Exploiting the Intranet With a Webpage
Is JavaScript the New Shellcode?

HITB SecConf 2007
05. September 2007

Martin Johns
Me, myself and I

Martin Johns

- johns at informatik.uni-hamburg.de
- Security researcher at the University of Hamburg
- Member of the secologic project
  - Research project carried out by SAP, Commerzbank, Eurosec and the University of Hamburg
  - Sponsored by the German Ministry of Technology (BMWi)
  - Goal: Improving software security
  - Visit us at http://www.secologic.org
JavaScript Malware?

Term coined by Jeremiah Grossman

Describes a class of browser-based-attacks that target intranet resources

A lot of ongoing research since late 2006

All attacks covered in this talk are “legal” actions according to the HTTP, HTML and JavaScript specs/drafts/RFCs

No browser bugs required
Agenda

- The Basics
- Intranet Attacks
- DNS Rebinding
- Client Side Protection
- Conclusion
Agenda

- The Basics
- Intranet Attacks
- DNS Rebinding
- Client Side Protection
- Conclusion
The Same Origin Policy (SOP)

Designed to prevent cross-domain read/write access

- Applies to JavaScript
- Affects cookie-access, cross-document interaction and networking communication
- The SOP is satisfied iff
  - the protocol,
  - the domain and
  - the port
  of two elements match
- Java and Flash have similar policies
So, the SOP provides a nice sandbox:

1. No direct access to the local file system  
   (Protocol-rule)
2. No direct access to other hosts  
   (Domain-rule)
3. No direct access to other applications on the same host  
   (Port- and protocol-rule)
A loophole in the SOP

JavaScript can initiate network communication through dynamic inclusion of elements in the DOM-tree:

- The script includes a HTML element in the page which references a network resource
  - IMG, STYLE, SCRIPT, IFRAME
- By this inclusion of such an element the browser creates an HTTP request

The targets of such requests are not restricted by the SOP

- This in fact enables indirect cross-domain communication
- Outgoing data:
  - URL parameters
- Incoming data:
  - Side effects of the inclusion process
Circumventing the SOP (I)

Remember: The SOP prevents cross-domain data-retrieval

- Does it?

The basic reconnaissance attack (BRA)

- Question: Does the element with URL U exist?

Method:

- Construct URL U pointing to the target of the examination
- Start a timeout-event t
- Include a suiting network aware element using U
- Use JavaScript’s eventhandler-framework to determine the result:
  - The timeout t occurs ⇒ The target does not exist
  - onload() event ⇒ The target exists
  - onerror() event ⇒ (specific result depends on the element and target - stay tuned)
Circumventing the SOP (II)

Remember: The SOP prevents cross-domain write access

- Does it?

Cross Site Request Forgery

- CSRF aka XSRF
- aka Session Riding
- aka Sea Surf

Implicit authentication

- Auth. mechanisms that are executed by the browser without user interaction
  - Cookies, HTTP Auth, NTLM, client-side SSL

CSRF exploits implicit authentication mechanisms

- Creation of hidden, state-changing cross-domain requests
- These requests are automatically outfitted with the user’s credentials
Cross Site Request Forgery (CSRF)

www.bank.com

Cookie: auth_ok
Cross Site Request Forgery (CSRF)

www.bank.com

GET transfer.cgi?am=10000&an=3422421

Cookie: auth_ok

www.attacker.org
Agenda

- The Basics
- Intranet Attacks
- DNS Rebinding
- Client Side Protection
- Conclusion
IP based authentication

Firewall

Intranet webserver
Firewall == implicit mean of authentication
⇒ Susceptible to CSRF
Putting it all together

By looking at a webpage we allow JavaScript-execution within the intranet...

As we have just discussed, JavaScript can do

- Reconnaissance (BRA)
- Exploiting (CSRF)
Intranet exploration with the BRA

<img src="http://10.10.10.10">

onload()/onerror()/timeout()-events,

Host / URL / element exists

Intranet Server (10.10.10.10)

Firewall

Malicious host
But where to start?

“My hosts are NATed and use obscure private IPs”

Java to the rescue:

- Java applets provide low level sockets
- The target of these sockets is restricted by the SOP
- This does not matter as we are interested in the origin of the connection (the local IP)
- On modern browsers even more convenient with LiveConnect:

```javascript
function natIP() {
    var w = window.location;
    var host = w.host;
    var port = w.port || 80;
    var Socket = (new java.net.Socket(host, port))
        .getLocalAddress().getHostAddress();
    return Socket;
}
```
Ping sweep / http-server discovery:
- Iterate through the subnet using the BRA
- `<IFRAME src="http://10.10.10.1"`,
  `<IFRAME src="http://10.10.10.2"`,
  `<IFRAME src="http://10.10.10.3"`,
  ...
  ◆ Timeout-event: Host does not exist
  ◆ OnLoad-event: Host runs a webserver
  ◆ OnError-event: Host exists but the port is closed (RST package)
- Varying the port might locate https or development servers
Intranet exploration with the BRA (III)

Server/application discovery/fingerprinting

- Known “special” DNS names
  - `<IFRAME src="http://fritz.box">` (home router)

- Known image-URLs
  - `<IMG src="http://10.10.10.10/icons/c.gif">` (Apache)

- Web page fingerprinting based on JavaScript errors
  - `<SCRIPT src="http://10.10.10.10/index.php">

```
<script>
function err(msg, url, code) {
    if ((msg == "missing } in XML expression") && (code == 1)) {
        // Wordpress
    } else if ((msg == "syntax error") && (code == 3)) {
        // Squirrelmail
    } else
        // unknown
}

window.onerror = err;
</script>
```
HTTP-authentication

- If the scanned server is protected by HTTP-auth the browsers displays a login-dialogue
- This should at least startle the browser’s user

Avoiding HTTP-authentication pop-ups (Stefan Esser)

- The trick is to cause the server to drop the request before it is processed
- This can be achieved by malformed URLs
- Incomplete entities:
  - http://host/
- Excessively long URLs
  - http://host/AAA ... AAA
- Breaks fingerprinting
Exploiting the intranet

The attacker is able to:
- locate intranet hosts and
- fingerprint applications/routers/devices

Several promising points for CSRF attacks:
- Unchanged default passwords on appliances
  - “Drive by Pharming”
- Unpatched servers
  - The old and almost forgotten IIS in the basement
- Outdated intranet applications
  - Wordpress 2.0 for internal communication
Some limitations...

Timing

- Working with timeout-events takes... Time
- Using parallelization can speed the process up
- But various restrictions on connections limits exit
  - Windows XP SP2 and later

Port restrictions

- Most browsers only allow HTTP and high-number ports
Further attacks

- Fingerprinting / attacking non http-protocols via multi-part HTML forms
- Attacks that don’t require JavaScript

Convenient attack tools exist

- E.g., Browser Exploitation Framework (BeEF)
  - One line XSS-payload
- Backframe
Agenda

- The Basics
- Intranet Attacks
- DNS Rebinding
- Client Side Protection
- Conclusion
Originally invented 1996 to subvert Java applets

General technique

- The attacker dynamically created DNS entries assigned to local IP addresses
- This way the SOP can be circumvented
DNS spoofing / anti-DNS-pinning / DNS rebinding

http://attacker.org/index.html

http://attacker.org/foo.html

Contents of the server

10.10.10.10 = attacker.org

10.10.10.10 = attacker.org

⇒ The SOP is satisfied

attacker.org == attacker.org

200.200.200.200 = attacker.org
DNS pinning

Counter Measure

- Keep the DNS binding for the lifet ime of the browser session
- Breaks, e.g., dynamic DNS, certain load balancing techniques
- Further problem: nowadays our browser sessions are quite long
- Violates RFC 2616

“Anti-DNS-pinning”

- In Firefox, IE and Opera not fully implemented
  - Issue open since approx. one year
  - Unknown if and how it will be fixed
- Methods to cause the browser to drop the pinning
  - Close the original port on attacker.org
  - Request a resource on a closed port on attacker.org
- Browsers take different amount of time to drop the pin
  - IE is the fastest
Countering DNS rebinding attacks

Host header:
- All requests created through JavaScript are within the domain “attacker.org”
  - Dictated by the SOP
  ⇒ Host-header == “attacker.org”
  ⇒ Web content for other virtual hosts is unreachable for JavaScript

“Anti-Anti-Anti-DNS pinning”
- It used to be possible to forge the host header:
  - with XMLHttprequest
  - with Flash
- Both vectors are fixed and work only on outdated browsers
DNS rebinding with LiveConnect

Remember LiveConnect?

- JavaScript can dynamically create Java objects

```javascript
function natIP() {
    var w = window.location;
    var host = w.host;
    var port = w.port || 80;
    var Socket = (new java.net.Socket(host, port)).getLocalAddress().getHostAddress();
    return Socket;
}
```

- TCP sockets

- What happens if such an object is created AFTER the DNS entry has changed?
  - The Java-runtime has its own pinning table
  - Java only sees the changed mapping

⇒ TCP sockets can be used in the attack
DNS rebinding with Flash

Flash 9 also has TCP sockets

- ...what Flash does not provide (yet) is DNS Pinning
  - Flash also outfits the attacker with TCP socket-connection to intranet hosts
- Rebinding attacks are quite fast, determined by the entries TTL
DNS rebinding and TCP sockets

TCP sockets enable the attacker to recreate HTTP
- This implies creation of arbitrary HTTP headers
  ⇒ The host-header is useless, again.

Recreation of HTTP, part II
- 401 responses don’t cause pop-ups anymore
  ⇒ password brute-forcing

Further capabilities through TCP sockets:
- Other protocols
- Binary data
- Basically, everything

Check out Billy K. Rios talk tomorrow!!!!
Agenda

- The Basics
- Intranet Attacks
- DNS Rebinding
- Client Side Protection
- Conclusion
Why at the client side?

- The server receives correct http requests from valid intranet hosts

- Server side indicators:
  - External referrer header
  - Mismatching host header, in the case of DNS-based attacks

- Both indicators can be evaded
  - Referrer headers can be deterministically suppressed
  - Host headers can be spoofed
Concept: Segmentation based on the origin of webpages

- Local pages:
  - Retrieved from intranet locations
- Remote pages:
  - Retrieved from outside locations
- Classification is based on IP-address-ranges

Only local pages (== resources that have a local origin) are allowed to create requests to intranet locations
LocalRodeo (II)

Intranet Server (10.10.10.10)

Webpage is tagged as “local”

Malicious host

Firewall
Webpage is tagged as “remote”
DNS re-binding

- LocalRodeo classifies into local/remote

Rebinding attack == switch from “remote” to “local”

- Rather easy to spot and stop
Limitation:

- A local/remote classification is not in all cases possible
- Example:
  - DMZ resources that grant different access rights based on source IP address
Advantages
- Good protection against all specified attacks
- Easy configuration for "simple" networks

Disadvantages
- Complicated configuration for "sophisticated" networks
  - How should different network segments be treated?
  - E.g., protecting against attacks from the inside of the same company
- No protection against Java and Flash based attacks
Implementation

Extension for the Firefox browser

Get it: http://databasement.net/labs/localrodeo

Still in beta
  ■ if you would like to contribute, go ahead, it’s open source :-)

The NoScript-developers have announced to include the proposed techniques in a future release
Stanford researchers also investigated DNS rebinding

- They proposed / implemented a couple excellent countermeasures
- Firewall
  - Monitors DNS traffic
  - Denies external hostnames to resolve to internal IP addresses
- Check it out: [http://crypto.stanford.edu/dns/](http://crypto.stanford.edu/dns/)

Keep in mind:
- Does only protect against rebinding attacks
- BRA and CSRF are still possible
Agenda

- The Basics
- Intranet Attacks
- DNS Rebinding
- Client Side Protection
- Conclusion
What did I not tell you?

Attacks that do not rely on JavaScript
- CSS based ping-sweeping
- External timing-based reconnaissance attacks

Privacy attacks
- Browser History-Disclosure
- Local machine profiling
- Timing attacks

Using the browser as an attack proxy
- Click-Fraud
- Server scanning (Nikto)
- Helping worm propagation (puppetnets)

...check out the bibliography when you have time
Summary

A rogue webpage can:

- Obtain the (internal) IP address of the hosting web browser
  - Using Java or guessing based on other evidence (existing URLs)
- Portscan the LAN to locate intranet http servers
  - Using the BRA while suppressing HTTP auth dialogues
- Fingerprint these http servers using well known URLs
- (sometimes) exploiting them via CSRF or
  - In case the fingerprinting found a known and vulnerable application
- Access the servers content and leak it to the outside by breaking DNS pinning
  - And use sockets for more sophisticated attacks

Think: XSS payload

- Remember the Samy worm
Conclusion / Lessons learned / Outlook

**Intranet Servers**
- Do not solely rely on the firewall to protect sensitive intranet services
- Apply additional explicit authentication
- Do not leave intranet servers unpatched

**Client Side**
- Disable Flash!
- Disable Java
- Use NoScript (and/or LocalRodeo)
Conclusion

- The SOP is insufficient
- ... and so is DNS pinning
- Relying on DNS for security purpose is not a good idea, anyway
  - DNS is not controlled by the web application
- Soundly solving the problem exclusively on the client-side (i.e., in the browser) is not feasible
  - At least as long certain cross-domain requests are permitted
- Future work should investigate server based policies concerning cross-domain interaction
Thank you for your attention

Questions?

Comments?
The bibliography


The bibliography


The bibliography


The bibliography


