Ruijie Zhang
student id: r82zhang
id#: 20487924

cs458 Assignment 2 written questions and exploits explanation

Question 1

a)

1. has read access, no write access

2. no read access and no write access

3. has both read and write access

4. no read access, has write access.

5. no read access and no write access

b)

1. Alice's integrity level does not change, I(file) = {secret, {US, M}}

2. Carol's integrity level does not change, I(file) = {confidential, {US, G}}.

3. I(Bob) = {confidential, {G, M}}, file's integrity level does not change.

4. Eve' s integrity level does not change, I(Carol) = {Confidential, {}}, I(X) = {Confidential, {M}}

5. Eve's integrity level does not change, I(Alice) = {Confidential, {US}}, I(X) = {Confidential, {US}}, I(Y) = {confidential, {US}}

Question 2

a) The first factor the bank uses is "Something user knows".The pin number on the credit card is something user knows while others don't. The second factor used here is " Something user has", which the phone. The bank would send a text to user's phone, which is something user has, so only the user has access to the instant message.

Besides the bank authenticates Bob, Bod also needs to authenticate the Bank. There are many phishing websites that pretends to be some publicly known official websites, and try to phish sensitive information from user, like user bank account number and password.

b) **First way**: Compromises the phone such that all the message sent and received on a particular phone can be monitored on another device. The attacker can do one of the following:

1. 'borrow' the phone for a short while to install monitor software.
2. Cloning the phone. The attacker can do it by copying the SIM card and using an identical model. When a new message came, the phone network will update both of the devices and it does not realize that there two devices.

Second way:

Use specialized phone or device that acts like a radio receiver and it will pick up all text messages of every phone nearby. This works great if the message is not encrypted. If it is encrypted, there needs to be a process to decrypt the message to read the pin. This attack method works well in an place like the airport where the attacker can be physically near the target.

Source: https://nobullying.com/how-to-intercept-text-messages/
http://security.stackexchange.com/questions/11493/how-hard-is-it-to-intercept-sms-two-factor-authentication
https://www.quora.com/Phone-Security-What-are-the-most-common-techniques-for-spying-intercepting-text-messages-iPhone

c) Most questions regular people choose are questions that related to their family, friends, work, childhood and e.t.c. These information are either public discoverable facts that can be found on Google, or can be phished out in a casual conversation. Eve can use social engineering method to attack Bob, she can just find some way to talk to Bob or his friends, family and phish out these information.

It does not make the account any more secure because the two factors used here are both "something user knows". Most regular people like Bob are not secure-conscious. Just like the password can be given out to others by Bob himself, the answer to the security question can also be given out to others by Bob himself. As long as Bob is not secure-conscious, information Bob knows can be easily phished out. The attacker can use social engineering to get the password, they can also use the same technique to get the answer to the security questions.

Question 3:

a) The least privilege is not being followed here. Because the creator of the file can give permission to other users. That means an administrator can give access of some important system files to users. That means the users will potentially have access to system files that is beyond his/her legit purposes of usage. Users should not be able to touch any important system files. So this system design does not follow the least privilege principle.

b) An unsigned program may does unexpected things or even being malicious. Since Alice is running the program as the administrator, that makes it even more dangerous as the program is having the root privilege. For example, the program could install a backdoor and since it is running as administrator, there is hardly something that is there to stop it. Now the attacker can use this backdoor to gain all the information on this computer anytime he/she wants.

c) It violates the "Permission based / Fail-safe principle". The default should be denial of service. In the case of creating a file, the default behaviour should be that it does not have any access unless it is specified.

d) It violates the "Ease of use". Because the way the system handles invalid inputs. It kills all the children and abort. It makes the user hard to use the system as any typo could make user

lose all the existing process and data, and cause frustration. Eventually, the user will be tired of this validation and try to get rid of this check and make the system vulnerable.

To fix this: the validation has to handle the invalid input gracefully. The validation program should save some important information before existing or just ignore invalid input and check again later, instead of just crushing and kill all the processes.

Question 4:

a) I would achieve the second requirement by simply eliminate the access to internet from the web server.
    1. Disconnect the web server from ethernet
    2. All computers are disconnect from the ethernet as well, only connection left is between the web server and the computers.
    3. Forbidden employees to connect their devices to the computers or the server.
    4. Don't allow wi-fi in the company.

In this way, the server and the computer forms a local network that is completely isolated from the internet. Any other one on the internet can not access this web server because the server is not connected to the internet, thus the second requirement is fulfilled.

b) i: {E U H}, {E U Internet}, {H U Internet}

ii: {E U H} , {E U Internet} , {H U Internet} , {R}

iii: {R}, {E}

iv: {E U H}, {R U H}, {E U Internet} , {R U Internet}, {H U Internet}, {R U E}

For the star shape arrangement, I will put the IDS on the router. In this case, no two computers can attack each other. In this case, no user can defeat it.

Explanation for the exploits:

first exploit:

The process:
The first exploit is attacking user nnasresfahani and post an article on his behalf. The exploit first get the cookie of by making a post request with username nnasrefahani and the password 'bluejays'. This cookie is saved on the local directory. Then, the script makes another post request using that cookie and supply the necessary content for the article.

The reason it works:

I am able to find the password for nnasrefahani is bluejays because nnasrefahani is a big fan of bluejays. He made a post that made this very clear. I guessed that it is very likely people use their favourite team name for the password, so I tried to login as nnasrefahani with password bluejays, and then successfully post an article n his behalf.

Which vulnerability it represents:
This represents the A6 vulnerability of the OWASP top 10 vulnerabilities, which is sensitive data exposure. The user's uses favourite team's name as the password, that makes this a sensitive data. The user talks about his favourite team publicly, that makes this sensitive data open and public. The better approach would be, when the user sets the password to be "bluejays", the administrator of the system should later try to alert the user that this password is not secure.

second exploit:

The process:
This second exploit is attacking user Bob and post an image on his behalf. The exploit first get the cookie by making a post request with username to be Bob and the password to be 'aaaaaa'. The script makes another post request to post the image.

Why it works:
I am able to get the password of bob because I went to the robot.txt on this website and use the information there to find their Data.db file. I open the db with a sql database reader and then is able to find the hashed string of the password of bob. After getting the hashed result of bob's password, I used an dictionary attack which is going online and try to find if any existing hashed result matches this. Then on the site https://www.hashkiller.co.uk/sha1-decrypter.aspx, I was able to find the string is the sha1 hashed result with origin plain text being 'aaaaa'.

The vulnerability is represents:
This vulnerability represents the A4- Insecure Direct Object References of the 10 OWASP web vulnerabilities. The Data.db file is crucial component that should not be exposed, however I can easily download the database file without anything stopping me. The insecure expiation of this file leaded to this attack, to fix that, we should not expose any internal key objects.

Third exploit:

The process:
The third exploit is attacking user uhengartener and post an link on his behalf. The exploit first get the cookie by making a post request with username to be **test' or 1=1 —** and the password to be a random value. After that, the user can use this cookie to post an link on be half of the uhengartner.

Why it works:
This attack is using sql injection to bypass the the authentication process. The username string is **test' or 1=1 —** and this will work because there is no secure user input validation for the user supplied string. My guess is that the query used for the user validation is something like this:

SELECT id FROM users WHERE username = '+username+' and password = '+hashed passoword'

And then the system check if the result query's has at least 1 tuple in the result, if yes, it uses the first user to log in, else, the validation failed.

Now the first ' in the username string is to complete the first quote in the query string. The **1=1** makes the result query always have result because 1=1 will always be true. And the **—** make everything after it become a comment so that the query string will not result in a syntax error. In this way, all the users will be in our result query because 1=1 is true all the time. And then the system use the first user as the one we are logged in as, because there are usually one tuple in the result query. The first user in the database is uhengartener, and thus we are able to be authenticate as uhengarterner and post an article on his behalf.

Which vulnerability it represent:
It represents the vulnerability A1- injection in the OWASP top 10 vulnerabilities. Because we are doing sql injection to bypass the authentication. To prevent this, we need check the data before we execute it in the sql database.

Fourth exploit:

The attack process:
The fourth exploit is attacking user mmazmudar and make a down vote on the first post on his behalf. The script first makes a get request on the link "confirm.php?hash=deadbeef" and saves the cookie I get on the local directory. And then use the cookie to make a post request to make a post request to make the down vote.

Why it works:
There is an post on the website that reveals some information about the authentication process. If an user has not been confirmed, then we can use make an get request to the confirm.php with the correct parameter, then we will be able to login as that in user. In this case, the user is mmazmudar and the parameter is hash=deadbeaf. we know this information from the data.db file we downloaded. After we have the cookie, we can just make a post request and make the down vote.

Which vulnerability it represents:
A2- Broken authentication and Session Management in OWASP top 10 vulnerabilities. This vulnerability is an implementation flaw that allows attackers to bypass the the password authentication. For unconfirmed accounts, there needs be better way to confirmed it other than just hitting a link. Because the attacker could also send a get request to that link and presented to be that user.

Fifth exploit:

The attack process:
The fifth exploit us attacking user cmcknigh and post a comment on the first post on his behalf. There is no cookie needed as there is no authentication on posting the comment. To impersonate as cmcknigh, we only need to make our post request and set uid = 3, where 3 is the id in the users table for cmcknigh.

Why it works:
Apparently, there is no validation process on post the comment. The posting comment identifies a user by the uid passed in the post request body, which everyone can do. So there is no authentication here, that is why the attack is successful.

Which vulnerability it represents:
“A2- Broken Authentication and Session management” in OWASP 10 web vulnerabilities. Clearly, there is no authentication at all in the post comment section and attackers can impersonate any user to post a comment and that is easily done by changing the body of the post request. To properly fix that, we need to add authentication on the post comment feature.