Protecting Java EE Web Apps with Secure HTTP Headers

JavaOne 2012

About

- Frank Kim
  - Consultant, ThinkSec
  - Author, SANS Secure Coding in Java
  - SANS Application Security Curriculum Lead
- Shout out
  - Thanks to Jason Lam who co-authored these slides

JavaOne Rock Star

Outline

➡ XSS
- Session Hijacking
- Clickjacking
- Wrap Up

Cross-Site Scripting (XSS)

- Occurs when unvalidated data is rendered in the browser
- Types of XSS
  - Reflected
  - Stored
  - Document Object Model (DOM) based

XSS Demo

HttpOnly Flag

- Ensures that the Cookie cannot be accessed via client side scripts (e.g. JavaScript)
  - Set by default for the JSESSIONID in Tomcat 7
- Configure in web.xml as of Servlet 3.0

```
<session-config>
  <cookie-config>
    <http-only>true</http-only>
  </cookie-config>
</session-config>
```

- Programmatically

```
String cookie = "mycookie=test; Secure; HttpOnly";
response.addHeader("Set-Cookie", cookie);
```

X-XSS-Protection

- Blocks common reflected XSS
  - Enabled by default in IE, Safari, Chrome
  - Not supported by Firefox
    - Bug 528661 open to address
- X-XSS-Protection: 1
  - Browser modifies the response to block XSS
- X-XSS-Protection: 0
  - Disables the XSS filter
- X-XSS-Protection: 1; mode=block
  - Prevents rendering of the page entirely

Java Code

- X-XSS-Protection: 1

```
response.addHeader("X-XSS-Protection", "1");
```

- X-XSS-Protection: 0

```
response.addHeader("X-XSS-Protection", "0");
```

- X-XSS-Protection: 1; mode=block

```
response.addHeader("X-XSS-Protection", "1; mode=block");
```

X-XSS-Protection Demo

Content Security Policy

- Helps mitigate reflected XSS
  - Originally developed by Mozilla
  - Currently a W3C draft
    - https://dvcs.w3.org/hg/content-security-policy/raw-file/tip/csp-specification.dev.html
- Supported browsers
  - Firefox and IE 10 using X-Content-Security-Policy
  - Chrome and Safari using X-WebKit-CSP header

CSP Requirements

- No inline scripts
  - Can't put code in `<script>` blocks
  - Can't do inline event handlers like

    `<a onclick="javascript">`
- No inline styles
  - Can't write styles inline

CSP Directives

- default-src
- script-src
- object-src
- style-src
- img-src
- media-src
- frame-src
- font-src
- connect-src

CSP Examples

1) Only load resources from the same origin

```
X-Content-Security-Policy: default-src 'self'
```

2) Example from mikewest.org

```
x-content-security-policy:
   default-src 'none';
   style-src https://mikewestdotorg.hasacdn.net;
   frame-src
      https://www.youtube.com
      http://www.slideshare.net;
   script-src
      https://mikewestdotorg.hasacdn.net
      https://ssl.google-analytics.com;
   img-src 'self'
      https://mikewestdotorg.hasacdn.net
      https://ssl.google-analytics.com data:;
   font-src https://mikewestdotorg.hasacdn.net
```

Report Only

- Facebook Example

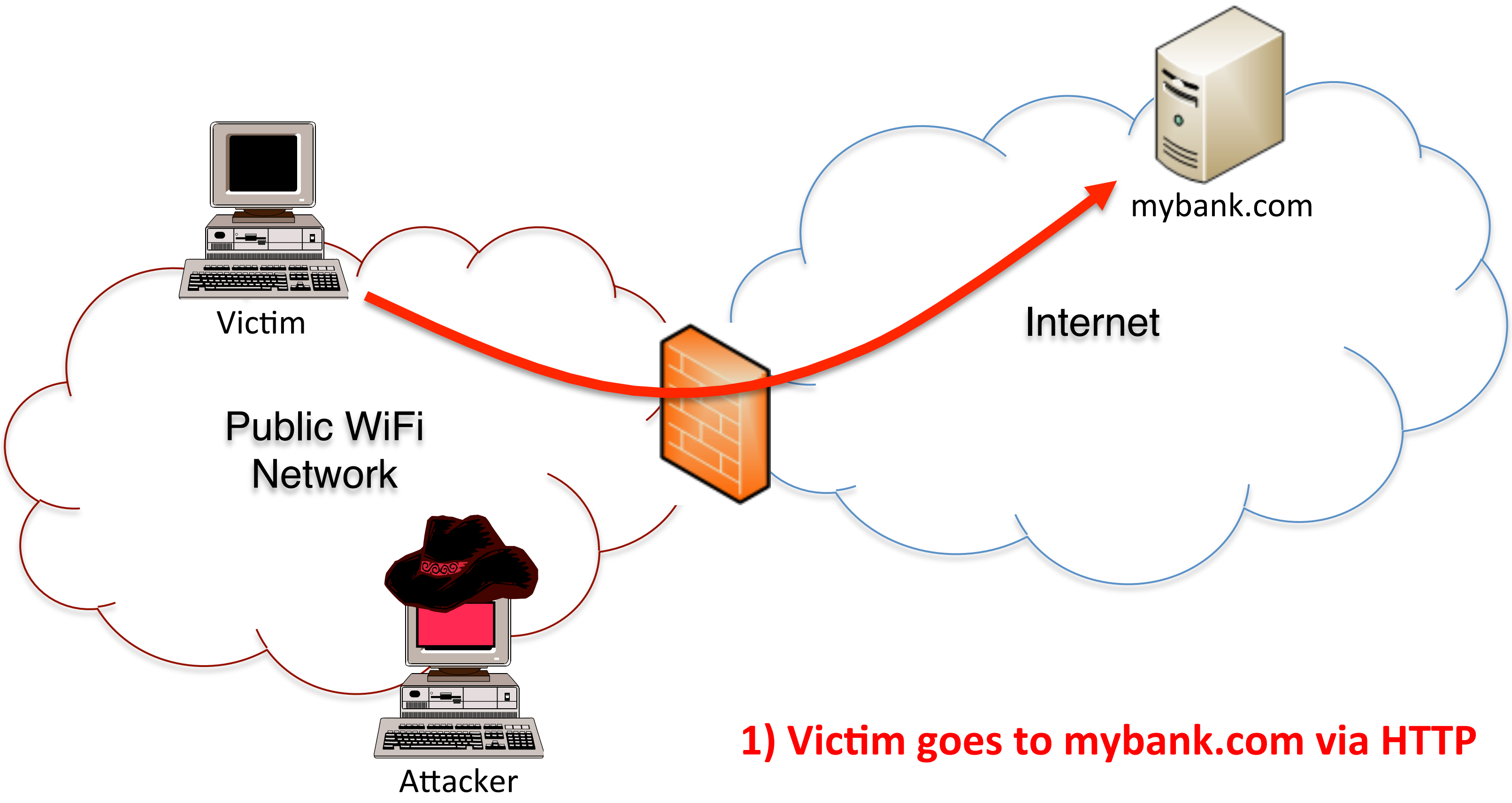
```
x-content-security-policy-report-only:
   allow *;
   script-src https://*.facebook.com
              http://*.facebook.com
              https://*.fbcdn.net
              http://*.fbcdn.net
              *.facebook.net
              *.google-analytics.com
              *.virtualearth.net
              *.google.com
              127.0.0.1:*
              *.spotilocal.com:*;
   options inline-script eval-script;
   report-uri https://www.facebook.com/csp.php
```

Content Security Policy Demo

Outline

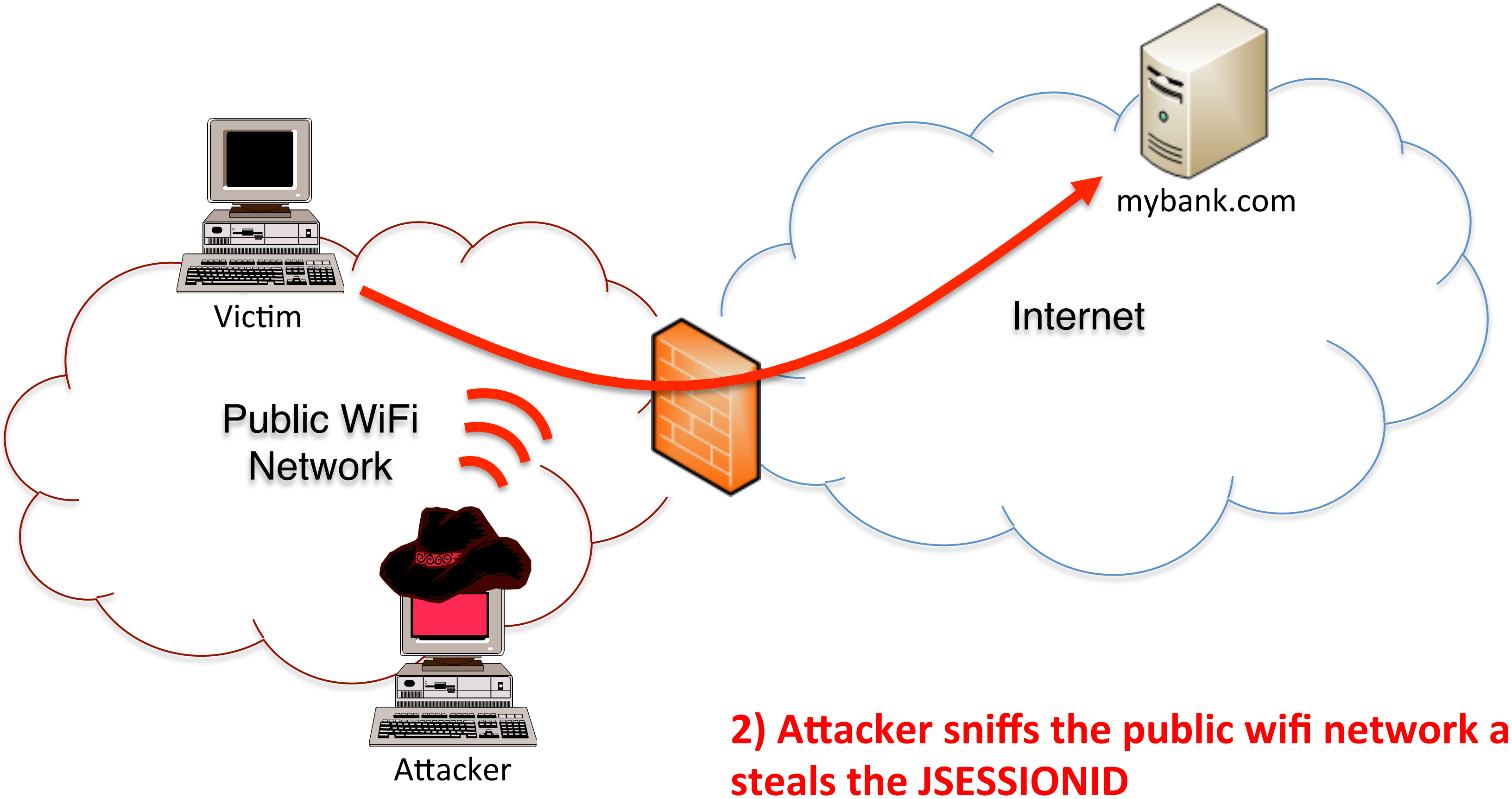
- XSS
- ➡ Session Hijacking
- Clickjacking
- Wrap Up

Session Hijacking



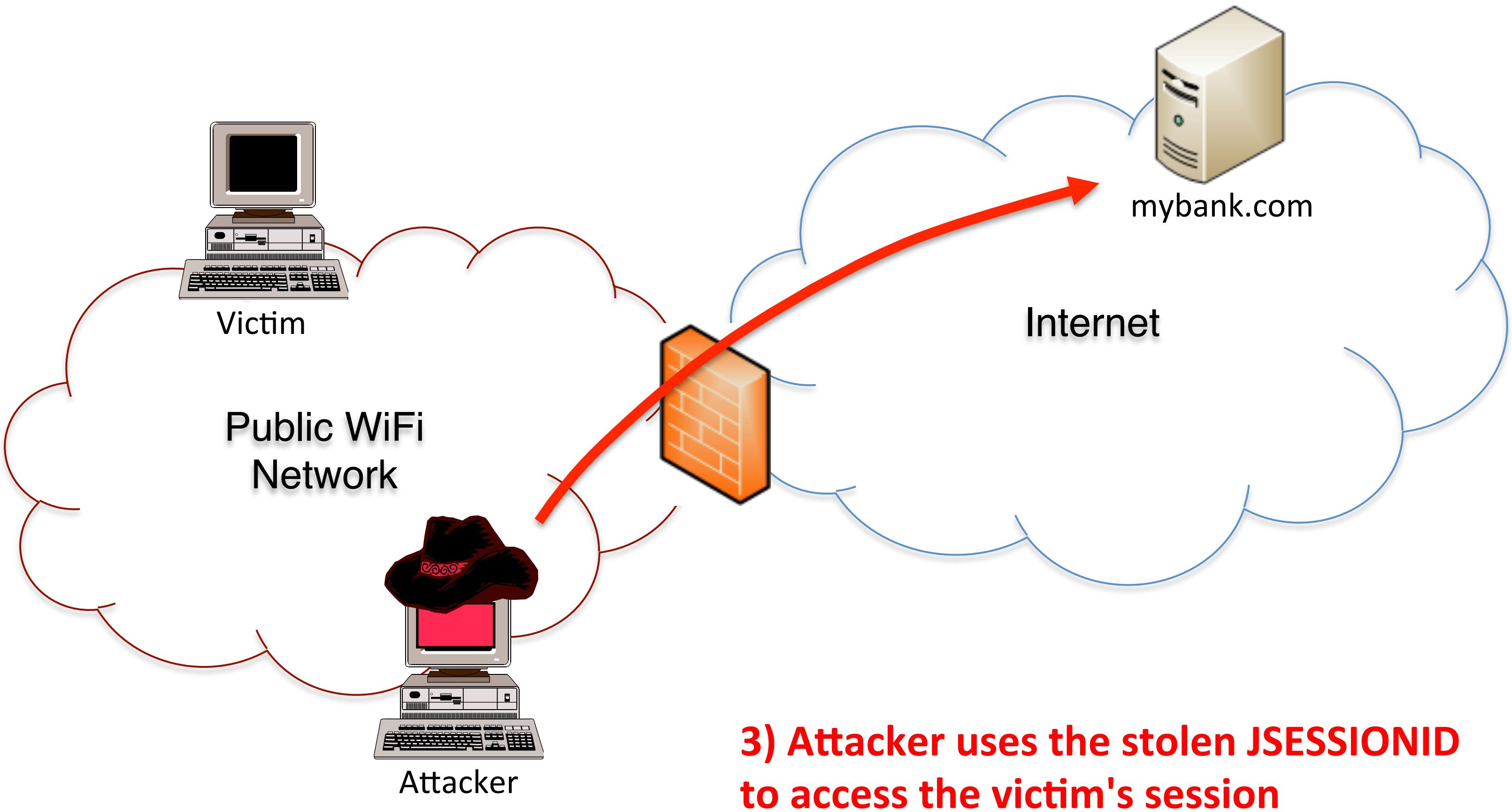
1) Victim goes to mybank.com via HTTP

Session Hijacking



2) Attacker sniffs the public wifi network and steals the JSESSIONID

Session Hijacking



3) Attacker uses the stolen JSESSIONID to access the victim's session

Secure Flag

- Ensures that the Cookie is only sent via SSL
- Configure in web.xml as of Servlet 3.0

```
<session-config>
  <cookie-config>
    <secure>true</secure>
  </cookie-config>
</session-config>
```

- Programmatically

```
Cookie cookie = new Cookie("mycookie", "test");
cookie.setSecure(true);
```

Strict-Transport-Security

- Tells browser to only talk to the server via HTTPS
  - First time your site accessed via HTTPS *and* the header is used the browser stores the certificate info
  - Subsequent requests to HTTP automatically use HTTPS
- Supported browsers
  - Implemented in Firefox and Chrome
  - Currently an IETF draft

```
Strict-Transport-Security: max-age=seconds
                                [; includeSubdomains]
```

Outline

- XSS
- Session Hijacking
- ➡ Clickjacking
- Wrap Up

Clickjacking

- Tricks the user into clicking a hidden button
  - User has no idea the button was clicked
- Works by concealing the target site site
  - Victim site placed in an invisible iframe
  - Attacker site overlays the victim site

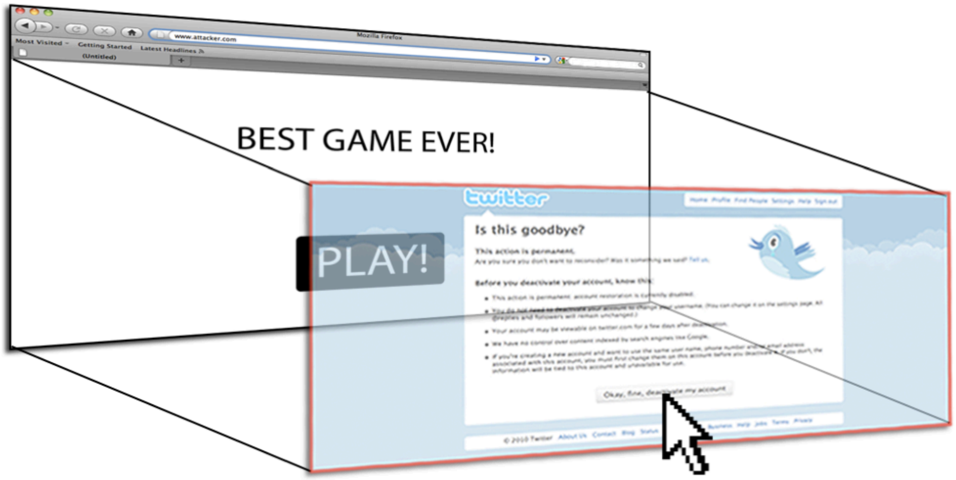

www.attacker.com
BEST GAME EVER!
PLAY!
twitter
Is this goodbye?
This action is permanent.
Before you deactivate your account, know this:
Okay, fine, deactivate my account
© 2010 Twitter

Image source: http://seclab.stanford.edu/websec/framebusting/framebust.pdf

Clickjacking Demo

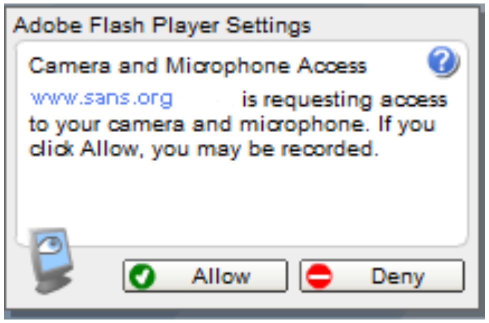
Clickjacking Code

- Put the victim in an invisible iframe

```
<iframe id="attacker" width=1000 height=400
  src="http://victim" style="opacity:0.0;
  position:absolute;left:10;bottom:100">
</iframe>
```

Adobe Flash Example

- Clickjacking discovered by Jeremiah Grossman & Robert "Rsnake" Hansen
- Showed how to use Flash to spy on users
  - Use Clickjacking to trick users into enabling the mic and camera via Flash



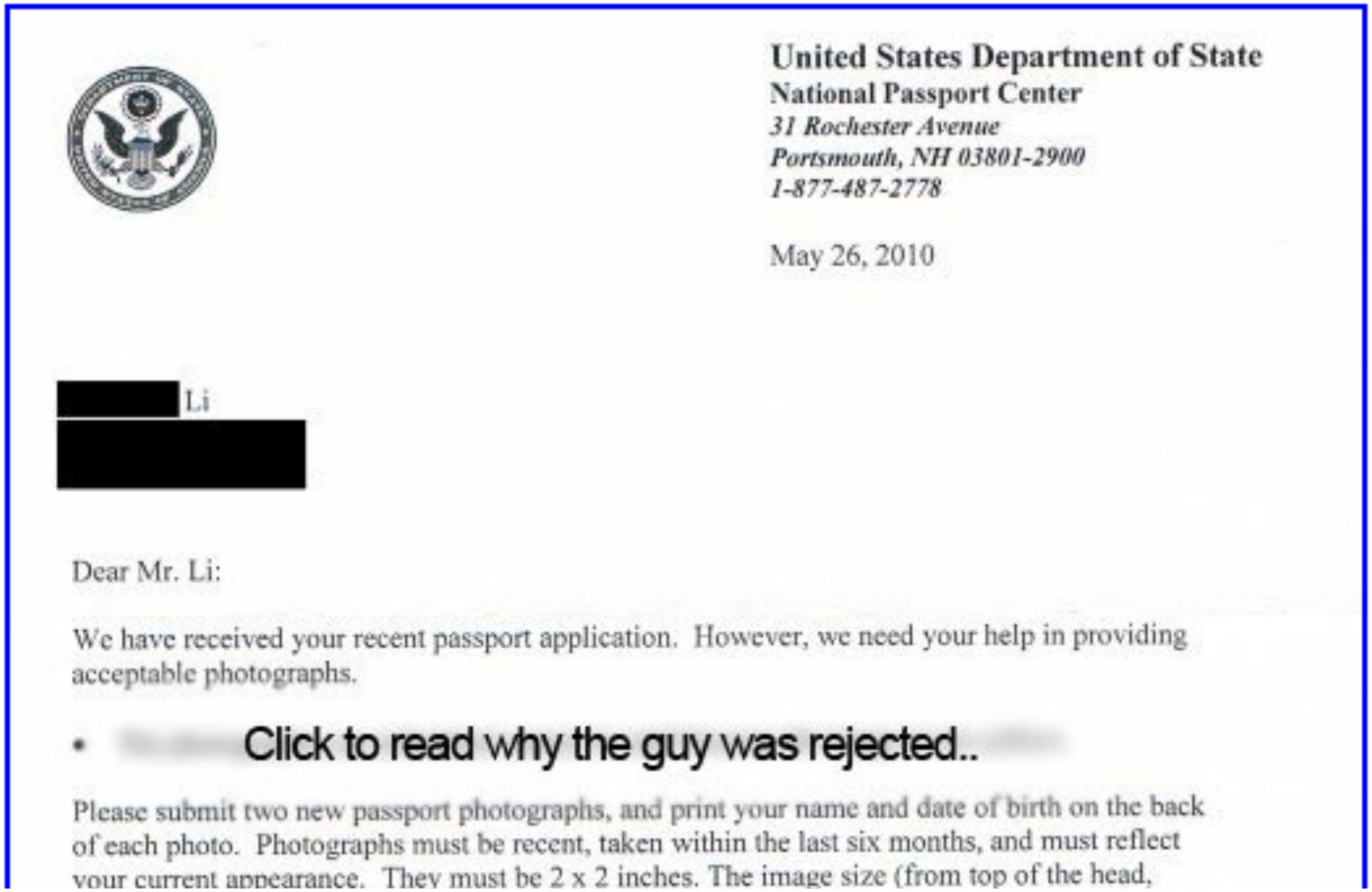
Adobe Flash Player Settings

Camera and Microphone Access

www.sans.org is requesting access to your camera and microphone. If you click Allow, you may be recorded.

Facebook Example

- The "best passport application rejection in history" became popular on Facebook



United States Department of State
National Passport Center
31 Rochester Avenue
Portsmouth, NH 03801-2900
1-877-487-2778

May 26, 2010

[redacted] Li
[redacted]

Dear Mr. Li:

We have received your recent passport application. However, we need your help in providing acceptable photographs.

- Click to read why the guy was rejected..

Please submit two new passport photographs, and print your name and date of birth on the back of each photo. Photographs must be recent, taken within the last six months, and must reflect your current appearance. They must be 2 x 2 inches. The image size (from top of the head,

Facebook Like Code

```
<div style="overflow:hidden; width:10px; height:12px;
filter:alpha(opacity=0); -moz-opacity:0.0; -khtml-opacity:
0.0; opacity:0.0; position:absolute;" id="icontainer">

<iframe src"http://www.facebook.com/plugins/like.php?
href=http://credittreport.info/the-best-passport-
application-rejection-in-history.html&amp;
layout=standard&amp;show_faces=false&amp;width=450&amp;act
ion=like&amp;font=tahoma&amp;colorscheme=light&amp;height=
80" scrolling="no" frame border="0" style="border:none;
overflow:hidden;width:50px; height:23px;"
allowTransparency="true" id="likee" name="likee">
</iframe>

</div>
```

Source: https://isc.sans.edu/diary.html?storyid=8893

Facebook Like Code

```
<div style="overflow:hidden; width:10px; height:12px;
filter:alpha(opacity=0); -moz-opacity:0.0; -khtml-opacity:
0.0; opacity:0.0; position:absolute;" id="icontainer">

<iframe src"http://www.facebook.com/plugins/like.php?
href=http://credittreport.info/the-best-passport-
application-rejection-in-history.html&amp;
layout=standard&amp;show_faces=false&amp;width=450&amp;act
ion=like&amp;font=tahoma&amp;colorscheme=light&amp;height=
80" scrolling="no" frame border="0" style="border:none;
overflow:hidden;width:50px; height:23px;"
allowTransparency="true" id="likee" name="likee">
</iframe>

</div>
```

Source: https://isc.sans.edu/diary.html?storyid=8893

Facebook Like Code

```
<div style="overflow:hidden; width:10px; height:12px;
filter:alpha(opacity=0); -moz-opacity:0.0; -khtml-opacity:
0.0; opacity:0.0; position:absolute;" id="icontainer">

<iframe src"http://www.facebook.com/plugins/like.php?
href=http://credittreport.info/the-best-passport-
application-rejection-in-history.html&amp;
layout=standard&amp;show_faces=false&amp;width=450&amp;act
ion=like&amp;font=tahoma&amp;colorscheme=light&amp;height=
80" scrolling="no" frame border="0" style="border:none;
overflow:hidden;width:50px; height:23px;"
allowTransparency="true" id="likee" name="likee">
</iframe>

</div>
```

Source: https://isc.sans.edu/diary.html?storyid=8893

Facebook Like Code

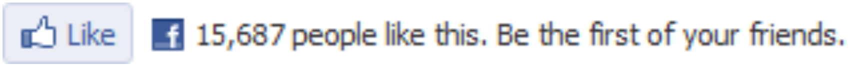
```
<div style="overflow:hidden; width:10px; height:12px;
filter:alpha(opacity=0); -moz-opacity:0.0; -khtml-opacity:
0.0; opacity:0.0; position:absolute;" id="icontainer">

<iframe src"http://www.facebook.com/plugins/like.php?
href=http://credittreport.info/the-best-passport-
application-rejection-in-history.html&amp;
layout=standard&amp;show_faces=false&amp;width=450&amp;act
ion=like&amp;font=tahoma&amp;colorscheme=light&amp;height=
80" scrolling="no" frame border="0" style="border:none;
overflow:hidden;width:50px; height:23px;"
allowTransparency="true" id="likee" name="likee">
</iframe>

</div>
```

Source: https://isc.sans.edu/diary.html?storyid=8893

Like Button Demo



15,687 people like this. Be the first of your friends.

Like Button Code

```
var like = document.createElement('iframe');

...

function mouseMove(e) {
    if (IE) {
        tempX = event.clientX + document.body.scrollLeft;
        tempY = event.clientY + document.body.scrollTop;
    } else {
        tempX = e.pageX;
        tempY = e.pageY;
    }

    if (tempX < 0) tempX = 0;
    if (tempY < 0) tempY = 0;

    like.style.top = (tempY - 8) + 'px';
    like.style.left = (tempX - 25) + 'px';

    return true
}
```

Like button moves with cursor

Source: http://erickerr.com/like-clickjacking

Why Likejacking?

- Send victims to evil sites with malware
- Trick users into signing up for unwanted subscription services
- Drive traffic to sites to increase ad revenue
- Adscend Media
  - Alleged to have made up to $1.2 million per month via Clickjacking
  - Facebook and Washington State filed lawsuits against them in January 2012

How to Fix?

- Use X-Frame-Options
  - HTTP Response Header supported by all recent browsers
- Three options
  - DENY
    - Prevents any site from framing the page
  - SAMEORIGIN
    - Allows framing only from the same origin
  - ALLOW-FROM *origin*
    - Allows framing only from the specified *origin*
    - Only supported by IE (based on my testing)
    - Firefox Bug 690168 - "This was an unintentional oversight"

Java Code

- DENY

```
response.addHeader("X-Frame-Options", "DENY");
```

- SAMEORIGIN

```
response.addHeader("X-Frame-Options", "SAMEORIGIN");
```

- ALLOW-FROM

```
String value = "ALLOW-FROM http://www.trustedsite.com:8080";
response.addHeader("X-Frame-Options", value);
```

X-Frame-Options Demo

Using X-Frame-Options

- You might not want to use it for the entire site
  - Prevents legitimate framing of your site (i.e. Google Image Search)
- For sensitive transactions
  - Use SAMEORIGIN
  - And test thoroughly
- If the page should never be framed
  - Then use DENY

Frame Busting Code

- What about older browsers that don't support X-Frame-Options?
- JavaScript code like this is commonly used

```
if (top != self)
    top.location = self.location;
```

- Not full-proof
  - – Various techniques can be used to bypass frame busting code

Some Anti-Frame Busting Techniques

- IE <iframe security=restricted>
  - Disables JavaScript within the iframe
- onBeforeUnload - 204 Flushing
  - Repeatedly send a 204 (No Content) response so the onBeforeUnload handler gets canceled
- Browser XSS Filters
  - Chrome XSSAuditor filter cancels inline scripts if they are also found as a parameter

```
<iframe src="http://www.victim.com/?v=if(top+!%3D
+self)+%7B+top.location%3Dself.location%3B+%7D">
```

Outline

- XSS
- Session Hijacking
- Clickjacking

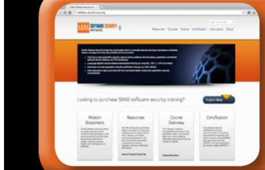
➡ Wrap Up

Summary

- Use the following HTTP Response Headers
  - ☑ Set-Cookie HttpOnly
  - ☑ X-XSS-Protection: 1; mode=block
  - ☑ Set-Cookie Secure
  - ☑ Strict-Transport-Security
  - ☑ X-Frame-Options: SAMEORIGIN
- Plan to use the following
  - ☑ Content Security Policy

SANS Software Security

CURRICULUM



Website:

http://software-security.sans.org

Free resources, white papers, webcasts, and more

Courses

Attack

SEC 542 – Web App Penetration Testing and Ethical Hacking

Assess Your Web Apps in Depth Web applications are a major point of vulnerability in organizations today. Web app holes have resulted in the theft of millions of credit cards, major financial and reputational damage for hundreds of enterprises, and even the compromise of thousands of browsing machines that visited Web sites altered by attackers. In this intermediate to advanced level class, you'll learn the art of exploiting Web applications so you can find flaws in your... Read More

Defense

DEV 522 – Defending Web Applications Security Essentials

This is the course to take if you have to defend web applications! Traditional network defenses, such as firewalls, fail to secure web applications. The quantity and importance of data entrusted to web applications is growing, and defenders need to learn how to secure it. DEV522 covers the OWASP Top 10 and will help you to better understand web application vulnerabilities, thus enabling you to properly defend y... Read More

Courses:

http://software-security.sans.org/courses

Blog

Blog:

http://software-security.sans.org/blog

Twitter:

@sansappsec

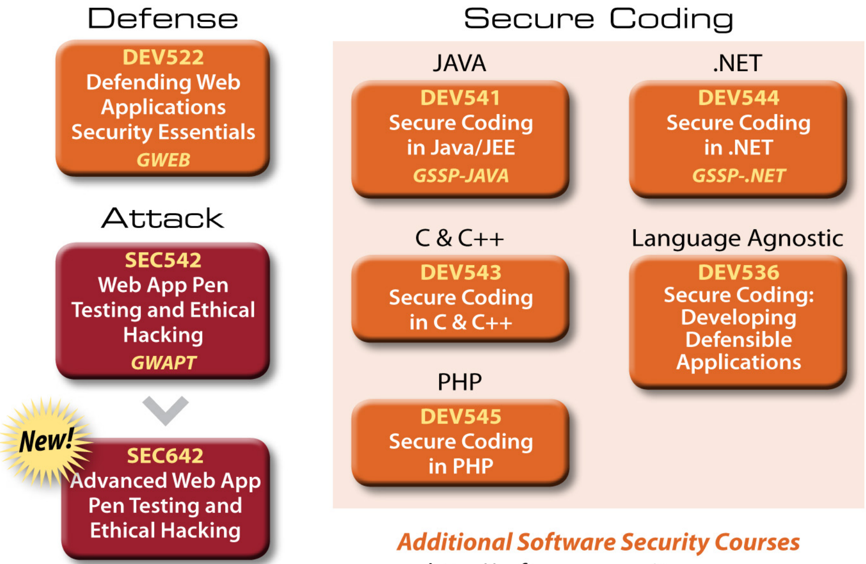
Latest news, promos, and other information

Secure Coding Assessment:

http://software-security.sans.org/courses/assessment

Security Developer

The security-savvy software developer leads all developers in the creation of secure software, implementing secure programming techniques that are free from logical design and technical implementation flaws. This expert is ultimately responsible for ensuring customer software is free from vulnerabilities that can be exploited by an attacker.



Defense

DEV522 Defending Web Applications Security Essentials *GWEB*

Attack

SEC542 Web App Pen Testing and Ethical Hacking *GWAPT*

New! **SEC642** Advanced Web App Pen Testing and Ethical Hacking

Secure Coding

JAVA

DEV541 Secure Coding in Java/JEE *GSSP-JAVA*

.NET

DEV544 Secure Coding in .NET *GSSP-.NET*

C & C++

DEV543 Secure Coding in C & C++

Language Agnostic

DEV536 Secure Coding: Developing Defensible Applications

PHP

DEV545 Secure Coding in PHP

Additional Software Security Courses

http://software-security.sans.org

Frank Kim

frank@thinksec.com

@thinksec

@sansappsec

THINKSEC

SANS

References

- Content Security Policy
    - https://dvcs.w3.org/hg/content-security-policy/raw-file/tip/csp-specification.dev.html
- Busting Frame Busting: A Study of Clickjacking Vulnerabilities on Popular Sites
    - http://seclab.stanford.edu/websec/framebusting/framebust.pdf
- Like Clickjacking
    - http://erickerr.com/like-clickjacking
- Clickjacking Attacks on Facebook's Like Plugin
    - https://isc.sans.edu/diary.html?storyid=8893
- Lessons from Facebook's Security Bug Bounty Program
    - https://nealpoole.com/blog/2011/08/lessons-from-facebooks-security-bug-bounty-program/
- Google+ Gets a "+1" for Browser Security
    - http://www.barracudalabs.com/wordpress/index.php/2011/07/21/google-gets-a-1-for-browser-security-3/