CSCI 3403 **Erika Bailon**
Professor Daniel Massey **Homework 06**

Questions Not In Book:

Avoiding the Top Ten Security Design Flaws:

Read the paper on Avoiding the Top 10 Security Design Flaws. The paper is attached here: Link

Question 1: *List the Top 10 Flaws in your own words, using no more than two sentences per flaw. Note this must be in your own words, no points will be awarded for simply cutting and pasting the text from the paper*

1.- Earn or give, but never assume, trust - We assume the user would not misuse the system, or assume that all the data send/received would not be compromised, and therefore we do not create enough security barriers or components, or protections and restrictions for users to access the system. Some examples of what we could improve to avoid this are expiration dates for passwords or data stored, creating better APIs avoiding the assumption that all APIs will be called in the same order, don't share all secret or cryptographic material for all the clients.

2.- Use an authentication mechanism that cannot be bypassed or tampered with - The error is to assume that the user would be the only one using the system or that the resources that are going to be shared are not compromised because of the way they would be shared. Therefore, the error also takes us to forget about using the 4 important was of authentication we know about, which are something you know, something you are, or something you have. Not following these ways of authentication will give access to user to things they shouldn't have access to. The way of avoiding this, is not trying to use or invent new ways of authentication to grant access to users.

3.- Authorize after you authenticate - This one is really important because we cannot assume a system is in possession of the user at all times after we authenticate their identity. This is, accessing sensitive systems or data on a device, after authentication of the user, should never be assumed or taken for granted. As the article said, what if I unblock my computer but then go to the bathroom and someone is around and gets into my computer and tries to collect all my passwords or accounts information, the system should be good enough to ask for another authentication method or more methods to identify my identity before granting access to all that sensitive information.

4.- Strictly separate data and control instructions, and never process control instructions received from untrusted sources - The flaws in this point, is that we need to remember that code can be injected and with things like buffer overflow or changing some APIs all the system can be compromised. That is why it is important to keep separate the executable instructions from the data that comes into the system and how it is handled. Also it is important to not trust APIs that are from an untrusted source, so constantly checking the APIs and instructions' source it is crucial in this situation.

5. Define an approach that ensures all data are explicitly validated - Following the previous point, not having restrictions on the data we receive in our system is a big mistake. It is necessary to limit the data and to always check the source and the size of it because if we don't do this, the attacker can access the system by controlling the execution flow because we allowed untrusted data to access the system.

6. Use cryptography correctly - This point can easily fall out of our hands if we don't follow the known and trusted cryptographic algorithms or implementations. It is important to not try to create new ones because we can cause attackers to have access to the system since is not secured correctly. Some errors can also be caused by not using correctly the keys and therefore is easy for the attacker to get the key value and have access to encrypted data. Another thing is not using the right algorithms to create random character or things like the salt, if random is not working, then it will be easy to find a path and crack into the keys and encrypted information.

7. Identify sensitive data and how they should be handled - When it comes to knowing which data is sensitive and which one isn't , we should never assumed and we should look into the context of the data and acknowledge the sensitivity of it. The error comes when we don't use enough boundaries or methods to authenticate the access to some data only because we don't protect the data that needs to be protected in a correct way. We cannot assume confidentiality when it comes to protecting data and also we need to be really careful when we state and declared the boundaries between systems and organizations.

8. Always consider the users - We cannot trust in the users 100 percent and assumed that the users will know how to activate and use good authentication systems to access their data. Some of the errors when it comes to letting users handle security, is assuming they care, some users do not even care about security or do not know much about it. We need to create the security for them and in an easy way that they can use it without trouble, because another error is making it too complex that is not enjoyable. Another error is not thinking that the user who's using the system is not an attacker so we don't do a good job with authentication and authorization.

9. Understand how integrating external components changes your attack surface - The error comes when we don't take in consideration that external functionality or components will access our system. When we don't take them in consideration we can forget about the risks that are inherited and we do not apply patches and updates to these external component and this puts our system at risk because then we can inherit the security weaknesses, security limitations from those external components.

10. Be flexible when considering future changes to objects and actors - This oneI think is really important because everything in the world continuously changes so it is important to also improve the security systems and algorithms, not come up with new ones, but rather re enforced the existent once that have been working for a long time. In addition, the error on this point is to assume that data will continue to be the same and that is not true since the data can increase or decrease or secrete data can be compromised. Therefore, it is important so anticipate scaling changes and that data can be compromised eventually so we need to be checking it constantly and change cryptographic properties so the attackers cannot have enough time or patterns to crack the security barriers.

Question 2: *Pick one of the Top 10 Flaws and describe how code you produced suffered from that flaw. If you believe you have never written any code with one of the Top 10 Flaws, then invent an example (not one of the examples in the paper) that demonstrates the flaw.*

Last year we created a website that will allow users to have a database of plants they possessed and our website was in charge of telling the user when to water the plants so they won't die. I remember out data was compromised for many reasons, but I am going to pick **3.- Authorize after you authenticate** to explain our error. When the user will enter the site, we would ask them for email and password and that was IT, we never ask the user to authenticate the identity again, ever AGAIN! So now I see that it is a HUGE mistake because once the usr was logged in, the database of the user's plants could be change by anyone who had access to the computer, and change their plants!!! This will cause the user to not know when to water the plants the user actually had and therefore his/her plants will die!! In addition, we never created a time frame for the user to be logged in. This is, if the user was NOT in the website for 5 minutes or something like that, it will be logged out for inactivity, WE DIDN'T do this!! and now I can see how bad this is because the website could be running for hours and the user could forget he/she was logged in and leave it running and someone could come around and change the data of the plants and then again this would case the user to kill the plants by not knowing when to water them. In addition, the attacker could be really mean and just add A BUNCH of plants to the user data base so the user will receive a lot of notifications to water plants the user doesn't have because the attacker added them easily without trouble since the attacker didn't have to input a password or some type of validation because adding plants to the user database.
In conclusion, for my really bad website, not authenticating after authorizing access to the website could result in a catastrophic killing of plants because the attacker could access the database to add or delete data from the database without any problem because we never checked the identity of the user again.

Using the SWAMP: Watch the video on the SWAMP: Link

Stallings and Brown Chapter 16 Review Questions:

Review Question 16.1: *What are the principal concerns with respect to inappropriate temperature and humidity?*

Well, the computer-related equipment comes with its own temperature, but they are designed to work within certain frame of temperature. If the temperature goes TOO HIGH or TOO LOW then

the equipment won't be able to perform adequately because it will either not be able to cool off when the temperature is too hot, causing internals components to be damaged, or if it's too cold a undergo thermal shock will happen when the computer is turned on and will cause circuit boards or integrated circuits to crack.
When we talk about humidity, if we expose the equipment to HIGH humidity for a long time, corrosion will take place. Then the condensation can threaten magnetic and optical storage media. High humidity can also cause metal from one connector to slowly migrate to the mating connector. Very LOW humidity can cause material to change shape and the performance will be affected.

Review Question 16.7: *List and describe some measures for dealing with power loss.*

According to the book, to deal with short power interruptions, we should:
- Use an Uninterruptible Power Supply - which is UPS - for each piece of equipment we have. This UPS can maintain power to some equipment for a period of minutes after the power is gone. These also work as as power noise filters and automatic shutdown devices when the battery runs low.
-Emergency Power Resource- such as a generator. This one helps during longer blackouts or brownouts because it provides power for more time.

Review Question 16.8: *List and describe some measures for dealing with human-caused physical threats.*

Most of the damaged made by humans is because of physical access control, therefore, someone of the things we can do to restrict the access to the equipment are the following:
*Restricting the access to the resource, this is, where the resource is housed. This will help to deny access to outsiders, but doesn't really help much with unauthorized insiders or employees.
*Restricting access to the resource by locking it in a cabinet, safe, or room.
*Keep the machine in a permanent vault or an objet that is difficult to move. This will help with theft but not vandalism unauthorized access, or misuse.
*A tracking device to control the power switch.
Tracking device that will alert with the equipment is being moved.
*A tracking device that can be monitored to know always the current position of the equipment.
*Surveillance system so you can see who uses/misuses the equipment.