CSE 477: Introduction to Computer Security

Lecture – 25

Course Teacher: Dr. Md Sadek Ferdous
Assistant Professor, CSE, SUST
E-mail: ripul.bd@gmail.com

Outline

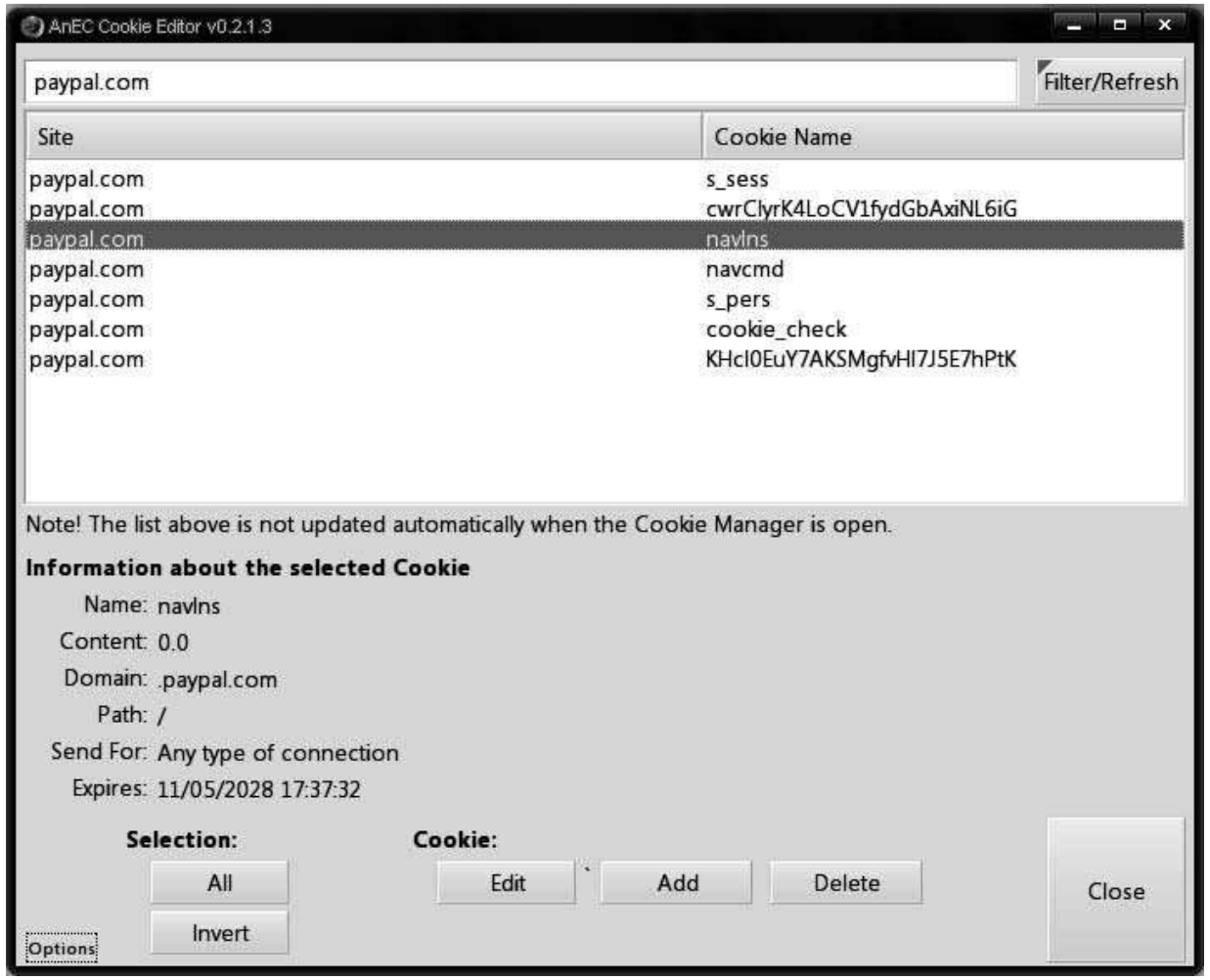
- Attacks on Web Protocol
- Mitigations Techniques

HTTP Session via Cookie

- Another common method of creating user sessions uses small packets of data, called ***cookies***
- Cookies are a small bit of information stored on a computer associated with a specific server
- This is then are sent to the client by the web server and stored on the client's machine
- When the user revisits the web site, these cookies are returned, unchanged, to the server, which can then "remember" that user and access their session information
- Cookies are set on a client's system when a server uses the Set-Cookie field in the header of an HTTP response

HTTP Session via Cookie

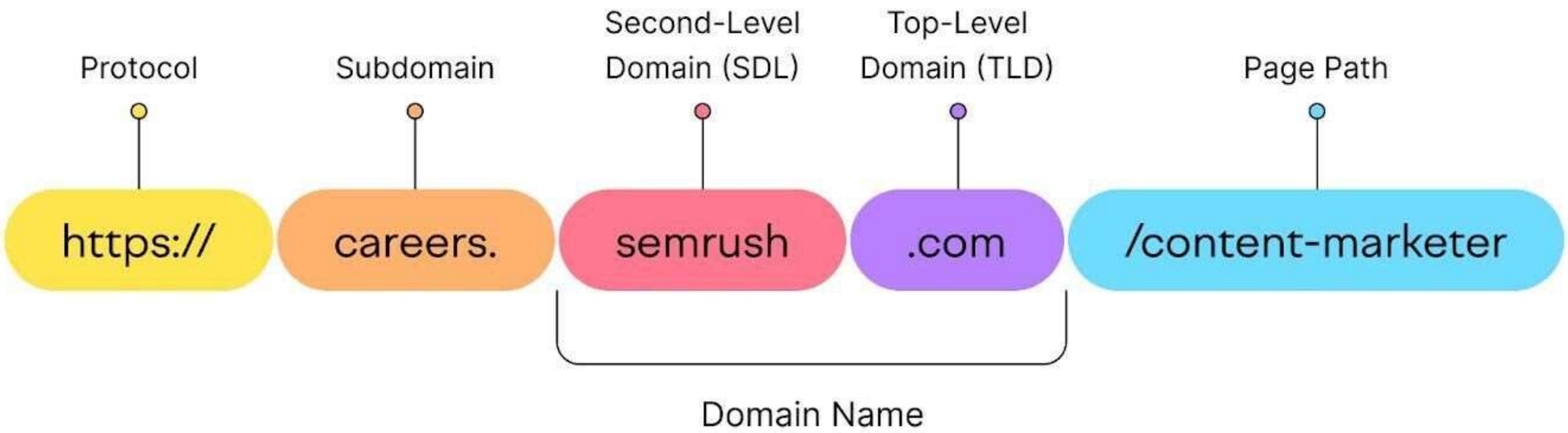
- The name and content fields correspond to the key-value pair of the cookie
- The domain name .paypal.com specifies that this cookie is valid for this top-level domain and all subdomains
- The path / indicates that it applies to the root directory of the site
- The send for value indicates that this is not a secure cookie
- The expiration date specifies when this cookie will be automatically deleted
  - If no expiration date is specified, the cookie is deleted when the user exits the browser



Cookie properties

- The domain field can be specified for a top-level domain or subdomains of a web site
- Only hosts within a domain can set a cookie for that domain
- A subdomain can set a cookie for a higher-level domain, but not vice versa
- Similarly, subdomains can access cookies set for the top-level domain, but not the other way around
  - mail.example.com could access cookies set for example.com or mail.example.com
  - example.com could not access cookies set for mail.example.com
- Hosts can access cookies set for their top-level domains, but hosts can only set cookies for domains one level up in the domain hierarchy
  - one.mail.example.com could read (but not set) a cookie for .example.com

Parts of a URL



Cookie properties

- The path field specifies that the cookie can only be accessed within a specific subdirectory of the web site
  - defaults to the root directory of a given domain if path is absent
- By default, cookies are transmitted unencrypted using HTTP, and as such are subject to the same man-in-the-middle attacks as all HTTP requests
- To remedy this weakness, a secure flag, which requires that a given cookie be transmitted using HTTPS, can be set
- Situations have been disclosed where web sites using HTTPS to encrypt regular data transfer failed to properly set the secure cookie flag
  - Resulting in the possibility of session hijacking
- A sensitive cookie can be further protected by encrypting its value and by using an opaque name
- Thus, only the web server can decrypt the cookie and malware that accesses the cookie cannot extract useful information from it

Cookie properties

- Finally, cookies can set an HTTP-Only flag
- If enabled, scripting languages are prevented from accessing or manipulating cookies stored on the client's machine
- This does not stop the use of cookies themselves
  - The browser will still automatically include any cookies stored locally for a given domain in HTTP requests to that domain!
- In addition, the user still has the ability to modify cookies through browser plugins

HTTP session via cookie

- To let the server access previously set cookies, the client automatically includes any cookies set for a particular domain and path in the Cookie field of any HTTP request header being sent to that server
- Notably, a user's cookies are accessible via the DOM, and therefore can be accessed by many scripting languages
- The cookie specification is built directly into the HTTP protocol, which is interpreted by the browser
  - As a result, the mechanism for setting and accessing cookies is different for each scripting language

HTTP session via cookie

- Many languages have their own built-in cookie APIs that provide convenient means of using cookies
- But other languages, including JavaScript, treat cookies as simple strings of text stored in the DOM
- All of these properties of cookies are managed by the browser, rather than the operating system
- Each browser sets aside space for storing this information, and allows the possibility of a user having separate sets of cookie information for each of multiple browser

Cookie security

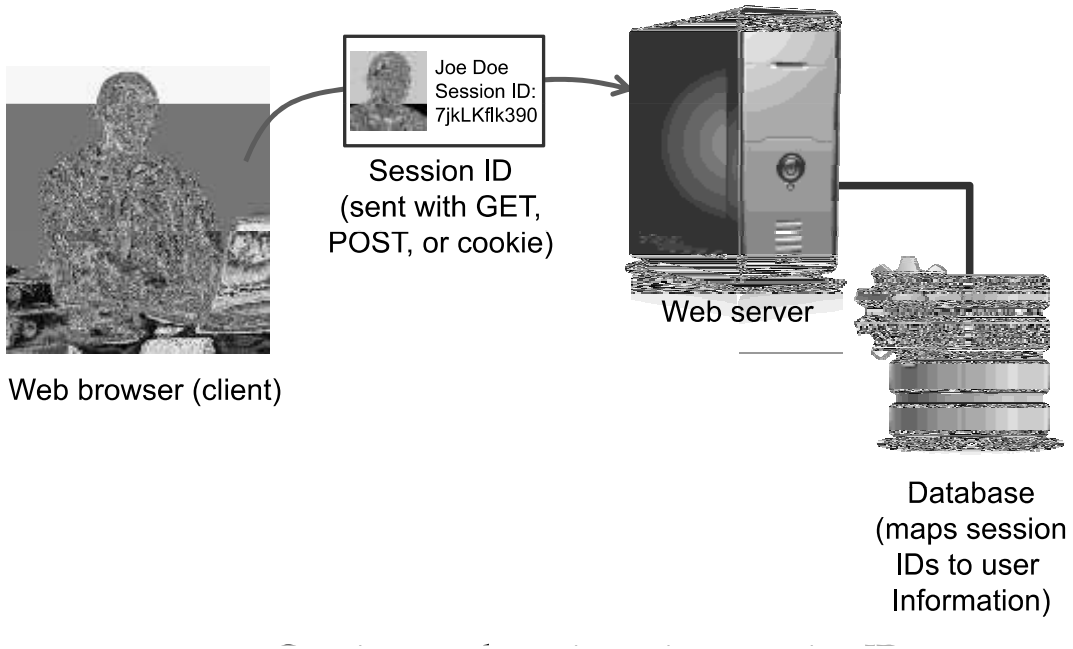
- Cookies have profound implications for the security of user sessions
- For instance, it is dangerous to store any sensitive information unencrypted in the body of a cookie, since cookies can typically be accessed by users of the system on which they are stored
- Even if sensitive information is encrypted, however, accessing a user's cookies for a web site may allow an attacker to assume that user's session
- Because of this, there is a need for users to protect their cookies as they would any login information
- The expiration date built into cookies is a good preventive measure, but it is still recommended that users erase their cookies on a regular basis to prevent such attacks
- In addition to these security concerns, cookies also raise several issues related to user privacy discussed later

Server side session

- A final method of maintaining session information is to devote space on the web server for keeping user information
- This model reduces several risks for the user, because compromise of the user’s system no longer necessarily results in compromise of their web sessions
- Widely used in e-commerce websites for maintaining shopping carts

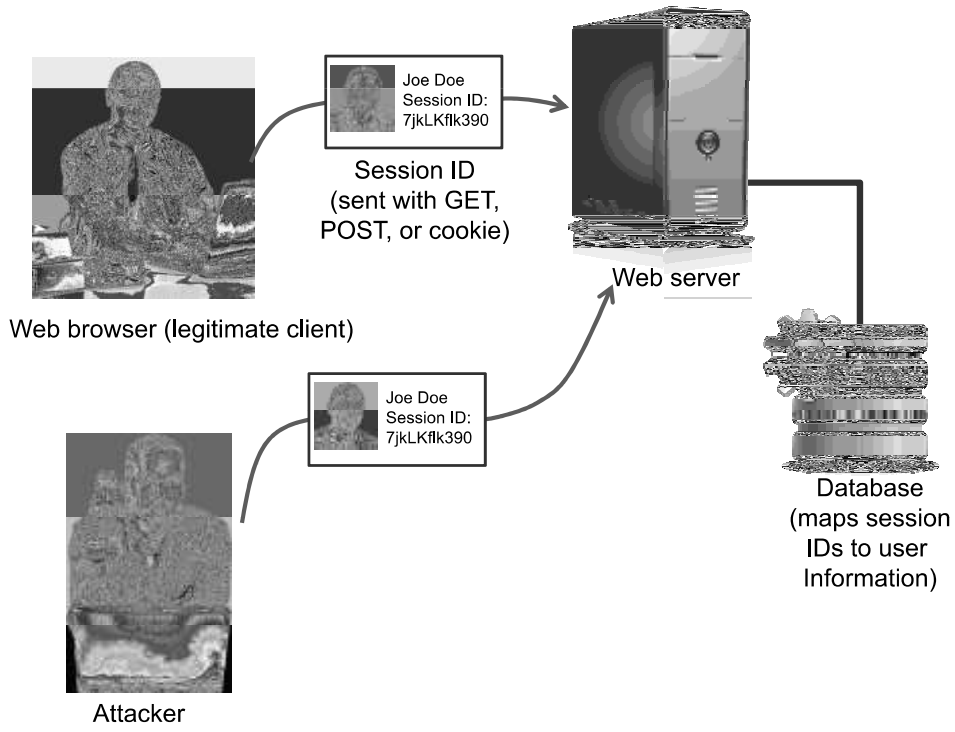
Server side session

- For this, servers typically use a ***session ID*** or ***session token*** — a unique identifier that corresponds to a user's session
- The server then employs one of the two previous methods (GET/POST variables or cookies) to store this token on the client side
- When the client navigates to a new page, it transfers this token back to the server, which can then retrieve that client's session information
- A session ID should be hard to guess by an attacker
- Thus, a typical mechanism for issuing session IDs involves the use of a random number generator or of a message authentication code
- Note that, if the client's computer is ever compromised, then all the attacker learns is an old session ID that is likely to have expired by the time the attack occurs



Session hijacking

- Similar to TCP ***session hijacking,*** HTTP sessions can also be taken over in session hijacking attacks
- Such an attack can be especially damaging if strong authentication is used at the beginning of an HTTP session but communication between the client and server is unencrypted after that
- Performing an HTTP session hijacking attack not only requires that the attacker intercept communication between a web client and web server
- But also requires that the attacker impersonates whatever measures are being used to maintain that HTTP session



Session hijacking

- If the attacker is utilising a packet sniffer, then he might be able to discover any session IDs that are being used by a victim
- Likewise, he might also be able to mimic session tokens encoded in cookies or GET/POST variables
- Given this information, an attacker can hijack an HTTP session
- If an attacker can reconstruct a valid server-side session token, or mimic a client-side token, then he can assume the identity of the legitimate user with that token
- Thus, a first line of defence against HTTP session hijacking is to protect against packet sniffers and TCP session hijacking

Session hijacking defence

- To prevent session hijacking attacks in client-side tokens, it is important for servers to encrypt session tokens
- Likewise, server-side session IDs should be created in ways that are difficult to predict, for instance, by using pseudo-random numbers.
- In addition, it is also important for servers to defend against possible ***replay attacks***
  - which are attacks based on reusing old credentials to perform false authentications or authorizations
- In this case, a replay attack would involve an attacker using an old, previously valid token to perform an attempted HTTP session hijacking attack
- A server can protect against such attacks by incorporating random numbers into client-side tokens, as well as server-side tokens, and also by changing session tokens frequently, so that tokens expire at a reasonable rate
- Another precaution is to associate a session token with the IP addresses of the client so that a session token is considered valid only when connecting from the same IP address

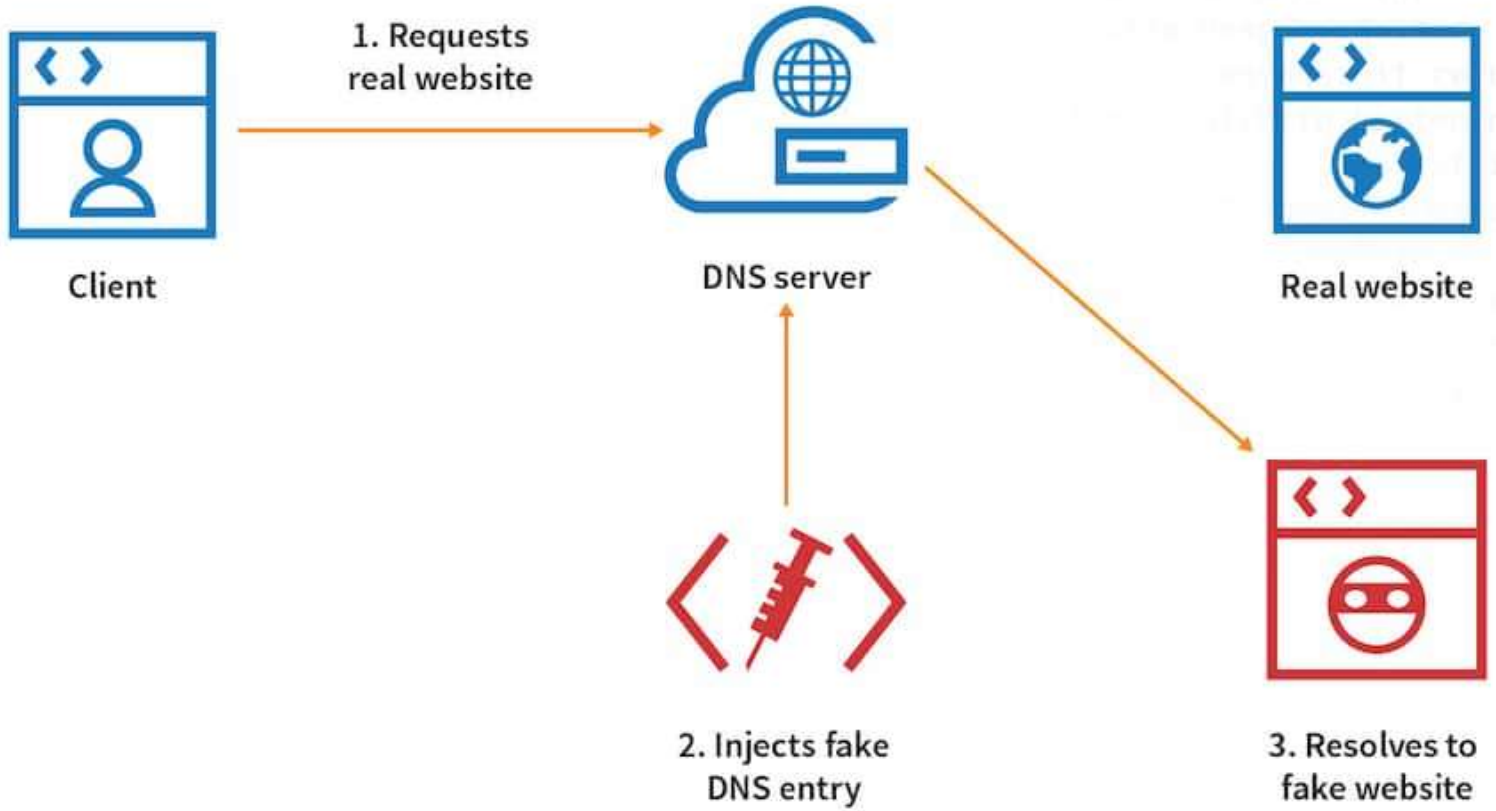
Phishing attack

- Forged web pages created to fraudulently acquire sensitive information
- User typically solicited to access phished page from spam email
- Most targeted sites
  - Financial services (e.g., Citibank)
  - Payment services (e.g., PayPal)
  - Auctions (e.g. eBay)
- 45K unique phishing sites detected monthly in 2009 [APWG Phishing Trends Reports]

Phishing attack

Phishing attack

- Phishing typically relies on the fact that the user will not examine the fraudulent page carefully, since it is often difficult to recreate pages exactly
- Also, unless the URL is falsified as a result of **DNS cache poisoning**, a simple glance at the address bar could provide clues that the site is a fake
- In addition, viewing the source code of a web site carefully could give additional evidence of fraud
- One of the most popular phishing prevention techniques used by browsers is regularly updated blacklists of known phishing sites
- If a user navigates to a site on the list, the browser alerts the user of the danger



What is DNS cache poisoning?

Akamai

URL obfuscation

- A popular technique used by phishers is to somehow disguise the URL of the fake site, so as not to alert a victim of any wrongdoing
- For instance, a simple misspelling of a URL might not be noticed by a casual user
- Likewise, spam emails that are written in HTML are often displayed in formatted fashion by most email clients
- Another trick phishers use is to include a hyperlink in the email that appears real but actually links to a phishing site
- For instance, consider the HTML source of a spam email message

```
<p>Dear customer:<br>
We at Secure Bank of Total Trust care a great deal about
your financial security and have noticed some suspicious
activity on your account. We would therefore like to ask you
to please login to your account, using the link below, to
confirm some of the latest charges on your credit card.<br>

<a href="http://phisher225.com">http://www.securetotaltrust.com</a>

<br>Sincerely,<br>
The Account Security Team at Secure Bank of Total Trust</p>
```

URL obfuscation

- One variation of this URL obfuscation method is known as the ***Unicode attack***, more formally known as a ***homeograph attack***
- Unicode characters from international alphabets may be used in URLs in order to support sites with domain names in multiple languages
  - so it is possible for phishers to register domain names that are very similar to existing legitimate sites by using these international characters
- Even more dangerous, however, is the fact that there are many characters that have different Unicode values but are rendered identically by the browser
- A famous example involved a phishing site that registered the domain www.paypal.com using the Cyrillic letter *р*, which has Unicode value #0440, instead of the ASCII letter p, which has Unicode value #0070
  - With a valid certificate, the attacker can utilise the semantic attack on websites

Click-jacking

- Similar to the idea of URL obfuscation that is used in phishing attacks, ***click-jacking*** is a form of web site exploitation where a user's mouse click on a page is used in a way that was not intended by the user
- This piece of HTML code is a simple example that creates a link which appears to be point to www.trustedsite.com.
- Moreover, this code may even provide a false sense of security to the user, since many browsers show the target URL of a link in the status bar when the user hovers the mouse pointer on the hyperlink

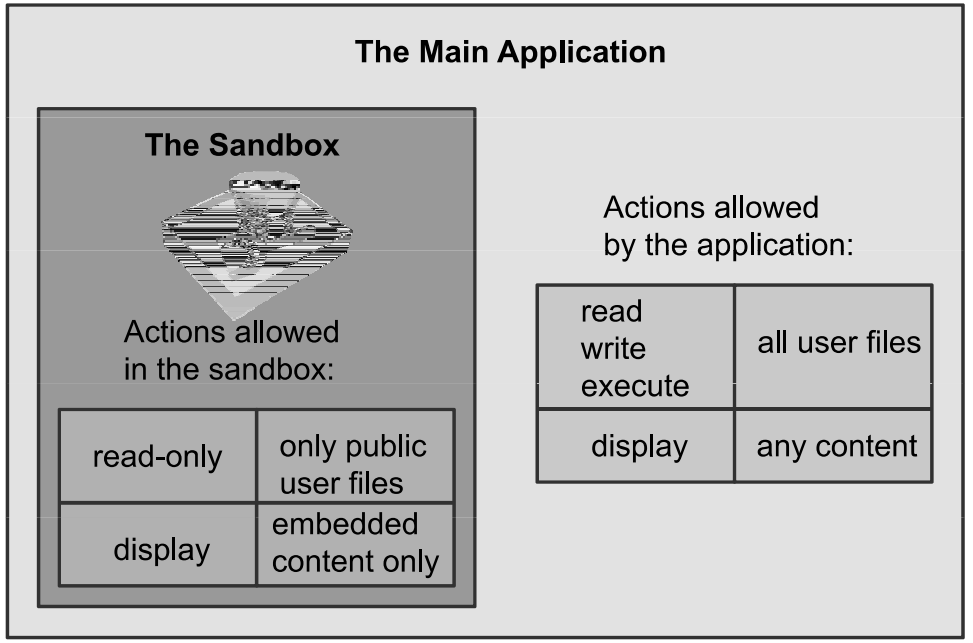
```
<a onMouseUp=window.open("http://www.evilsite.com")
href="http://www.trustedsite.com/">Trust me!</a>
```

Click-jacking

- Click-jacking extends beyond the action of actually clicking on a page, since it is possible for malicious sites to use other JavaScript event handlers such as onMouseOver
  - which triggers an action whenever a user simply moves their mouse over that element
- Another common scenario where click-jacking might be used is advertisement fraud
- Most online advertisers pay the sites that host their advertisements based on the number of ***click-throughs***
  - how many times the site actually convinced users to click on the advertisements
- Click-jacking can be used to force users to unwillingly click on advertisements, raising the fraudulent site's revenue, which is an attack known as ***click fraud***

Sandboxing

- A sandbox refers to the restricted privileges of an application or script that is running inside another application
- For example, a sandbox may allow access only to certain files and devices



Sandboxing

- Developers are often striving to create new ways of isolating code execution to reduce the impact of malicious behaviour
- For example, Google's Chrome browser runs each new tab as a new process, effectively sandboxing each tab at the operating system level
- This tactic mitigates the risk of vulnerabilities allowing browser tabs to access the contents of other tabs by creating a sandbox beneath the application layer

Adobe Flash

- Online media content can be another vector for attack
  - Increasingly, audio and video are embedded into web sites
- If an embedded media player used by a web browser to play this content has application-level flaws, malicious media files may be created to escape the sandbox of the victim's browser and execute code on the victim's machine
- This has been a recurring problem for streaming media technologies
  - One particularly popular media format is Adobe Flash (formerly known as Macromedia Flash, then Shockwave Flash)
- This technology is nearly ubiquitous, and is frequently used to create advertisements or other interactive web content
- Like all media content requiring a separate player, however, Flash presents potentials for security vulnerabilities in exploiting application flaws in the Flash media player
- Thus, one should always be using the latest version of this player, which will include patches to previously discovered vulnerabilities

Mobile code

- What is mobile code?
  - Executable program
  - Sent via a computer network
  - Executed at the destination
- Examples
  - JavaScript
  - ActiveX
  - Java Plugins
  - Integrated Java Virtual Machines

ActiveX vs Java

- Windows-only technology runs in Internet Explorer
- Binary code executed on behalf of browser
- Can access user files outside the sandbox
- Support for signed code
- An installed control can be run by any site (up to IE7)
- IE configuration options – Allow, deny, prompt – Administrator approval

- Via Java Applet
- Platform-independent via browser plugin
- Java code running within browser
- Sandboxed execution
- Support for signed code
- Applet runs only on site where it is embedded
- Applets deemed trusted by user can escape sandbox

ActiveX

- This signed ActiveX control ask the user for permission to run
  - If approved, the control will run with the same privileges as the user
- The “Always trust content from …” checkbox automatically accepts controls by the same publisher
  - Probably a bad idea

ActiveX

- Trusted publishers
  - List stored in the Windows registry
  - Malicious ActiveX controls can modify the registry table to make their publisher trusted
  - All future controls by that publisher run without prompting user
- Unsigned controls
  - The prompt states that the control is unsigned and gives an accept/reject option
  - Even if you reject the control, it has already been downloaded to a temporary folder where it remains
  - It is not executed if rejected, but not removed either

Privacy attacks

- In addition to privacy-invasive software, like adware and spyware, cookies create a number of specific privacy concerns
- For instance, since web servers set cookies through HTTP responses, if a web site has an embedded image hosted on another site, the site hosting the image can set a cookie on the user's machine
  - Cookies that are set this way are known as ***third-party cookies***
- Most commonly, these cookies are used by advertisers to track users across multiple web sites and gather usage statistics
- Some consider this monitoring of a user's habits to be an invasion of privacy, since it is done without the user's knowledge or consent
- Blocking third-party cookies does not automatically defend against tracking across different websites
- Indeed, an advertising network may have image servers hosting multiple domain names from participating websites

Privacy attacks

- Modern browsers include a number of features designed to protect user privacy
- Browsers now include the ability to specify policies regulating how long cookies are stored and whether or not third-party cookies are allowed
- In addition, private data such as the user's history and temporarily cached files can be set to be deleted automatically
- Finally, to protect a user's anonymity on the Web, proxy servers can be used
- Most modern web browsers have a "private browsing" mode
  - preventing the storage of any cookies and the recording of any browsing history while in this mode

The lecture slides can be found in the following location!