SCC.363 Security and Risk Coursework Part 2

Aleksandr Goidin (id:34833455)

Risk analysis and security measures for your AAA service

Asset 1

The first asset of PHR (Patient Health Record) system is client application. This applies for all employees – regulator, staff, and admin as well as patients. There are many different threat sources that can lead to unauthorized access into the system and manipulating users or stealing the personal information of patients. Some of them might be hackers with the intension to manipulate or steal information, ex-employers who can accidentally access the system with no longer permission to do so or patient lost the paper first-time password given by admin.

Evaluation asset

NR	Threat	Existing countermeasure	Impact	Likelihood	Risk
1	Unauthorized access using brute-force + dictionary attack	Strong password policy	High	High	High
2	Unauthorized access using rainbow table attack	Storing hash of the password + unique salt for each user	High	Medium	Medium
3	Lost paper given by admin	Password changing after first login	Medium	Low	Low
4	Ex-employer misuse	Admin can easily remove all user data from the database	High	Low	Low
5	Insecure Internet connection	N/A	High	Medium	Medium

Detailed description of selected countermeasures

Countermeasure 1

The system has the policy of using the password following rules: min 8 characters, upper-case and lower-case letters, numbers, and special characters. Using the strong password policy makes it much more difficult to succeed with dictionary attack and takes longer to brute-force it. If we reference to the Figure 1 below, it will take around 8 hours to brute-force the plain text combination with this policy.

Number of Characters	Numbers Only	Lowercase Letters	Upper and Lowercase Letters	Numbers, Upper and Lowercase Letters	Numbers, Upper and Lowercase Letters, Symbols
4	Instantly	Instantly	Instantly	Instantly	Instantly
5	Instantly	Instantly	Instantly	Instantly	Instantly
6	Instantly	Instantly	Instantly	1 sec	5 secs
7	Instantly	Instantly	25 secs	1 min	6 mins
8	Instantly	5 secs	22 mins	1 hour	8 hours
9	Instantly	2 mins	19 hours	3 days	3 weeks
10	Instantly	58 mins	1 month	7 months	5 years
11	2 secs	1 day	5 years	41 years	400 years
12	25 secs	3 weeks	300 years	2k years	34k years
13	4 mins	1 year	16k years	100k years	2m years
14	41 mins	51 years	800k years	9m years	200m years
15	6 hours	1k years	43m years	600m years	15 bn years
16	2 days	34k years	2bn years	37bn years	1tn years
17	4 weeks	800k years	100bn years	2tn years	93tn years
18	9 months	23m years	6tn years	100 tn years	7qd years

Figure 1

Countermeasure 2

Hashing passwords in combination with unique salt for each user means that pre-computed attacks which use Rainbow tables are not possible to perform. Rainbow table attacks usually rely on the system using just hashing algorithms thus making it easy to pre-compute hash values of any table of passwords.

Countermeasure 3

Since we use an administrator which creates accounts for other users, the users at hospital are given an initial password on a paper document. This document could be left in a public place and for that reason the system prompts the user to change their password after first successful login. After the user changes their password, leak of the initial password document does not present any risk.

Countermeasure 4

The system allows the admin user to delete old users from the database. Every succeeding login for that user will fail and all current sessions will be revoked (the user role won't be available anymore to make decisions) leading to effective revocation of permissions in case someone stops working at the hospital or his account is compromised.

Countermeasure 5

No existing countermeasure provided.

Risk acceptance

Threat Nr	**Proposed countermeasure including expected impact at the risk level.**
1	It might be unnecessary for the patients to have stronger password, because in case of unauthorized access threat agent will only get access personal info of one person. Making the password policy stricter for all employees will reduce the likelihood (from high to medium) of unauthorized access to all users/patients data. By using 10 character-long passwords will require 5 years to find the combination instead of 8 hours.
2	Adding obligatory usage of one-time password, secret word/question, 2FA or SMS authentication will add one more layer of security and reduce the likelihood (from high to low) of restricted access to the client application therefore the risk will be also reduced.
5	By using insecure internet connection in public places makes it very dangerous for data to be leaked. Even if patients would use public network, and their data would be compromised, threat agent will get access to only his personal data. However, employees are managing a huge data sets, which shouldn't be leaked it is essential that they only use private secure network to work with. This will lower the likelihood and therefore risk from medium level to low.

Asset 2

The second asset is Users Table stored in PHR database with users email, password hash and salt. This asset could be compromised by hackers who might want to access system using stolen credentials or misused by employees.

Evaluation asset

NR	Threat	Existing countermeasure	Impact	Likelihood	Risk
1	Excessive privileges	Only admins can create/delete/read users information	Medium	Low	Low
2	Compromised user credentials	No plain-text passwords are stored in the database	High	Medium	Medium
3	Entry deletion – privilege abuse	Logs record who executed an event	Medium	High	Medium

Detailed description of selected countermeasures

Countermeasure 1

All users once created are assigned to the specific role. In case of Users Table admin is the only one who can interact with that, moreover, he can only do that partially, like add or delete entry, and read some of the information and can't actually see hashed password, automatically generated hash or even modify user credentials.

Countermeasure 2

For the hacker to log-in using stolen users credentials it is essential to steal email and password to log-in. All values stored in password column are the hashed plain password plus random unique salt, which makes it more difficult for hacker to use stolen data.

Countermeasure 3

If admin may decide to abuse with his privileges (or that may happen accidently) and delete a user from the record, the logs will record all the actions, which clearly show who executed an event.

Risk acceptance

Threat Nr	Proposed countermeasure including expected impact at the risk level.
2	Although passwords aren't stored directly, email and salt are still stored in plain text which makes the work a bit easier for malicious actions. Encryption of that data shall add one more level of security and lower risk to low by lowering the likelihood (from medium level to low)
3	Although it is possible to see from the logs who has modified the users database it still doesn't save from losing the user data. One of the solutions might be to create auto back-ups of the database, which will help to restore the data after accident.

Asset 3

The third asset is patient personal data such as name, date of birth and health information. It is all stored in patients table in PHR database. That data might be misused by other employees or table data compromised by hackers or destroyed by hacktivists.

Evaluation asset

NR	Threat	Existing countermeasure	Impact	Likelihood	Risk
1	Excessive privileges	Only staff can modify patients personal information	High	Low	Low
2	Obtain private data	N/A	High	High	High
3	Change or destroy private data	N/A	High	High	High

Detailed description of selected countermeasures

Countermeasure 1

Only staff is assigned with the privilege to interact with patients table and modify patients personal data, however, even staff can't fully delete patients entries from the database. Regulators are also able to read patients personal data, but can't modify which makes it safe in terms of confidence and misuse.

Countermeasure 2

No existing countermeasure provided.

Countermeasure 3

No existing countermeasure provided.

Risk acceptance

Threat Nr	Proposed countermeasure including expected impact at the risk level.
2	Private data could be obtained using SQL Injection attacks. Most instances of SQL injection can be prevented by using parameterized queries (also known as prepared statements) instead of string concatenation within the query. Because this countermeasure prevents some of the attacks and there are still chance of successful injection the likelihood is lowered from high to medium level, therefore risk is also becomes medium.
3	In case of successful attack and destroy or modification of database it is very expensive and time-consuming to retrieve the data. One of the countermeasures might be automated creation of back-ups. So, in case the data is completely lost it can be partially or fully recovered from the back-up version, depending on the frequency of the back-ups. Because it might be consuming in terms of space to make back-ups few times a day, some up-to-date data might be still missing, therefore the impact of this countermeasure is becoming medium and not low. So, the risk reduced to medium level too.

Asset 4

The fourth asset is server, which is responsible for secure communication with client application and information exchange with database. Possible threat agents might be hackers, who might want to stop the server working or steal information while communicating with the client application or database. Also

Evaluation asset

NR	Threat	Existing countermeasure	Impact	Likelihood	Risk
1	Intercepted communication	Cryptographic signatures validation	High	Low	Low
2	Lost private key	N/A	High	Medium	Medium
3	Stop working or slow response time	N/A	Medium	Medium	Medium

Detailed description of selected countermeasures

Countermeasure 1

The client verifies the server's identity by validating cryptographic signatures by using a public-private key algorithm (RSA). Also, the communication between the server and client is encrypted using a symmetric AES encryption using shared key negotiated for the session and generated by client and transmitted in an encrypted form in which only the server can decrypt it. Encryption ensures that "man-in-the-middle" cannot read any of the communication between client and the server. Integrity checks ensure that third parties cannot add extra information or manipulate the information while in transit on an unsecured network (requests which may have been tampered are rejected on the server)

Countermeasure 2

No existing countermeasure provided.

Countermeasure 3

No existing countermeasure provided.

Risk acceptance

Threat Nr	Proposed countermeasure including expected impact at the risk level.
2	The loss of private key used for server secure communication with client will produce dramatic losses and costs for recovery. It is essential to store a copy of the key in a cloud or other secure storage, so it will be easier to recover it. In case of key duplicate the impact becomes low and risk level low too.
2	Successful DoS attack might slow down the server or even make unavailable. Strong anti-virus and anti-spyware software might help to partially solve the problem of having a malware on server. Some other methods to handle Denial of service attacks are increase the size of the TCP connection queue, decrease the connection establishment period, and employ dynamic backlog mechanisms to ensure that the connection queue is never exhausted. So, the likelihood and risk level will become low.

Additional marks

Vulnerability 1

The first vulnerability is data in database stored in plain text and even database itself is not encrypted. In case of compromising the database all data, except passwords, will be shown in plain text, which makes it an easy malicious target for attackers. Stolen data of all users and patients personal data and health records will negatively affect the reputation of organization and may jeopardize patients.

One of the first steps that should be done is encrypting the database and information stored in it, especially patients personal data. It would be still possible to steal the decryption key, however it still adds an extra layer of security.

Vulnerability 2

The second major vulnerability is not enough secure log-in process. A threat agent might get a peek at monitor and see password, or for instance, user will have keylogger or any other malware that will release the password. It will be very easy for threat agent to log-in into the system and depending on the privileges read, modify or steal some data. That will break the act of personal details enclosure and might harm users of the health organization.

On of the most effective things will be to add Two-Factor authentication. Even if the passwords will be disclosed it won't be possible to log-in since attacker don't have the access for the second type of authentication. One more layer of security will add the IP tracking of device trying to log-in. Notification will be sent to the user if suspicious actions was recorded, so user might change password for more confidence.

Vulnerability 3

The third vulnerability is string concatenation within the query to database. Most of the SQL Injections are caused because attacker can modify an SQL query to return an additional result. A successful input injection attack can give an attacker unrestricted access to an entire database. They can obtain private data, change, or destroy it. It is one of the most dangerous privacy issues for data confidentiality and might cause a big money loss for the organisation to retrieve the data.

Example piece of code from the program:

```
public void deleteUser(int id) throws SQLException {
        String query =   "DELETE FROM " + DBConstants.USERS_TABLE +
                                " WHERE " + DBConstants.ID_COLUMN + " = " + id + ";"
        Statement statement = connection.createStatement();
        ResultSet resultSet = statement.executeQuery(query);
}
```

The following code is vulnerable to SQL injection because the user input is concatenated directly into the query.

So we can change it to:

```
public void deleteUser(int id) throws SQLException {
        PreparedStatement statement = connection.prepareStatement(
                                        "DELETE FROM Users WHERE id = ?");
        statement.setString(1, id);
        ResultSet resultSet = statement.executeQuery();
}
```

The above example prevents the user input from interfering with the query structure.