Web Application Vulnerabilities

Sara Khanchi
INCS 745 – NYIT

Objectives

- Understand challenges of the Web security
- Recognize Web server vulnerabilities
- Discuss ways to protect Web servers against vulnerabilities
- Pinpoint Web browser vulnerabilities – by you
- Understand session ID exploits – by you
- List several protective measures for Web browsers – by you

Web Applications

- Internet is a collection of interconnected networks
- Users can access many different kinds of servers
- Most users are not aware of the sort of applications they are contacting
- The only time the average user is aware of a Web server is when they see error messages
  - See HTTP error messages table

Informational Status Codes

100 – Continue [The server is ready to receive the rest of the request.]

101 – Switching Protocols [Client specifies that the server should use a certain protocol and the server will give this response when it is ready to switch.]

Client Request Successful

200 – OK [Success! This is what you want.]

201 – Created [Successfully created the URI specified by the client.]

202 – Accepted [Accepted for processing but the server has not finished processing it.]

203 – Non-Authoritative Information [Information in the response header did not originate from this server. Copied from another server.]

204 – No Content [Request is complete without any information being sent back in the response.]

205 – Reset Content [Client should reset the current document. Ie. A form with existing values.]

206 – Partial Content [Server has fulfilled the partial GET request for the resource. In response to a Range request from the client. Or if someone hits stop.]

Request Redirected

300 – Multiple Choices [Requested resource corresponds to a set of documents. Server sends information about each one and a URL to request them from so that the client can choose.]

301 – Moved Permanently [Requested resource does not exist on the server. A Location header is sent to the client to redirect it to the new URL. Client continues to use the new URL in future requests.]

302 – Moved Temporarily [Requested resource has temporarily moved. A Location header is sent to the client to redirect it to the new URL. Client continues to use the old URL in future requests.]

303 – See Other [The requested resource can be found in a different location indicated by the Location header, and the client should use the GET method to retrieve it.]

304 – Not Modified [Used to respond to the If-Modified-Since request header. Indicates that the requested document has not been modified since the the specified date, and the client should use a cached copy.]

305 – Use Proxy [The client should use a proxy, specified by the Location header, to retrieve the URL.]

307 – Temporary Redirect [The requested resource has been temporarily redirected to a different location. A Location header is sent to redirect the client to the new URL. The client continues to use the old URL in future requests.]

Client Request Incomplete

400 – Bad Request [The server detected a syntax error in the client's request.]

401 – Unauthorized [The request requires user authentication. The server sends the WWW-Authenticate header to indicate the authentication type and realm for the requested resource.]

402 – Payment Required [reserved for future.]

403 – Forbidden [Access to the requested resource is forbidden. The request should not be repeated by the client.]

404 – Not Found [The requested document does not exist on the server.]

405 – Method Not Allowed [The request method used by the client is unacceptable. The server sends the Allow header stating what methods are acceptable to access the requested resource.]

406 – Not Acceptable [The requested resource is not available in a format that the client can accept, based on the accept headers received by the server. If the request was not a HEAD request, the server can send Content-Language, Content-Encoding and Content-Type headers to indicate which formats are available.]

407 – Proxy Authentication Required [Unauthorized access request to a proxy server. The client must first authenticate itself with the proxy. The server sends the Proxy-Authenticate header indicating the authentication scheme and realm for the requested resource.]

408 – Request Time-Out [The client has failed to complete its request within the request timeout period used by the server. However, the client can re-request.]

409 – Conflict [The client request conflicts with another request. The server can add information about the type of conflict along with the status code.]

410 – Gone [The requested resource is permanently gone from the server.]

411 – Length Required [The client must supply a Content-Length header in its request.]

412 – Precondition Failed [When a client sends a request with one or more If… headers, the server uses this code to indicate that one or more of the conditions specified in these headers is FALSE.]

413 – Request Entity Too Large [The server refuses to process the request because its message body is too large. The server can close connection to stop the client from continuing the request.]

414 – Request-URI Too Long [The server refuses to process the request, because the specified URI is too long.]

415 – Unsupported Media Type [The server refuses to process the request, because it does not support the message body's format.]

417 – Expectation Failed [The server failed to meet the requirements of the Expect request-header.]

Server Errors

500 – Internal Server Error [A server configuration setting or an external program has caused an error.]

501 – Not Implemented [The server does not support the functionality required to fulfill the request.]

502 – Bad Gateway [The server encountered an invalid response from an upstream server or proxy.]

503 – Service Unavailable [The service is temporarily unavailable. The server can send a Retry-After header to indicate when the service may become available again.]

504 – Gateway Time-Out [The gateway or proxy has timed out.]

505 – HTTP Version Not Supported [The version of HTTP used by the client is not supported.]

Unused status codes

306- Switch Proxy

416- Requested range not satisfiable

506- Redirection failed

HTTP protocol version 1.1 Server Response Codes

http://www.w3.org/Protocols/rfc2616/rfc2616.html

Chart created September 5, 2000 by Suso Banderas(suso@suso.org). Most of the summary information was gathered from Appendix A of "Apache Server Administrator's Handbook" by Mohammed J. Kabir.

Why the Web Is Vulnerable?

- Complexity
  - The web is a complex system with multiple layers of hardware, software, and protocols.
- Openness
  - The web is an open and public network
- Human factor
  - Humans are often the weakest link in web security.
  - Many security breaches are the result of human error, such as weak passwords or falling for phishing scams.

Why the Web Is Vulnerable?

- Insecure Software Configuration
  - Microsoft server operating systems are shipped using an easy-to-implement, but unsecured, configuration
  - Majority of network traffic on the Web is not encrypted
    - In 2012, the average time between bringing an unsecured server (or client) onto the Internet and its being infected by one of the thousands of circulating Internet worms was measured in minutes rather than hours.
  - Applications used on Web servers require very specialized knowledge to configure properly
- Lack of security training
  - Many web developers and users lack proper security training and are not aware of the best practices for securing web applications and systems.

Why the Web Is Vulnerable?

- Legacy systems
  - Many web applications and systems are built on outdated technologies and may have security vulnerabilities that are difficult to patch or fix.
- Ease of Information Distribution
  - Internet is primarily an avenue for distributing information
  - Novel exploits and newly discovered vulnerabilities are widely known upon disclosure
- Increasingly Sophisticated Hacking Tools Available
  - Network security professionals and hackers alike develop and discover new tools
    - And innovative methods of attacks that apply to new features of security systems and software

Why the Web Is Vulnerable?

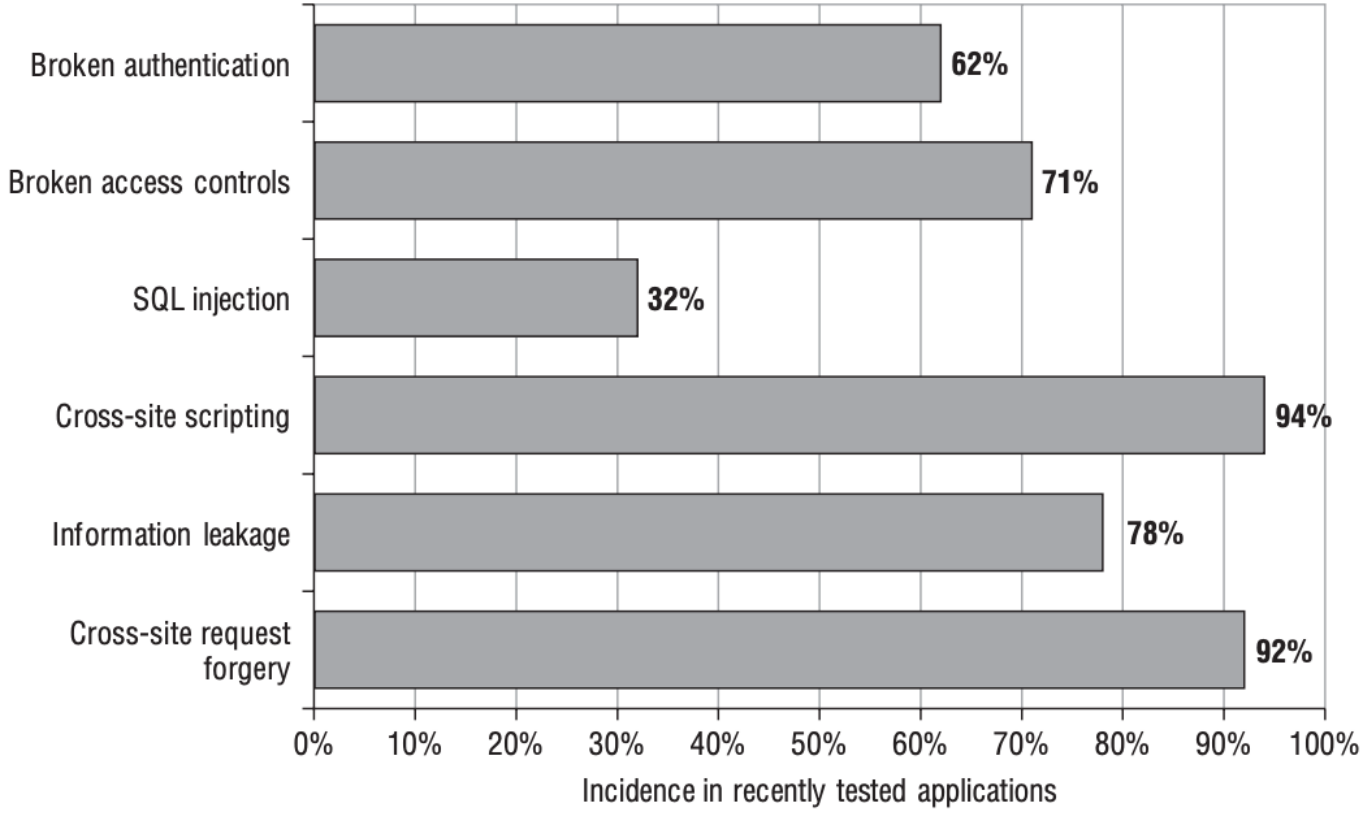
- Increasingly Sophisticated Hacking Tools Available
  - Tools used to exploit Web vulnerability include:
    - Network scanners
    - Password-cracking tools
    - Packet sniffers
    - Trojan horse programs
    - Tools for modifying system log files
    - Tools for automatically modifying system configuration files

Web Application Vulnerabilities

- If a web application is using SSL/TLS for their communication, can we say it's secure?

Web Application Vulnerabilities

- If a web application is using SSL/TLS for their communication, can we say it's secure?



Web Application Vulnerabilities

- SSL/TLS is an excellent technology that
  - protects the confidentiality and integrity of data in transit between the user's browser and the web server.
  - It helps defend against eavesdroppers, and it can provide assurance to the user of the identity of the web server he is dealing with.
- But it does not stop attacks that directly target the server or client components of an application, as most successful attacks do.

Web Application Vulnerabilities - Key Problem Factors

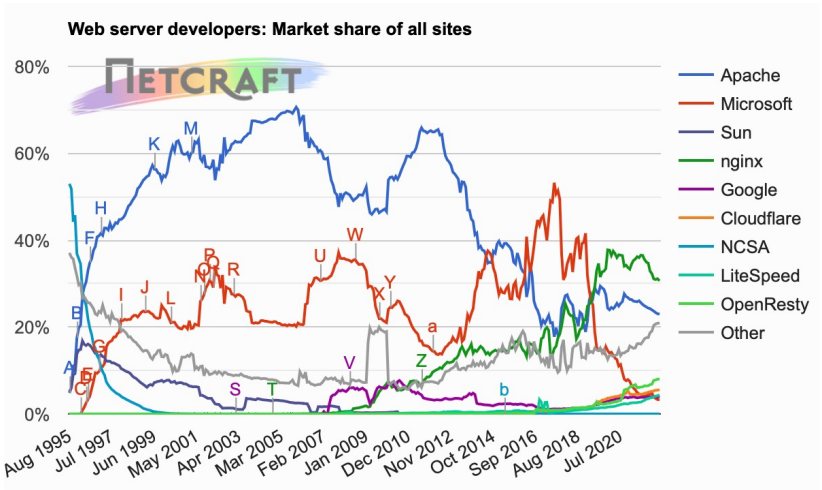
- Underdeveloped Security Awareness
  - Although awareness of web application security issues has grown in recent years, it remains less well-developed than in longer-established areas such as networks and operating systems.
- Custom Development
  - Most web applications are developed in-house by an organization's own staff or third-party contractors.
  - Every application is different and may contain its own unique defects.
- Deceptive Simplicity
  - With today's web application platforms and development tools, it is possible for a novice programmer to create a powerful application from scratch in a short period of time.
  - There is a huge difference between producing code that is functional and code that is secure.
- Rapidly Evolving Threat Profile
  - Research into web application attacks and defenses continues to be a thriving area in which new concepts and threats are conceived at a faster rate than is now the case for older technologies.

Web Application Vulnerabilities - Key Problem Factors

- Resource and Time Constraints
  - Most web application development projects are subject to strict constraints on time and resources, arising from the economics of in-house, one-off development.
- Overextended Technologies
  - Many of the core technologies employed in web applications began life when the landscape of the World Wide Web was very different.
- Increasing Demands on Functionality
  - Applications are designed primarily with functionality and usability in mind.
  - A few years ago an application designer may have been content with implementing a username and password challenge to create the login functionality. Modern sites may include password recovery, username recovery, password hints, and an option to remember the username and password on future visits

Web Server Vulnerabilities

- Some of the most important Web server vulnerabilities
  - Unsecure networks
  - Unsecure hardware
  - Threats from insiders
  - Weaknesses in site administration tools
  - Weaknesses in application or protocol design
  - Weaknesses in operating system software



These kinds of statistics are useful for hackers

- Note: layers of security is needed along the way to make the whole Web experience secure

Unsecure Network

- When the network of an organization is not secure
  - No data transmission over the Internet or local area network (LAN) is secure
- Users who have access to the network
  - Can intercept messages over the network with the use of packet sniffers

Unsecure Hardware

- If the Web server hardware is not securely protected from unauthorized physical access
  - No amount of software security can protect that server's data
  - After 10 unmonitored minutes, a cracker with the right tools can reconfigure a server

Threats from Insiders

- Most effective computer crime originates within the organizations targeted
- Motives include boredom, idle curiosity, the challenge, revenge, or financial reward

Weaknesses in Site Administration Tools

- Web sites are designed to be dynamic
- A server upon which Web sites are hosted is regularly monitored
- If you administer your server locally, it is simple to keep your administration tools secure
- Configuring the server to allow for remote administration, or allowing multiple administrators to host from multiple sites, can increase the challenge of securing the admin tools.
  - The easier you make it for authorized users to access their sites, the easier it becomes for unauthorized users to access pieces of the Web server

Weaknesses in Application or Protocol Design

- At the time that software is designed, security is often not of the highest priority
  - Intending to address security controls at a later stage of development
  - This strategy typically produces software that presents unexpected vulnerabilities
- If a protocol has a fundamental design flaw, then it is vulnerable to various exploits, essentially forever
  - E.g. Buffer overflow in C/C++

Weaknesses in System Software

- All operating system software has vulnerabilities
- System software is very complicated
  - And intended to supply the base for all subsequent application layer and presentation layer software
- System software is the foundation upon which the software is laid
- The same issues of security as an afterthought apply to system software
  - As they do to application software

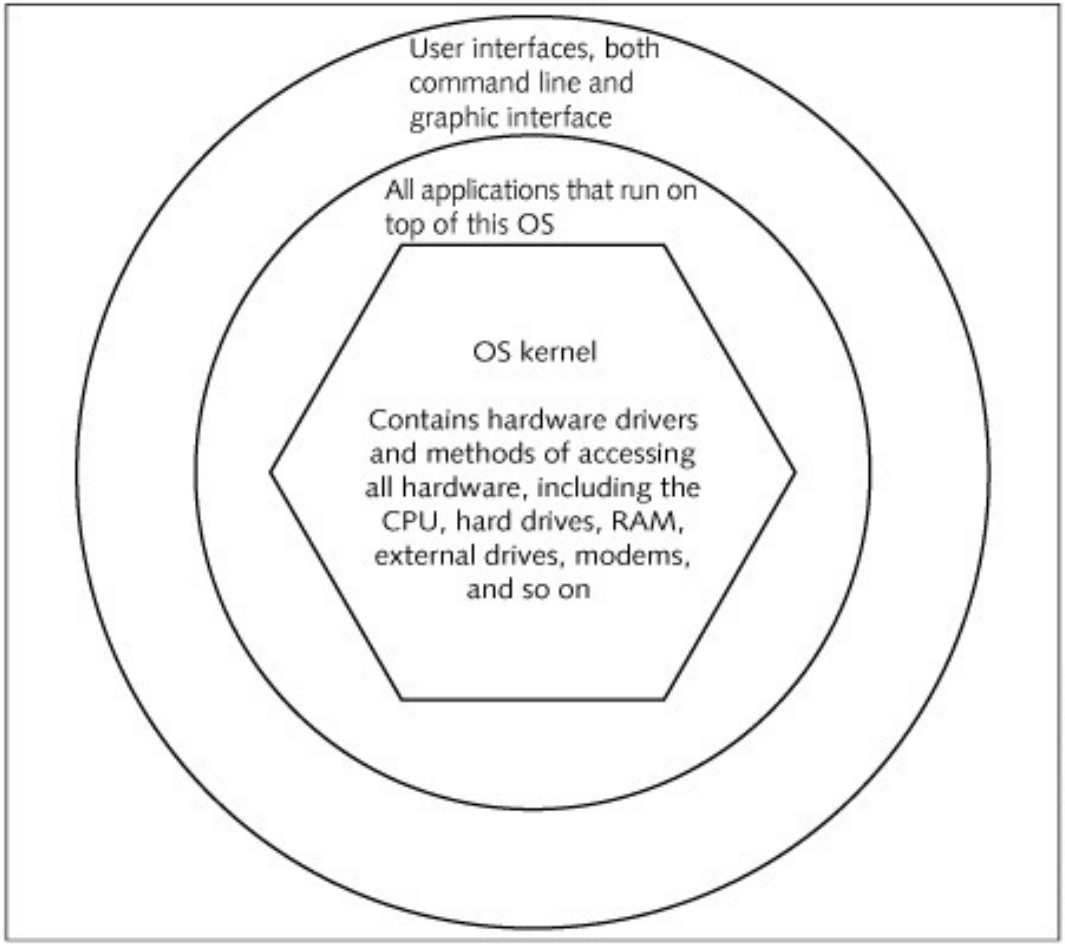


Figure 15-1 Computer architecture

Weaknesses in System Software

- Coding Vulnerabilities
  - API abuse
  - Access control vulnerability
  - Authentication vulnerability
  - Code permission vulnerability
  - Code quality vulnerability
  - Cryptographic vulnerability
  - Environmental vulnerability
  - Error-handling vulnerability
  - General logic error vulnerability
  - Input validation vulnerability

Weaknesses in System Software

- Implementation Vulnerabilities
  - Improper Web server access configuration
    - A Web server is itself treated as a user of the operating system and hardware on a physical server
  - Administrative privileges
    - All users are able to access all files on the physical server
  - Default user accounts
    - Network protocol analyzers can return to an attacker the list of users who have never logged in.
  - Misconfigured file permissions

Protection Against Web Application Vulnerabilities

- This section describes protection methods for
  - The physical server
  - The network architecture
  - The operating system on that server
  - The Web server application

Securing the Operating System and the Web Server

- Place your Web server in a demilitarized zone
- Demilitarized zone (DMZ)
  - A neutral zone between the private LAN and the public network of an organization
  - Designed to prevent external users from gaining direct access to any internal servers
  - Protects LAN from the possibility that your Web server will be hacked by some insider or some outsider

Securing the Operating System and the Web Server

- Security measures
  - Check for all default configurations in the operating system and in the Web server
  - Dump any default user profiles
  - Shutdown or even uninstall any services that the server does not need to be running
  - Modify user groups to guarantee that authorized users have only as much access as they require
  - Shut down Telnet and anonymous FTP

Securing the Operating System and the Web Server

- Security measures
  - Use encrypted services like secure shell (SSH) and authenticated FTP
  - Set your network firewall to ignore HTTP connections to all ports except HTTP and HTTPS ports
  - Automate OS patch updates so that patches are installed as soon as they are available

Monitoring the Server for Suspicious Activity

- Measures
  - Learn what suspicious traffic looks like and monitor system logs for it
  - Install Snort on your server to search for signature attacks
  - Install some scripts to watch for attacks on the server
  - Use tools such as Tripwire, that can run unattended
    - Maintain integrity of password files and registry entries
  - Set tools to send an e-mail to the server administrator or a page to her cell phone

Controlling Access to Confidential Documents

- Measures
  - Limit the number of users having administrative or root-level access
  - Allow only secure shell encrypted remote administration
    - Or authenticated user access through the GUI control panels
  - Always maintain Web page on a server on the intranet
    - And make all changes to your Web pages from there

Controlling Access to Confidential Documents

- Setting Up Remote Authoring and Administration Facilities
  - Allows you to monitor all user activity on your private development machine
  - And keep a record of Web server logs on a protected machine
- Frequently remove unnecessary files from the scripts directory
  - And remove default documents

Protecting the Web Server on a LAN

- Prior to connecting the Web server to the Internet
  - Make certain it has been hardened
  - And cannot be used as a staging area to attack other computers on the network
- If the organization has several Web servers and they are maintained by different departments
  - Remove trust relationships that might exist between them

Checking for Security Issues

- Periodically, scan Web server with tools such as Nmap or Nessus
  - To check for possible new vulnerabilities
- Add a software firewall such as Zone Alarm Pro to your Windows machine
  - Monitor unexpected activities

Web Browser Vulnerabilities

- Client side issues are similar to the server side
- Physical tampering and operating system vulnerabilities do exist
  - For most users, the main focus is the Web browser
- The most common source of Web-browser exploits is physical tampering
  - The best way to avoid is to use password-protection and always lock the screen when it is unattended

Cache File

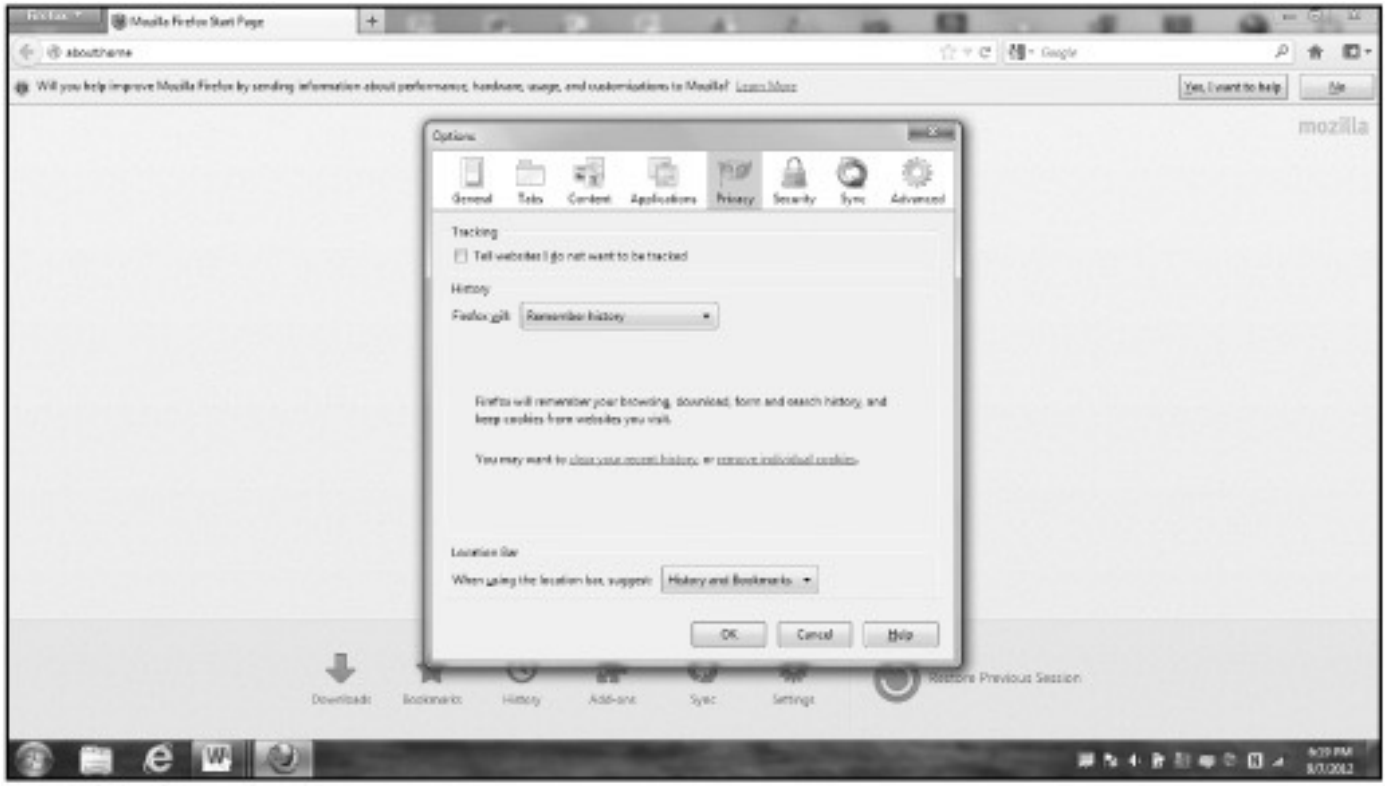
- When a Web site is accessed
  - The browser receives files from the Web server that the browser interprets
    - And presents the data to the best of its ability
- Everything accessed on the Internet is copied to a cache file
- If the file is available in the cache
  - The browser displays it in preference to displaying the file available on the server

Cache File

- The information saved in the cache files, history file, or bookmarks on a browser
  - Might pose a threat if accessed by someone intending to gather information about the user
- If your browser supports HTML 3.0 extensions and Java, and you are not properly configured
  - Your history file, cache, and other files can be copied from your hard drive
    - And directly uploaded to an attacker's server by using Java, JavaScript, or ActiveX

History File

- Allows you to view the pages you have visited in the last user-defined number of days
- Information regarding the forms you submit on a Web page is also included in the history file
- History file may include credit card details, user name, or password



Source: Firefox

Figure 15-2 Privacy tab in Firefox

Bookmarks

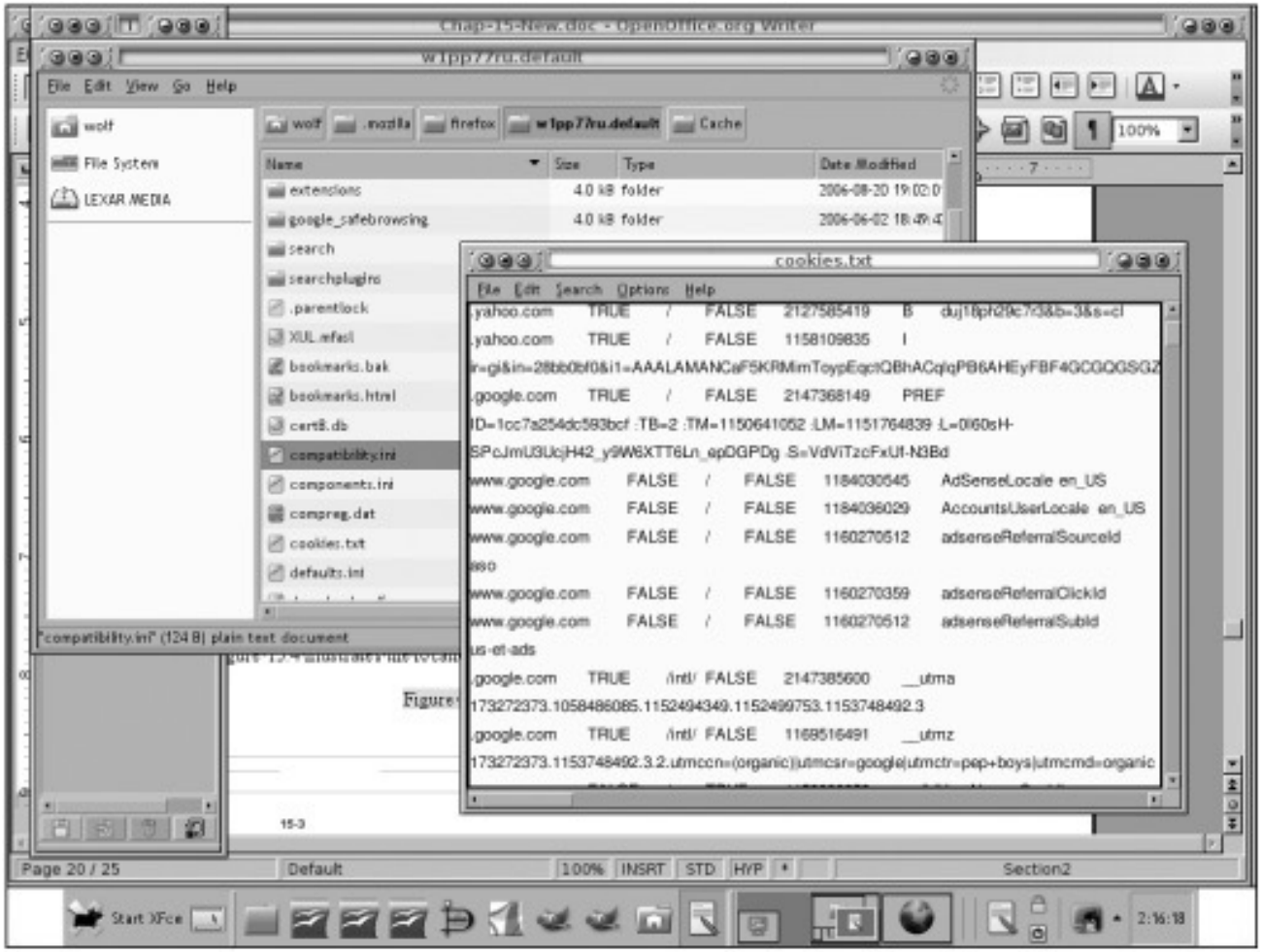
- Store information about Web pages you have visited
  - Bookmarks do not expire like history files
- If you bookmark a Web site that requires entering a password
  - You can save the username and password
- An attacker who can access your machine may be able to access your controlled-access sites

Cookies

- Cookie
  - Small text file stored on a computer by Web servers
  - Contains information about the last session when you visited the site
- Cookies store followed link information and may store username and password information
- Cookies are stored on well known directories

Cookies

- Two flavors of cookies
  - Session cookies
    - Temporary cookies that are erased when you close your browser at the end of your session
  - Persistent cookies
    - Remain on hard drive until erased or expired



Source: Linux

Figure 15-3 Linux cookie file

Location of Web Files Cache

- Cache information is located in various directories
  - Depending on the operating system, the browser, and the version of the browser
- Cache information is typically stored in a subdirectory of the Web browser's working directory
- Can change how often browser updates the cache

Browser Information

- Whenever you log onto a Web site
  - Browser automatically sends information
- Logon credentials that are sent to a Web server may compromise the privacy of a computer
- One of the sites that can be used to acquire information from the Web browser is BrowserSpy

Browser Information

- Every time a Web site is visited, the browser automatically sends the following data:
  - Host address
  - Web browser's version
  - Web browser's language
  - Files the Web browser accepts
  - Characters your Web browser accepts
  - Browser encoding
  - Username
  - HTTP port of the computer

Browser Information

```
GET /login.php HTTP/1.1
Host: testphp.vulnweb.com
Connection: keep-alive
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_7) AppleWebKit/
537.36 (KHTML, like Gecko) Chrome/103.0.0.0 Safari/537.36
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/
avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.9
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.9
```

Browser Information

- The following information about a computer's settings may be acquired if JavaScript is enabled:
  - JVM or Java plug-ins
  - FTP password
  - Current resolution
  - Maximum resolution
  - Version
  - Color depth
  - Platform
  - Anti-aliasing fonts

Session ID Exploits

- Once establishing a connection with a server
  - A user provides authentication information
- Session ID is generated and then sent to the client
  - Shows that the user can communicate with the server until that session expires
- Based on the session ID, the client computer is given access to a variety of services on that server

Session ID Exploits

- Sometimes, when sessions expire
  - Servers permit the same session ID to be used for the next session
- An attacker can use the same server behavior to access account details
  - By borrowing the session key and connecting to the server

Web Browser Protection

- Precautions include (cache)
  - Disable the cache, or set its size to zero
  - Set browser to clear cache every time you close the browser
    - Look into the file system to see if it is actually doing that
  - Set the History preference to save for 0 days or, even better, delete the file at the end of the session
  - Do not set vulnerable pages in your bookmarks
  - Do not save passwords or set the master password

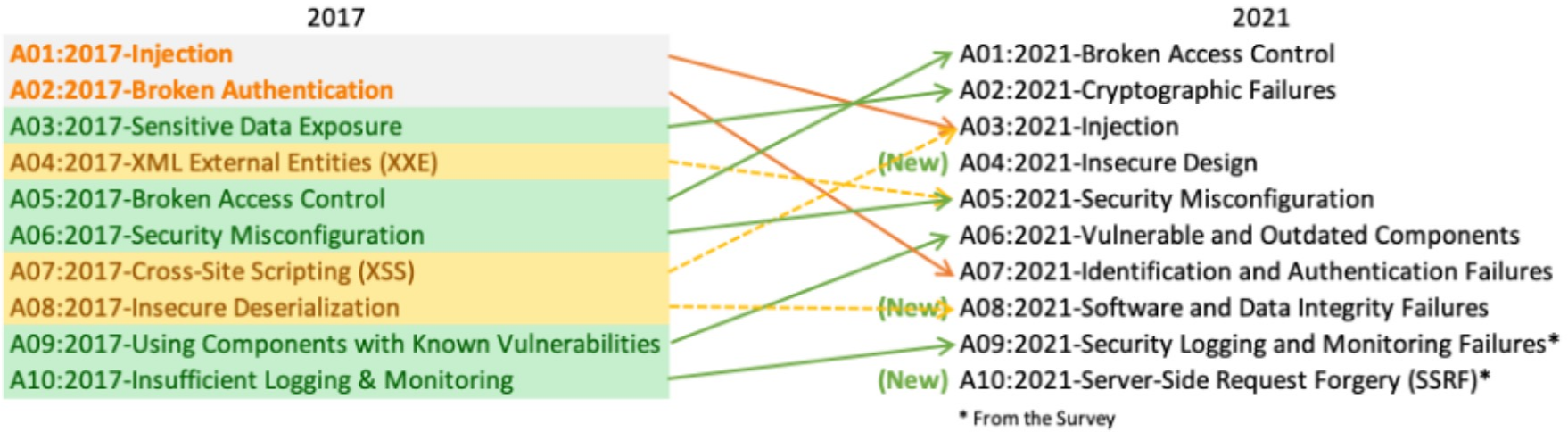
Web Browser Protection

- Precautions include (cookies)
  - Clear cookies file to remove cookies, and make the cookie.txt file read only
  - Disable JavaScript support and cookies on your browser
  - Use Firefox browser
  - Set browser to accept only cookies from trusted sites and the originating Web site
  - Set Internet security to High, requiring all scripts to ask for permission to run

OWASP Top 10 Web App Vulnerabilities for 2021

A1: Broken Access Control
A2: Cryptographic Failures
A3: Injection
A4: Insecure Design
A5: Security Misconfiguration
A6: Vulnerable and Outdated Components
A7: Identification and Authentication Failures
A8: Software and Data Integrity Failures
A9: Security Logging and Monitoring Failures
A10: Server-Side Request Forgery

OWASP Top 10: 2021 vs 2017



1. Broken Access Control

- CWEs:
  - CWE-200: Exposure of Sensitive Information to an Unauthorized Actor
  - CWE-201: Insertion of Sensitive Information Into Sent Data
  - CWE-352: Cross-Site Request Forgery
- Centered around vulnerabilities that allow a user to have access to data and application functionality that the developers did not intend
- Privilege Escalation
- Motivation: Gaining rights, accesses and control you are not supposed or authorized to have
- Prevention:
  - After logout, volumes and cookies should not be valid
  - Restricting resources on the server side

Scenario #1: The application uses unverified data in a SQL call that is accessing account information:

```
pstmt.setString(1, request.getParameter("acct"));
ResultSet results = pstmt.executeQuery( );
```

An attacker simply modifies the browser's 'acct' parameter to send whatever account number they want. If not correctly verified, the attacker can access any user's account.

```
https://example.com/app/accountInfo?acct=notmyacct
```

Scenario #2: An attacker simply forces browses to target URLs. Admin rights are required for access to the admin page.

```
https://example.com/app/getappInfo
https://example.com/app/admin_getappInfo
```

If an unauthenticated user can access either page, it's a flaw. If a non-admin can access the admin page, this is a flaw.

2. Cryptographic Failures

- CWEs:
  - CWE-259: Use of Hard-coded Password
  - CWE-327: Broken or Risky Crypto Algorithm
  - CWE-331 Insufficient Entropy.
- Covers the display of data, data at rest, and data in transit.
- Sensitive data that does not need to be kept, should not be. Data should be protected in accordance with how sensitive it is.
- Can be done via man in the middle attack. Usually when the application has no strong encryption and Transport Layer Security
- Motivation: Identifying sensitive data bits and exploit them
- Prevention:
  - Regularly use safe and secure protocols and algorithms
  - Encryption of all data, both static and transit

Example Attack Scenarios

Scenario #1: An application encrypts credit card numbers in a database using automatic database encryption. However, this data is automatically decrypted when retrieved, allowing a SQL injection flaw to retrieve credit card numbers in clear text.

Scenario #2: A site doesn't use or enforce TLS for all pages or supports weak encryption. An attacker monitors network traffic (e.g., at an insecure wireless network), downgrades connections from HTTPS to HTTP, intercepts requests, and steals the user's session cookie. The attacker then replays this cookie and hijacks the user's (authenticated) session, accessing or modifying the user's private data. Instead of the above they could alter all transported data, e.g., the recipient of a money transfer.

Scenario #3: The password database uses unsalted or simple hashes to store everyone's passwords. A file upload flaw allows an attacker to retrieve the password database. All the unsalted hashes can be exposed with a rainbow table of pre-calculated hashes. Hashes generated by simple or fast hash functions may be cracked by GPUs, even if they were salted.

3. Injection

- CWEs:
  - CWE-79: Cross-site Scripting
  - CWE-89: SQL Injection
  - CWE-73: External Control of File Name or Path

- Occurs anytime untrusted input is used as an execution command.
  - Examples are in SQL queries, PHP queries, OS command, etc.
- Where injected: Username & password fields, textbox field, comment box, URL, etc.
- Motivation: To check if the application is vulnerable

- Prevention:
  - Using safe APIs
  - Sanitization or whitelisting

Example Attack Scenarios

Scenario #1: An application uses untrusted data in the construction of the following vulnerable SQL call:

```
uery = "SELECT \* FROM accounts WHERE custID='" + request.getParameter("id") + "'";
```

Scenario #2: Similarly, an application's blind trust in frameworks may result in queries that are still vulnerable, (e.g., Hibernate Query Language (HQL)):

```
ssion.createQuery("FROM accounts WHERE custID='" + request.getParameter("id") + "'")
```

In both cases, the attacker modifies the 'id' parameter value in their browser to send: ' or '1'='1. For example:

```
http://example.com/app/accountView?id=' or '1'='1
```

This changes the meaning of both queries to return all the records from the accounts table. More dangerous attacks could modify or delete data or even invoke stored procedures.

4. Insecure Design

- CWEs:
  - CWE-209: Generation of Error Message Containing Sensitive Information
  - CWE-256: Unprotected Storage of Credentials
  - CWE-501: Trust Boundary Violation
  - CWE-522: Insufficiently Protected Credentials
- A broad category representing different weaknesses, expressed as "missing or ineffective control design."
- There is a difference between insecure design and insecure implementation.

- Prevention:
  - Secure Development Lifecycle
  - Establish and use a library of secure design patterns
  - Use threat modeling for critical authentication, access control, business logic, and key flows

5. Security Misconfiguration

- CWEs:
  - CWE-16 Configuration
  - CWE-611 Improper Restriction of XML External Entity Reference
- Occurs anytime an insecure default setting goes ignored or a server or application is configured without security in mind
- Examples include the application returning stack traces or other default messages to the client and vulnerabilities such as Web Cache Deception
- Default Configurations on the Application
- Prevention:
  - Hardening applications and hardware
  - Check configurations from time to time (Updates)

6. Using Components With Known Vulnerabilities

- CWEs:
  - CWE-1104: Use of Unmaintained Third-Party Components

- Vulnerabilities can pop up in 3rd party code and tools. If the code is still supported, generally a patch can be applied. If it's no longer supported, a replacement or work-around may be required.

- Libraries, coding frameworks, network frameworks vulnerable functions, etc.

- Prevention:
  - Patch frequently
  - Be aware of new vulnerabilities(CVE Numbers) by always checking online, check whether any include components in your system and fix it

7. Identification and Authentication Failures

- CWEs:
  - CWE-297: Improper Validation of Certificate with Host Mismatch
  - CWE-287: Improper Authentication
  - CWE-384: Session Fixation

- A broad category that covers issues such as Credential Stuffing, Insecure Password Reset, Session Management Issues, Insufficient Password Complexity, Session Cookies, etc.

- Motivation: Either get into someone else's session or use a session which has been ended by the user or steal session related information.

- Prevention:
  - Using two or more factor authentications
  - Session expiry after idle moments

8. Software and Data Integrity Failures

- CWEs:
  - CWE-829: Inclusion of Functionality from Untrusted Control Sphere
  - CWE-494: Download of Code Without Integrity Check
  - CWE-502: Deserialization of Untrusted Data

- Relates to code and infrastructure that does not protect against integrity violations.
  - Application relies upon plugins, libraries, or modules from untrusted sources, repositories, and content delivery networks (CDNs)
  - Auto-update functionality without integrity checking

- Motivation: Unauthorized access, malicious code, or system compromise

- Prevention:
  - Use digital signatures
  - Ensure libraries and dependencies
  - Ensure that there is a review process for code and configuration changes

9. Insufficient Logging & Monitoring

- CWEs:
  - CWE-117 Improper Output Neutralization for Logs
  - CWE-223 Omission of Security-relevant Information
  - CWE-532 Insertion of Sensitive Information into Log File

- Logging and monitoring is often overlooked. Proper logging provides valuable information to developers and security teams that can be used to improve weak points.

- In the event of a breach, logging and monitoring data can be used to assist with quicker response times, reducing impact.

- 100% security is not feasible. Goal is to secure the system as hard as possible, frustrate the attacker and buy time for detection and recovery.

- Prevention:
  - Monitoring log analysis and application traffic 24/7
  - Being ready to react effectively to any security breach any time of the day and night

10. Server-Side Request Forgery

- Occurs whenever a web application is fetching a remote resource without validating the user-supplied URL

- Prevention:
  - **From Network layer**
    - Segment remote resource access functionality in separate networks
    - Enforce “deny by default” firewall policies
  - **From Application layer:**
    - Sanitize and validate all client-supplied input data
    - Do not send raw responses to clients
    - Disable HTTP redirections

Summary

- Protocols upon which the Internet rest are insecure
- Absence of a fundamentally secure infrastructure, coupled with constantly evolving user expectations, results in quick, easy, and inexpensive Web attacks
- Factors that lead to vulnerability of data and applications on the Web include weak passwords, and insecure software configuration
- Hundreds or thousands of Web server programs

Summary

- Web server vulnerabilities include an insecure network, insecure hardware, threats from insiders, and weaknesses in site administration tools
- System software vulnerabilities can be divided into two categories: coding and implementation
- Several layers require protection in relation to Web services
- Actions to take for protecting Web servers include securing the operating system and Web server and monitoring the server for suspicious activity

Summary

- Primary Web browser vulnerabilities include physical tampering, operating system vulnerabilities, and vulnerabilities inherent in the browser itself
- Hackers can learn a lot about individuals and organizations due to browser vulnerabilities
- A session ID serves as a key between a client computer and a server
- Actions to protect against various browser vulnerabilities include password-protect your screensaver, lock the screen when you are away from your computer, and disable the cache