3).

As soon as the victim's Flash player examines the policy, it establishes a network connection with the attacker's server. Through that connection, `malicious.swf` receives control messages describing the URLs for which it will issue HTTP GET requests (steps 2-3 in Figure 6). Since, malicious.swf is executed inside the victim's web browser, the victim is the one who places those HTTP requests on the network (step 4). As soon as the request is served back to `malicious.swf` (step 5), it forwards the received content through the network socket back to the attacker's control server (step 6). At this point the attacker receives the response to an HTTP request that the victim has placed on his behalf.

The set of URLs the attacker is able to instruct the victim to fetch on his behalf, is dictated by the set of sites on the web with weak cross-domain policies. While this may seem as a serious restriction on the attacker's request-issuing capabilities, in section 4 we found more than 6.5K sites containing weak policies. Furthermore, in special categories such as Greek e-shopping sites, more than 80% of those that have a deployed policy are vulnerable to this attack.

Steps 4 and 5 can differ, depending on the actual attack carried out. In the next sections we describe two possible attack cases. Both cases were implemented and tested in real world scenarios, using a proof-of-concept deployment of the Figure 6 components. Section 5.4 describes real-world examples found during our study.

5.2 Abusing cookies and credentials

In the first attack scenario, the attacker abuses the victim's web credentials when accessing sites with weak policies. By instructing `malicious.swf` to issue an HTTP request towards `http://site.requiring.login.com`, the victim's browser will in fact send along any cookies it has, if the victim is already authenticated for that site (e.g. victim's web mail, e-shopping account). So, for instance, the attacker could instruct the victim to fetch `https://mail.webmail.com` and receive a response with the victim's web mailbox contents or `https://shopping-site.com/cart?action=add&item=foo` to add an item to the basket. Some sites employ Cross-site Request Forgery (CSRF) tokens which are, in essence, random nonces returned to the (legitimate) user upon logging in to a site. Thus, any attacker cannot issue a direct request without knowing the nonce. However, as with login cookies, CSRF tokens are also appended by the victim's browser in the upstream path from `malicious.swf` to the destination site or can be read from the HTTP headers included in the site's responses. Let it be noted that such requests are not recorded in the browsing history and don't leave other evidence, such as cache files, on the victim's computer.

5.3 Laundering web attacks

An attacker can launder various types of attacks, such as XSS and SQL injections, by issuing them through the victim's browser. The attacker's target site must have a weak policy so that the victim can issue the malicious requests even if the attacker does not require the receipt of HTTP responses. Furthermore, sites with weak security policies are also likely to be vulnerable to other attacks.

On a different note, an attacker could frame the victim by associating him with frowned-upon activity. For instance, he could abuse the victim's credentials for posting offending messages on a web discussion group, with a weak policy, that the victim is a member of or place network requests towards controversial content through the victim's browser.

5.4 Examples

Here we discuss two real-world examples of weakly implemented cross-domain policies we came across during our experiments.

Global Telecommunications Company. We have identified the case of a major mobile telecommunications carrier's site in Spain, in which country it holds more than 30% of the market share. The carrier offers its subscribers the ability to manage their account online, via a portal located under the carrier's official localized domain. However, the domain implements a weak policy, allowing access from any Flash object located under any foreign domain. As a result, using the attack platform presented above, one could gain unauthorized access to the information of the portal's users. This portal allows subscribers to check, among other things, their call log, billing and consumption information.

Major E-Shopping Site. We have also identified the case of a major e-shopping site located in Greece, which also holds more than 30% of the market share. The site's weak cross-domain policy gives the attacker unrestricted access to the victim's account, including the shopping cart and personal information. By exploiting such access attackers can review the victim's history of purchases and extract sensitive information such as the user's name, address and phone number. Moreover, the attacker can also place new orders or add/remove items while the victim is placing his own order.

6. RELATED WORK

Balduzzi et al. [6] conducted a large-scale study to assess the extent of websites that are vulnerable to HTTP Parameter Pollution (HPP) attacks. Their vulnerability assessment study determined how 5,000 popular websites behaved upon receiving duplicate HTTP parameters. Their results showed that 30% of the websites contained vulnerable parameters. Also, 14% of the sites were susceptible to HPP attacks that allowed the modification of the web application in a way that enabled a series of client-side attacks. As the authors report, these vulnerabilities are due to developers not being aware of the problem or not taking it seriously. This also seems to be the case with the flash cross-domain policy vulnerability we explored in our study, as many developers are not aware of the security implications.

Jackson et al. [9] focus on ways to deal with DNS rebinding attacks. In such an attack, the same-origin policy is subverted by manipulating the IP addresses the attacker's host names are bound to. In a DNS rebinding attack against Flash cross-domain policy checking, the attacker responds to the user's initial DNS query with the IP address of the attacker's server, using a short TTL. That way, `site.com` is temporarily bounded to the IP address of the attacker's server so that a spoofed `crossdomain.xml` is provided from there and, as the TTL will expire, a subsequent DNS request for `site.com` will bind to the IP address of the actual server, therefore providing access to the contents of the site.

In [8, 5] the authors expose the use of transparent proxies in networks as a mechanism to break the same-origin restriction imposed by the Adobe Flash player. They demonstrate that such proxy caches may be poisoned as a direct result of their confusion regarding the handling of cases, where the HTTP `host` header and the destination IP address of the intercepted connection do not match.

7. DISCUSSION

In this section, we discuss possible mitigation techniques for the security issues arising from the weak cross-domain policies highlighted in the previous sections.

A first step in mitigating such weaknesses, is to educate site developers on the security aspects of weak cross-domain access policies for RIA applications. Beside the common sense of not having a policy when it is not needed, site operators should also be motivated, though awareness, to correctly implement policies when needed. Nonetheless, while policy files work well when correctly implemented, there are cases where that is hard to do or not possible at all. For those cases, we describe ways to avoid creating security vulnerabilities.

Domain Separation. Consider a site `www.domain.com` which wants to allow access to its HTTP REST API from anywhere `(*)`. A solution is to move the API to `api.domain.com` (with a `domain="*"` policy) and restrict or completely remove the policy file for the main domain. This way, arbitrary user Flash applications can place requests towards the API from anywhere, but cannot access the site itself. As a result, the site is isolated from CSRF requests through Flash objects and the API-hosting domain can be more easily monitored and managed. This is the solution implemented by Flickr, which moved `flickr.com/api` with an allow-from-anywhere policy to `api.flickr.com` and removed the policy file entirely from the www-prefixed domain. This can also be applied in cases when user-generated content is hosted. Such files can be placed on a separate domain that is not included in the policy file.

Embedded Sources. If a site wants to distribute some of its content to a large (or unknown) number of other sites, it can provide embeddable content hosted on its own domain. This way, people can pull specific pieces of content from the site without being allowed unrestricted access to its pages. This technique is followed by YouTube, which doesn't allow access to `youtube.com` from arbitrary destinations but at the same time allows anyone to include videos in their own sites.

8. CONCLUSION

In this paper we conducted an extensive study regarding the adoption and implementation of the cross-domain policy for RIA object access. When the policy is not specifically crafted to match a website's design and structure, an adversary can leverage the weak implementation to deploy various attacks targeting the site's users. We found that 14% of the top 100K sites have adopted cross-domain policy files, and almost 50% of them have a weak implementation. Furthermore, the top 500 sites we examined from a country-level point of view present an adoption rate of 50%, with weak implementation percentages ranging from 28% to 39%. When examining top e-shopping websites in various countries, we found up to 83% of them implementing a weak policy. We also presented real world examples of weak policy implementations and attacks that can be carried out against them. Finally, our proof-of-concept attack implementation highlighted the security implications of misconfigured cross-domain access policies.

Acknowledgments

The research leading to these results has received funding from the European Union Seventh Framework Programme (FP7/2007-2013) under grant agreement 257007. This work is supported in part by Herakeitos II PhD Scholarship in the area of "Internet traffic classification". We would like to thank the anonymous reviewers for their valuable comments. The authors are also with the University of Crete.

9. REFERENCES

[1] Adobe Flash Platform Pervasiveness. `http://www.adobe.com/flashplatform/statistics/`.

[2] Jeremiah Grossman - crossdomain.xml statistics. `http://jeremiahgrossman.blogspot.com/2006/10/crossdomainxml-statistics.html`.

[3] Rich Internet Application Market Share. `http://www.statowl.com/custom_ria_market_penetration.php`.

[4] wikipedia.org - Rich Internet Application. `http://en.wikipedia.org/wiki/Rich_Internet_application`.

[5] R. Auger. Socket capable browser plugins result in transparent proxy abuse, 2010. `http://www.thesecuritypractice.com/the_security_practice/TransparentProxyAbuse.pdf`.

[6] M. Balduzzi, C. T. Gimenez, D. Balzarotti, and E. Kirda. Automated discovery of parameter pollution vulnerabilities in web applications. In *Proceedings of the 18th Network and Distributed System Security Symposium, 2011*.

[7] A. Barth, C. Jackson, and J. C. Mitchell. Robust defenses for cross-site request forgery. In *Proceedings of the 15th ACM Conference on Computer and Communications Security, 2008*.

[8] L.-S. Huang, E. Y. Chen, A. Barth, E. Rescorla, and C. Jackson. Transparent proxies: Threat or menace? `http://www.adambarth.com/experimental/websocket.pdf`.

[9] C. Jackson, A. Barth, A. Bortz, W. Shao, and D. Boneh. Protecting browsers from dns rebinding attacks. In *Proceedings of 14th ACM conference on Computer and Communications Security, 2008*.