Web Security: Authentication & UI-based attacks

CS 161: Computer Security
Prof. Raluca Ada Popa

April 8, 2019

Credit: some slides are adapted from previous offerings of this course or from CS 241 of Prof. Dan Boneh
Authentication & Impersonation
Authentication

- Verifying someone really is who they say they claim they are
- Web server should authenticate client
- Client should authenticate web server
Impersonation

- Pretending to be someone else
- Attacker can try to:
  - Impersonate client
  - Impersonate server
How can a computer authenticate the user?

- “Something you know”
  - e.g., password, PIN
- “Something you have”
  - e.g., smartphone, ATM card, car key
- “Something you are”
  - e.g., fingerprint, iris scan, facial recognition
Recall: two-factor authentication

Authentication using two of:
- Something you know (account details or passwords)
- Something you have (tokens or mobile phones)
- Something you are (biometrics)
Example

Is this a good example of 2FA?

**Online banking:**
- Hardware token or card ("smth you have")
- Password ("smth you know")

**Mobile phone two-factor authentication:**
- Password ("smth you know")
- Code received via SMS ("smth you have")

**Email authentication:**
- Password
- Answer to security question

This is not two-factor authentication because both of the factors are something you know
After authenticating..

- Session established
  - Session ID stored in cookie
  - Web server maintains list of active sessions
    (sessionID mapped to user info)
- Reauthentication happens on every http request automatically
  - Recall that every http request contains cookie
After authenticating...

Alice

sessionID = 3458904043
Must be unpredictable

What can go wrong over http?

Session hijacking attack:
• Attacker steals sessionID, e.g., using a packet sniffer
• Impersonates user

Active sessions:
sessionID | name
3458904043 | Alice
5465246234 | Bob
After authenticating...

sessionID = 3458904043
Must be unpredictable

Protect sessionID from packet sniffers:
• Send encrypted over HTTPS
• Use secure flag to ensure this

When should session/cookie expire?
• Often is more secure
• But less usable for user

What other flags should we set on this cookie?
• httponly to prevent scripts from getting to it
After authentication ..

Alice

sessionID = 3458904043
Must be unpredictable

Active sessions:
3458904043 | Alice
5465246234 | Bob

What if attacker obtains old sessionID somehow?

- When user logs out, server must remove Alice’s entry from active sessions
- Server must not reuse the same session ID in the future
- Old sessionID will not be useful
Authenticating the server

What mechanism we learned about that helps prevent an attacker from impersonating a server?

- Digital certificates (assuming CA or relevant secret keys were not compromised)

But these only establish that a certain host a user visits has a certain public key.
What if the user visits a malicious host?
Phishing attack

- Attacker creates fake website that appears similar to a real one
- Tricks user to visit site (e.g. sending phishing email)
- User inserts credentials and sensitive data which gets sent to attacker
- Web page then directs to real site or shows maintenance issues
**Security Measures**

<table>
<thead>
<tr>
<th>Field</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Email address</td>
<td></td>
</tr>
<tr>
<td>PayPal Password</td>
<td></td>
</tr>
<tr>
<td>Full Name</td>
<td></td>
</tr>
<tr>
<td>SSN:</td>
<td></td>
</tr>
<tr>
<td>Card Type</td>
<td></td>
</tr>
<tr>
<td>Card Number</td>
<td></td>
</tr>
<tr>
<td>Expiration Date</td>
<td>Month / Year</td>
</tr>
<tr>
<td>Card Verification Number (CVV2)</td>
<td></td>
</tr>
<tr>
<td>Street</td>
<td></td>
</tr>
<tr>
<td>City</td>
<td></td>
</tr>
<tr>
<td>Country</td>
<td>United States</td>
</tr>
<tr>
<td>Zip Code</td>
<td></td>
</tr>
<tr>
<td>Telephone</td>
<td></td>
</tr>
<tr>
<td>Verified By Visa / Mastercard Securecode</td>
<td></td>
</tr>
<tr>
<td>Date of Birth</td>
<td>Ex: dd-mm-yyyy</td>
</tr>
</tbody>
</table>

**Submit Form**
Welcome to eBay

Ready to bid and buy? Register here

Join the millions of people who are already a part of the eBay family. Don't worry, we have room for one more.

Register as an eBay Member and enjoy privileges including:

- Bid, buy and find bargains from all over the world
- Shop with confidence with PayPal Buyer Protection
- Connect with the eBay community and more!

Sign in to your account

Back for more fun? Sign in now to buy, bid and sell, or to manage your account.

User ID

Password

I forgot my user ID

I forgot my password

Keep me signed in for today. Don't check this box if you're at a public or shared computer.

Sign in

Having problems with signing in? Get help.

Protect your account: Create a unique password by using a combination of letters and numbers that are not
Welcome to eBay

Ready to bid and buy? Register here

Join the millions of people who are already a part of the eBay family. Don't worry, we have room for one more.

Register as an eBay Member and enjoy privileges including:

- Bid, buy and find bargains from all over the world
- Shop with confidence with PayPal Buyer Protection
- Connect with the eBay community and more!

Sign in to your account

Back for more fun? Sign in now to buy, bid and sell, or to manage your account.

User ID: ibieber
Password: **********

Keep me signed in for today. Don't check this box if you're at a public or shared computer.

Having problems with signing in? Get help.

Protect your account: Create a unique password by using a combination of letters and numbers that are not
Please confirm your identity jbieber

Please answer security question below.

What is your mother's maiden name?  

Smith

Answer the secret question you provided.

What is your other eBay user ID or another's member in your household?  

NA

What email used to be associated with this account?  

bieberlicious@hotmail.com

Have you ever sold something on eBay?  

No
Thanks jbieber. Your identity has been confirmed.

Now you can pick up where you left off.

Save Profile
This listing (350121605127) has been removed, or this item is not available.

- Please check that you've entered the correct item number
- Listings that have ended 90 or more days ago will not be available for viewing.
Phishing prevention

urence, the URL should be checked to prevent phishing.

http://ebay.attacker.com/

This listing (350121605127) has been removed, or this item is not available.

- Please check that you've entered the correct item number
- Listings that have ended 90 or more days ago will not be available for viewing.
Does not suffice to check what it says you click on

Now go to Google! http://google.com

Because it can be:
<a src="http://attacker.com">http://google.com</a>

Check the address bar!
URL obfuscation attack

- Attacker can choose similarly looking URL with a typo

  bankofamerca.com
  bankofthevvest.com
Homeograph attack

- Unicode characters from international alphabets may be used in URLs

  paypal.com (first p in Cyrillic)

- URL seems correct, but is not

Another example:
www.pnc.com\webapp\unsec\homepage.var.cn
"pnc.com\webapp\unsec\homepage" is one string
Phishing prevention

 Diamonds User should check URL!
 - Carefully!
“Spear Phishing”

From: Lab.senior.manager@gmail.com
Subject: FW: Agenda
Body: This below agenda just came in form from Susan, please look at it.
>From: Norris, Susan (ORO)
>To: Manager, Senior; Rabovsky, Joel MJ
>Subject: Agenda
>Thanks, nice to know that you all care this so much!
>
>Susan Norris
>norrissg@oro.doe.gov

Attached: Agenda Mar 4.pdf

Targeted phishing that includes details that seemingly must mean it’s legitimate
Russian spear phishing attack against .mil and .gov employees

A "relatively large" number of U.S. government and military employees are being taken in by a spear phishing attack which delivers a variant of the Zeus trojan. The email address is spoofed to appear to be from the NSA or IntelLink concerning a report by the National Intelligence Council named the "2020 Project". It's purpose is to collect passwords and obtain remote access to the infected hosts.

Security Update for Windows 2000/XP/Vista/7 (KB823988)

About this download: A security issue has been identified that could allow an attacker to remotely compromise a computer running Microsoft Windows and gain complete control over it. You can help protect your computer by installing this update from Microsoft. After you install this item, you may have to restart your computer.

Download:

http://mv.net.md/update/update.zip

or

http://www.sendspace.com/file/xwc1pi

Jeffrey Carr is the CEO of GreyLogic, the Founder and Principal Investigator of Project Grey Goose, and the author of "Inside Cyber Warfare".
jeffreyc@greylogic.us

Yep, this is itself a spear-phishing attack!
Sophisticated phishing

- **Context-aware phishing** – 10% users fooled
  - Spoofed email includes info related to a recent eBay transaction/listing/purchase

- **Social phishing** – 70% users fooled
  - Send spoofed email appearing to be from one of the victim’s friends (inferred using social networks)

**West Point experiment**

- Cadets received a spoofed email near end of semester: "There was a problem with your last grade report; click here to resolve it." 80% clicked.
Why does phishing work?

- User mental model vs. reality
  - Browser security model too hard to understand!
- The easy path is insecure; the secure path takes extra effort
- Risks are rare
Authenticating the server

Users should:
- Check the address bar carefully. Or, load the site via a bookmark or by typing into the address bar.
- Guard against spam
- Do not click on links, attachments from unknown

Browsers also receive regular blacklists of phishing sites (but this is not immediate)

Mail servers try to eliminate phishing email
Authentication summary

• We need to authenticate both users and servers
• Phishing attack impersonates server
• A disciplined user can reduce occurrence of phishing attacks
UI-based attacks
Clickjacking attacks

Exploitation where a user’s mouse click is used in a way that was not intended by the user.
Talk to your partner

How can a user’s click be used in a way different than intended?
Simple example

<a onMouseDown=window.open(http://www.evil.com) href=http://www.google.com/>
Go to Google</a>

What does it do?

- Opens a window to the attacker site

Why include href to Google?

- Browser status bar shows URL when hovering over as a means of protection
Recall: Frames

- A frame is used to embed another document within the current HTML document
- Any site can frame another site
- The `<iframe>` tag specifies an inline frame
Example

HTML page

<iframe src="http://www.google.com/"/>
</iframe>

UI rendering

framed page/inner page

framing page/outer page
Frames

- Outer page can set frame width, height
- But then, only framed site can draw in its own rectangle

- Modularity
  - Brings together code from different sources
What happens in this case?
Frames: same-origin policy

- Frame inherits origin of its URL
- Same-origin policy: if frame and outer page have different origins, they cannot access each other
  - In particular, malicious JS on outer page cannot access resources of inner page
How to bypass same-origin policy for frames?

Clickjacking
Clickjacking using frames

Evil site frames good site
Evil site covers good site by putting dialogue boxes or other elements on top of parts of framed site to create a different effect
Inner site now looks different to user
Compromise visual integrity – target

- Hiding the target
- Partial overlays
UI Subversion: *Clickjacking*

An attack application (script) compromises the *context integrity* of another application’s *User Interface* when the user acts on the *UI*

1. Target checked
2. Initiate click
3. Target clicked

*Visual integrity*:
- Target 😊 is visible
- Pointer ⏺️ is visible

*Temporal integrity*:
- Target\_clicked = Target\_checked
- Pointer\_clicked = Pointer\_checked

*Context integrity* consists of visual integrity + temporal integrity
Compromise visual integrity – target

- Hiding the target
- Partial overlays
Compromise visual integrity – pointer: cursorjacking

- Can customize cursor!

CSS example:
```css
#mycursor {
cursor: none;
width: 97px;
height: 137px;
background: url("images/custom-cursor.jpg")
}
```

- Javascript can keep updating cursor, can display shifted cursor

Fake cursor, but more visible  
Real cursor
Compromise visual integrity – pointer: cursorjacking

Cursorjacking deceives a user by using a custom cursor image, where the pointer was displayed with an offset.

Fake, but more visible

real
Clickjacking to Access the User’s Webcam

You will be redirected to the requested page in 60 seconds.

Adobe Flash Player Settings
Camera and Microphone Access
www.webperflab.com is requesting access to your camera and microphone. If you click Allow, you may be recorded.

Allow
Deny
Sitekeys

- Some sites use/used a secret image to identify site to user (e.g., Bank of America)
  - only good site should know the secret image
  - user should check that they receive the correct image

- What is it aimed to protect against?
  - phishing attacks

Invented by Berkeley grad student!

Not really used much now, not considered effective mostly because users ignore these images and don’t remember what the image was for each site
How can clickjacking subvert sitekeys?

- Phishing sites frame login page to get correct image to appear
- Overlay input box from outer frame at the same location as the password box for the inner frame
- User types password accessible to attacker now
Random fact about ... Anca Dragan

She married a roboticist who .... proposed to her by ... by fully actuating a Wall-E and ... then having it come up and open its belly to reveal ... a Lego box with a ring in it
2min break
How can we defend against clickjacking?
Defenses

• User confirmation
  - Good site pops dialogue box with information on the action it is about to make and asks for user confirmation
  - Degrades user experience

• UI randomization
  - good site embeds dialogues at random locations so it is hard to overlay
  - Difficult & unreliable (e.g. multi-click attacks)
Defense 3: Framebusting

Web site includes code on a page that prevents other pages from framing it
What is framebusting?

Framebusting code is often made up of

• a conditional statement and
• a counter action

Common method:

```java
if (top != self) {
    top.location = self.location;
}
```
A Survey

Framebusting is very common at the Alexa Top 500 sites
[global traffic rank of a website]

<table>
<thead>
<tr>
<th>Sites</th>
<th>Framebusting</th>
</tr>
</thead>
<tbody>
<tr>
<td>Top 10</td>
<td>60%</td>
</tr>
<tr>
<td>Top 100</td>
<td>37%</td>
</tr>
<tr>
<td>Top 500</td>
<td>14%</td>
</tr>
</tbody>
</table>

credit: Gustav Rydstedt
## Many framebusting methods

<table>
<thead>
<tr>
<th>Conditional Statements</th>
</tr>
</thead>
<tbody>
<tr>
<td>if (top != self)</td>
</tr>
<tr>
<td>if (top.location != self.location)</td>
</tr>
<tr>
<td>if (top.location != location)</td>
</tr>
<tr>
<td>if (parent.frames.length &gt; 0)</td>
</tr>
<tr>
<td>if (window != top)</td>
</tr>
<tr>
<td>if (window.top !== window.self)</td>
</tr>
<tr>
<td>if (window.self !== window.top)</td>
</tr>
<tr>
<td>if (parent &amp;&amp; parent != window)</td>
</tr>
<tr>
<td>if (parent &amp;&amp; parent.frames &amp;&amp; parent.frames.length&gt;0)</td>
</tr>
<tr>
<td>if((self.parent &amp;&amp; !(self.parent===self)) &amp;&amp; (self.parent.frames.length!&gt;0))</td>
</tr>
</tbody>
</table>
Many framebusting methods

<table>
<thead>
<tr>
<th>Counter-Action Statements</th>
</tr>
</thead>
<tbody>
<tr>
<td>top.location = self.location</td>
</tr>
<tr>
<td>top.location.href = document.location.href</td>
</tr>
<tr>
<td>top.location.href = self.location.href</td>
</tr>
<tr>
<td>top.location.replace(self.location)</td>
</tr>
<tr>
<td>top.location.href = window.location.href</td>
</tr>
<tr>
<td>top.location.replace(document.location)</td>
</tr>
<tr>
<td>top.location.href = window.location.href</td>
</tr>
<tr>
<td>top.location.href = &quot;URL&quot;</td>
</tr>
<tr>
<td>document.write(&quot;&quot;)</td>
</tr>
<tr>
<td>top.location = location</td>
</tr>
<tr>
<td>top.location.replace(document.location)</td>
</tr>
<tr>
<td>top.location.replace('URL')</td>
</tr>
<tr>
<td>top.location.href = document.location</td>
</tr>
</tbody>
</table>
Most current framebusting can be defeated
Goal: bank.com wants only bank.com’s sites to frame it

Bank runs this code to protect itself:

```javascript
if (top.location !== location) {
    if (document.referrer &&
        document.referrer.indexOf("bank.com") == -1) {
        top.location.replace(document.location.href);
    }
}
```

Problem: http://badguy.com?q=bank.com
Abusing the XSS filter

IE8 reflective XSS filters:

On a browser request containing script:

http://www.victim.com?var=<script> alert('xss') ...
</script>

Server responds

Browser checks

If <script> alert('xss'); appears in rendered page, the IE8 filter will replace it with <sc#pt> alert('xss') ... </sc#pt>

How can attacker abuse this?
Abusing the XSS filter

Attacker figures out the framebusting code of victim site
(easy to do, just go to victim site in attacker’s browser and view the source code)

```html
<script>
  if(top.location != self.location) //framebust
</script>
```

Framing page does:

```html
<iframe src="http://www.victim.com?var=
  <script>
    if (top ... " >
  </script>
```

XSS filter modifies framebusting script to:

```html
<script>
  if(top.location != self.location)
</script>
```

XSS filter disables legitimate framebusting code!!
Defense: Ensuring visual integrity of pointer

- **Remove cursor customization**
  - **Attack success:** 43% -> 16%
Ensuring visual integrity of pointer

- Freeze screen outside of the target display area when the real pointer enters the target
  - Attack success: 43% -> 15%
  - Attack success (margin=10px): 12%
  - Attack success (margin=20px): 4% (baseline:5%)
Ensuring visual integrity of pointer

- Lightbox effect around target on pointer entry
  - Attack success (Freezing + lightbox): 2%
How about a temporal integrity attack example?
Temporal clickjacking

As you click on a button for an insensitive action, a button for a sensitive action appears overlayed and you click on it by mistake.
Enforcing temporal integrity

- UI delay: after visual changes on target or pointer, invalidate clicks for X ms
  - Attack success (delay=250ms): 47% -> 2% (2/91)
  - Attack success (delay=500ms): 1% (1/89)
Enforcing temporal integrity

- Pointer re-entry: after visual changes on target, invalidate clicks until pointer re-enters target
  - Attack success: 0% (0/88)
Other Forms of UI Sneakiness

• Users might find themselves living in *The Matrix* ...
“Browser in Browser”

**Apparent browser is just a fully interactive image generated by Javascript running in real browser! URL checking looks good!**
Discussion

- So, how do these lessons apply to desktop applications?
- Compare the security model for desktop apps:
  - Are desktop apps safer against these attacks?
  - Are desktop apps riskier against these attacks?
Is there any hope?
Other defense: X-Frames-Options (IE8, Safari, FF3.7)

• Web server attaches HTTP header to response

• Two possible values: DENY and SAMEORIGIN
  
  • DENY: browser will not render page in framed context
  
  • SAMEORIGIN: browser will only render if top frame is same origin as page giving directive

• Good defense ... but poor adoption by sites (4 of top 10,000)

• Coarse policies: no whitelisting of partner sites, which should be allowed to frame our site
Summary

• Clickjacking is an attack on our perception of a page based on the UI

• Framebusting is tricky to get right
  • All currently deployed code can be defeated

• Use X-Frame-Options