MockBlind: Examine Mylar and GDPR Compliance

Xunting Wang

Abstract

Social media platforms have the ability to acquire a large amount of data associated with specific users. Platforms and third-party advertising firms both benefit from using user data in direct marketing and learning user behavior. Users might not have direct control over how their data is being used and face the risk of exposing themselves to adversaries in the event of database breach. To address the above problems, this project demonstrates that Mylar, a client-side encryption tool built in MetoerJs, is suitable for allowing users to grant and revoke access to their platform specific data in a GDPR compliant way.

1. Introduction

1.1. Mylar

Mylar is a MIT research project that seeks to protect user data when an active adversary gains full control over the server. User data is encrypted on the client side by his or her private key, so not even the database administrator can gain access to user data. However, while Mylar has the ability to encrypt and protect user data, the current implementation of Mylar library in MeteorJs fails to delegate the power of defining privacy policy to the user dynamically. Software developers using Mylar are in charge of predefining a set of privacy policies.

Developers represent each access control entity by a principal, a named public-private key pair. Mylar API also requires the developers to specify fields of a document that will be encrypted and the principal whose keys will be used in such encryption. To allow sharing documents, developers also need to utilize **add_access** from Mylar API to grant access to a principal's encrypted documents [1].

Despite the users have no control over defining privacy policies, developers are able to create functionalities that return such power back to the users. To do so, developers should predefine a comprehensive set of privacy policies and create respective functionalities to allow users to choose which policy to enforce. This project will focus on **Lawfulness of Processing** of the GDPR, especially granting and revoking consent.

1.2. GDPR

GDPR, General Data Protection Regulation, is the first regulation that requires companies to protect user data by the EU, one of the largest legal entities in the world. At the heart of GDPR is Article 6 – Lawfulness of Processing. The first section of Article 6 defines that processing of user data will be considered lawful if the user gives explicit consent to such processing. Although other conditions such as processing for a legal obligation also justify processing of user data, acquiring consent is the most prevalent way of maintaining lawfulness of processing.

GDPR also specifies in Article 7 (3) that the user can revoke his or her consent at any time. This should not be confused with Article 17, Right of Erasure, since Article 7 (3) ensures that the processing prior to the revocation of consent shall not be affected by removing consent, meaning the data processor, the entity that determines means of processing and purpose of processing, is not required to remove user related data upon consent withdrawal. [2]

2. Background

Social media platforms usually provide free services, although some of them remain the most profitable companies in the world. Because of the sheer number of users on their platforms, they are able to provide data to marketing firms who seek ways to directly market to the audience who might be more interested in their products.

A blog post written by a digital marketing firm StrikeSocial, a direct beneficiary of the rapid development of social media data services, points out that the bulk of data provided by social media platforms is user behavioral **metadata**. [3] Marketing firms are not interested in the total history of a user's behavior on a social media platform. The overall trends are more worthy of their interest.

Academia has also noticed and taken advantage of user behavioral datasets provided by the platforms. A study published in AI & Society has compiled and categorized ways that social media platforms provide user data. [4] For example, Twitter provides a REST API which any entity can pay and acquire a set of user information, including username, created time, and metadata including recent tweets and types. Meanwhile, Twitter also provides a real-time streaming API, which can be used to acquire Tweets from users filtered by geographic locations or keyword.

Despite such innovations by social media platforms have sparked the birth of new social science research technologies such as sentiment analysis, there are incidents where large sets of user specific data have been abused. For example, Cambridge Analytica has acquired information of millions of Facebook users and caused one of the most controversial presidential election in U.S history. As the users become

more and more educated on their rights of privacy online, social media firms need to think about ways to return the power of privacy policy back to the users.

3. Design

As explained in section 2, marketing firms are more interested in the user's metadata. A sample application is built to demonstrate how developers can take advantage of Mylar's API to protect user's metadata and grant users the power to control who can use their data.

MockBlind is a simple mock-up of a popular social media network called Blind. Any user can ask questions in specific sections and the platform will maintain metadata that keeps tracks of the section in which user has posted content and the number of times the user has posted.

```
Sections {
  sectionName: the name of section
}

Messages {
  sectionName: name of section where this message exist
  message: text content of the message
  createdBy: the user who created this message
  time: the time user created this message
}

Summaries {
  blindUserId: unique identifier of user
  blindUserName: username of user
  blindUserPrinc: private-public key pair of user
  invitediD: list of users that can peek this summary
  summaryItems {
    section: section name
    count: number of times the user has posted in the section
  }
}
```

Figure 1: Design of database

3.1. Design of database

Figure 1 shows a view of the database design. Since Mylar is built on top of MeteorJs, which uses MongoDB as the underlying database, each of the documents in Figure 1 corresponds to a collection in MongoDB. Meteor has provided a convenient API to create user, which automatically creates a user collection in the underlying database. The only thing reader should know from the user collection is that Mylar adds fields **_pk**, **_wrapped_key**, and **blindUserPrinc** upon calling **create_princ** Mylar API. The field _pk is the public of the user and _wrapped_key is created when another principal has been granted access to the owner's encrypted data.

The only encrypted field of the database is **summaryItems** in Summaries collection. Mylar's principal library will intercept any data insertion, modification and deletion related communication between Meteor client and server, and encrypt the developer specified fields, in this case summaryItems, with the owner's private key. When any individual examines the underlying MongoDB database, he or she will not be able to find a summaryItems field in Summaries. Instead, a **summaryItem_enc**, the encrypted version of summaryItems, will be found.

3.2. Design of user interaction

To simulate the interaction between MockBlind users and marketing firm users, any client can choose to be a MockBlind user or an AdService user upon creating his or her account.

MockBlind user can create section, access all sections on the platform, post in all sections and grant and remove access to their post summary. AdService users' actions are much more limited, to mimic real interaction with social media platforms through API call. First, AdService users will request access to a MockBlind user's summary. Access will only be granted to them upon a mouse click event on "accept request" button by the MockBlind user. Once the request is accepted, MockBlind users are able to view who currently has access to his or her summary and can revoke such access upon clicking "remove access" button on their dashboard page.

Consent and withdrawal of consent specified in GDPR are accomplished here, since MockBlind users have the knowledge of which marketing firms have access to their summary, and they are able to explicit grant and remove consent by clicking certain buttons.

3.3. Potential Attacks

SummaryItems field contains the metadata of interest. As discussed in 3.1, no individual will have access to the unencrypted version of a user's post summary. However, summary of a user is computed over the user's posts over each section, which are not designed to encrypted.

Potentially, an adversary paid by the marketing firm could create a fake user account, which has access to all posts, and scrape the posts of a user. This is indeed a problem in this sample application, but also found in nearly all social media platforms. Solutions to mitigate this problem include adding anti-scraper logic for the website and limiting the number of requests per second per IP address. Furthermore, if the application has a large number of users, gaining metadata of a single user might become significantly time consuming.

4. Implementation

This project is implemented with MeteorJs and Mylar. Data representation, referred to by Figure 1, is located in model.js,

which is shared by both the client-side and server-side logic. Client-side logic consists of component manipulation in HTML template and API call to the backend. Since creating principal, adding access and removing access are both front-end logic in Mylar's API, the client-side code issues multiple API calls to Mylar's front-end library. Account creation is also done on the client-side. Server-side logic includes updating specific fields in Summaries and creating new entries in Messages. For more information, please refer to the GitHub repository of this project.

5. Evaluation and Future Work

5.1 Developer Effort

Since MeteorJs offers a convenient library for account creation, no significant development effort is spent in adding user to the system besides code for adding principals. The client-side logic consists of 328 lines of code, among which 38 lines of code are related to Mylar's API calls. On the server-side there is no significant portion of calls to Mylar API. For the code in charge of data representation and object mapping, 7 out of 86 lines are used to specify fields to be encrypted by Mylar.

For future work, developers can choose to encrypt messages as discussed in potential attacks in section three. This will likely incur more overhead in developer effort and performance. When a new MockBlind user joins the network, all users have to update their wrapped keys and re-encrypt their posts with the newly updated wrapped keys. On the other hand, if the developer chooses to add more metadata in Summaries, no significant overhead will be added, since the developer only needs to add one-line to the code that specifies which fields need encryption.

5.1 GDPR Compliance

To evaluate GDPR compliance, inspection of database is needed. A simple experiment is conducted to illustrate GDPR compliance:

1. MockBlind User "A" accepting AdService User "B" request for accessing summary
2. "A" grants access
3. "B" views user summary
4. "A" User removes access

After step 4, the view of A's post summary was removed from B's dashboard. When inspecting the underlying MongoDB database, B has been removed from the "invitedID" fields of A's entry in Summaries. At any point of the above experiment, summaryItems of A's entry are constantly encrypted.

6. References

[1] Raluca Ada Popa, E. S. (n.d.). Building web applications on top of encrypted data using Mylar.

[2] *REGULATION (EU) 2016/679 OF THE EUROPEAN PARLIAMENT AND OF THE COUNCIL*. (n.d.). Retrieved from https://eur-lex.europa.eu/legal-content/EN/TXT/HTML/?uri=CELEX:32016R0679#d1e2172-1-1

[3] Miller, J. (n.d.). *How do social media platforms handle and protect user data?* Retrieved from https://strikesocial.com/blog/how-do-social-media-platforms-handle-and-protect-user-data/

[4] Treleaven, B. B. (n.d.). Social media analytics: a survey of techniques, tools and platforms. *AI & Society*.

[5] Confessore, N. (n.d.). *Cambridge Analytica and Facebook: The Scandal and the Fallout So Far*. Retrieved from https://www.nytimes.com/2018/04/04/us/politics/cambridge-analytica-scandal-fallout.html