PrivacyAsst: Safeguarding User Privacy in Tool-Using Large Language Model Agents

Xinyu Zhang, Huiyu Xu, Zhongjie Ba*, *Member, IEEE*, Zhibo Wang, *Senior Member, IEEE*, Yuan Hong, *Senior Member, IEEE*, Jian Liu, Zhan Qin, *Senior Member, IEEE*, and Kui Ren, *Fellow, IEEE*

Abstract—Swift advancements in large language model (LLM) technologies lead to widespread research and applications, particularly in integrating LLMs with auxiliary tools, known as tool-using LLM agents. However, amid user interactions, the transmission of private information to both LLMs and tools poses considerable privacy risks to users. In this paper, we delve into current privacy-preserving solutions for LLMs and outline three pivotal challenges for tool-using LLM agents: generalization to both open-source and closed-source LLMs and tools, compliance with privacy requirements, and applicability to unrestricted tasks.

To tackle these challenges, we present PrivacyAsst, the first privacy-preserving framework tailored for tool-using LLM agents, encompassing two solutions for different application scenarios. First, we incorporate a homomorphic encryption scheme to ensure computational security guarantees for users as a safeguard against both open-source and closed-source LLMs and tools. Moreover, we propose a shuffling-based solution to broaden the framework's applicability to unrestricted tasks. This solution employs an attribute-based forgery generative model and an attribute shuffling mechanism to craft privacy-preserving requests, effectively concealing individual inputs. Additionally, we introduce an innovative privacy concept, t-closeness in image data, for privacy compliance within this solution. Finally, we implement PrivacyAsst, accompanied by two case studies, demonstrating its effectiveness in advancing privacy-preserving artificial intelligence. [1]

Index Terms—Large language model (LLM), Tool-using LLM agent, Privacy, Homomorphic encryption, t-Closeness.

1 Introduction

With the rapid advancement in the field of Natural Language Processing (NLP), Large Language Models (LLMs) have become a breakthrough force driving digital innovation in a variety of fields [1]–[3]. Extensions and applications centered on these large language models are also evolving, of particular note is *Tool-using LLM agent* [4]–[6]. Tool-using LLM agent is a pioneering framework that uses the LLM as a core controller, allowing the LLM to interact seamlessly with a variety of tools and APIs. The framework greatly improves the performance of the LLM itself by calling external tools that can access to additional information missing from the model weights. Moreover, the framework greatly stimulates the potential capabilities of LLMs to plan, execute, and accomplish complex tasks. Thus, tool-using LLM agents are being widely studied and developed as a fundamental framework for LLM applications.

Nevertheless, the fundamental framework of tool-using LLM agents raises new privacy considerations. Specifically, within the tool-using LLM agent usage process (refer to Figure 1), the user initially submits a public query and a private input to the LLM. Subsequently, the LLM orchestrates the task and invokes relevant tools to execute it. After receiving privacy results from the tools, the LLM generates and delivers the response to the user [7]. In this process, the LLM possesses knowledge of the user's private input and inadvertently shares this information with the corresponding tools during task execution. For instance, when a user employs a tool-using LLM agent to describe the content of an image, they provide a private image to LLM, which subsequently shares that image with a third-party model deployment platform. This interaction results in the exposure of the user's privacy to both the LLM and the third-party tool provider platform.

To the best of our knowledge, there is currently no solution for privacy issues under the tool-using LLM agent. While privacy-preserving approaches under LLM (i.e., local deployment [8], anonymous communication [9], and data masking [10]) can be extended and applied to tool-using LLM agents, these approaches have shortcomings under both LLMs and tool-using LLM agents. (1) *Local deployment* deploys the open-source large language model and uses open-source tools that are also deployed locally to solve problems [8]. This approach guarantees privacy by ensuring that data does not leave the local execution environment. However, this approach demands substantial user-side expertise and hardware resources to deploy the large language model and tools. In addition, this approach is limited to using only open-source large language models and tools; many powerful closed-source large language models or

This work is partially supported by the National Key R&D Program of China (2023YFB2904000), the National Natural Science Foundation of China (62172359, U20A20178, and 62072395), Zhejiang Provincial Natural Science Foundation of China (LD24F020010), the Fundamental Research Funds for the Central Universities (2021FZZX001-27), Hangzhou Leading Innovation and Entrepreneurship Team (TD2020003), and the National Science Foundation grants (CNS-2308730).

Xinyu Zhang, Huiyu Xu, Zhongjie Ba, Zhibo Wang, Jian Liu, Zhan Qin, and Kui Ren are with the State Key Laboratory of Blockchain and Data Security, School of Cyber Science and Technology, College of Computer Science and Technology, Zhejiang University, Hangzhou, Zhejiang 310007, China (e-mail: {xinyuzhang53, huiyuxu, zhongjieba, zhibowang, liujian2411, qinzhan, kuiren}@zju.edu.cn).

Yuan Hong is with the University of Connecticut, Stamford, CT 06901 USA (e-mail: yuan.hong@uconn.edu).

**Corresponding author: Zhongjie Ba.*

1. Code is available at https://github.com/Eyr3/PrivacyAsst.

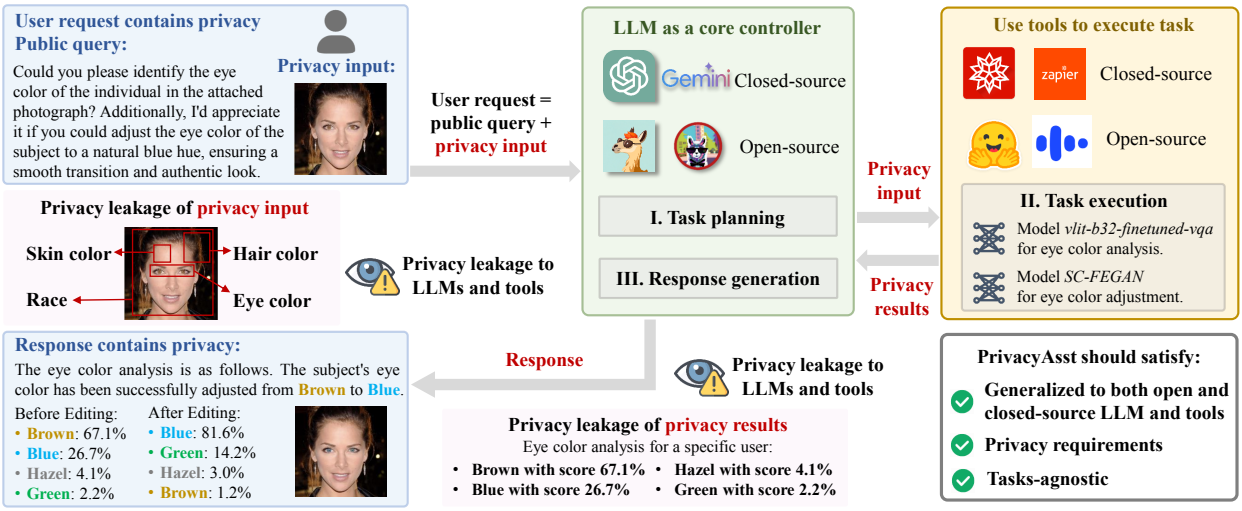

Fig. 1. The whole process of tool-using LLM agent and the privacy leakage within the process.

closed-source tools are not available. (2) *Anonymous communication* approaches do not require local deployment and are not affected by the fact that the model or tool is closed-source. This approach transmits information or data in a way that hides the identity of the sender and the receiver, making it difficult for third parties to trace or identify the participants involved in the communication. Users can use the Onion Router (i.e., Tor) [9] to communicate anonymously with LLM so that LLM and the tools cannot associate the user ID with the specific input. However, this approach remains impractical for most users who lack professional expertise in Tor deployments. In addition, there is no privacy guarantee for anonymous communication. Existing attacks illustrate that users' real IP addresses can still be traced even under Tor [11], [12]. (3) *Data masking* achieves protection of sensitive information by replacing original data values with fictitious, altered, or scrambled values while preserving the overall structure and characteristics of the data. For instance, PrivateGPT [10] identifies and removes over 50 types of personally identifiable information (i.e., PII) within the application before sending it through to ChatGPT. However, this approach is task-bound, i.e. it cannot be accomplished when the task itself requires the input of private information.

Guided by the shortcomings of privacy-preserving solutions within the LLM domain, we summarize three key challenges relevant to the development of a privacy-preserving framework for tool-using LLM agents (refer to Figure 1): First, how to achieve a high level of generality in the framework. This includes ensuring that the framework is not limited to specific types of LLMs and tools, whether they are open-source or closed-source. Second, how to ensure privacy guarantees within the framework, aligning with existing privacy requirements. Third, how to enable the framework to be adaptable and agnostic to a spectrum of tasks, free from the constraints of specific tasks.

In the paper, we introduce PrivacyAsst, the first privacy-preserving framework for tool-using LLM agents, addressing the aforementioned challenges. Distinguishing itself from conventional data masking solutions, which cannot operate on private input, our encryption-based solution incorporates an encryption scheme. This solution enables the framework to encompass tasks that can operate on encrypted inputs, including neural network-based classification tasks. Employing the homomorphic encryption mechanism, the original input is encrypted and subsequently distributed to third-party tools through the LLM, enabling the acquisition of accurate encrypted outcomes. This basic solution requires that third-party tools be compatible with encrypted data and provide encrypted counterparts of their functionalities. To further enhance the framework's adaptability for more intricate tasks like *generative models*, the framework also incorporates a shuffling-based solution. This solution leverages a forgery generative model to generate a set of forgery inputs, subsequently employing an attribute shuffling mechanism to construct privacy-preserving requests from these forgery inputs, effectively concealing individual inputs.

Specifically, we develop a privacy-preserving framework that includes two solutions. The basic encryption-based solution ensures computational security privacy protection for both open and closed-source LLMs and tools, given the availability of an encrypted tool variant. The solution is suitable for a wide range of tasks that can operate on encrypted inputs, including computations, classifications, and counting tasks. In addition, the shuffling-based solution provides t-closeness privacy protection for both open and closed-source LLMs and tools, and it's task-agnostic.

In summary, our contributions are as follows:

- We present PrivacyAsst for tool-using LLM agents, the first privacy-preserving framework that is generalized across open and closed-source LLMs/tools, adheres to established privacy requirements, and offers applicability across a diverse range of tasks.
- We design two innovative privacy-preserving solutions within PrivacyAsst. The first, an encryption-based solution, incorporates an encryption scheme into tool-using LLM agents to ensure user privacy in compliance with computational security standards. The second, a shuffling-based solution, leverages attribute-based forgery generative models and an attribute shuffling mechanism to generate a set of privacy-preserving requests. These requests effectively obfuscate the user's

private inputs while still adhering to the principles of t-closeness.

- We implement PrivacyAsst and provide two examples for each solution to evaluate its effectiveness. Extensive experiments show that PrivacyAsst enables accurate tool usage with LLMs and addresses three of the aforementioned challenges.

2 Background

2.1 Language Models

A language model (LM) is a probabilistic machine learning model that estimates the probabilities of word sequences based on its training text corpora. Given an input text, an LM iteratively predicts the next token or word per the maximum probability. LMs play a fundamental role in numerous natural language processing (NLP) tasks, encompassing text classification [13] (e.g., sentiment analysis, news topic classification, email spam filtering), machine translation [14] (e.g., translating business content into different languages), and natural language generation [15] (e.g., question-answering, abstract summarization, report writing).

A large language model (LLM) is distinguished by its huge number of model parameters and its extremely large-scale training data. Such powerful LLMs, exemplified by models like ChatGPT [1], LLaMA [2], [16], and Google Gemini [17], signify a significant leap in AI capabilities [18]. These models not only outperform traditional LMs in conventional NLP tasks, but also demonstrate high potential in comprehending human intentions, automating intricate processes, and generating coherent responses [19]. Moreover, prompt engineering techniques enable effective communication with LLMs, enabling control over their behavior to achieve desired outcomes without necessitating updates to the model weights. This further enhances the LLM's language comprehension and generation abilities [20].

2.2 Tool-Using LLM Agents

Tool-using LLM agent [5], [6] is a new paradigm of today's LLM applications, addressing inherent limitations of the LLM, such as limited multimodal skills, rudimentary mathematical skills, and challenges in maintaining the up-to-date information [21]. Tool-using LLM agents empower LLMs to harness external tools (such as calculators [22], search engines [23], and machine learning models [7]) without necessitating changes to the underlying model parameters.

The landscape of tools that can be manipulated by instructions in conjunction with LLMs is increasingly diversified, allowing the completion of complex tasks. And tool-using LLM agents can be categorized into three interaction-based genres from a user-centric perspective [5], ranging from the most tangible to the least tangible:

Physical interaction-based agents [24] engage directly with the physical world, interfacing with devices such as sensors, wearables, cameras, and robots. Tools within these agents possess the ability to sense physical environments, collect data through devices, perform data analysis via programs, and subsequently react to the next step in operations.

Graphical user interface (GUI)-based agents [25], [26] facilitate tool manipulation through GUIs, including graphical icons, indicators, and menus found in websites, computer games, and similar platforms. Tools within these agents do not exert direct influence on the physical world but capture and store users' actions and preferences within the agents, which are subsequently leveraged for data analysis and further interactions.

Program-based agents [7], [22], [27] grant users access to the innermost layers of tool source code, encompassing diverse forms, such as programming libraries, software development environments, and machine learning models. These tools offer enhanced flexibility and customization, allowing them to address complex challenges. Tools in these agents require the user input for required parameters and subsequently produce final results.

2.3 Sensitive Information Leakage in Tool-Using LLM Agents

According to previous research [28]–[31], the leakage of sensitive information poses a significant concern in the domain of LLM, as well as tool-using LLM agents. On one hand, LLMs are typically trained on extensive textual datasets, which may potentially contain sensitive information such as personally identifiable information (PII) (e.g., names, addresses, and email addresses), financial data, health records, and location information [16]. In cases where LLMs excessively rely on specific sensitive data points, there is an inadvertent risk of reproducing or exposing them, thereby heightening the likelihood of data leakage [32], [33]. Deliberately crafted prompts by malicious users or inadequate filtering of sensitive content during training may lead to privacy breaches of the training data [34], [35]. On the other hand, users may directly input documents or files containing sensitive information into LLMs or tool-using LLM agents. Such sensitive data is subsequently exposed to the LLM and the tool provider [30]. Besides, as with sensitive training data in the former scenario, where user-supplied files encompassing sensitive data traverse into the LLM and the tool provider's domain, there is a risk of unauthorized users accessing this content during generation.

This paper focuses on the latter aspect of sensitive information leakage. Building upon the taxonomy in Section 2.2, we discuss privacy leakage in different tool-using LLM agents, as outlined in Table 1.

Physical interaction-based agents facilitate the collection of real-world data through mechanisms such as image-capturing cameras and audio-capturing microphones. Previous attacks have highlighted the potential severe leakage of sensitive information when AI analysis is applied to real-time real-world data. For instance, analysis conducted in a coffee shop context [36] demonstrates that real-world sensors can directly expose user facial and location details.

GUI-based agents have the capacity to directly capture user inputs, preferences, and online activities on web and Android interfaces. Due to the bridging nature of GUIs that connect the physical and virtual realms, users often transmit sensitive information, including bank account details and social media account information, through GUIs. Existing GUI attacks [37]–[39] have demonstrated that adversaries can intercept GUI communications, gaining access to private information without authorization.

TABLE 1
Sensitive Information Leakage in Tool-Using LLM Agents

Agent taxonomy	Agent example	Application and task	Content of privacy leaks	Intercepting entities
Physical interaction-based	VoxPoser [24]	Language understanding, Motion planning, Code generation.	Sensitive information in the physical world.	LLMs, devices that receive real-world data, unauthorized users.
GUI-based	Voyager [25]	Prompt generation, Motion planning, Code generation.	Initial state of the software/system.	LLMs, software that receives user interactions, unauthorized users.
	Mantella [26]	Speech-to-text	Speech content.	LLMs, speech recognition system (e.g., Whisper), game software, unauthorized users.
Program-based	Toolformer [22]	Question answering, Wikipedia search, Machine translation.	Sensitive information in user-provided text.	LLMs, third-party tools, unauthorized users.
	HuggingGPT [7]	Object detection, Image classification, Image-to-text, etc.	Sensitive information in user-provided images.	LLMs, AI community/ machine learning platform (e.g., HuggingFace), unauthorized users.

Program-based agents possess access to source code and its corresponding inputs, encompassing text, speech, and images. The sensitive information of these inputs is directly exposed to the source code provider.

In summary, the aforementioned sensitive information leakage within the three types of agents can be interpreted as sensitive information leakage within program-based agents. For instance, in the case of physical interaction-based agents, data collected from the physical world is transmitted to the internal programs within these agents. Similarly, GUI-based agents transmit data collected from graphical interfaces to the internal programs within the agents. Therefore, establishing a privacy-preserving framework for program-based agents can directly extend the protection to address the privacy concerns associated with both physical interaction-based and GUI-based agents.

3 Problem Formulation

3.1 System Model

The system comprises three entities: the user, the LLM provider, and the tool provider (see Figure 2). We follow a typical and widely used program-based agent, HuggingGPT [7], and consider OpenAI as the LLM provider to offer ChatGPT [1] and Hugging Face [40] as the tool provider to offer a wide range of AI models. The user has a request $Q = \{q, x\}$ containing a public query q and the private input x, where the private information in x is $p(x)$. The user wants to know the answer to the request for input, but the user lacks the computation source and professional knowledge. So the user wants to employ the LLM to obtain the answer while not leaking private information. The LLM serves as a controller, which can plan tasks and select AI models to suit different tasks. The tool provider can execute AI models according to the input arguments.

There is a two-way communication channel between the user and the LLM and between the LLM and the AI models, respectively. The user sends the user request Q to the LLM and wants to obtain the correct response. After receiving the user request Q, the LLM leverages its ability to parse the user request and decomposes the user request into multiple tasks $\texttt{task}_i = \{\texttt{id}, \texttt{goal}, \texttt{agrs}\}$. The LLM plans the task order and dependency based on knowledge and then generates the response template r according to all tasks. The response template r is generated by inputting pre-determined prompts into the LLM. The LLM distributes the parsed tasks to AI models according to the model description in Hugging Face. Each AI model executes the assigned tasks on the input $\texttt{agrs}$ and returns the prediction result c_i to the LLM. After receiving all results $c_{\{i\in[m]\}}$, the LLM fills $c_{\{i\in[m]\}}$ into the template r to generate the response R and returns it to the user.

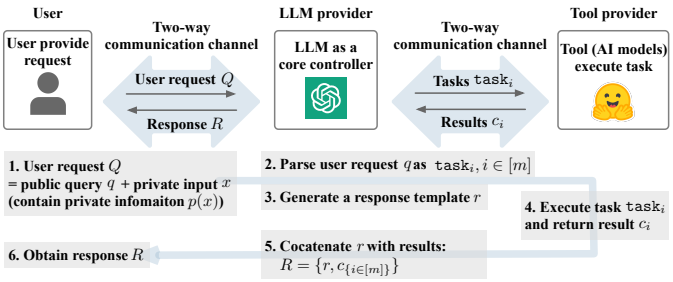

Fig. 2. The system model. The entire process involves 6 steps.

3.2 Threat Model

We assume both the LLM and AI models to be semi-honest adversaries characterized as honest but curious. This implies that they adhere to the protocol but have a motivation to learn as much private information as possible. *Importantly, we assume there is no collusion between any entities, including the LLM and AI models, as well as among different AI models.* Specifically, the LLM possesses knowledge of the user ID and faithfully executes the user request Q, while also exhibiting curiosity towards the content of x and attempting to infer the private information $p(x)$. On the other hand, AI models remain unaware of the user ID and faithfully execute the assigned task $\texttt{task}_i$.

3.3 Design Goal

We aim to design a privacy-preserving framework for tool-using LLM agents that can ensure accurate responses to user requests and be applicable to both open-source and closed-source LLMs and tools. In particular, we propose two solutions that cater to different privacy requirements and task ranges.

3.3.1 Design Goals of the Encryption-based Solution

- **Correctness:** The framework should guarantee the correctness of user requests, ensuring that the final response remains unchanged compared to a scenario without privacy safeguards.

- **Generalization:** The framework should be compatible with both open-source and closed-source large language models and tools.
- **(Privacy Requirement) Computational Security:** The entities involved, including the LLM and AI models, should only possess knowledge of the ciphertext of the private input x and the associated private information $p(x)$. It is important to note that there is no computationally feasible method to recover the ciphertext in a reasonable timeframe.

3.3.2 Design Goals of the Shuffling-based Solution

In addition to the **Correctness** and **Generalization** goals outlined in Section 3.3.1, the shuffling-based solution should satisfy the following objectives:

- **(Privacy Requirement) t-Closeness:** The entities (i.e., LLMs and AI models) are aware that the private input and private information of a particular user are part of a set of private inputs and private information. Furthermore, the set of private inputs and private information should satisfy t-closeness.
- **Task-agnostic/ Independent of AI models:** Given the diverse range of challenges encompassing classification, object detection, segmentation, generation, and more, the framework should remain compatible with any AI model, allowing for flexibility in task selection.

4 Encryption-based Solution

In this section, we present a solution using homomorphic encryption to ensure computational security. The proposed solution involves performing operations on ciphertext throughout the process of tool-using LLM agents.

4.1 Homomorphic Encryption-based AI Inference

In this solution, *we do not alter the training process of the AI model - we train the AI model with plaintext samples. Rather, we make modifications solely to the inference process.* We employ homomorphic encryption schemes to perform inference on ciphertext and obtain accurate encrypted results. This methodology enables computations on ciphertext without prior decryption, thereby protecting the privacy of the input data. To illustrate this approach, we provide an example of a classification model, utilizing the TenSEAL library [41].

Homomorphic Encryption and Decryption. TenSEAL library encrypts and decrypts integers vectors using the Brakerski/Fan-Vercauteren (BFV) scheme [42], [43] and real numbers vectors using the Cheon-Kim-Kim-Song (CKKS) scheme [44].

Homomorphic Operations. TenSEAL provides fundamental operations of homomorphic encryption schemes, including element-wise addition, subtraction, and multiplication for both encrypted-encrypted vectors and encrypted-plain vectors. Additionally, essential operations in AI inference, such as dot product and vector-matrix multiplication (e.g., 2-D convolutions), are supported. By leveraging the supported operations in TenSEAL, we can perform inference on ciphertext for AI models such as the Convolutional neural network and Deep neural network.

It is worth noting that other encryption-based AI inference techniques are also applicable. Further advancements in encryption techniques are expected to contribute to the continued development of this solution.

4.2 System Design

The overview of the encryption-based solution is depicted in Figure 3. The user first encrypts the private input x using a homomorphic encryption scheme, such as leveled homomorphic encryption [45]. The user sends the plaintext query q and the encrypted input x^* to the LLM. The LLM parses the plaintext query q into multiple tasks $\texttt{task}_i = \{\texttt{id}, \texttt{goal}, \texttt{agrs}\}$ and generates the response template r. It is important to note that the request parsing and the response template generating is similar to the original one, i.e., step 2 and step 3 in Figure 2, except for the task's input is the ciphertext, i.e., $\texttt{agrs} = x^*$. The LLM distributes the parsed tasks to AI models. Each AI model executes the task on the encrypted input data. Specifically, encrypted versions of the private input are fed into the encrypted neural network. The network performs the encrypted inference, combining different encrypted computations on the ciphertext using homomorphic operations (the detail can be referred to Section 4.1). After the inference is completed, the AI model returns the encrypted form of the final output of the neural network c_i^* to the LLM. The LLM fills the encrypted results into the template r and obtains the response. Finally, the user decrypts c_i^* using the secret decryption key to reveal the results and response.

Note that to ensure that the user's input privacy is preserved at all stages of the request and computation, we assume that the user has knowledge of encryption and decryption with a homomorphic encryption scheme.

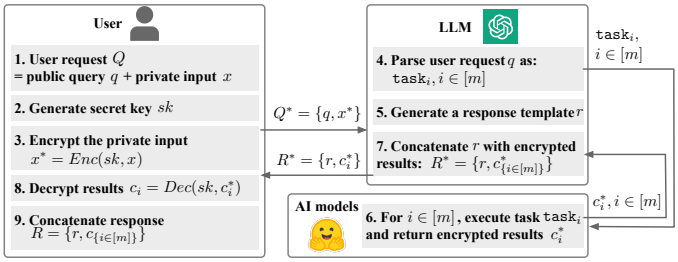

Fig. 3. An overview of the encryption-based solution, including 9 steps.

4.3 Security Analysis

The LLM receives the encrypted inputs from the user and receives encrypted results from AI models. AI models receive the encrypted images from the LLM and obtain the encrypted results after inference.

Theorem 1 (Computational Security)**.** *Let λ denote the security parameter in the homomorphic encryption scheme, then any adversary attempting to recover the plaintext (x) from the ciphertext (x^*) should undertake a computational effort of at least $\Omega(2^\lambda)$ bit operations [44]. In the encryption-based solution, both LLM and AI models own x^*, which means that the solution satisfies a λ-bit security level [46], [47].*

5 Shuffling-based Solution

In this section, we present a shuffling-based solution for ensuring t-closeness. Our solution leverages a generative model and an encryption mechanism to generate a set of encrypted forged inputs, while preserving the confidentiality

of the input content within the LLM. We then employ an attribute shuffling mechanism to generate privacy-preserving queries that effectively obscure the user's private input and private information. We provide the preliminary knowledge for our solution from Section 5.1 to Section 5.4, followed by the detailed description of the solution in Section 5.5.

5.1 Text-to-Image Generative Model

Text-to-image models are a form of machine learning models that, based on natural language descriptions provided as input, generate images that align with those descriptions. These models typically comprise a language model and a generative model. The language model transforms the natural language input description into vectors in the latent space, and subsequently, the generative model leverages the textual vectors to produce the corresponding descriptive image. Currently, state-of-the-art text-to-image models such as DALL·E [48], Midjourney [49], and open-source Stable Diffusion [50], enable users to enter simple prompts and create visually realistic images.

5.2 Key Exchange Protocol

The Key Exchange Protocol plays a crucial role in facilitating secure communications over insecure channels by allowing two or more parties to securely establish a shared secret key. The shared secret key can then be used to encrypt and decrypt messages to ensure confidentiality, integrity, and authenticity during communication. A widely used key exchange protocol is Diffie-Hellman (DH) key exchange [51], which allows two parties to establish a shared secret key without prior knowledge of each other. As depicted in Figure 4, this protocol involves Alice and Bob, who aim to derive a shared secret key, while the eavesdropper, Eve, attempts to intercept the secret key. The protocol consists of the following steps:

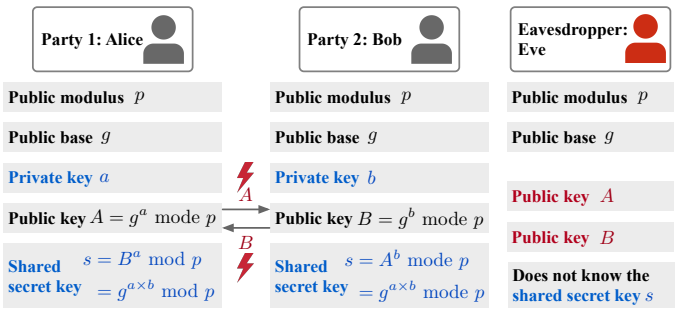

Fig. 4. The process of Diffie-Hellman key exchange protocol [51].

1) **Public Modulus and Base Agreement:** Alice and Bob agree to use a public modulus (p) and a public base (g).
2) **Key Generation:** Alice and Bob each select a private key (a and b) and compute their respective public key (A and B). The private keys are held confidential, while the public keys are openly exchanged.
3) **Key Exchange:** Alice and Bob engage in the exchange of their public keys (A and B).
4) **Shared Secret Key Derivation:** Following the key exchange, both Alice and Bob independently compute a shared secret key (s) utilizing their private keys and the received public key from the other party.

TABLE 2
Categories and examples of data record attributes stored in tables

Attributes categories	Examples
Explicit identifiers	Social Security Number, Address, and Name
Quasi-identifiers	Skin color, Hair color, Age group, and Eye color
Sensitive features	Gender and Race

5) **Shared Secret Key Establishment:** Finally, Alice and Bob obtain the same shared secret key (s), which can be used for secure communication employing symmetric encryption algorithms.

The security of the Diffie-Hellman key exchange relies on the difficulty of solving the discrete logarithm problem in finite fields [51]. By exchanging only public keys, the protocol ensures that even in the event of interception and acquisition of the public keys by the eavesdropper, Eve, it remains computationally infeasible for Eve to deduce the private keys and consequently deduce the shared secret key.

5.3 The t-closeness Principle

For a table where each record corresponds to an individual, the attributes of each record can be divided into three categories [52]–[54]. These categories, along with examples are detailed in Table 2. (1) *Explicit identifiers* are attributes that can directly and uniquely identify individuals. (2) *Quasi-identifiers* are attributes that, in isolation, do not uniquely identify an individual, but when used in combination with other quasi-identifiers, could potentially do so. This combination of these attributes could potentially narrow down the group of individuals to a smaller subset. In particular, the **equivalence class** of a table is defined as a set of records that share the same values for the quasi-identifiers. (3) *Sensitive features* are attributes that are often considered private or sensitive to an individual.

The concept of t-*closeness*, as proposed by Li et al. [54], focuses on preventing the disclosure of sensitive attributes within the table. The goal of t-closeness is to ensure that the distribution of sensitive attributes in each equivalence class closely resembles their distribution across the entire table. In other words, for each sensitive attribute, the distance (e.g., measured by the Earth Mover's Distance [55]) between the attribute distributions in each equivalence class and in the overall table should not exceed a predefined threshold t.

5.4 Privacy Attributes in Images

The privacy of an image refers to the personal identity, location data, and other sensitive content contained within the image. To establish a quantifiable privacy-preserving framework for image privacy, we refer to the standard list of privacy attributes defined by Orekondy et al. [56]. The list comprises privacy attributes obtained by integrating and pruning from relevant guidelines for handling personally identifiable information contained in the European Union Data Protection Directive [57] and the United States Privacy Act of 1974 [58], as well as the prohibitions on sharing personal information across various social networking sites. Table 3 provides a comprehensive list of privacy attributes.

5.5 System Design

As depicted in Figure 5, the overview of the shuffling-based solution can be divided into two phases: Phase I. attribute-based forged image generation, and Phase II. shuffling-based privacy preserving request.

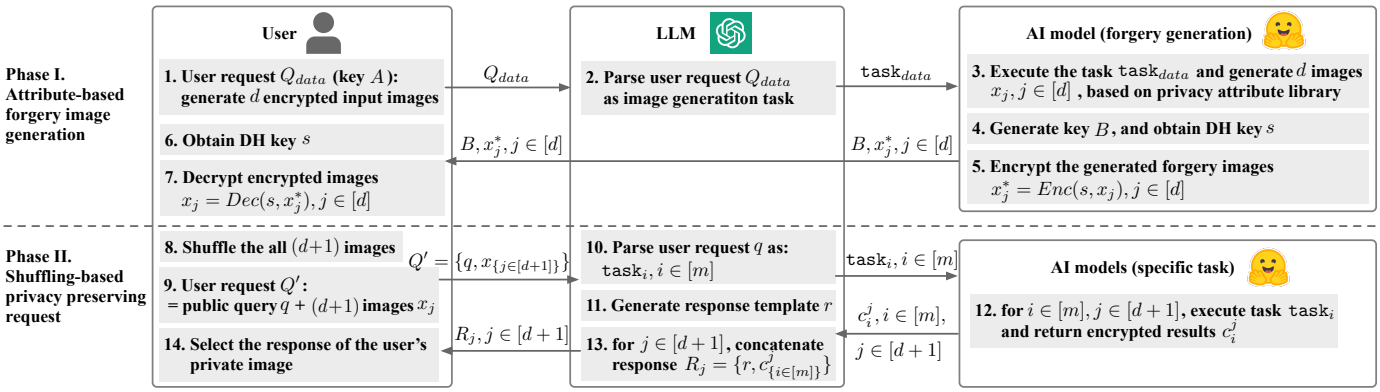

Fig. 5. An overview of the shuffling-based solution, including 14 steps.

TABLE 3
List of 68 Privacy Attributes in [56].

Privacy Attributes ('C' and 'P' stand for complete and partial)
Skin Color, Gender, Hair Color, Race, Age Group, Safe, Face (C), Face (P), Eye Color, Weight Group, Height Group, Culture, Landmark, Occupation, Visited Location (P), Date/Time of Activity, Social Circle, Spectators, Professional Circle, Semi-nudity, Full Name, Similar view, Sports, General Opinion, Personal Relationships, Work Occasion, Tickets, Handwriting, Sexual Orientation, Tattoo, Hobbies, License Plate (C), Traditional clothing, Medical Treatment, Competitors, Signature, Religion, Passport, Receipts, First Name, Nationality, Last Name, Personal Occasion, Physical Disability, License Plate (P), Date of Birth, Vehicle Ownership, Mail, Phone, no. Visited Location (C), Education History, Online Conversations, Username, Marital Status, Home Address (C), Political Opinion, Credit Card, Email Address, Student ID, Drivers License, Medical History, Place of Birth, Email Content, Legal Involvement, Nudity, Fingerprint, National Identification, Home Address (P)

In Phase I, the user aims to obtain d forged images to conceal the privacy attributes in the private image. Meanwhile, the plaintext of the generated image should remain unknown to the LLM, as the LLM could potentially acquire the user's private image by analyzing two phases. To achieve this, the user initiates a new request to the LLM, denoted as Q_{data}, to obtain d encrypted forged images. The user also includes a public key A along with the request for encryption. The LLM then parses and forwards the task to the forgery generative AI model. Subsequently, the AI model generates a set of forged images based on the privacy attribute library, as explained in Section 5.5.1. To encrypt the images, the AI model generates another public key B and then combines it with A to derive the shared secret key s (through DH key exchange), which is used for encryption. The AI model then sends the encrypted images, along with public key B, to the user through the LLM. Upon receiving the encrypted images, the user can derive the shared secret key s and then use s to decrypt images.

In Phase II, the user aims to obtain the response of the private image without revealing the privacy attributes. To achieve this, the user shuffles the $d+1$ images and submits them as a request to the LLM. The process in Phase II is similar to the original process depicted in Figure 2, with the difference being that the number of private images has increased from 1 to $d+1$. Both LLM and the AI model (specific task) possess knowledge of the $d+1$ images, but they are unaware of which image corresponds to the user's private image, thereby satisfying the t-closeness.

5.5.1 *Image Generation based on Privacy Attribute Library*

We provide a detailed example of the image generation process, utilizing the creation of a facial image for illustration, as depicted in Figure 6. As per the privacy attributes enumerated in Section 5.4, the privacy attributes pertaining to facial images encompass 6 distinct attributes (i.e., skin color, gender, hair color, race, age group, face, and eye color). Within the image generation process, we first randomly select elements in each privacy attribute category from the existing library. These selected attributes are subsequently concatenated into an attribute constraint. The attribute constraint is then input into the LLM with guided prompts to produce text-to-image prompts. The detailed generation process of text-to-image prompts is illustrated in Table 8. The text-to-image model leverages this prompt to generate corresponding images that align with the textual description. The combination of privacy attributes as the key prompt allows the generated facial forgeries to obfuscate the user's private image attributes.

5.5.2 *Privacy Attribute Library Extensibility*

The solution is designed to be easily extensible, allowing for the addition or modification of the privacy attribute library. If new privacy attributes emerge in real-world scenarios, such as emotions, they can be added during the attribute constraint generation phase (i.e., Step 1 in Figure 6), without affecting subsequent steps.

5.6 Security Analysis

In Phase I, the forgery generative model generates images according to the LLM's task and then transmits d encrypted images to the LLM. In Phase II, the LLM receives $d+1$ images from the user, which are then passed on to a specific task AI model. Notably, neither the LLM nor the specific task AI model can infer the user's private image by intersecting the sets of images from both phases. This is due to the fact that the images sent to the LLM during Phase I are encrypted. In addition, our threat model assumes that different AI models do not collude.

To prove that our solution satisfies the principle of t-closeness, we introduce the first definition of t-closeness in the domain of computer vision, specifically tailored for a set of images. This novel definition incorporates a privacy requirement designed to safeguard the confidentiality of sensitive features contained in images. According to the foundational concept of t-closeness for a table in Section 5.3,

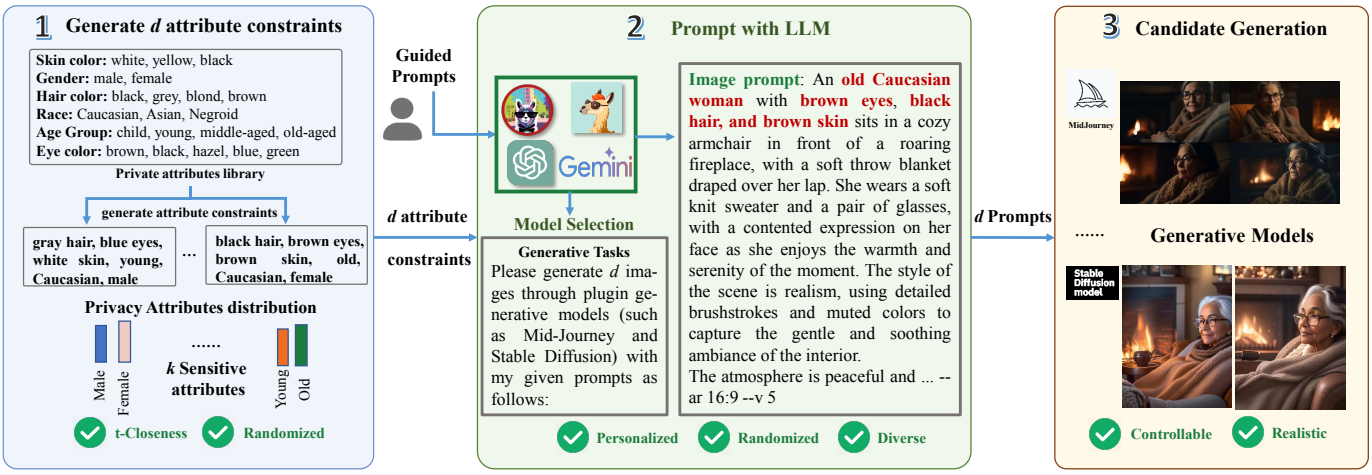

Fig. 6. The process of the forged image generation based on privacy attribute library in Phase I.

we extend t-closeness to the domain of computer vision. Within this context:

- *Explicit identifiers*, *quasi-identifiers*, and *sensitive features* of an image are defined as different privacy attributes derived from the image utilizing attribute classification models (refer to Table 2)[2].
- The *equivalence class* of an image set is defined as a collection of images that share the same values for the quasi-identifiers.
- The core objective of t-closeness for image sets is to prevent the inadvertent disclosure of sensitive features inherent within the image set.

Definition 1. *(The t-closeness principle for a set of images.) If, within an equivalence class, the distribution of a sensitive feature closely aligns with the overall distribution of these features across the entire image set (i.e., the distance between the two distributions does not exceed a threshold t), then the equivalence class is deemed to ensure t-closeness. If this condition holds true for all equivalence classes within the set, then the entire image set is said to have t-closeness.*

According to the classical measurement in t-closeness for a table [54], we also employ the Earth Mover's Distance (EMD) [55] as the measurement to quantify the distribution of sensitive attributes in each equivalence class and the distribution in the entire image set. In a set containing n images, we represent the probability distributions of sensitive features in an equivalence class and the entire set as

$$P = \{(1, p_1), \cdots, (n, p_n)\} = \{(v, p_v)_{v=1}^{n}\} \quad (1)$$
$$Q = \{(1, q_1), \cdots, (n, q_n)\} = \{(w, q_w)_{w=1}^{n}\} \quad (2)$$

where p_v is the weight (probability mass) associated with bin v in distribution P and q_w is the weight linked to bin w in distribution Q. Then the Earth Mover's Distance between P and Q is calculated as

$$\begin{aligned} &\mathrm{EMD}(P, Q) = \min \sum_{v=1}^{n} \sum_{w=1}^{n} d_{vw} f_{vw} \\ &\text{s.t.} \quad f_{vw} \geq 0 \ \text{ for all } \ 1 \leq v \leq n, 1 \leq w \leq n \\ &\qquad p_v - \sum_{w=1}^{n} f_{vw} + \sum_{w=1}^{n} f_{wv} = q_v \ \text{ for all } \ 1 \leq v \leq n \\ &\qquad \sum_{v=1}^{n} \sum_{w=1}^{n} f_{vw} = \sum_{v=1}^{n} p_v = \sum_{w=1}^{n} q_v = 1 \end{aligned} \quad (3)$$

where d_{vw} is the ground distance between bin v and bin w and f_{vw} is a flow variable of how much probability mass is moved from bin v to bin w. The objective is to find the optimal flow values f_{vw} that minimize the Equation (3) while satisfying the constraints.

According to the definition of privacy attributes in images, these attributes are all categorical, with the categorized values being mutually independent. Therefore, we employ the Equal Distance [54] to measure the EMD between two distributions. Within this metric, the distance between any two values within a categorical attribute is defined as 1. This distance is formulated as follows:

$$\mathrm{EMD}(P, Q) = \frac{1}{2} \sum_{v=1}^{n} |p_v - q_v| = \sum_{p_v \geq q_v} (p_v - q_v) \quad (4)$$

Wherein, for each point v, where $p_v - q_v > 0$, any extra is simply redistributed to the other points in the distribution.

Theorem 2. *(t-Closeness for a set of images $x_{\{j \in [d+1]\}}$.) In the shuffling-based solution, both the LLM and the specific task AI model are aware that the private image is in a set of images, i.e., $x_{\{j \in [d+1]\}}$. Furthermore, this set of images satisfies t-closeness.*

Proof. In the shuffling-based solution, image set $x_{\{j \in [d+1]\}}$ satisfy **Definition 1**. We provide the specific t value in the quantitative evaluation of Section 6.3. □

6 Experiments

6.1 Experimental Setup

Datasets. To conduct our evaluation, we construct the user request dataset, as depicted in Figure 7. As described in

2. It is important to note that these categories may change depending on the situation. For instance, in particular situations, quasi-identifiers like age groups may be considered sensitive features if their disclosure could lead to discriminatory practices or other harmful consequences. Similarly, sensitive feature gender could also be a quasi-identifier in certain situations, where it can help to identify an individual in combination with other attributes.

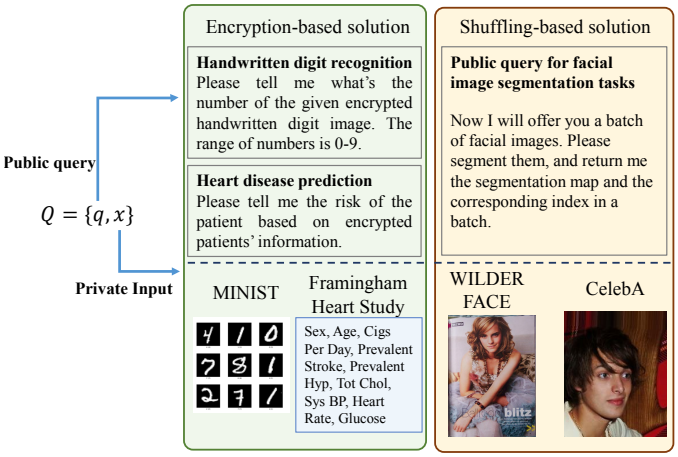

Fig. 7. The request dataset in our experiments.

Section 3, the user request consists of public query q and private input x. We first manually set two query templates as the public query for the encryption-based solution and the shuffling-based solution correspondingly. To ensure our validation is easily reproducible, we populate the private inputs x with public datasets: MNIST [59] and Framingham Heart Study [60] for the encryption-based solution, WILDER FACE [61] and CelebA (CelebFaces Attributes Dataset) [62] for the shuffle-based solution.

MNIST [59] is a 10-class digit image dataset used for image classification tasks. It comprises $60,000$ training images and $10,000$ test images, with each having dimensions of 28 pixels in both height and width.

The Framingham Heart Study [60] is a pivotal dataset utilized for cardiovascular research, originating from an ongoing study focused on the residents of Framingham, Massachusetts. The dataset, which is publicly accessible on the Kaggle website, aims to predict whether a patient has a 10-year risk of future coronary heart disease and encompasses over $4,000$ records and 15 attributes, each being a potential risk factor. These attributes span demographic, behavioral, and medical risk factors, both from historical and current perspectives. We have excluded the attributes of education, current smoker, BPMeds, diabetes, and BMI due to the significant imbalance in their category distributions (e.g., almost all the data is labeled as 0 on BPMeds).

WILDER FACE [61] is a benchmark dataset of face detection, collected from the publicly available WILDER dataset. This collection consists of $32,203$ images with $393,703$ faces, labeled as 61 distinct event classes. These faces exhibit a broad spectrum of variations in terms of scale, pose, and occlusion. Within each event class, data is partitioned into training (40%), validation (10%), and testing (50%) subsets.

CelebA [62] is a large-scale face attributes dataset, which is characterized by a wide range of pose variations and background clutter within the images. This dataset comprises $10,177$ unique identities and a total of $202,599$ face images. Each image is annotated with 5 landmark locations and 40 binary attribute labels.

Large Language Models. Our framework is independent of open-source or closed-source LLMs. So we implement the framework on more difficult settings, i.e., closed-source LLMs (GPT-4 [1], Google Gemini [17] and Claude models [63]) for privacy-preserving tool learning of LLMs. We conduct the inference using the official APIs provided.

TABLE 4
Accuracy (Correctness) of non-privacy baseline and PrivacyAsst's encryption-based solution.

Dataset	Baseline	PrivacyAsst
MNIST	98.25%	98.14%
Framingham Heart Study	69.76%	69.76%

AI Models as Tools. For the encryption-based solution, we employ publicly available AI models as tools, enabling LLMs to perform inference on user requests containing ciphertext. We specifically focus on two tasks: handwritten digit recognition and heart disease prediction. We implement a convolutional neural network, comprising a single convolutional layer followed by two fully connected layers with square activation functions between them, to test handwritten digit recognition. For the heart disease prediction task, we employ a fully connected neural network consisting of one layer. For the shuffling-based solution, we focus on LLMs solving the facial image segmentation task. We employ the BiSeNet [64] model as the tool to complete the segmentation of facial images. To facilitate batch processing of images from user requests, we resize the images to dimensions 256×256 before facial segmentation.

Implementation Details. For the encryption-based solution, we adopted the CKKS [44] as the homomorphic encryption algorithm for secure inference on private data. We leverage the TenSEAL library [41] and follow the parameter setup in the tutorial. We set the security parameter to $\lambda = 26$ for the precision of the fractional component and the polynomial modulus degree to 8192 to set the length of the polynomial. For the shuffling-based solution, we use open-sourced LLM [2], [16] to locally generate text prompts that satisfy our selected facial attribute constraints to generate the shuffling fake facial images. Then, we utilize MidJourney [49] as the text-to-image tool, whose text prompts are generated from the local LLM.

Baselines. We compare our encryption-based solution with non-encryption tool-using LLM agents on the tasks of handwritten digit recognition and heart disease prediction, in which all computations and reasoning are performed directly on the plaintext. For the shuffling-based solution, we draw a comparison with the tool-using LLM agents that conduct inference on a single query, without incorporating any privacy protection mechanism. To ensure a fair and direct comparison, both our solutions and the baselines utilized the same tools, including identical model architectures, model weights, and hyperparameters.

6.2 Evaluation of Encryption-based Solution

Quantitative Results. We compare the performance of our solution with baselines and the results are detailed in Table 4. Our results indicate that our encryption solution yields prediction accuracy that is comparable to baselines, with only a slight drop caused by the approximations introduced during encryption and decryption processes, demonstrating that our solution maintains performance consistency with existing tool-using LLM agents in responding to correct answers to users.

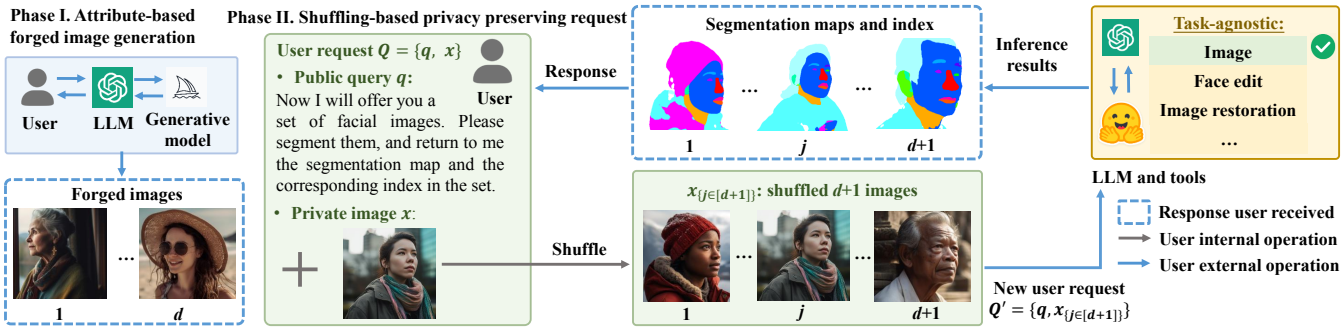

Fig. 8. Qualitative analysis of the shuffling-based solution to the face segmentation task. We omit the encryption and decryption of forged images in Phase I.

TABLE 5
Computational cost/ model inference times (milliseconds) of non-privacy baseline and PrivacyAsst's encryption-based solution.

Dataset	Baseline	PrivacyAsst
MNIST	0.138 ms/sample	152 ms/sample
Framingham Heart Study	0.003 ms/sample	1.78 ms/sample

Our encryption-based solution achieves the three design goals in Section 3.3.1. First, our encryption-based solution guarantees correctness by providing consistent responses, which are equivalent to those obtained from tool-using LLM agents that lack privacy preservation guarantees. Second, our solution does not conflict with existing tool-using LLM agents, which can be extended to open-source and closed-source LLMs, aligning with the generalization objective of PrivacyAsst. Third, both LLM and AI models receive the homomorphic encryption version of the private input, and the private key is owned by the user. Thus, the encryption-based solution is computationally secure.

Communication and Computational Cost. The communication of our encryption-based solution has the complexity of $O(1)$ compared to the non-privacy baseline. We measure the end-to-end running time including the time of transferring homomorphic encryption ciphertexts through the network, as shown in Table 5. The primary expense of our solution lies in the computation of ciphertexts. While homomorphic encryption indeed requires more inference time compared to the baseline, the time taken to infer a single sample remains under 0.2 second. The computational cost is within an acceptable range, ensuring that our solution remains efficient and practical for real-world tool-using LLM agents.

6.3 Evaluation of Shuffling-based Solution

Qualitative Results. Figure 8 shows the qualitative analysis of the face segmentation task. In Phase I, the user's objective is to generate d forged images to obfuscate the private image. The results of the generated forged images are presented (see Section 5.5.1 for more generation details). Notably, to enhance the realism of the generated images, we utilize the local LLM to generate text-to-image prompts, as described in Table 8. As shown in the last row, we create a specific text-to-image prompt, which is subsequently used by simply filling in the attribute constraints. The shuffled images produced by MidJourney, guided by the prompts generated from local LLM, exhibit remarkable realism, as depicted in Figure 8 and Figure 6, featuring diverse content and robust attribute controllability. In Phase II, the user aims to obtain segmentation maps and corresponding index, then select their private image's segmentation maps, i.e., $Q = \{q, x\}$. The user combines x with the forged images to construct a set of $d + 1$ images, which are then shuffled to derive the new user request $Q' = \{q, x_{\{j\in[d+1]\}}\}$. After receiving Q', LLM control tools execute the image segmentation task and provide a response containing inference results to the user.

TABLE 6
Privacy attributes of forged images that have 0.5-closeness w.r.t. gender and 0.8-closeness w.r.t. race

	Skin color	Hair color	Age group	Eye color	Gender	Race
1	White	Black	Middle-aged	Hazel	Male	Negroid
2	Brown	Blond	Child	Brown	Male	Negroid
3	Brown	Black	Child	Brown	Male	Negroid
4	Black	Black	Middle-aged	Brown	Male	Negroid
5	Black	Brown	Child	Hazel	Female	Asian
6	Black	Gray	Young	Blue	Male	Negroid
7	White	Black	Old-aged	Brown	Male	Asian
8	Black	Brown	Young	Blue	Male	Caucasian
9	Black	Blond	Young	Hazel	Female	Asian
10	White	Brown	Old-aged	Blue	Male	Negroid

TABLE 7
The t values of t-closeness w.r.t different sensitive attributes.

Forgery number (d)	Quasi-identifier number	Gender	Race
10	4	0.50	0.80
100	4	0.52	0.70
1000	4	0.51	0.67
1000	2	0.13	0.14
1000	3	0.30	0.37
1000	4	0.51	0.67

Quantitative Evaluation. We give an example of privacy attributes of 10 forged images in Table 6. Next, we employ the pyCANON library [65] to calculate the t values for t-closeness concerning various sensitive attributes, under different numbers of forged images (i.e., d) and quasi-identifiers, as listed in Table 7. A smaller t value signifies a closer resemblance between the distributions, indicating that the sensitive attribute's distribution in each equivalence class is more similar to its distribution in the entire dataset. So a smaller t value represents a stronger level of privacy protection. We note that for the gender attribute, the t value is smaller when $d = 10$ compared to $d = 100$, due to a certain randomness. In general, the results suggest increasing the number of forged images enhances the effectiveness

TABLE 8
Detailed methodology of text-to-image prompts generation, showcasing the interaction between our system and LLM to create prompts for MidJourney (text-to-image generative model). Selected prompts from the prompt library [66] are utilized as our guided prompts.

PrivacyAsst	(Guided Prompts): You will now act as a prompt generator for a generative AI called "Midjourney". Midjourney AI generates images based on given prompts. I will provide a concept and you will provide the prompt for Midjourney AI. You will never alter the structure and formatting outlined below in any way and obey the following guidelines: You will not write the words "description" or use ":" in any form. Never place a comma between [ar] and [v]. You will write each prompt in one line without using return. Structure: - [1] = [[Topic]] - [2] = a detailed description of [1] that will include very specific imagery details. - [3] = with a detailed description describing the environment of the scene. - [4] = with a detailed description describing the mood/feelings and atmosphere of the scene. - [5] = A style, such as photography, painting, illustration, sculpture, Artwork, paperwork, 3d, and more. - [6] = A description of how [5] will be realized. (e.g. Photography (e.g. Macro, Fisheye Style, Portrait) with camera model and appropriate camera settings, Painting with detailed descriptions about the materials and working material used, rendering with engine settings, digital Illustration, Woodburn art (and everything else that could be defined as an output type) - [ar] = "–ar 16:9" if the image looks best horizontally, "–ar 9:16" if the image looks best vertically, "–ar 1:1" if the image looks best in a square. (Use exactly as written) - [v] = If [5] looks best in a Japanese art style use, "–niji". Otherwise, use, "–v 5" (Use exactly as written) Formatting: What you write will be exactly as formatted in the structure below, including the "/" and ":" This is the prompt structure: "/imagine prompt: [1], [2], [3], [4], [5], [6], [ar] [v]". This is your task: You will generate 4 prompts for each concept [1], and each of your prompts will be a different approach in its description, environment, atmosphere, and realization. The prompts you provide will be in English*. Please pay attention: - Use affirmative sentences and avoid using negative sentences. - Describe what you want clearly and avoid using abstract vocabulary. - Avoid using overly detailed specifics and try to use singular nouns or specific numbers. - Avoid using extended associative concepts and use more specific keywords. - Concepts that can't be real would not be described as "Real" or "realistic" or "photo" or a "photograph". for example, a concept that is made of paper or scenes that are fantasy-related. - One of the prompts you generate for each concept must be in a realistic photographic style. you should also choose a lens type and size for it. Don't choose an artist for the realistic photography prompts. - Separate the different prompts with two new lines - Make only one prompt per query and delete the index(such as [6]) If you understand, please respond YES. And the next round I will give you some attributes to generate prompts.
LLM	YES, I understand. Please provide me with the attributes for generating prompts.
PrivacyAsst	black skin, female, brown hair, Negroid, young, brown eyes
LLM	/text-to-image prompt: A young Negroid woman with black skin, brown hair, and warm brown eyes stands in front of a colorful street art mural. She's wearing a bold red jacket and has a confident stance, her hair blowing slightly in the breeze. The atmosphere is lively and artistic, with a sense of creativity and expression in the air. The style is a mix of street art and graphic design, using playful shapes and bright colors to create a dynamic and eye-catching scene. Realized by using a combination of digital and analog techniques, including photography, drawing, and digital software to create a layered and visually interesting texture. –ar 16:9 –v 5

of PrivacyAsst in safeguarding user privacy. An increase in the number of quasi-identifiers weakens privacy protection because it makes individuals more unique and less similar to the distribution of the entire dataset.

Our shuffling-based solution achieves the four design goals in Section 3.3.2. Since the shuffle-based solution keeps the content of each individual request unchanged, it ensures correctness and generalization consistent with that of tool-using LLM agents without privacy protection. Then, due to the randomness of the attributes and our analysis in the quantitative evaluation, the shuffled image set effectively obscures the user's private image, ensuring compliance with t-closeness. Finally, our shuffling-based solution is task-agnostic because it operates independently of specific application domains, allowing for broad adaptability and integration across different tasks.

Communication and Computational Cost. Our shuffling-based solution requires the request and execution of the task on $d+1$ images, as opposed to the baseline solution, which requires only one image. Thus, the communication and computation of our shuffling-based solution is of complexity $O(N)$ compared to the non-privacy baseline. For a more practical evaluation, we assess the runtime performance of our shuffling-based solution on the segmentation task.

TABLE 9
Computational cost/ model inference times (seconds) of non-privacy baseline and PrivacyAsst's shuffling-based solution.

Dataset	Baseline	Number of forgeries d	PrivacyAsst
WILDER FACE	3.67 s	10	4.20 s
		100	13.46 s
		1000	87.85 s
CelebA	3.59 s	10	4.18 s
		100	13.03 s
		1000	87.84 s

We batch the images with a batch size of 8 and measure the execution time of a single run for processing varying numbers of images $d = \{10, 100, 1000\}$. The results in Table 9 show that our solution incurs a minimal increase in inference time especially when $d = \{10, 100\}$, demonstrating its practicality and efficiency for real-world applications without compromising significantly on performance.

Storage (Memory). To evaluate the storage cost on the user's local device, we calculate the total memory cost for obfuscated images in scenarios of $d = \{10, 100, 1000\}$. The results presented in Table 10 reveal that although the induced memory is significantly more than the original image (by a factor of d times), the task itself does not demand

TABLE 10
Total storage/ memory (MB) of non-privacy baseline and PrivacyAsst's shuffling-based solution.

Dataset	Baseline	Number of forgeries d	PrivacyAsst
WILDER FACE	1.42 MB	10	13.7 MB
		100	141057 MB
		1000	1411504 MB
CelebA	1.34 MB	10	14.2 MB
		100	141050 MB
		1000	1411503 MB

substantial memory in baseline scenarios. Consequently, our solution's cost remains acceptable, staying within a 1.4GB limit even when $d = 1000$. This demonstrates that our shuffling-based solution, while introducing an additional memory overhead due to the obfuscation of images, still maintains a feasible and practical memory footprint for real-world deployment, ensuring that user devices are not overly burdened with excessive storage demands during the execution of the desired task.

7 Related Work

Anonymous Communications. These methods safeguard the privacy of user communications by concealing key details, including the sender's identity, the recipient's identity, the time of communication, and its duration [67]. Anonymous communication was originally introduced by Chaum [68] with his proposal to implement an anonymous e-mail service using a *mix network*, which aimed to conceal the identities of message senders and recipients. Building upon the *mix network*, Dingledine et al. [9] introduced the widely used onion-routing anonymity network, i.e., *Tor*. However, such networks are vulnerable to traffic analysis attacks by monitoring user traffic [69]–[71]. Chaum [72] proposed *dc-net*, based on the Dining Cryptographers problem, which can prevent traffic analysis attacks. However, due to the high bandwidth overhead of this method, systems based on *dc-net* could not scale to more than a few dozen clients [73]. Another anonymity network proposed by Kesdogan et al. [74] applies the technique of *private information retrieval (PIR)* [75]. However, this method incurs significant computational overhead and has a weaker threat and system model than other methods [75]. To overcome the significant bandwidth and computational overhead of the aforementioned methods, Kwon et al. [76] proposed *Riffle* using a new hybrid verifiable shuffle technique and PIR.

Nevertheless, previous research has demonstrated that an effective attack can be launched by observing the edges of the anonymous communication networks over a long period of time [76]–[78]. For instance, Berthold et al. [79] proposed the intersection attack, which makes it possible to extract the identity of a message sender by intersecting the sender anonymity sets of successive messages from the same user. Kesdogan et al. [11] proposed the disclosure attack which identifies mutually disjoint sets of message recipients based on the sequence of recipient anonymity sets corresponding to the sender's messages. To reduce the time spent analyzing messages, Danezis and Serjantov [12], [80] developed the statistical disclosure attack, which allows an attacker to identify all potential message recipients for the sender's messages and even determine the exact recipients of specific messages.

Network trace anonymization represents another line of work for anonymizing IP addresses. Xu et al. introduced *Crypto-PAn* [81], a prefix-preserving anonymization technique based on cryptography, aimed at ensuring consistent preservation of prefixes in large distributed scenarios. However, this technique exhibits vulnerability to injection and fingerprinting attacks [82], [83], which leverage frequency analysis to de-anonymize IP addresses. To counter these attacks, Mohammady et al. [84] and Xie et al. [85] proposed *multi-view* methods. These methods generate and analyze multiple anonymized views of the original network traces, preventing adversaries' attempts to recover the same IP across partitions. However, these methods pose challenges due to their complex deployment and the computational overhead, which scales linearly with the number of partitions [84]. These issues render them impractical when processing large amounts of network data [86].

Data Masking. These methods aim to hide or anonymize sensitive information, such as PII or corporate data, by replacing real values with substitutes [87]. For instance, the *blank* method involves deleting sensitive information and replacing the field with NULL. *Substitution* replaces sensitive information with random values that may or may not bear relation to the original data [88]. *Shuffling* entails the random rearrangement of the distinct values within one or more columns [89]. The *number and date variance* method replaces specific values with meaningful ranges and is valuable in fields involving financial and data-driven information.

However, these methods also come with their limitations [90]. *Substitution* may struggle to generate unique random values for each original value, especially in the context of large datasets. *Shuffling* may not be effective when dealing with a limited number of records within the dataset. The *numeric and date variance* method is exclusively applicable to numeric values and cannot be extended to other data formats. Moreover, it is important to note that data masking irreversibly obfuscates sensitive information [91]. So the applicability of this technique is limited to specific tasks. Many tasks require access to plaintext data, including situations involving reversibly ciphertext [92], making data masking unsuitable for such scenarios.

8 Discussion and Limitations

8.1 Extending to LLM-powered Autonomous Agents

LLM-powered autonomous agents are inspiring applications of LLM that aim to build a powerful general problem solver [6]. These agents are designed to autonomously perform a wide variety of tasks, from answering questions to engaging in user interactions, and even mastering complex games. Examples of such agents include such as Auto-GPT [93] and GPT-Engineer [94]. The core functions of these agents typically include three key components: planning, tool utilization, and memory management, effectively extending the capabilities of tool-using LLM agents.

To effectively address a given task, the LLM-powered autonomous agent first leverages the analytical reasoning capabilities inherent in the LLM. This process involves task decomposition, in which the LLM dissects intricate tasks

TABLE 11
True positive rate and false positive rate of fake image detection by GPT-4 across different datasets. Absence of positive (fake) samples in WILDER FACE and CelebA Datasets indicated by "-".

Dataset	True positive rate	False positive rate
WILDER FACE	-	78.7%
CelebA	-	81.1%
Forgery Images	10.8%	89.2%

into smaller, more manageable subgoals, allowing complex tasks to be handled efficiently. Various techniques have been developed for task decomposition, such as chain-of-thought prompting [95], tree-of-thought prompting [96], and LLM+P [97]. Throughout the planning phase, the agent may interact with a memory module, where it can access and retrieve pertinent information. This module serves as a repository for storing and retaining data that aids in task execution. After the planning phase, the agent employs various tools to execute specific tasks and subsequently delivers results — a process similar to tool-using LLM agents.

Our PrivacyAsst framework is also available for enhancing the privacy protection of LLM-powered autonomous agents. When plaintext input is unnecessary for planning, and tasks can be executed using ciphertext, the encryption-based solution can be employed to ensure computational security. For more complex tasks, the shuffling-based solution is able to handle a wide range of agnostic tasks, regardless of the specific planning or tools used, while ensuring that the private input among a set of forged inputs has t-closeness.

8.2 Forgery Detection in Shuffling-based Solution

In the shuffling-based solution, we generate forged images to obfuscate the user's private image, ensuring that the user privacy adheres to t-closeness. While generative models have the capability to generate realistic images, it is noteworthy that existing forgery detection methods [98]–[100] exhibit potential in distinguishing between real and forged images. To assess the threat posed by these methods to our solution—specifically, whether LLMs or tools can detect real user images by filtering out forged ones, we employ GPT-4 [1] (with a standard prompt of '*Is this image real?*') for conducting fake image detection on all images used during a single run of our shuffling-based solution, including both real images and forged images.

The results are presented in Table 11, indicate that indicate that the detection model frequently misclassifies real images as fake. Moreover, out of $1,000$ shuffled images, only 8 samples are detected as fake. This implies that the detection model has limited generalization capabilities, making it challenging for LLMs and tools to derive meaningful information from the detection outcomes. Therefore, we conclude that the current state of fake image detection exhibits limited generalization and thus poses a limited threat to the security of our shuffling-based solution.

8.3 Limitations of Encryption-based Solution

User Knowledgement. In the encryption-based solution, users should possess the knowledge required to generate a public and private key pair (i.e., Step 2 in Figure 3). Users need to understand how to encrypt their private input using a private key (i.e., Step 3 in Figure 3) and decrypt the encrypted results with the private key (i.e., Step 8 in Figure 3).

Application Scope. The encryption-based solution involves performing operations on ciphertexts, rendering it suitable for scenarios where an AI model can be transformed into an encrypted version. Specifically, when employing the TenSEAL library for homomorphic encryption operations, the solution is applicable to scenarios where AI model inference operations fall within a certain set of operations, including element-wise addition, subtraction, multiplication, dot product, and vector-matrix multiplication.

Computational Cost. Task execution on ciphertext results in increased computational costs, as detailed in Table 5 of Section 6.2.

8.4 Limitations of Shuffling-based Solution

User Knowledgement. In the shuffling-based solution, users are required to understand the Key Exchange Protocol (i.e., Step 6 in Figure 5) and the process of decrypting images using symmetric encryption (i.e., Step 7 in Figure 5).

Computational and Communication Cost. The execution of tasks on $d+1$ images leads to increased computational and communication costs, both scaling at $O(N)$ as evaluated in Table 9 of Section 6.3.

Identity Disclosure. While t-closeness offers protection against attribute disclosure, it does not address identity disclosure. Thus, future enhancements to the shuffling-based solution should encourage the development of stronger identity protection mechanisms.

9 Conclusion

In this paper, we propose PrivacyAsst, a generalized privacy-preserving framework designed to safeguard user privacy in tool-using LLM agents, encompassing both open-source and closed-source LLMs and tools. We develop encryption-based and shuffling-based solutions. The former, based on the principle of the homomorphic encryption scheme, assures computational security. While the latter, incorporating forgery generation and attributes shuffling, ensures t-closeness for preserving user privacy. The shuffling-based solution generates privacy-preserving queries by applying a forgery generation model that effectively transforms an individual user input into a set of generation inputs. Coupled with attribute shuffling, this solution demonstrates applicability to a wide range of tasks, thereby accommodating agnostic use cases. Furthermore, this paper underscores the critical importance of user privacy within the swiftly advancing landscape of LLMs and their multifaceted applications. The innovative privacy framework, PrivacyAst, not only safeguards user privacy within the realm of tool-using LLM agents but also paves the way for the development of a more privacy-preserving landscape for artificial intelligence.

References

[1] OpenAI, "Gpt-4," https://openai.com/research/gpt-4, 2023.
[2] H. Touvron, T. Lavril, G. Izacard, X. Martinet, M.-A. Lachaux, T. Lacroix, B. Rozière, N. Goyal, E. Hambro, F. Azhar *et al.*, "Llama: Open and efficient foundation language models," *arXiv preprint arXiv:2302.13971*, 2023.

[3] K. Singhal, S. Azizi, T. Tu, S. S. Mahdavi, J. Wei, H. W. Chung, N. Scales, A. Tanwani, H. Cole-Lewis, S. Pfohl *et al.*, "Large language models encode clinical knowledge," *Nature*, pp. 1–9, 2023.

[4] Y. Huang, J. Shi, Y. Li, C. Fan, S. Wu, Q. Zhang, Y. Liu, P. Zhou, Y. Wan, N. Z. Gong *et al.*, "Metatool benchmark: Deciding whether to use tools and which to use," in *ICLR*, 2023.

[5] Y. Qin, S. Liang, Y. Ye, K. Zhu, L. Yan, Y. Lu, Y. Lin, X. Cong, X. Tang, B. Qian *et al.*, "Toolllm: Facilitating large language models to master 16000+ real-world apis," *ICLR*, 2024.

[6] L. Weng, "Llm-powered autonomous agents," *lilianweng.github.io*, Jun 2023. [Online]. Available: https://lilianweng.github.io/posts/2023-06-23-agent/

[7] Y. Shen, K. Song, X. Tan, D. Li, W. Lu, and Y. Zhuang, "Hugginggpt: Solving ai tasks with chatgpt and its friends in hugging face," *NeurIPS*, vol. 36, 2024.

[8] I. Martínez Toro, D. Gallego Vico, and P. Orgaz, "PrivateGPT," May 2023. [Online]. Available: https://github.com/imartinez/privateGPT

[9] R. Dingledine, N. Mathewson, P. F. Syverson *et al.*, "Tor: The second-generation onion router." in *USENIX security symposium*, vol. 4, 2004, pp. 303–320.

[10] "Privategpt: The privacy layer for chatgpt by private ai," https://www.private-ai.com/products/privategpt-chatbot/.

[11] D. Kedogan, D. Agrawal, and S. Penz, "Limits of anonymity in open environments," in *Information Hiding: 5th International Workshop, Revised Papers 5*. Springer, 2003, pp. 53–69.

[12] G. Danezis, "Statistical disclosure attacks: Traffic confirmation in open environments," in *IFIP TC11 18 th International Conference on Information Security (SEC2003)*. Springer, 2003, pp. 421–426.

[13] X. Zhang, H. Hong, Y. Hong, P. Huang, B. Wang, Z. Ba, and K. Ren, "Text-crs: A generalized certified robustness framework against textual adversarial attacks," in *2024 IEEE Symposium on Security and Privacy (SP)*. IEEE Computer Society, 2023, pp. 53–53.

[14] D. He, Y. Xia, T. Qin, L. Wang, N. Yu, T.-Y. Liu, and W.-Y. Ma, "Dual learning for machine translation," *Advances in neural information processing systems*, vol. 29, 2016.

[15] R. van Zoest, "6 nlp tasks for natural language generation — innerdoc.com — medium," https://medium.com/innerdoc/6-nlp-tasks-for-natural-language-generation-8a2a1dead1df, 2021.

[16] H. Touvron, L. Martin, K. Stone, P. Albert, A. Almahairi, Y. Babaei, N. Bashlykov, S. Batra, P. Bhargava, S. Bhosale *et al.*, "Llama 2: Open foundation and fine-tuned chat models," *arXiv preprint arXiv:2307.09288*, 2023.

[17] "Gemini," https://gemini.google.com/app, 2024.

[18] W. X. Zhao, K. Zhou, J. Li, T. Tang, X. Wang, Y. Hou, Y. Min, B. Zhang, J. Zhang, Z. Dong *et al.*, "A survey of large language models," *arXiv preprint arXiv:2303.18223*, 2023.

[19] S. Bubeck, V. Chandrasekaran, R. Eldan, J. Gehrke, E. Horvitz, E. Kamar, P. Lee, Y. T. Lee, Y. Li, S. Lundberg, H. Nori, H. Palangi, M. T. Ribeiro, and Y. Zhang, "Sparks of artificial general intelligence: Early experiments with gpt-4," 2023.

[20] J. White, Q. Fu, S. Hays, M. Sandborn, C. Olea, H. Gilbert, A. Elnashar, J. Spencer-Smith, and D. C. Schmidt, "A prompt pattern catalog to enhance prompt engineering with chatgpt," *arXiv preprint arXiv:2302.11382*, 2023.

[21] M. Komeili, K. Shuster, and J. Weston, "Internet-augmented dialogue generation," in *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics*, 2022, pp. 8460–8478.

[22] T. Schick, J. Dwivedi-Yu, R. Dessì, R. Raileanu, M. Lomeli, E. Hambro, L. Zettlemoyer, N. Cancedda, and T. Scialom, "Toolformer: Language models can teach themselves to use tools," *NeurIPS*, vol. 36, 2024.

[23] Y. Mehdi, "Reinventing search with a new ai-powered microsoft bing and edge, your copilot for the web - the official microsoft blog," https://blogs.microsoft.com/blog/2023/02/07/reinventing-search-with-a-new-ai-powered-microsoft-bing-and-\edge-your-copilot-for-the-web/, Feb 7, 2023.

[24] W. Huang, C. Wang, R. Zhang, Y. Li, J. Wu, and L. Fei-Fei, "Voxposer: Composable 3d value maps for robotic manipulation with language models," in *Conference on Robot Learning*. PMLR, 2023, pp. 540–562.

[25] G. Wang, Y. Xie, Y. Jiang, A. Mandlekar, C. Xiao, Y. Zhu, L. Fan, and A. Anandkumar, "Voyager: An open-ended embodied agent with large language models," *arXiv:2305.16291*, 2023.

[26] "Mantella (chatgpt in skyrim vr) - improved voices & long-term memory update : r/skyrimvr," https://www.reddit.com/r/skyrimvr/comments/14cp2sp/mantella_chatgpt_in_skyrim_vr_improved_voices/.

[27] T. Cai, X. Wang, T. Ma, X. Chen, and D. Zhou, "Large language models as tool makers," *ICLR*, 2024.

[28] N. Lukas, A. Salem, R. Sim, S. Tople, L. Wutschitz, and S. Zanella-Béguelin, "Analyzing leakage of personally identifiable information in language models," in *2023 IEEE Symposium on Security and Privacy (SP)*. IEEE Computer Society, 2023, pp. 346–363.

[29] H. Yao, J. Lou, K. Ren, and Z. Qin, "Promptcare: Prompt copyright protection by watermark injection and verification," *arXiv preprint arXiv:2308.02816*, 2023.

[30] A. Moktali, "Generative ai data privacy with skyflow llm privacy vault - skyflow," https://www.skyflow.com/post/generative-ai-data-privacy-skyflow-llm-privacy-vault, 2023.

[31] L. Sun, Y. Huang, H. Wang, S. Wu, Q. Zhang, C. Gao, Y. Huang, W. Lyu, Y. Zhang, X. Li *et al.*, "Trustllm: Trustworthiness in large language models," *arXiv preprint arXiv:2401.05561*, 2024.

[32] N. Carlini, F. Tramer, E. Wallace, M. Jagielski, A. Herbert-Voss, K. Lee, A. Roberts, T. Brown, D. Song, U. Erlingsson *et al.*, "Extracting training data from large language models," in *30th USENIX Security Symposium*, 2021, pp. 2633–2650.

[33] H. A. Inan, O. Ramadan, L. Wutschitz, D. Jones, V. Rühle, J. Withers, and R. Sim, "Training data leakage analysis in language models," *arXiv preprint arXiv:2101.05405*, 2021.

[34] H. Duan, A. Dziedzic, M. Yaghini, N. Papernot, and F. Boenisch, "On the privacy risk of in-context learning," in *The 61st Annual Meeting Of The Association For Computational Linguistics*, 2023.

[35] S. Kim, S. Yun, H. Lee, M. Gubri, S. Yoon, and S. J. Oh, "Propile: Probing privacy leakage in large language models," in *NeurIPS*, 2024.

[36] L. Ekenstam, "Linus on x: "if you think this is creepy… you should know what all major stores in the world knows about you as a shopper they use in-store, online, cell-tower, and more to keep track of everything… everything. cambridge analytics is a joke in comparison." https://twitter.com/LinusEkenstam/status/1692602911518343502, Aug 19, 2023.

[37] K. S. Balagani, M. Conti, P. Gasti, M. Georgiev, T. Gurtler, D. Lain, C. Miller, K. Molas, N. Samarin, E. Saraci *et al.*, "Silk-tv: Secret information leakage from keystroke timing videos," in *23rd European Symposium on Research in Computer Security, ESORICS 2018, Proceedings, Part I 23*. Springer, 2018, pp. 263–280.

[38] J. Chen, H. Chen, E. Bauman, Z. Lin, B. Zang, and H. Guan, "You shouldn't collect my secrets: Thwarting sensitive keystroke leakage in mobile ime apps," in *24th USENIX Security Symposium*, 2015, pp. 657–690.

[39] C. Ren, Y. Zhang, H. Xue, T. Wei, and P. Liu, "Towards discovering and understanding task hijacking in android," in *24th USENIX Security Symposium*, 2015, pp. 945–959.

[40] H. Face, "Hugging face – the ai community building the future." https://huggingface.co/.

[41] A. Benaissa, B. Retiat, B. Cebere, and A. E. Belfedhal, "Tenseal: A library for encrypted tensor operations using homomorphic encryption," 2021.

[42] Z. Brakerski, "Fully homomorphic encryption without modulus switching from classical gapsvp," in *Annual Cryptology Conference*. Springer, 2012, pp. 868–886.

[43] J. Fan and F. Vercauteren, "Somewhat practical fully homomorphic encryption," *Cryptology ePrint Archive*, 2012.

[44] J. H. Cheon, A. Kim, M. Kim, and Y. Song, "Homomorphic encryption for arithmetic of approximate numbers," in *Advances in Cryptology-ASIACRYPT 2017: 23rd International Conference on the Theory and Applications of Cryptology and Information Security, Proceedings, Part I 23*. Springer, 2017, pp. 409–437.

[45] J. W. Bos, K. Lauter, J. Loftus, and M. Naehrig, "Improved security for a ring-based fully homomorphic encryption scheme," in *Cryptography and Coding: 14th IMA International Conference, IMACC 2013. Proceedings 14*. Springer, 2013, pp. 45–64.

[46] R. Lindner and C. Peikert, "Better key sizes (and attacks) for lwe-based encryption," in *Topics in Cryptology–CT-RSA 2011: The Cryptographers' Track at the RSA Conference 2011, San Francisco, CA, USA, February 14-18, 2011. Proceedings*. Springer, 2011, pp. 319–339.

[47] C. Gentry, S. Halevi, and N. P. Smart, "Homomorphic evaluation of the aes circuit," in *Annual Cryptology Conference*. Springer, 2012, pp. 850–867.

[48] "Dall·e 3," https://openai.com/dall-e-3.

[49] "Midjourney," https://www.midjourney.com/home/?callbackUrl=%2Fapp%2F.

[50] "Stable diffusion — stability ai," https://stability.ai/stable-diffusion.

[51] W. Diffie and M. E. Hellman, "New directions in cryptography," in *Democratizing Cryptography: The Work of Whitfield Diffie and Martin Hellman*, 2022, pp. 365–390.

[52] L. Sweeney, "k-anonymity: A model for protecting privacy," *International journal of uncertainty, fuzziness and knowledge-based systems*, vol. 10, no. 05, pp. 557–570, 2002.

[53] A. Machanavajjhala, D. Kifer, J. Gehrke, and M. Venkitasubramaniam, "l-diversity: Privacy beyond k-anonymity," *ACM Transactions on Knowledge Discovery from Data (TKDD)*, vol. 1, no. 1, pp. 3–es, 2007.

[54] N. Li, T. Li, and S. Venkatasubramanian, "t-closeness: Privacy beyond k-anonymity and l-diversity," in *2007 IEEE 23rd international conference on data engineering*. IEEE, 2006, pp. 106–115.

[55] Y. Rubner, C. Tomasi, and L. J. Guibas, "The earth mover's distance as a metric for image retrieval," *International journal of computer vision*, vol. 40, pp. 99–121, 2000.

[56] T. Orekondy, B. Schiele, and M. Fritz, "Towards a visual privacy advisor: Understanding and predicting privacy risks in images," in *ICCV*, 2017, pp. 3686–3695.

[57] C. Dwyer, S. Hiltz, and K. Passerini, "Trust and privacy concern within social networking sites: A comparison of facebook and myspace," *AMCIS 2007 proceedings*, p. 339, 2007.

[58] E. McCallister, *Guide to protecting the confidentiality of personally identifiable information*. Diane Publishing, 2010, vol. 800, no. 122.

[59] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[60] A. Bhardwaj, "Framingham heart study dataset," https://www.kaggle.com/datasets/aasheesh200/framingham-heart-study-dataset.

[61] S. Yang, P. Luo, C.-C. Loy, and X. Tang, "Wider face: A face detection benchmark," in *CVPR*, 2016, pp. 5525–5533.

[62] Z. Liu, P. Luo, X. Wang, and X. Tang, "Deep learning face attributes in the wild," in *ICCV*, 2015, pp. 3730–3738.

[63] Anthropic, "Anthropic claude 2," https://www.anthropic.com/index/claude-2, July 11 2023.

[64] C. Yu, J. Wang, C. Peng, C. Gao, G. Yu, and N. Sang, "Bisenet: Bilateral segmentation network for real-time semantic segmentation," in *ECCV*, 2018, pp. 325–341.

[65] J. Sáinz-Pardo Díaz and Á. López García, "A python library to check the level of anonymity of a dataset," *Scientific Data*, vol. 9, no. 1, p. 785, 2022.

[66] SinCode, "Prompt library - by sincode ai," https://www.sincode.ai/prompt-tool/midjournei-prompt-generator, 2023.

[67] F. Shirazi, M. Simeonovski, M. R. Asghar, M. Backes, and C. Diaz, "A survey on routing in anonymous communication protocols," *ACM Computing Surveys (CSUR)*, vol. 51, no. 3, pp. 1–39, 2018.

[68] D. L. Chaum, "Untraceable electronic mail, return addresses, and digital pseudonyms," *Communications of the ACM*, vol. 24, no. 2, pp. 84–90, 1981.

[69] J.-F. Raymond, "Traffic analysis: Protocols, attacks, design issues, and open problems," in *Designing Privacy Enhancing Technologies: International Workshop on Design Issues in Anonymity and Unobservability, 2000 Proceedings*. Springer, 2001, pp. 10–29.

[70] A. Back, U. Möller, and A. Stiglic, "Traffic analysis attacks and trade-offs in anonymity providing systems," in *International Workshop on Information Hiding*. Springer, 2001, pp. 245–257.

[71] S. J. Murdoch and G. Danezis, "Low-cost traffic analysis of tor," in *2005 IEEE Symposium on Security and Privacy (S&P'05)*. IEEE, 2005, pp. 183–195.

[72] D. Chaum, "The dining cryptographers problem: Unconditional sender and recipient untraceability," *Journal of cryptology*, vol. 1, pp. 65–75, 1988.

[73] H. Corrigan-Gibbs and B. Ford, "Dissent: accountable anonymous group messaging," in *Proceedings of the 17th ACM conference on Computer and communications security*, 2010, pp. 340–350.

[74] D. Kesdogan, M. Borning, and M. Schmeink, "Unobservable surfing on the world wide web: is private information retrieval an alternative to the mix based approach?" in *Privacy Enhancing Technologies: Second International Workshop, PET 2002, Revised Papers 2*. Springer, 2003, pp. 224–238.

[75] B. Chor, E. Kushilevitz, O. Goldreich, and M. Sudan, "Private information retrieval," *Journal of the ACM (JACM)*, vol. 45, no. 6, pp. 965–981, 1998.

[76] A. Kwon, D. Lazar, S. Devadas, and B. A. Ford, "Riffle: An efficient communication system with strong anonymity," in *Proceedings on Privacy Enhancing Technologies Symposium*, vol. 2016, no. 2, 2016, pp. 115–134.

[77] N. Mathewson and R. Dingledine, "Practical traffic analysis: Extending and resisting statistical disclosure," in *International Workshop on Privacy Enhancing Technologies*. Springer, 2004, pp. 17–34.

[78] G. Danezis and C. Diaz, "A survey of anonymous communication channels," 2008.

[79] O. Berthold, A. Pfitzmann, and R. Standtke, "The disadvantages of free mix routes and how to overcome them," in *Designing Privacy Enhancing Technologies: International Workshop on Design Issues in Anonymity and Unobservability, 2000 Proceedings*. Springer, 2001, pp. 30–45.

[80] G. Danezis and A. Serjantov, "Statistical disclosure or intersection attacks on anonymity systems," in *International Workshop on Information Hiding*. Springer, 2004, pp. 293–308.

[81] J. Xu, J. Fan, M. H. Ammar, and S. B. Moon, "Prefix-preserving ip address anonymization: Measurement-based security evaluation and a new cryptography-based scheme," in *10th IEEE International Conference on Network Protocols, 2002. Proceedings.* IEEE, 2002, pp. 280–289.

[82] T. Brekne, A. Årnes, and A. **O**slebø, "Anonymization of ip traffic monitoring data: Attacks on two prefix-preserving anonymization schemes and some proposed remedies," in *International Workshop on Privacy Enhancing Technologies*. Springer, 2005, pp. 179–196.

[83] T.-F. Yen, X. Huang, F. Monrose, and M. K. Reiter, "Browser fingerprinting from coarse traffic summaries: Techniques and implications," in *6th International Conference, 2009. Proceedings 6*. Springer, 2009, pp. 157–175.

[84] M. Mohammady, L. Wang, Y. Hong, H. Louafi, M. Pourzandi, and M. Debbabi, "Preserving both privacy and utility in network trace anonymization," in *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security*, 2018, pp. 459–474.

[85] S. Xie, M. Mohammady, H. Wang, L. Wang, J. Vaidya, and Y. Hong, "A generalized framework for preserving both privacy and utility in data outsourcing," *IEEE Transactions on Knowledge and Data Engineering*, vol. 35, no. 1, pp. 1–15, 2021.

[86] A. Aleroud, F. Yang, S. C. Pallaprolu, Z. Chen, and G. Karabatis, "Anonymization of network traces data through condensation-based differential privacy," *Digital Threats: Research and Practice (DTRAP)*, vol. 2, no. 4, pp. 1–23, 2021.

[87] M. Cobb, "What is data masking? techniques, types and best practices," https://www.techtarget.com/searchsecurity/definition/data-masking#:~:text=Data%20masking%20is%20a%20method,real%20data%20is%20not%20required.

[88] X.-B. Li, L. Motiwalla *et al.*, "Protecting patient privacy with data masking," in *WISP*, 2009, pp. 7–11.

[89] DATPROF, "Data masking: techniques & best practices," https://www.datprof.com/data-masking/.

[90] G. Sarada, N. Abitha, G. Manikandan, and N. Sairam, "A few new approaches for data masking," in *ICCPCT*, 2015, pp. 1–4.

[91] "Data masking definition," https://mask-me.net/datamaskingwiki/wiki/26/data-masking-definition.

[92] J. Simpson, "Data masking and encryption are different," https://www.iri.com/blog/data-protection/data-masking-and-data-encryption-are-not-the-same-things/.

[93] "Significant-gravitas/autogpt: An experimental open-source attempt to make gpt-4 fully autonomous." https://github.com/Significant-Gravitas/AutoGPT.

[94] "Antonosika/gpt-engineer: Specify what you want it to build, the ai asks for clarification, and then builds it." https://github.com/AntonOsika/gpt-engineer.

[95] J. Wei, X. Wang, D. Schuurmans, M. Bosma, F. Xia, E. Chi, Q. V. Le, D. Zhou *et al.*, "Chain-of-thought prompting elicits reasoning in large language models," *NeurIPS*, vol. 35, pp. 24 824–24 837, 2022.

[96] S. Yao, D. Yu, J. Zhao, I. Shafran, T. L. Griffiths, Y. Cao, and K. Narasimhan, "Tree of thoughts: Deliberate problem solving with large language models," *ICLR*, 2024.

[97] B. Liu, Y. Jiang, X. Zhang, Q. Liu, S. Zhang, J. Biswas, and P. Stone, "Llm+p: Empowering large language models with optimal planning proficiency," *arXiv:2304.11477*, 2023.
[98] Z. Ba, Q. Liu, Z. Liu, S. Wu, F. Lin, and K. Ren, "Exposing the deception: Uncovering more forgeries clues for deepfake detection," in *AAAI*, 2024, pp. 51–51.
[99] C. Shuai, J. Zhong, S. Wu, F. Lin, Z. Wang, Z. Ba, Z. Liu, L. Cavallaro, and K. Ren, "Locate and verify: A two-stream network for improved deepfake detection," in *Proceedings of the 31st ACM International Conference on Multimedia*, 2023, pp. 7131–7142.
[100] K. Pan, Y. Yin, Y. Wei, F. Lin, Z. Ba, Z. Liu, Z. Wang, L. Cavallaro, and K. Ren, "Dfil: Deepfake incremental learning by exploiting domain-invariant forgery clues," in *Proceedings of the 31st ACM International Conference on Multimedia*, 2023, pp. 8035–8046.

Xinyu Zhang received the B.E. degree in information security from Xidian University, Xi'an, China, in 2019. She is currently pursuing the Ph.D. degree at the School of Cyber Science and Technology, College of Computer Science and Technology, Zhejiang University, Hangzhou, China. Her research interests include trustworthy AI, federated learning, and data privacy.

Huiyu Xu received the B.E. degree in Information Security from Nankai University, China, in 2021. He is currently pursuing his Ph.D at School of Cyber Science and Technology, Zhejiang University, China. His research interest focuses on privacy protection and AI security.

Zhongjie Ba (Member, IEEE) received the PhD degree in computer science and engineering from the State University of New York at Buffalo, USA, in 2019. He is currently a professor at the School of Cyber Science and Technology, College of Computer Science and Technology, Zhejiang University, China. He was a postdoctoral researcher with the School of Computer Science, McGill University, Canada. His current research interests include the security and privacy aspects of the Internet of Things, forensic analysis of multimedia content, and privacy-enhancing technologies in the context of collaborative deep learning. Results have been published in peer-reviewed top conferences and journals, including S&P, CCS, NDSS, INFOCOM, ICDCS, and the IEEE Transactions on Information Forensics and Security. Currently, he serves as an associate editor of the IEEE Internet of Things Journal and the technical program committee of several conferences in the field of Internet of Things and wireless communication.

Zhibo Wang (Senior Member, IEEE) received the B.E. degree in automation from Zhejiang University, Hangzhou, China, in 2007, and the Ph.D degree in electrical engineering and computer science from the University of Tennessee, Knoxville, TN, USA, in 2014. He is currently a Professor at the School of Cyber Science and Technology, College of Computer Science and Technology, Zhejiang University. His research interests include AI security, the Internet of Things, network security, and privacy protection.

Yuan Hong (Senior Member, IEEE) received the PhD degree in information technology from Rutgers University. He is currently an associate professor with the Department of Computer Science and Engineering, University of Connecticut. His research interests focus on data privacy, trustworthy AI, cyber-physical systems security and privacy, and optimization. He is a recipient of the National Science Foundation (NSF) CAREER Award and Cisco Research Award.

Jian Liu received the PhD degree from Aalto University, in July 2018. He is a ZJU100 Young professor with Zhejiang University. Before that, he was a post-doctoral researcher with UC Berkeley. His research is on applied cryptography, distributed systems, blockchains, and machine learning. He is interested in building real-world systems that are provably secure, easy to use and inexpensive to deploy.

Zhan Qin (Senior Member, IEEE) is currently a ZJU100 Young Professor, with both the College of Computer Science and Technology and the School of Cyber Science and Technology at Zhejiang University, China. He was an assistant professor at the Department of Electrical and Computer Engineering at the University of Texas at San Antonio after receiving the Ph.D. degree from the Computer Science and Engineering Department at the State University of New York at Buffalo in 2017. His current research interests include data security and privacy, secure computation outsourcing, artificial intelligence security, and cyber-physical security in the context of the Internet of Things. His works explore and develop novel security sensitive algorithms and protocols for computation and communication in the general context of Cloud and Internet devices.

Kui Ren (Fellow, IEEE) received the PhD degree in electrical and computer engineering from Worcester Polytechnic Institute. He is the Dean and Professor of the College of Computer Science and Technology, at Zhejiang University, where he also directs the Institute of Cyber Science and Technology. Before that, he was SUNY Empire Innovation professor at the State University of New York at Buffalo, USA. His current research interests include data security, IoT security, AI security, and privacy. He received many recognitions including Guohua Distinguished Scholar Award of ZJU, IEEE CISTC Technical Recognition Award, SUNY Chancellor's Research Excellence Award, Sigma Xi Research Excellence Award, NSF CAREER Award, etc. He has published extensively in peer-reviewed journals and conferences and received the Test-of-time Paper Award from IEEE INFOCOM and many Best Paper Awards, including ACM MobiSys, IEEE ICDCS, IEEE ICNP, IEEE Globecom, ACM/IEEE IWQoS, etc. His h-index is 93, with a total citation exceeding 48,000 according to Google Scholar. He is a fellow of the ACM. He is a frequent reviewer for funding agencies internationally and serves on the editorial boards of many IEEE and ACM journals. Among others, he currently serves as chair of SIGSAC of ACM China Council, a member of ACM ASIACCS steering committee, and a member of S&T Committee of Ministry of Education of China.