Assignment IV - Web Security Programming Problem Set

Computer Security

1 Compile and run the program.

a) What happens if a client connects to SimpleWebServer, but never sends any data and never disconnects? Test this out with your running server. What type of an attack is this?

Doing that will make service unavailable. This type of attack is called Denial of Service (DoS).

b) Try the DoS attack described in class: See if you can download /dev/random (assuming you are running on a Linux system). Rewrite serveFile() as discussed in class to guard against this type of attack.

- We need to canonicalize and validate pathname. So, we write a new function "checkPath" to ensure target path is below current path and no .. in pathname

```
private static String checkPath (String pathname) throws Exception {
   File target = new File(pathname);
   File cwd = new File(System.getProperty("user.dir"));
   /* User's current working directory stored in cwd */
   String targetStr = target.getCanonicalPath();
   String cwdStr = cwd.getCanonicalPath();
   if (!targetStr.startsWith(cwdStr))
      throw new Exception("File Not Found");
   else return targetStr;
}
```

Then edit the serveFile() function:

Use:
fr = new FileReader(checkPath(pathname));
instead of:
fr = new FileReader(pathname);

c) Implement logging in SimpleWebServer. For each client that connects to the server, obtain information about that client and write the information in a log file.

Modify function run() as the following code.

```
public void run() throws Exception {
       while (true) {
          /* wait for a connection from a client */
          Socket s = dServerSocket.accept();
          /* then process the client's request */
          String timeStamp = new SimpleDateFormat("dd/MM/yyyyHH:mm:ss")
              .format(Calendar.getInstance().getTime());
          String socketStr = s.toString();
          buff = timeStamp+" "+socketStr.substring(7,socketStr.length()-1);
          processRequest(s);
          /* re-write log file */
         Files.write(Paths.get("log.txt"), buff.getBytes(),
                      StandardOpenOption.APPEND);
     }
}
```

Then declare `private static String buff = "";`

and add `buff = buff + ",command=" + command + ",path=\"" + pathname + "\"\n";` in processRequest() before `if (command.equals("GET")) { …`

d) HTTP supports a mechanism that allows users to upload files in addition to retrieving them through a PUT command.

• What threats would you need to consider if SimpleWebServer also provided functionality for uploading files? For each of the specific threats you listed, what security mechanisms must be added to mitigate these threats?

- The file space of the Web server could be exhausted by the attacker uploading a huge file.
  - The application should set a maximum length for the file name, and a maximum size for the file itself.
- Overwrite another file that already exists with the exact same name on the server. The new file could be used to deface the website by replacing an existing page, or it could be used to edit the list of allowed file types in order to make further attacks simpler.
  - The application should not use the file name supplied by the user. Instead, the uploaded file should be renamed according to a predetermined convention.
  - The directory to which files are uploaded should be outside of the website root.
- The new file could be used to deface the website by replacing an existing page, or it could be used to edit the list of allowed file types in order to make further attacks simpler.
  - The application should use a whitelist of allowed file types. This list determines the types of files that can be uploaded, and rejects all files that do not match approved types.
  - All uploaded files should be scanned by antivirus software before they are opened.

• Implement uploading capability in SimpleWebServer. You will need to research how to send an HTTP PUT command to the server. You also need to write a storeFile() function.

Implement as much security as you feel is needed to guard against the threats you specified above.

```
public void storeFile (OutputStreamWriter osw, String pathname)
              throws Exception {
    if (pathname.charAt(0)=='/') pathname=pathname.substring(1);
       File f = new File(pathname);
       // Only allow file that smaller than MAX_FILE_LENGTH
       // Still doesn't work for /dev/random, since it's a special file
             whose length is reported as 0
       if (f.length() > MAX_FILE_LENGTH) throw new Exception();

       BufferedReader br = null;
       FileWriter fw = null;
       try {
             br = new BufferedReader(new FileReader(pathname));
             fileType = getFileType(pathname);
             // Check if file type is in whitelist
             if(!isInWhitelist(fileType)) throw new Exception();
             // Generate new name which is not already exist
             filename = generateFilename();
             // Specify the path to store the file. The STORE_PATH is
                   outside of the website root
             pathToStore = STORE_PATH + filename;

             fw = new FileWriter(pathToStore);
             String s = br.readLine();

             // To properly defend against /dev/random attack, need to
                   impose max upload limit
             while (s != null && (sentBytes < MAX_UPLOAD_LIMIT)) ) {
                   fw.write(s);
                   sentBytes++;
                   s = br.readLine();
             }
             fw.close();
             osw.write("HTTP/1.0 201 Created");
       } catch(Exception e) {
             osw.write("HTTP/1.0 500 Internal Server Error");
       }
}
```

• Once you have logging and storeFile() implemented, launch an attack to deface index.html, that is, replace it with another index.html that you have created.

This cannot be done base on how we implement the storeFile(). Since the user's file are store outside the website root, you can't replace our index.html with another index.html.

2 What are the most important steps you would recommend for securing a new web server? A new web application?

Web Server Security:

- Update/Patch the web server software
- Minimize the server functionality disable extra modules
- Delete default data/scripts
- Increase logging verboseness
- Update Permissions/Ownership of files

Web Application Security:

- Make sure Input Validation is enforced within the code - Security QA testing
- Configured to display generic error messages
- Implement a software security policy
- Remove or protect hidden files and directories

3 What is "Cross-Site Scripting"? What is the potential security impact to servers and clients?

Cross-Site Scripting: (Acronym XSS) An attack technique that forces a web site to echo client-supplied data, which execute in a users web browser. When a user is Cross-Site Scripted, the attacker will have access to all web browser content (cookies, history, application version, etc). XSS attacks do not typically directly target the web server or application, but are rather aimed at the client. The web server is merely used as a conduit for the XSS data to be presented to the end client.

4 What are phishing and pharming? What are some ways to protect against such attacks?

Phishing and pharming are two different ways hackers attempt to manipulate users via the Internet. Phishing involves getting a user to enter personal information via a fake website. Pharming involves modifying DNS entries, which causes users to be directed to the wrong website when they visit a certain Web address.

The way can protect against phishing and pharming is

1. Check the email Carefully
2. Never Enter Financial or Personal Information
3. Identify a Fake Phone Call
4. Protection through Software
5. Never Send Personal Information through emails
6. Check Bank Details Regularly
7. Never Download Files from Unreliable Sources

5 Explore the cert.org website.

(This website are no longer exist, so we try to search from other websites)

• What are some of the vulnerabilities of web browsers discussed in the Securing your Web Browser section?

ActiveX technology and Java applet that has been plagued with various vulnerabilities and implementation issues like **Cross-Site Scripting, Detection evasion, Cross-Zone and Cross-Domain Vulnerabilities**

• What are some modes of attack used to implement a Denial of Server? What preventive measures can be implemented?

Modes of attacks for DoS come with variety ways, but there are three basics modes.

1. Consumption of Scarce Resources
2. Destruction or Alteration of Configuration Information
3. Physical Destruction or Alteration of Network Components

To prevent Dos attack, we have to consider the prevention which our organization can afford. For example

- Implement router filters
- Disable any unused or unneeded network services
- Install patch to prevent TCP SYN flooding