Software Vulnerability Analysis

Assignment 1

Sagar Wani (swani1@jhu.edu)
Prashanth Venkateswaran (pvenkat7@jhu.edu)
Chanakya Gaur (cgaur1@jhu.edu)

IDENTIFY ASSETS

Assets : Description

Reputation (Abstract Asset): KDE is a well known organization. Reputation of the organization is a direct asset.
Cookies (Physical Asset): Cookies stored by the browser to maintain user sessions and browsing data.
JavaScript Interpreter (Physical Asset): The javascript interpreter parses and runs the javascript on web pages.
CSS Parser (Physical Asset): The CSS parser will parse the web pages for styling and layout.
Users (Abstract Asset): The users that are using web browser to access the internet..
Saved Passwords (Physical Asset): Browser will store the user password for the specific web pages upon the request of user.
Browser Cache (Physical Asset): Browser uses some portion of memory to cache the web page data for faster browsing experience.
Browser History (Physical Asset): All the website and web links that are browsed by user in the normal mode are saved for future reference in browser history.
XML Parser (Physical Asset): This will take the XML data on the web page and parse it in the XML parser to display the contents appropriately.
HTML Parser (Physical Asset): This will take the HTML data on the web page and parse it in the HTML parser to display the contents appropriately.
URI Handler (Physical Asset): URI handler takes the input as the string of characters and identifies the resource as URL, telnet connection, ftp etc.
Frame Handling (Physical Asset): The Frame handler handles all the content displayed in the or between different frames.
Content Window or Tab (Physical Asset): The content windows or tabs contain the web pages which the user is browsing.
Browser Extensions (Physical Asset): Extensions can change the user interface of the web browser without directly affecting viewable content of a web page; for example, by adding a browser toolbar.
Secure Socket Layer (Physical Asset): It is the standard security technology for establishing an encrypted link between a web server and a browser.

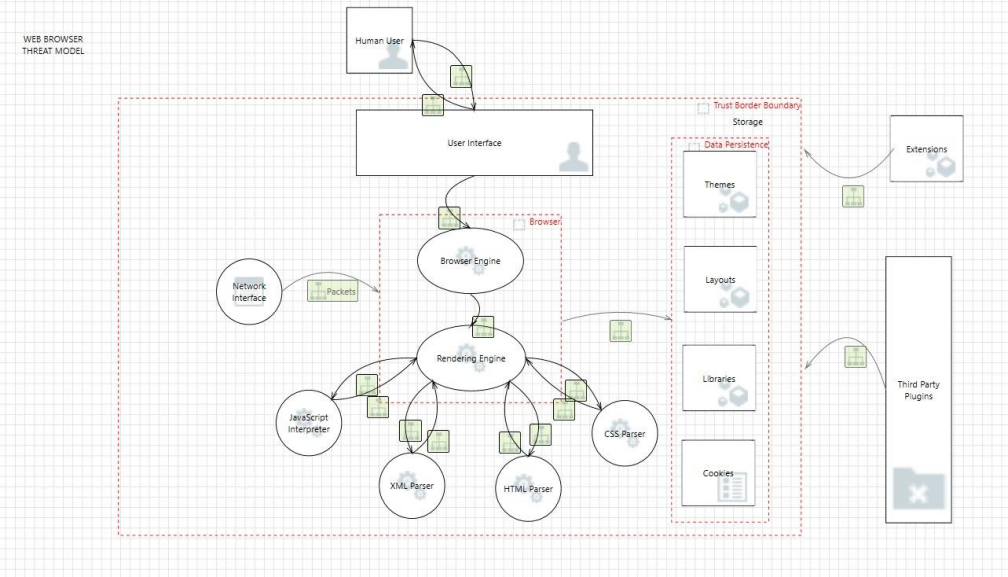
CryptoAPI (Physical Asset): It is an application programming interface that provides services to enable developers to secure applications using cryptography.

CREATE AN ARCHITECTURE OVERVIEW

Following are the basic architectural components of Konqueror:

- User Interface
- Browser Engine
- Rendering Engine
- Network Interface
- Java Interpreter
- XML Parser
- HTML Parser
- CSS Parser
- Data Persistence:
  - Layouts
  - Themes
  - Libraries
  - Cookies
- Extensions
- Plugins

The High-Level threat model can be viewed in the file name conqueror.tm7



DECOMPOSE THE APPLICATION

Category	Considerations

Input Validation	- The URL entered by the user should be validated. - Is it possible to inject the malicious code from an input field?
Sensitive Data	- Are the saved login usernames and passwords encrypted? - If yes, what kind of encryption is being used? - Is the browsing history protected? - Is the browser certificate trust store protected?
Session Management	- How are the session cookies generated? - Are the cookies protected from hijacking? - How are the persistent session state secured? - Are the session IDs being encrypted or secured? - Are the HTTP cookies being validated?
Cryptography	- Is the sensitive data stored by the browser being encrypted or not? - Are the cryptographic algorithms being used the latest/most secure ones? - How often are encryption keys recycled?
Parameter Manipulation	- Does the application detect tampered parameters? - The browser must validate all the parameters in the form fields, HTTP headers etc.
Exception Management	- The application must be programmed to handle all kinds of exceptions. - The exceptions if propagated back to client must not reveal internal methods of application.
Auditing and logging	- The application must audit all the activity of all windows and frames. - The log files generated by the application must be secured and not easily accessible.
Network Interface Management	- Network interfaces must validate if the HTTP packet is malformed/crafted. - Does the ftp/http protocol implementation allows clients to remotely connect to unknown servers?
External Extensions and Plugins	- Is the data coming from 3rd party plug-ins and extensions validated? - The extensions run with full admin privileges, so do the untrusted extensions ask for permission and warning before installation?
Parsing Validation	- The HTML parser must parse the content properly and should not allow any XSS based attacks.

	- The javascript parser must validate for any kind of malicious code invoked by third party plugin/extension. - XML parser must detect the malformed/invalid XML documents.
Storage Management	- Is the sensitive data from storage being encrypted or not? - The application must have access privileges defined to access the storage.
Secure Socket Layer	- Is the SSL protocol version up to date? - Does the SSL capability of the browser verify all the basic constraints required for signed CA/intermediate CA certificate exchange?
Memory Management	- The browser must be programmed to prevent any background third party process to either read or modify the browser memory space.
Frame & Window Management	- Are the frames in one domain can protect the content from being injected into another frame of another domain? - Similarly does the browser protect content injection from any window to any target window?
Open Source Libraries	- The HTML, JAVA, XML etc. libraries being used by the browser must be updated to the latest version.

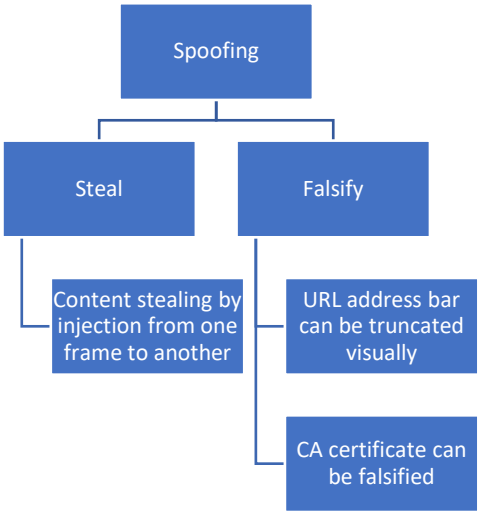
IDENTIFY THE THREATS

Spoofing

Property	Threat Definition	Examples	Mitigation
Spoofing	one person or program successfully masquerades as another by falsifying data, thereby gaining an illegitimate advantage.	1. An attacker can spoof the domain and other entries in an invalid CA certificate that seems legitimate. e.g. CVE-2007-6591 2. The URL address bar can be spoofed with the help of visual truncation vulnerability as well as window.location property. e.g. CVE-2007-4225	- Authentication based key exchange between endpoints/websites. - ACLs to block unknown or out of range Ips. - Certificate validation techniques for all connections over SSL. - Privilege levels defined for accessing and modifying any kind of source files for package.

		3. Attackers can spoof to arbitrary websites by injecting content from one window/frame to another. e.g. CVE-2004-1158	

Name	**Description**	**Countermeasures**
Spoofing	A spoofing attack is when an attacker or malicious program successfully acts on another person's (or program's) behalf by impersonating data. Attacks can be spoofed in different ways for a browser for *Example*: - injecting content from one window to another, spoof CA certificates that are legitimate as their own.	Use cryptographic network protocols: Transport Layer Security (TLS), Secure Shell (SSH), HTTP Secure (HTTPS) and other secure communications protocols bolster spoofing attack prevention efforts by encrypting data before it is sent and authenticating data as it is received. Access lists be implemented in the firewalls to block IP ranges that are unknown. This is generally done by tracking the daily network data and historical data flowing through the network.

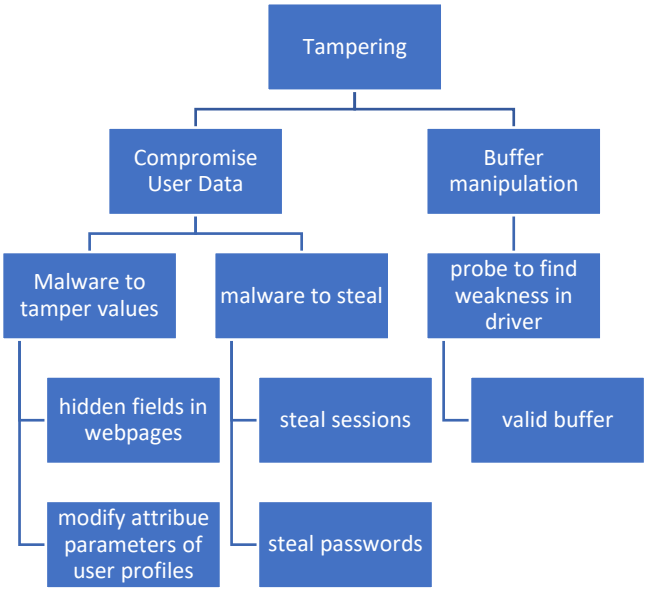


Goal: Exploit the visual truncation vulnerability to spoof the URL address bar.
Precondition: Victim must be visiting a page with a login form or with username /password fields.
Attack: 1. Find the browser with the version 3.5.7 and below.
2. Track the user activity to identify when the user visits the login page.
3. Spoof the URL address bar via an http URI with a large amount of whitespace in the user/password portion.
Postcondition: The URL address bar can be spoofed with the desired output.

Tampering

Property	Threat Definition	Examples	Mitigation
Tampering	act of deliberately modifying (destroying, manipulating or editing) data through unauthorized channels.	- Web browsers use hidden fields on forms to store status and other informations, an attacker can tamper with these values stored on the browser and change the referred information. - An attackers can modify the attribute parameters of a page from the URL bar if the users use customer profiles to log in the browser, and delete/modify the web page. - A malicious application could send a valid buffer to the driver, and then subsequently modify the data in an attempt to probe and find the weaknesses of the driver. Examples, CVE-2007-1564, CVE-2007-0537,	- Effective input field filtering. - Application firewalls. - Variable integrity checking.

Name	**Description**	**Countermeasures**
Integrity	Integrity involves maintaining the consistency, accuracy, and trustworthiness of data over its entire life cycle. Data must not be changed in transit, and steps must be taken to ensure that data cannot be altered by unauthorized people. Web browsers data can be tampered in multiple ways such as a malicious application could send a valid buffer to the driver, and then subsequently modify the data in an attempt to probe and find the weaknesses of the driver.	Web servers/certificate authorities should not give away their private SSL keys, similarly web browsers should only trust a minimal set of respected certificate authorities. Integrating strong firewalls in the system/network to ensure the data is not tampered by hash checking and session information. This ensures reliability



Goal: Cause denial of service attack using alert function of JavaScript

Precondition: Attacker must be able to invoke the alert function

Attack: Use alert function of JavaScript containing an overly large string of URL-encoded invalid Characters

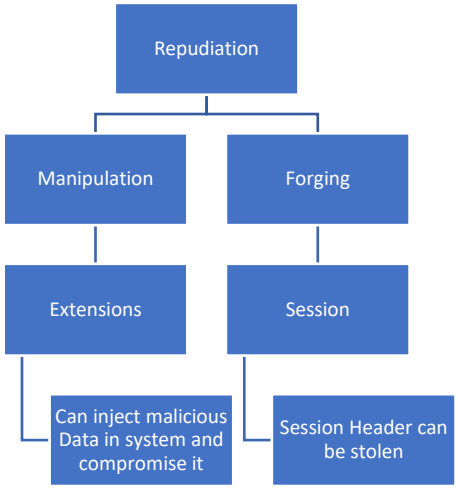
Postcondition: The browser must not be able to resolve the characters

Repudiation

Property	Threat Definition	Examples	Mitigation
Repudiation	an application or system does not adopt controls to properly track and log users' actions, thus permitting malicious manipulation or forging the identification of new actions.	- browser makes access control and authorization based on JSESSIONID, but registers user actions based on a user parameter defined on the Cookie header, an attacker can make use of local proxy and change this parameter to other username. -browser extensions if vulnerable, could allow an unauthenticated, remote attacker to execute arbitrary code with the privileges of the	- Users must be careful while using any third party extensions. - third party extensions must show a warning/permission before acquiring the privileges of a browser to perform any kind of task. -

		affected browser on an affected system. Examples, CVE-2004-0746, CVE-2004-0527	

Name	Description	Countermeasures
Repudiation	Users who deny performing an action without other parties having any way to prove otherwise.	Web browsers can ensure non-repudiation when they connect using a HTTPS connection. The browsers validate a server certificate against an existing a root CA hardcoded into the browser. Ensure that the website are connected through https.



Goal: Exploit browser extensions to execute arbitrary code with the privileges of the browser on a system.

Precondition: Attacker can execute code on browser to gain privilege access

Attack: 1. Find vulnerability in extension of the browser

2. Convince a user to visit the attacker controlled web page
3. Execute arbitrary code with the privileges of the affected browser

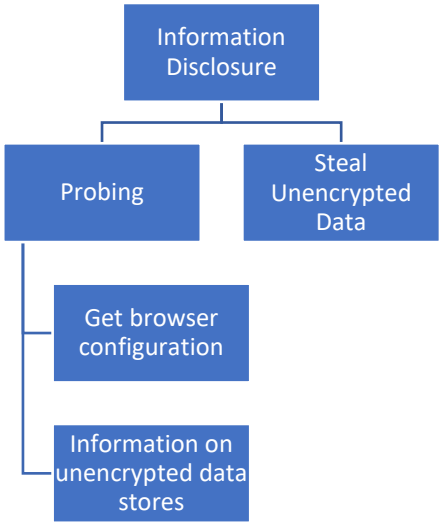
Postcondition: Arbitrary code is executed on target system

Information Disclosure

Property	Threat Definition	Examples	Mitigation
Information Disclosure	an application fails to properly protect sensitive information from parties that are not supposed to have access	- attackers can send requests to the system they are attempting to attack in order to gather more information about it. If the browser is not	- Make sure that browser does not send information that revel internal methods or information.

	to such information in normal circumstances.	well configured, it may leak information about itself, such as the server version, PHP/ASP.NET version, OpenSSH version, etc. - If the sensitive data stored by the browser is not encrypted, attacker might find it easy to steal it. Examples, CVE-2002-0862, CVE-2002-1151	- save and encrypt session data and storage - Make sure all the input is processed correctly as well as the pages parsed by the parser.

Name	Description	Countermeasures
Session Hijacking	Session hijacking deceives a server or a client into accepting the upstream host as the actual legitimate host. Instead the upstream host is an attacker's host that is manipulating the network so the attacker's host appears to be the desired destination. For Example:- Malware can intercept the HTTP Post data sent by the browser and gain form data. Or Inject JavaScript onto certain webpages to gain form data.	Use VPNs that allow secure access to corporate networks by using an encrypted tunnel through the internet. Ensure that all inputs are validated and session information are checked appropriately. Use IPSec (Internet Protocol Security) to transmit sensitive information across unprotected networks such as the Internet.



Goal: To steal cookies from the victim's browser

Precondition: Cross Site scripting protection should fail to initialize the domains on sub-(i)frames correctly

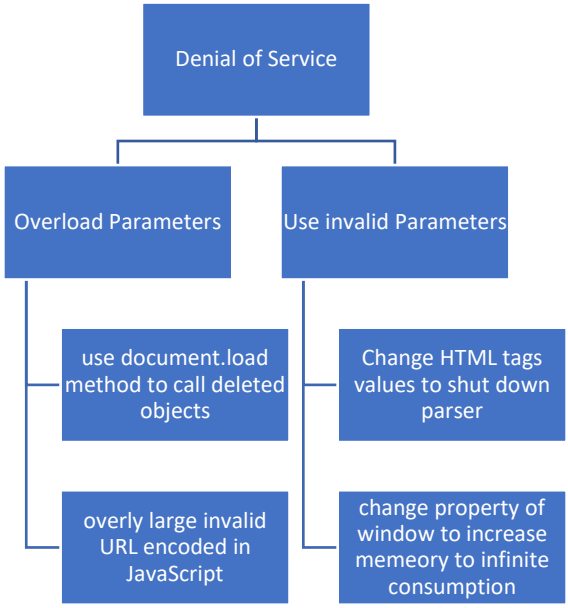
Attack: JavaScript can access any foreign subframe which is defined in the HTML source

Postcondition: Cookies should be stored in the victim's browser

Denial of Service

Property	Threat Definition	Examples	Mitigation
Denial of Service	The attacker or a group of attackers take an action to make the browser application inaccessible	- the attackers can change the property parameters of the window/frame objects that reserve memory to increase upto infinite consumption. e.g. CVE-2009-2537 - invalid use of document.load methods that trigger the use of deleted objects can also result in such condition. e.g. CVE-2008-5698 - the adversary can also trigger this condition while invoking the javascript alert containin URL encoded with large number of invalid characters. e.g. CVE-2008-4382	- Variable/Parameter integrity validation. - Defining access privileges to access the internal methods of program. - Network/Application based access control mechanism. - Traffic rate limiting.

Name	**Description**	**Countermeasures**
Denial of Services using large HTTP Cookies Parameters of the HTML Parser	Major aim in this case is not to gain access or data from the victim but more along lines of breaking the application such that it cannot be used by the victim. For Example: - The browsers can be DoSed by attacking various modules such as sending extra-large HTTP cookies. Or by sending a COLOR attribute that it is too large for the HTML Parser to understand.	Extensive Input/Parameter/Attribute verification needs to be performed before product release. By using session protection. Extensive testing of the product by validating various modules in the code.



Goal: Cause Denial of Service via large HTTP cookie parameters.
Precondition: Attacker should be able to send a cookie to the browser
Attack: Send an argument containing an over long cookie causing the browser of the victim to crash.
Postcondition: Victim's browser must accept cookies from attacker

DOCUMENT THE THREATS

Threat Description	**Attacker spoofs the URL address bar**
Threat Target	Konqueror's URI Handler
Risk	**Low**
Attack techniques	http URI altering with large amount of whitespace in username/password portion.
Countermeasures	User must use SSL all the times to encrypt the communication over channel along with certificate validation techniques for all connections over SSL.

Threat Description	**Attacker tempers javascript alert function with large number of invalid characters**
Threat Target	Konqueror's Javascript Parser
Risk	**Medium**
Attack techniques	Use of JavaScript alert function
Countermeasures	Effective input validation along with parameter integrity validation.

Threat Description	**Attacker exploits browser extensions to execute arbitrary code with the privileges of the browser on a system.**
Threat Target	Konqueror's extensions
Risk	**High**
Attack techniques	Use trusted third party extensions and show warnings before any kind of change.
Countermeasures	Sanitize and validate all user inputs

Threat Description	**Attacker tries to steal cookies from victims browser**
Threat Target	Konqueror's JavaScript Parser
Risk	**High**
Attack techniques	Idea is to use the JavaScript to steam cookie as it can access any foreign subframe which is defined in the HTML source.
Countermeasures	Use of SSL and VPN is recommended for 100% protection from such attacks.

Threat Description	**Attacker tries to cause denial of service**
Threat Target	Konqueror's handling of HTTP Cookies
Risk	**Medium**
Attack techniques	To send an argument containing an over long cookie causing the browser of the victim to crash.
Countermeasures	Use of SSL. Network/Application based Access Control along with Parameter integrity check would be able to counter such attacks.

Thank you!