Veli-Pekka Eloranta

13.1.2020

D 1.3.1 Data protection and security checklist

Table of Contents

Table of Contents

Disclaimer: Authors of this document are not lawyers and own judgement should be used when considering GDPR related topics.

Introduction

For creating great visualizations we need to collect data. This data might be usage data of the system, data from the development flow or demographic data about the users or something else. However, common concern for all data is that it needs to be kept secured. Unauthorized use of data might result in remarkable losses for the company as users will lose trust on the company and the system and thus stop using it. In the worst case, the company might need to pay compensations to parties whose data has been compromised.

This document contains a checklist of attack vectors that anyone implementing data collection and visualizations should take care of. The list is not all-inclusive, but a starting point for building secure systems. One should still analyze the system at hand and carefully consider if other attack vectors exist.

Attack vectors to take into the account

Consider at least the following attack vectors when taking care of the system's data security. This list is created basing on the top 10 application security risks list by OWASP Foundation [1].

- Injections such as SQL, NoSQL, OS, LDAP injections
- Broken Authentication
- Sensitive Data Exposure
- XML External Entities (XXE)
- Broken Access Control
- Security Misconfiguration
- Cross-Site Scripting (XSS)
- Insecure Deserialization
- Insufficient Logging & Monitoring

Checklist for starting data collection in a software system

If you collect data, here is a checklist you can use to see if you are good to go.

- If any personal data or data that can be used to identify a person is collected, you need to comply with the General Data Protection Regulation (GDPR) [2] by asking permission from the person whose data is being collected:
  - You need to explain in detail what data will be stored
  - You should provide the rationale why the data is collected
  - You should explain how the data is being used
  - If the data will be moved to external system, it needs to be explained where the data will be transferred and why it is transferred.
  - If data will be transferred from external system to subcontractors, you need to inform users how subcontractors will use the data
  - Rationalize for users what is done with the collected data
- If health related data is gathered, the system needs to ask a separate permission for that
- If new data about the user is gathered, or there are changes how the data is used, the system should ask permission for data collection again.'
- There is clear contact information of controller of the data registry.
- If a user asks for what information about him/her has been stored, there should be an easy way to find out this information
- If a user wants all data concerning him/her deleted, there should be a way to do this in timely manner
  - There should be no data that can identify the user left behind
    - NOTE: All kind of identification information should be deleted.
  - User should not be visible in log data, you should not be able to identify the user from the log entry.
  - When removing user data, note that this also concerns possible backups of database and other information.

        - NOTE: If backups are deleted anyway within reasonable timeline, it is acceptable and no further action needed. However, if backups are to be stored for a long time, user data needs to be removed from backups as well. What reasonable time means is determined case by case.
    - Data should be removed from staging environment too.
        - NOTE: Might be a good idea not to use production data in staging because of this.
- Data should not be stored if there is no business reason for it.
- To collect usage data, you need permission. Usually this is implemented with a pop up banner
- One should not make it hard for user to prevent data collection.
- Once data is gathered, you need to make sure it is kept secured.
- If it is found out that the system is hacked and user data might be compromised, there should be a way to inform users about the breach.
- In case the system has been hacked and personal data has been stolen, you need to inform data protection authorities within 72 hours.
    - If you are unable to provide notification for data protection authorities, you need to provide rationale for not sending the notification
- If your system is hacked it is good practice to inform the users, even when there is no evidence that their data got stolen.
- When writing tests or showing demos, use generated data.
- Limit access to the real data only to a few carefully selected persons. If misuses occur, it is easier to find the person who is guilty. Do not provide access to real users' data for the whole development team.

Additional tips for handling privacy

If you are using an external analytics tools such as Google Analytics or MixPanel, it is useful to use separate UUID for analytics. Do not mix this analytics UUID with users UUID. In this way, if there is a vulnerability in the analytics tool and the data will be leaked, your user information cannot be analyzed against the leaked data and in this way the users actions are harder to track.

Keep data related to a user only in one place and keep them separated from all other data. Only refer to user data only with UUID. In this way, it will be easier to remove the data when the user asks for it.

When designing the system and thinking about which demographics will be used take into account that if you have low number of users, asking a lot of demographical information will make it pretty straight forward to identify the users. For example, if you know that the user is a male, his name, his age is 38 and lives in a certain postal number, it is pretty easy to identify the person with this information and thus this information is considered as a personal data.

Design guidelines

When designing the system you can use these principles to avoid the problem of private information in the first place

- If the personal infromation is not needed for the visualization to function, it should be by default obscured or anonymized.
- Think about how much developers or customers can see from the visualization about their colleagues or competitors. You should not be able to identify persons, or companies from the visualizations.

References

[1] OWASP Foundation, website, Top 10 application security risks 2017, https://www.owasp.org/index.php/Top_10-2017_Top_10, visited 11th November 2019

[2] REGULATION (EU) 2016/679, General Data Protection Regulation, https://eur-lex.europa.eu/legal-content/EN/TXT/HTML/?uri=CELEX:32016R0679&from=EN