Chatbot Security

Jiang-Hao Chen
Group 3
b07902013@ntu.edu.tw

Po-Heng Chen
Group 3
b07902114@ntu.edu.tw

Shih-Han Chang
Group 3
b07902111@ntu.edu.tw

Yi-Min Lin
Group 3
b07902016@ntu.edu.tw

Abstract

Chatbots are increasingly popular among texting applications like Line, Messenger, etc. Though chatbots are convenient and user-friendly, it's notable that chatbots actually have access to all plaintexts in a chat room, and thus users may unintentionally leak their sensitive information, e.g., cellphone numbers, credit card ID, etc., to chatbots, violating the **confidentiality** of security requirements. In this project, we propose several possible solutions including chatbot authentication, application-side terms of privacy and alert and message modification, and these methods can be adopted together with each other. Our solutions can mitigate the security problem of chatbots with little effect on the functionality of chatbots.

1 Introduction

1.1 Introduction to chatbots

Chatbots are programs that read users' inputs and make corresponding responses. In recent years, the number of chatbots increases with the rapid development of the social network, such as Facebook or Line. Chatbots can either finish specific tasks for users just like typical applications or simply chat with users. The reason users would like to use chatbots is that users can have their tasks done via a more user-friendly way like natural language rather than rigid user interfaces provided by the application designer.

1.2 Categories of chatbots

In this project, we categorize the chatbots from two perspectives. Based on the way of interaction between users and chatbots, we can categorize chatbots into 3 categories.

Menu/Button Based. To interact with this type of chatbots, users click on buttons or menus to send messages to chatbots. This type of chatbots is similar to applications, but menu/based chatbots respond to users more friendly.

Keyword Recognition. To interact with this type of chatbots, users send text messages to chatbots, and chatbots check whether there are any keywords exist in the messages. For example, a fortune chatbot may recognize names of zodiac signs and gives corresponding fortune today.

contextual analysis. To interact with this type of chatbots, users also send text messages to chatbots, and chatbots analyze users' input via natural language processing techniques and make the corresponding response. For example, user may input "What's the fortune of Gemini this week?", and the chatbot will know the intent of user and responds to user correspondingly.

Based on chatbots' functionality, we can also categorize chatbots into 3 categories.

Information-providing. This type of chatbots provides information required by users' input like weather condition, bus schedule, etc.

Task-oriented. This type of chatbots helps users complete specific tasks like booking tickets, transferring money, etc.

Chit-chat. This type of chatbots simply chats with users. For instance, chatbots may greet to users, tell jokes, discuss on whatever users are interested in, etc.

1.3 Security issues of chatbots

Users send messages to ask the chatbot to do the corresponding task, and chatbots will have plaintexts of the messages. Suppose users accidentally sends a message containing their personal information, such as the credit card number or the PIN, the chatbot can get these information even though they are not needed to complete the task. Besides, establishing chatbots is easy nowadays, and the authentications are not strict. In other words, the attackers can establish their own malicious chatbots effortlessly. Even worse, the attacker can collect several users' personal information once he gets control of the chatbot.

Hence, to solve the chatbot's security issue, we propose frontend and backend protection. In this project, our contribution contains four parts. 1.) Raise awareness of chatbot privacy issues 2.) Propose privacy protection methods 3.) Implement a prototype for our methods 4.) Evaluate the effectiveness of each method.

2 Problem definition

2.1 Privacy definition

To define chatbot security, we first define what kind of information is concerned about privacy when sending messages with chatbot. Some researches [9, 13] has pointed out that a lot of people regret sending messages because the messages expose too much personal or sensitive information. Thus, we think the information we want to protect is somehow "personal".

PII. Here we use personally identifiable information (PII) to define the "sensitive information". Any situation that chatbots can get user's PII that is irrelevant to the service it provide is considered a violation of chatbot privacy. PII indicates the information which can necessarily identifies a person. According to the NIST definition[7], we can categorize PII into some categories:

Table 1. PII used in our project

Category	PII
Personal	Name, address, phone number, email
Financial	Bank account number, credit card number, PIN
Technical	Username, password, IP address, URL

2.2 Threat model

In this section, we define the threat model. We assume messages from users should first be sent to a trusted server. When the server receives messages, it can operate on original messages before sending them to the chatbot. There are two threat models.

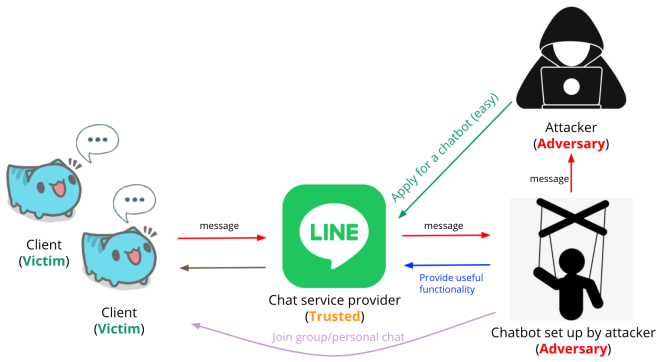

Figure 1. Malicious Chatbot

As **Figure 1** shows, an attacker can establish a chatbot and pretends to be a normal chatbot. However, users do not know whether the establisher of the chatbot is malicious. As mentioned before, the attacker can see messages from users. In other words, users may unintentionally reveal their personal information to the attacker.

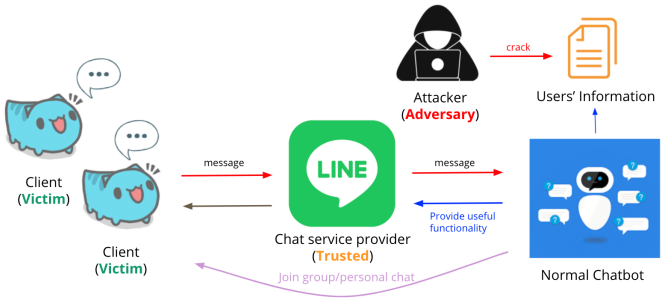

Figure 2. Attacked Chatbot

As **figure 2** shows, to provide a better-personalized service, the chatbot may record the conversation or users' information, such as mapping UUIDs to phone numbers. Therefore, even though the chatbot itself is not malicious, the attacker can get personal information by cracking into the records of the chatbots.

2.3 Current problems about the chatbot

The problems about the chatbot nowadays mainly contain two parts. One is the establishment of the chatbot lacks strict authentication. For example, everyone can establish a chatbot in Line app, which means users should consider the chatbot's background. The other is that messages from users are forwarded to chatbots without any processing. In other words, messages unrelated to chatbots' services are also forwarded to chatbots directly, and chatbots do not need most of the information. Hence, it is vital to solve these problems to prevent users from revealing their privacy.

3 Results

3.1 System Design

Figure 3 shows our chatbot security design. In our proposed system, all messages are forwarded by the server when clients and chatbots communicate with each other. We divide the system into three parts, frontend protection, backend protection, and message modification. Frontend will do some basic checks and reminders to prevent users from sending out messages containing privacy information. Once the user really sends out the messages, the server will analyze the them and try to find out wether the text contains PII. If PII is detected, the server will operate on the original message and sent it to the chatbot.

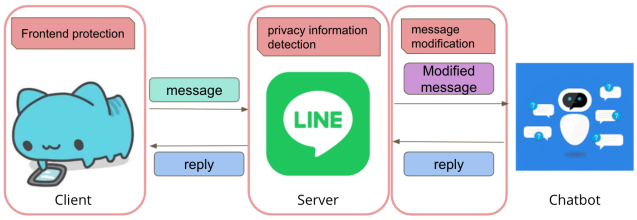

Figure 3. System Design

3.1.1 Chatbot authentication.

Basic idea. To solve the problem that the establishment of chatbot lacks strict management, we propose an authentication mechanism for chatbot. Every chatbot with access to PII has to undergo stricter application process before deployment. With such mechanism, we hope that it will be harder for attackers to easily set up an adversarial chatbot.

For task-oriented chatbot. For task-oriented chatbots that use users' PII to complete task, the server should reach an agreement with the chatbot developer before establishing a chatbot. The agreement should contain the developer's personal or enterprise document/information for verification, tasks provided by the chatbot, and the PII that the chatbot needs. Hence, the server can detect and remove unrelated PII from the messages according the content of the agreement. In addition, users can also be less worried about the chatbot's background.

For regular chatbot. For chatbots that don't use users' PII, server will detect and remove all PII entities from the messages sent to the chatbot. Thus, we think the authentication for such chatbot is not necessarily to be changed.

Possibility to automate the approving process. During the authentication process, we have to check whether the verification documents are legitimate, whether the requested PII is matched with the task of chatbot, etc. Even there's ML based approach that may automate the whole process, it's known that ML-based model is prone to adversarial attack. To achieve the effectiveness of this mechanism, we think that the more PII needed by a chatbot, the more enrollment of manual verification is needed.

3.1.2 Frontend (client side) protection.

Basic idea. Since there are researches [9, 13] showed that a lot of people regret sending their messages due to sensitive information leak, we want to develop a mechanism that detects and prevents users from sending sensitive information in advance. Besides, to integrate with the authentication mechanism, we want to show all chatbot information, like which PII it will uses, to the users. So that users can get better control of their personal (sensitive) information.

User agreement for chatbot. In our system design, task-oriented chatbot needs to undergo strict application to get access to users' PII information. For client side, such information (which PII the chatbot has applied for) is important for users to know. First, users must be aware of which and how their personal information is being used by chatbot, just like the permission mechanism for mobile APPs.

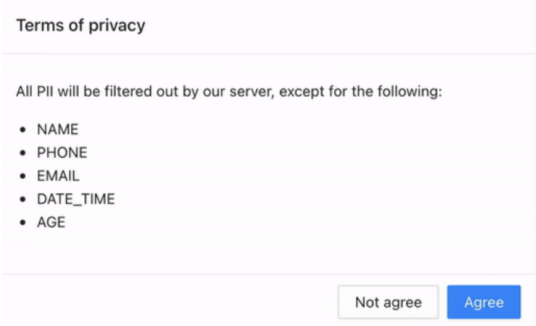

Figure 4. User agreement for chatbot

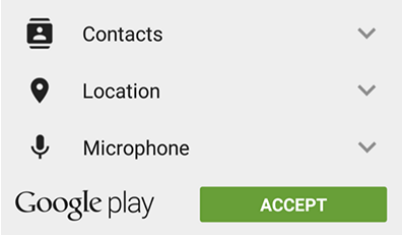

Figure 5. Permission request for Android APP [12]

Besides, with such information, users can decide whether to add a chatbot by themselves. Users can decline to add the chatbot if the chatbot is asking too much or irrelevant personal information. For example, it's suspicious that a daily-fortune chatbot is asking users' phone number or address.

Sensitive information alert. To prevent users from sending sensitive information in advance, we apply a simple detection mechanism on the client side. Consider that the computation resource is limited, we use much simpler rules to detect common patterns fro sensitive information. For example, we use five or more consecutive numbers to detect phone number, bank account, PIN, etc.

```
/[0-9-]{5,}/
```

When sensitive information is detected, the system will prompt a alert, telling user that he/she is going to send some sensitive information. But the user can decide whether to proceed to send the message.

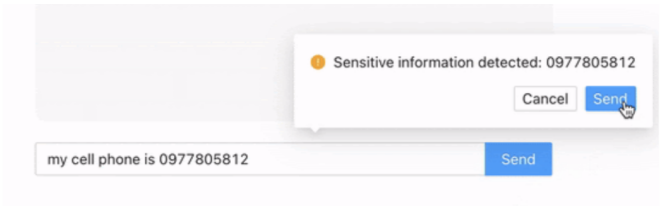

Figure 6. System alert for sensitive information

3.1.3 Backend (server side) protection - privacy information detection.

Basic idea. Although basic alert can be done at frontend, there is still a chance that users send out messages containing PII. Therefore, we survey for a few methods to detect privacy information at backend. If server can detect the PII in messages and identify if such information are needed by the chatbot according to the needs for PII the chatbot developer made during verification, server can do modification to the message(**section 3.1.4**) to avoid private information leakage. We divide PII detection methods into two categories, rule-based detection and ML-based detection.

Rule based detection. The concept of rule-based detection is similar to how we make alert at frontend. We use more complicated pre-defined rules than those at frontend because server have more computing resources than client. For each type of PII, we can define specific rules according PII's features, keywords, or production rules. For example, we can use regular expression to detect PII with fixed format, like cellphone number, address, email, etc. In addition, other keywords such as 'name' or 'age' can also be used to detect particular private information. [4] also mentions that there are already some rule based applications for such detection task and give out the key characteristics of them.

ML-based detection. When it comes to text analysis, natural language processing(NLP) is always an option. We first use tokenization tool to split text into numbers of tokens, and then determine whether it contains PII for each token and what types of PII it contains. There are already many effective and mature tokenization tools in current research, such as NLTK, coreNLP and spaCy. [8] discusses pros and cons of the tools mentioned above and the impact of them on the downstream model. As for detecting PII in each token, different research convert it to different ML problem. [11] sees the task as binary classification problem. In other words, it only focuses on distinguishing whether a token contains private information. [8] and [3] see the task as named entity recognition(NER) problem, which means that they would like to identify what types of PII a token contains. For our proposed methods, the last two studies are more useful because we need to determine whether to modify the information through the category of PII.
In our project, we use amazon comprehend service. Amazon comprehend is a NLP service provided on aws. It is good at analyzing text, extract insights about the content of documents, and recognizing the entities on unstructured text. PII detection is one of the services it provides. It can effectively and efficiently analyze text and detect several types of PII in message.

3.1.4 Backend (server side) protection - message modification.

basic idea. After detecting the messages containing PII, we proposed the following methods to modify the messages. Suppose the messages is $M = w_1w_2w_3...w_n$, w_k is the word in the message. The words which contain PII form a set $W = \{w|\ w \text{ contains PII}, w \in M\}$

- **Do not forward**
  Literally, the server does not forward the message to the chatbot if W is not an empty set.
- **Replace privacy information with specific tokens**
  The server replaces each $w \in W$ with specific tokens, such as * or -.
- **Hide senders**
  The server needs to send the messages without revealing clients' name. In order to communication between the clients and the chatbot, the server may need to maintain extra information. Suppose the number of clients is K, the server needs to hide clients in a K-anonymity set.
- **Replace privacy information with similar words**
  By using word2vec [5], we can represent each word w as a vector $v \in \mathbb{R}^d$ where d is the dimension of the latent space. Suppose there are word w_1 and w_2, and we use u and v to represent each of them in the latent space. When the word vector is trained on a great amount of corpus, the distance between u and v in the latent space is small if w_1 and w_2 have similar meanings [5]. Also, we use cosine similarity to measure the distance. Therefore, we randomly choose one word from the top K similar words to replace each $w \in W$ by measuring distances. Each word $w \in W$ is hide in a K-anonymity set. For example, '明天' may be replaced with '昨天', and '台北' may be replaced with '台南'.

3.2 Evaluation

In this section, we discuss on the advantages, disadvantages, effectiveness and effect on the functionality of chatbots of all protection methods mentioned in **3.1**.

3.2.1 Pros and cons of protection methods.

Chatbot authentication. Chatbot authentication is relatively easy to implement and deploy. Basic authentication including verifying the developer's personal or enterprise information is already used by a few social software, like Facebook Messenger and Line official account. Service providers

negotiating terms of privacy with third-party developers can also be seen in many smartphone apps , cloud services and so on. However, how to design suitable and effective verification mechanism to prevent third-party developers from over-claiming the PII they need through is a problem that we need to spend a lot of time thinking and planning.

Frontend (client side) protection. Frontend protection methods can operate simultaneously when the user is entering a message, which means that it would not affect the time required for message transmission. In addition, Because it is purely done in frontend, it can easily be plugged into a general app, which means that even if the chatbot is not authenticated, we still can use it to do basic PII alert. However, since clients have less computing resources than the server, complex functions may be inappropriate in this method.

Backend (server side) protection - privacy information detection.

- **Rule-based.** As we mentioned in **section 3.1.3**, rule-based PII detection has already had many studies and implementations for it. Besides, many PIIs have fixed formats, so it's easy to use production rule to detect them. However, there aren't efficient ways to define rules for each PII. In addition, if we only focus on format rather than actual message semantic, false positives may occur frequently.
- **ML-based.** To achieve a better performance, lots of data regarding PIIs are needed. However, it is difficult to collect abundant data containing private information for training, which makes these kinds somehow impractical. On the other hand, finding PIIs in unstructured text is actually a very new and immature research field, so how to make accurate judgments and avoid false positive issue is still a problem that needs further discussion.

Backend (server side) protection - message modification.

- **Do not forward/Replace privacy information with specific token.** Although sounds naive, these two methods are the most direct and effective way to avoid the leakage of privacy information. Nevertheless, it is easier to affect the functionality of chatbots.
- **Hide sender.** For server, this method is even more simple to implement but provides less protection. For the chatbots that do not offer personal services, e.g, information-providing or chit-chat chatbots, hiding sender shouldn't affect the quality of the service. With the sender hidden, connection between PIIs and users should be weaker and thus leaks less information about users.
- **Replace privacy information with similar words.** If we can replace the words containing PII with other similar words, we can protect privacy and maintain the readability of the message simultaneously. Keeping message readable is essential for those contextual analysis chatbots to understand text. However, it's not trivial to attain a good substitution. In the method we proposed, word embeddings are used to find similar word, which sometimes makes the modified message strange or ungrammatical. Thus, it's worth studing that how to substitute words that have more ambiguous meaning for those words containing PII (Taipei → somewhere in Taiwan), which may provide better results for preserving semantics.

3.2.2 Influence on different types of chatbot. Generally, chatbots' functionality depends on received messages. Hence, modifying message would inevitably affect chatbots' functionality when error occur in detection step. For example, it's possible that we detect too many PII and mask them in the message, preventing the chatbot's service from working properly. In this section, we talk about the influence of modification methods on different types of chatbot when false positive occur in detection procedure.
(1) for "Do not forward", (2) for "Replace with specific token", (3) for "Hide sender", (4) for "Replace with similar words"

menu/button based. Basically, the buttons only have basic selection functions and does not contain private information. Therefore, the instruction sent by pressing the button wouldn't be masked. In addition, doing any modification on the messages user send by accident would not affect the function because button based chatbot don't need them at all.

Table 2. Influence on menu/button based chatbot

(1)	(2)	(3)	(4)
Not affected	Not affected	Not affected	Not affected

keyword recognition. This type of chatbots rely on keywords in the message to operate. If a false positive detection occurs and the server drop or modify the message, service provided by these type of chatbots would be affected. For example, the words in the content needed to be translated by chatbot may be replaced or masked, resulting in incorrect translation. Hiding sender may be the best approach for this type of chatbots because it won't modify the content of messages.

Table 3. Influence on keyword recognition chatbot

(1)	(2)	(3)	(4)
Medium+	Medium	**Little**	Medium-

contextual analysis. Funtionality of this type of chatbot is prone to be affected if the message text is modified, especially when the modified message lacks readability. Furthermore, many contextual analysis chatbots serve as assistants, like google assistant, Apple siri, etc. These assistants provides personalized services, and thus hiding sender may not be the best solution to this type of chatbots. Ideally, replacing with similar words is the best way to maintain message readability. However, as mentioned in **section 3.2.1**, there are still problems in replacing similar words, which is worth researching in the future.

Table 4. Influence on contextual analysis chatbot

(1)	(2)	(3)	(4)
Medium+	Medium+	**Little**	**Medium-**

4 Related work

Our project incorporates ideas based on these researches [1, 2, 6, 10, 11]. Biswas and Debmalya first addressed the problem of chatbot privacy security and proposed entity-based privacy filtering. Besides, they used searchable encryption to preserve user chat privacy without requiring any knowledge of the chatbot design [2]. Furthermore, several researchers have studied the problem of detecting PII [10, 11]. They mainly focused on the challenges in detecting privacy revealing information and used machine-learning-based techniques that detect privacy-sensitive information in user-generated unstructured texts. For modifying sensitive messages, there is a research proposing a novel information-theoretic approach to text sanitization and discussing the concept of text anonymization[1]. In addition, a recent paper [6] adopt reinforcement learning on text anonymization. Those papers motivate us to come up with the ML-based K-anonymity idea.

5 Conclusion and future work

In this project, we have proposed several methods from different perspectives against our threat model. Our methods can be divided into three parts, frontend protection, backend protection, and message modification. Frontend protection, including chatbot authentication and chatbot terms of privacy, is effective and easy to deploy for all types of chatbots. Backend protection mainly focuses on how to efficiently and correctly detect PII. There are two methods for detecting PII, rule-based and ML-based, respectively. Detection is easy for rule-based, but definition of rules is tricky. ML-based needs no rules, but false-positive and false-negative issues and data collection issues are still unsolved. Finally, we adopt four ways to modify detected messages. However, the proposed methods have different influences on different types of chatbots. There are no universal method that can apply to all types of chatbots. As mentioned in **section 3.2.2**, influence on chatbots' function is inevitable when applying our methods, especially if false positive occurs in PII detection. Therefore, to make our proposal more practical, it's essential to research on more general message modification and precise PII detection in the future.

References

[1] Balamurugan Anandan, Chris Clifton, Wei Jiang, Mummoorthy Murugesan, Pedro Pastrana-Camacho, and Luo Si. 2012. t-Plausibility: Generalizing Words to Desensitize Text. *Trans. Data Priv.* 5, 3 (2012), 505–534.

[2] Debmalya Biswas. 2020. Privacy Preserving Chatbot Conversations. In *Proceedings of the 3rd IEEE Conference on Artificial Intelligence and Knowledge Engineering (AIKE)*.

[3] Fadi Hassan, David Sánchez, Jordi Soria-Comas, and Josep Domingo-Ferrer. 2019. Automatic Anonymization of Textual Documents: Detecting Sensitive Information via Word Embeddings. In *2019 18th IEEE International Conference On Trust, Security And Privacy In Computing And Communications/13th IEEE International Conference On Big Data Science And Engineering (TrustCom/BigDataSE)*. IEEE, 358–365.

[4] John Kingston. 2017. Using artificial intelligence to support compliance with the general data protection regulation. *Artificial Intelligence and Law* 25, 4 (2017), 429–443.

[5] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781* (2013).

[6] Ahmadreza Mosallanezhad, Ghazaleh Beigi, and Huan Liu. 2019. Deep reinforcement learning-based text anonymization against private-attribute inference. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*. 2360–2369.

[7] Special Publication 800-122. NIST. [n.d.]. Guide to Protecting the Confidentiality of Personally Identifiable Information (PII). https://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-122.pdf

[8] Paulo Silva, Carolina Gonçalves, Carolina Godinho, Nuno Antunes, and Marilia Curado. 2020. Using NLP and Machine Learning to Detect Data Privacy Violations. In *IEEE INFOCOM 2020-IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*. IEEE, 972–977.

[9] M. Sleeper, Justin Cranshaw, Patrick Gage Kelley, Blase Ur, A. Acquisti, L. Cranor, and N. Sadeh. 2013. "i read my Twitter the next morning and was astonished": a conversational perspective on Twitter regrets. *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (2013).

[10] Welderufael B Tesfay, Jetzabel Serna, and Sebastian Pape. 2016. Challenges in Detecting Privacy Revealing Information in Unstructured Text.. In *PrivOn@ ISWC*.

[11] Welderufael B Tesfay, Jetzabel Serna, and Kai Rannenberg. 2019. PrivacyBot: detecting privacy sensitive information in unstructured texts. In *2019 Sixth International Conference on Social Networks Analysis, Management and Security (SNAMS)*. IEEE, 53–60.

[12] How to Request Permissions in Android Application? [n.d.]. https://www.geeksforgeeks.org/android-how-to-request-permissions-in-android-application/

[13] Yang Wang, Gregory Norcie, Saranga Komanduri, A. Acquisti, P. Leon, and L. Cranor. 2011. "I regretted the minute I pressed share": a qualitative study of regrets on Facebook. In *SOUPS*.