Research Plan: Secure Delegation of Computation for Distributed Services

Yu-Chi Chen

Yuan Ze University

wycchen@saturn,yzu.edu.tw

1 Research background

The developments of information technology always go beyond our expectation. With the significant progression of computer architecture[1], computers have become very powerful and gradually changed the life of human being. Simultaneously, contributed by the progress of network efficiency and quality, the mobiles also have been widely used in the last decade, since their price became much more acceptable (even the new iPhone XS is not really cheap). Recently, many exciting techniques and applications of computer science and engineering (i.e., artificial intelligence, cryptocurrency, e-voting, ... etc) were introduced to gradually digitalize our real life and also bring us to an incredible level. As an interface of providing services, cloud computing always acts as a paradigm where the user can ubiquitously access to shared resources and enjoy higher services offered by the cloud server, often over the Internet. In fact, many IT companies such as Amazon Web Services (AWS), Oracle, Microsoft and Google paid lots of effort to develop cloud services.

There are some trade-off of cloud service and personal workstation. Renting cloud servers for a time slot must spend huge and fixed cost (no matter they are idle or not). However, workstations are needed to be maintained attentively and also take the risk. Both of them requires *constant* cost and risk, which implies that if those resources are not fully applied, they waste money and time without doubt. There is a new type of services, so-called distributed services ($\mathcal{DS}$) [7, 11, 43], which is implemented by a distributed model. Roughly speaking, in the model, a service is distributed across one or more service providers. $\mathcal{DS}$ is sort of similar to cloud services but the service provider is not a fixed party in $\mathcal{DS}$. In particular, they are referred to as a new solution that does not require constant cost. Indeed, the cost only depends on the use of services. As a result, in $\mathcal{DS}$, there are many providers would like to serve, but they are in a competitive or collaborative relationship[2]. Smart contracts [] that run on the top

[1] 2017 Turing Award winners, John L. Hennessy and David A. Patterson, for pioneering a systematic, quantitative approach to the design and evaluation of computer architectures with enduring impact on the microprocessor industry.

[2] Relationship exactly depends on the service.

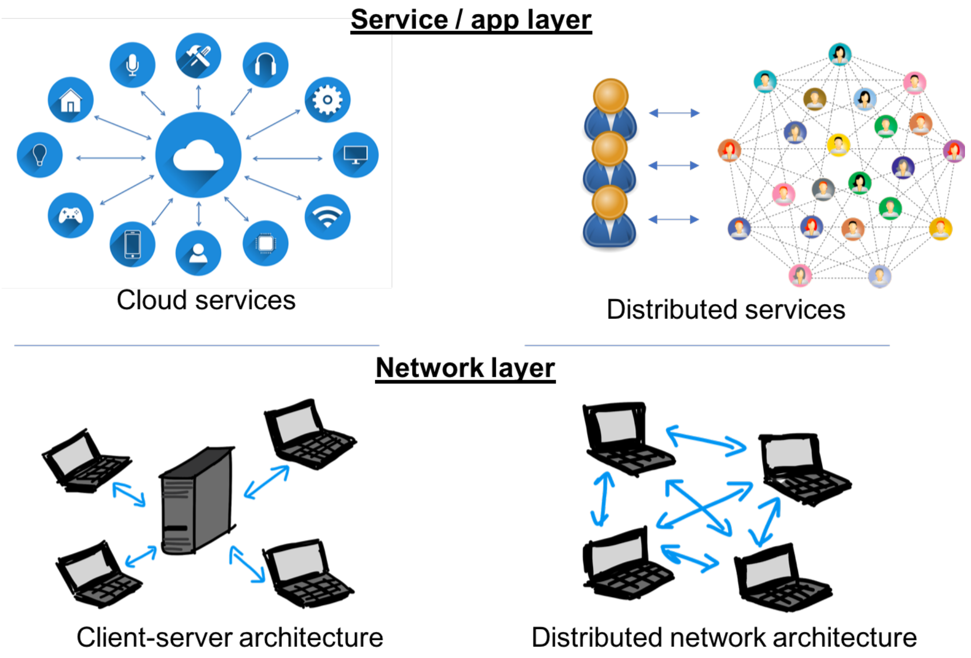


Figure 1: Model description and comparison.

of blockchain are viewed as a useful service interface, and require only on-line cost. More precisely, the on-line cost is the fee induced by changing the state of a smart contract. We believe that in the next generation, the trends lead the cloud services to the distributed services. One direction comes from blockchain revolution.

Achieving distributed services, distributed network architecture is an ideal model in which components are located on networked computers communicate and coordinate their actions by passing messages. The components interact with each other in order to achieve a goal. Three significant characteristics of distributed systems are: concurrency of components (like parallelism), lack of a global clock, and independent failure of components [37]. They makes the system work well, and provide consistency for offering the correct functionality. As known, some systems in distributed network are like service-oriented architecture-based systems, massively multiplayer online games, and peer-to-peer applications. In addition to theoretical side, lots of distributed system projects (i.e., Climateprediction.net, Quake-Catcher Network, and Folding@home [1, 2, 3]) rely on volunteers' idle computers in the world to help to realize a goal with a collaborative manner.

Secure delegation of computation in distributed services. Delegating computation has been discussed in the traditional cloud service model, and in the past, privacy, confidentiality, authentication, and some other security issues have also been well-studied. The main direction of this proposal is toward realizing secure delegation of computation in distributed services. The original security tasks in the cloud service model can be reconsidered in the distributed service model.

With distributed components in distributed network setting, some conventional issues (i.e., privacy, authentication, ...) will be re-considered, since the potential adversaries' power is different. In fact, there are some others that are only discussed in distributed network (i.e., consensus). With the identical analogy, the security tasks of cloud services will move to those of distributed services.

In the following, we will very briefly highlight a few tasks (relevant to this proposal and my previous research).

- **Privacy**: Provide confidentiality for distributed services. Secure delegating computation is also under the consideration of privacy of distributed services if the underlying computation model is implemented by the distributed service model. Moreover, anonymity is also a kind of privacy for hiding user identity.
- **Authentication**: Provide integrity and identification of users for distributed services. Moreover, certificates in the such setting is also considered as the authentication issue.
- **Consensus**: Provide an agreement or protocol to enforce every party to follow, if there are multiple parties jointly require the distributed service.

This suffices to point out the following fundamental question we will tackle in this proposal.

> *Can we design efficient solutions that achieve privacy, authentication, or consensus for specific distributed service models.*

Here we emphasize specific distributed service, since for general models, we already have many clever solutions for privacy and authentication.

In the past five years, I have worked on user identification [27] in distributed network models and *certificateless signatures* [25, 32, 33, 34] which efficiently provide the PKI-like authentication. I have worked on computation privacy in general parallel RAM models in the past three years, where joint with my collaborators, we initiated the research of *cryptography for the parallel computation model* [4, 22], developed some basic tools, and provided the feasibility result for many cryptographic tasks. These results somehow give us solutions to computation privacy, but the disadvantage of them is based on "non-standard" hardness (postpone to discuss this part). I have also worked on *secret sharing* for a long time, where my collaborators and I designed some solutions to improve efficiency and prevent cheating attacks [21, 26, 28, 30, 31], and I consider techniques of secret sharing could be used to overcome some issues in distributed network models, which is exactly an ambitious goals we desire to study in this project. In addition, I have also worked on privacy preserving search and on-line shopping in the cloud storage [20, 23, 24, 51, 52]. Our recent works and results were supported by the MOST grant (106-2218-E-155-008-MY3) 分散式網路架構下保有隱私一致性之計算–新的任務、挑戰與應用that mainly focuses on generic solutions to secure data gathering and privacy in distributed network architecture. This also motivates this project direction to work on security for distributed services.

However, in distributed service models, many interesting research questions or directions remain open or not well-studied yet. I plan to lead my research group to tackle these questions and further develop the solutions in the next five years, and in particular to find new research directions. In general, security of distributed service models is a huge field where we are not able to consider all of security issues, and thus this proposal aims for the way to solving questions we saw and discussing any potential issues we were not aware of now (here we will understand the such issues, state the problems, and try to develop new full-fledged solutions and methods or prove impossibility if needed). Even we have worked on security in the distributed network model for couple of years, the distributed service model is a new area for us. Thus, in the proposal we will start from some secure delegated computation in cloud services, and step by step move to that in distributed services. Let us describe the background and problem statements below.

1.1 Statement of research problems

In this proposal, we start from our some ongoing works (years 1 and 2 focuses) in secure delegated computation in cloud services. Based on the proposed techniques, we will move to a specific direction of preserving privacy in distributed services (years 2 and 3 focuses). Our final goals are to realize secure delegated computation in distributed services (years 4 and 5 focuses). By the way, looking for potential research topics and keeping open problem in mind are always our ambitious goals without any plan, schedule, or time constraint.

Research Problem 1 (Years 1 and 2): Secure Delegated Computation in Cloud Services

We elaborate the topics and ongoing works for secure delegated computation in cloud services.

Privacy-preserving outsourced similarity test (PPOS). Privacy-preserving outsourced similarity test (PPOS) is a simple application of cloud storage. It is introduced to meet a specific scenario where the user A (denoted by $\mathbb{A}$) uploads the encrypted original data to the sharing cloud, and the encrypted feature to the searching cloud. Particularly, such the feature is extracted from the original data. The user B (by $\mathbb{B}$) searches data by producing the search request from search feature and then sends the access token to the searching cloud. Then, the search cloud will return the result of similarity test between the search feature and the original feature; i.e., L2-norm). $\mathbb{B}$ takes the result to retrieve data from the sharing cloud.

Zhang et al. [91, 92] proposed a PPOS scheme based on ciphertext-policy attribute based encryption (CP-ABE) [12, 41, 49, 59, 82], additive homomorphic encryption (HE) [5, 15, 16, 46, 72, 80] and garbled circuits [10, 44, 47, 48, 53, 58, 62, 88]. We briefly summarized their main idea in their protocol.

- $\mathbb{A}$ generates a public key, a secret key of HE, and a garbling key. Then he runs key generation for CP-ABE, sets the access structure, and gives $\mathbb{B}$ an access key (note that

$\mathbb{B}$ can perform decryption of CP-ABE for one ciphertext if his access key matches the attributes of it).

- $\mathbb{A}$ uploads the ciphertext of key of HE and the garbling key encrypted by CP-ABE.
- $\mathbb{A}$ generates the $XOR(x_{\mathbb{A}}(k), \cdot)$ garbled circuits for each dimension k of $x_{\mathbb{A}}$, and the output is the HE ciphertext consisting of bit.
- $\mathbb{B}$ fetches the ciphertext of CP-ABE, decrypts it to get the garbling key, and generates the garbled input from the garbling key and his search request $x_{\mathbb{B}}$.
- Finally, the searching cloud server receives the garbled input uploaded by $\mathbb{B}$ and uses it to get $\mathbf{HE}.\mathbf{E}(XOR(x_{\mathbb{A}}(k), x_{\mathbb{B}}(k)))$ in each dimension. Then computes the summation of all dimensions to get $\mathbf{HE}.\mathbf{E}(d(x_{\mathbb{A}}, x_{\mathbb{B}}))$

Our motivation and observation are that their scheme works with two individual computations (including XOR operations in garbled circuit and summation in HE). Intuitively, we would like to get rid of the use of HE, since garbled circuits can also be used to take summation computation. We propose a new scheme based on CP-ABE and garbled circuit.

Witness-based searchable encryption (WBSE). Public key encryption with keyword search (PEKS) is a notion that given a trapdoor of keyword T_m (produced by a receiver), the server can search keyword ciphertexts $C_{m'}$ (sent by a sender) where m, m' are keywords. If the test considers that a pair of ciphertext and trapdoor matches (implicitly denotes $\mathsf{Test}(T_m, C_{m'}) = 1$ if $m = m'$), this concludes the ciphertext and trapdoor are exactly generated from the same keyword. Upon searching, the receiver will obtain the encrypted data, referred to as an original ciphertext of a message from the server. Boneh et al. [14] first introduced PEKS. They defined the security models for searchable ciphertext indistinguishability, and proposed a PEKS scheme which must depend on a secure channel. However, Baek et al. [8] indicated that any outsider can link the matching trapdoor and ciphertext without a secure channel. Then, they defined another security model for secure-channel-free searchable ciphertext indistinguishability. On the other hand, Byun et al. [18] showed that PEKS suffers from an attack, off-line keyword guessing attack (OF-KGA). Given a trapdoor, an attacker knows which keyword is used to generate the trapdoor. The attacker can generate searchable ciphertexts, and then knows the link for the trapdoor. Recently, Rhee et al. [75] considered OF-KGA to defined a security model for trapdoor indistinguishability, and proposed a new public key encryption with keyword search for a designated server (dPEKS) scheme to withstand OF-KGA. However, Yau et al. [89] found there is a loophole of dPEKS, which means dPEKS will suffers from another attack, the on-line keyword guessing attack (ON-KGA). This is referred to as an open problem to construct a secure dPEKS scheme against ON-KGA. Chen [20] presented a new model called SPEKS to capture ON-KGA.

However, we have done lots of effort for the keyword guessing attack. The well-known open problem in public key encryption with keyword search is how to avoid internal adversaries as the server. To overcome this problem, the original framework (PEKS) [14] must changed slightly. A fundamental goal is creates a secure *bridge* between the sender and receiver. Witness-based searchable encryption (WBSE) is a manner to realize the design goal. WBSE is there is a $R(w, t) = 1$, t can be generated by w, and it is impossible to derive w from t. The sender holds witness w, and the receiver retrieves the corresponding instance t and then use it to produce the trapdoor of keyword. The internal attacker cannot imitate the a sender because it does not have the correct witness w.

Our starting point is motivated by observing that the existing WBSE scheme of Ma et al. [67] relies on trapdoor size proportional to n where n is the number of senders. However, the factor n is a barrier on size of trapdoor in WBSE, since the receiver must obtain all distinct instances (produced by distinct senders) from the server or public board. Shrinking size from $O(n)$ to n is our goal in this work.

Reversible data hiding in encrypted image by using secret sharing. Reversible data hiding (RDH) is a notion that allows to embed the additional and secret message into cover media, such as military or medical images, and to perform a reversible procedure that extracts the hidden secret message and perfectly reconstructs the original cover content. Numerous reversible data hiding methods have been introduced over the last two decades. Two seminal ideas of RDH are difference expansion (proposed by Tian [79]) and histogram shifting (proposed by Ni et al. [71]). In the difference expansion method [79], the differences between two adjacent pixels are doubled to release a new least significant bit (LSB) plane for carrying the secret message. In the histogram shifting method [71], the zero and peak points are used to embed the secret message by slightly modifying the pixel values.

Recently, a new direction of RDH known as RDH over an encrypted image (RDHEI) has been introduced. This novel RDHEI notion was firstly introduced by Zhang in 2011 [94], and captures the following real-life scenario regarding *owner privacy* known as image privacy [94]. An inferior assistant or a channel administrator is in the middle of a workflow and is authorized to insert some additional data such as the origin information, image notations or authentication data, within the encrypted image, where the original image content is unknown to this party. Indeed, medical images are encrypted for preserving the patient privacy, and a database administrator only embeds a few data into the corresponding encrypted images. For the consistency of a medical image, it must guarantee that the original content can be *perfectly* reconstructed after decryption-then-extraction of the secret message by the receiver. That is, RDHEI not only ensures the accuracy of the reconstructed cover-image and extracted secret message which are two basic tasks of RDH, but also preserves the privacy of the cover-image. More precisely, the work of Zhang [94] formalizes the model to describe the aforementioned scenario. The image provider $\mathcal{P}$ intends to preserve the privacy of the cover-image, but still desires a data hider $\mathcal{H}$ to embed a secret message. Therefore, $\mathcal{H}$ embeds the message into the encrypted image which is generated by $\mathcal{P}$ from the cover-image. Finally, the receiver $\mathcal{R}$ can recover the original cover-image and then extract the secret message correctly. The procedure run by $\mathcal{R}$ is known as decryption-then-extraction. However, the receiver also can be divided into two steps (decryption and extraction). We specify these two steps to two kinds of receivers, $\mathcal{R}_{\mathsf{dec}}$ and $\mathcal{R}_{\mathsf{ext}}$, and $\mathcal{R}_{\mathsf{dec}}$ performs decryption, and $\mathcal{R}_{\mathsf{ext}}$ takes $\mathcal{R}_{\mathsf{dec}}$'s decrypted image to extract the secret message.

However, inspired by the factor of key setting, the present studies identify the following two notions of RDHEI.[3]

- **Share independent secret keys (SIK).** $\mathcal{R}$ shares independent keys, $\mathsf{key}_{\mathcal{P}}$ and $\mathsf{key}_{\mathcal{H}}$, with $\mathcal{P}$ and $\mathcal{H}$ respectively. Notably, these keys $(\mathsf{key}_{\mathcal{P}}, \mathsf{key}_{\mathcal{H}})$ are secret and used to run image encryption and embedding algorithms. Numerous insightful works [50, 56, 64, 93, 94, 95] have proposed this type of RDHEI schemes.
- **Share no secret key (SNK).** In contrast to SIK, $\mathcal{R}$ does not need to share any secret key. This can be easily achieved through public key encryption where $\mathcal{R}$ has a public/secret key pair, and $\mathcal{P}$ ($\mathcal{H}$, resp.) can use the public key to do image encryption (data embedding, resp.). The first solution, proposed by Chen et al. [29], is to use Paillier homomorphic encryption [73] to encrypt each pixel and rely on specific techniques to complete data embedding. With the use of the homomorphic encryption, the follow-up works of Zhang at al. [96], Li and Li [60], and Shiu et al. [76] respectively implement some reversible data hiding techniques under the public key encryption associated with the homomorphic property.

To summarize the flexibility of key setting, it is clear that only the designated party who has the secret key can be $\mathcal{P}$ or $\mathcal{H}$ in SIK. However, the advantage of SNK is that anyone can be $\mathcal{P}$ or $\mathcal{H}$, since the keys of encryption or embedding are exactly the public key. In addition, as known, those *homomorphic encryption-based* SNK-type RDHEI schemes are practically inefficient since the underlying encryption schemes usually rely on complicated algebra structures and spend high computational cost. Thus, in this project, we will aim for constructing an efficient scheme to satisfy the intermediate notion (between SIK and SNK) where $\mathcal{P}$ and $\mathcal{R}$ share a secret key, but no secret is shared with $\mathcal{H}$.

Research Problem 2 (Years 2 and 3): Preserving Privacy in Distributed Services

Decentralized privacy with distributed services. With the popularity of third-party applications, privacy concerns focused on personal data gains more and more attention [61, 69].

[3] The key setting offers the framework among $\mathcal{P}$, $\mathcal{H}$, and $\mathcal{R}$. For example, if a RDHEI scheme is under public key encryption, any one can be $\mathcal{P}$ and $\mathcal{H}$. If under a symmetric encryption between $\mathcal{P}$ and $\mathcal{R}$ ($\mathcal{H}$ and $\mathcal{R}$, resp.), only specific party who holds the shared secret key can be $\mathcal{P}$ ($\mathcal{H}$, resp.). The key use is referred to as *party flexibility*. In the following, we formalize the key setting for more details.

The third-party application always requires the user to grant a set of permissions about data indefinitely. The recent work of Zyskind et al. [98] addressed the privacy concerns for using third-party services. It mainly focuses specifically on mobile platforms, where services deploy applications for users to install. These applications constantly collect high resolution personal data of which the user has no specific knowledge or control. This work assumes that the services are honest-but-curious (i.e., they follow the protocol). Precisely, this system protects against the following common privacy issues:

- *Data Ownership.* Our framework focuses on ensuring that users own and control their personal data. As such, the system recognizes the users as the owners of the data and the services as guests with delegated permissions.
- *Data Transparency and Auditability.* Each user has complete transparency over what data is being collected about her and how they are accessed.
- *Fine-grained Access Control.* One major concern with mobile applications is that users are required to grant a set of permissions upon sign-up. These permissions are granted indefinitely and the only way to alter the agreement is by opting-out. Instead, in our framework, at any given time the user may alter the set of permissions and revoke access to previously collected data. One application of this mechanism would be to improve the existing permissions dialog in mobile applications. While the user-interface is likely to remain the same, the access-control policies would be securely stored on a blockchain, where only the user is allowed to change them

Zyskind et al. [98] gave a solution based on blockchain and some cryptographic tools. However, along with the help of blockchain to provide some security, but the whole system is in cloud services, since there is a centralized management server. This leads us a direction to implement decentralized privacy in distributed services, where the service is not provided by only one party.

Reversible data hiding in encrypted image with distributed services. Very recently, a new model formalized by Wu et al. [81] is similar to the pure model by Zhang [94] except for the setting of the service provider. This model is referred to as a *distributed model* where, instead of one single P, many specific parties jointly provide the service. Indeed, these parties are one control center (denoted by C) and n storage and processing centers (denoted by S_1,...,S_n). Note that the such storage centers are duplicated to work with the identical task. In the model, each S_i will perform data embedding. However, summarizing the security definition of the model, the semi-honest adversary[4] is assumed to able to only corrupt a *threshold* number (by c) of storage centers, and C and the other uncorrupted centers are honest. In addition, Wu et al. [81] relied secret sharing as image encryption, and also present the first secret sharing-based

[4] An adversary A is semi-honest if A can only eavesdrop but cannot tamper. In cryptography, we say if A can tamper, then A is malicious.

RDHEI scheme. The key point of [81] is to use Shamir t-out-of-n secret sharing to generate shares to storage centers. However, since t-out-of-n secret sharing must rely on a field $\mathbb{F}$ and the$\mathbb{F}$ closest to 255 is 251, the scheme of [81] needs to compression of shrinking for some pixels. This works but waste some kinds of pixel values. In this project, we plan to propose a new scheme without compression of shrinking, and it also can work over distributed service setting.

Research Problem 3 (Years 4 and 5): Secure Delegated Computation in Distributed Services

In the last two years, we will rely on the new results and techniques from the first three years, and lift our past research works in cloud services to distributed services. This is a giant step, since there is no discussion of those topics for distributed services. We will focus on PPOS, WBSE (plan to study in Years 1 and 2), public key encryption with equality test (PKEET), and plaintext checkable encryption (PCE)[5]. Except for the topics we plan to study, we are also open to work on any interesting on security of distributed services. In the following, we only recall backgrounds of PKEET and PCE, and omit to show those of WBSE and PPOS (since they are identical to Years 1 and 2). In summary, this is the first research work that studies secure delegation in distributed services.

Public key encryption with equality test (PKEET), and plaintext checkable encryption (PCE). Computation over ciphertext is viewed as a hard and achievable task since it must keep confidentiality. It becomes feasible to obtain some extra functionalities by losing security. Homomorphic encryption [45] can evaluate some operations between encrypted data such as the form

$$\mathsf{Evaluate}(\oplus, \mathsf{Enc}(m_1), \mathsf{Enc}(m_2)) = \mathsf{Enc}(m_1 + m_2).$$

In addition, public key encryption with equality test [51, 52, 65, 68, 78, 86] provides a capability of verifying the equivalence between the underlying plaintexts of two encrypted data without decryption

$$\mathsf{Verify}(\mathsf{Enc}(m_1), \mathsf{Enc}(m_2)) = 1 \text{ if } m_1 = m_2.$$

Also, public key encryption with keyword search [8, 14, 20, 54, 74, 84, 85, 90] is also an interesting topic in which the sender can upload the cloud a ciphertext of data associated with ciphertexts of keywords $\mathsf{Enc}(w)$, the receiver can produce a trapdoor of a keyword $\mathsf{Trapdoor}(w')$, and finally the cloud can test the equivalence between the underlying keywords

$$\mathsf{Test}(\mathsf{Enc}(w), \mathsf{Trapdoor}(w')) = 1 \text{ if } w = w'.$$

Those cryptographic solutions offering additional functionality usually have some impossibility results or barriers; for example, homomorphic encryption cannot achieve CCA2 security, and public key encryption with equality test cannot achieve CPA.

[5] PKEET and PCE are our previous research focuses.

Differing from computing over ciphertext, there is a new notion, plaintext-checkable encryption (PCE), which supports the specific functionality between ciphertext and plaintext. PCE, firstly introduced by Canard et al. [19], provides an add-on *check* service where, given a target plaintext, a ciphertext and a public key, it can check whether the ciphertext encrypts the target plaintext with the public key. Usually, we use a check algorithm to capture the above-mentioned procedure for check. However, in real-life application, PCE can be used to tackle the following scenario. Let us consider a semi-honest cloud storage which is not fully trusted by users[6]. However, a user can upload an encrypted data associated with a few ciphertexts of keywords $\mathsf{Enc}(w)$. The keyword search is publicly available for anyone. The server receives a plain keyword w', and then can check whether the underlying keyword of $\mathsf{Enc}(w)$ is identical to the request w' without decryption. If the check algorithm $\mathsf{Checkable}(\mathsf{Enc}(w), w')$ returns 1 (a.k.a $w = w'$), the server will return the corresponding encrypted data.

In the literature, a few works focused on PCE with defining security model and presenting new schemes. Canard et al. [19] initialized the notion of PCE, and showed the basic security model, called unlinkable CPA security. The general CPA security cannot achieved by PCE, since the adversary can access the check algorithm. Ma et al. [66] presented a follow-up to introduce a generic construction from smooth projective hash, and their scheme satisfies s-priv1-cca security, which is independent of unlinkable CPA security. However, for the goal to reach CPA security, Das et al. [40] modified the framework of PCE in which only the designated checker (who has been delegated check power) can perform check algorithm. They also introduced the new schemes and security model (for CPA security) in which the CPA security can be held if the adversary is external, not the designed checker.

1.2 Contributions

- This project studies the important research problems of secure delegated computation in distributed services. To our best knowledge, this is the first work that studies the secure delegated computation in distributed services with cryptographic and smart contract approaches.

- We start from some ongoing and familiar delegation in cloud services, as known as similarity test and searchable encryption. Some security issues will be addressed, and we aim for providing efficient solutions to overcome. Moreover, we expect to find open problems and attempt to deeply understand them.

- Preserving privacy in distributed services is our first step to study security in distributed services. However, privacy is one security concern, and it may still far from the final goal, delegation in distributed services. To achieve privacy preserving in distributed services,

[6] Semi-honesty means that the cloud server will follow the procedure of the system protocols and algorithms, and does not have any malicious behavior such as tampering.

we focus on decentralized privacy over smart contract-based distributed services. We plan to use smart contracts to implement privacy.

- From the experience of smart contracts above, we will be mature of blockchain and smart contract. In particular, we can foster talents who are good at blockchain techniques, smart contract programming skills, knowledge of secure delegation in smart contract-based distributed services. We believe that with the blockchain to flourish, the industry will need them without doubt.

- Our final goal is to overcome the barrier of distributed services. We expect to lift secure delegation from cloud to distributed services. However, smart contracts may or may not help. If not, we plan to have a new platform to realize our secure delegation in distributed services. We will focus on some distributed delegation, for example, similarity test and searchable encryption in distributed services.

2 Concrete Research Plan

We elaborate on our research methods/strategies, and schedule the yearly research plans.

2.1 Research Methodology and Strategies

This project is primarily in the field of cryptography, information security, and blockchain, where in general the research methodology consists of:

(i) formalizing models (security definition and requirement in theory and practice) that capture interesting questions to be investigated;

(ii) developing new ideas and techniques for constructing secure systems;

(iii) answering the questions by analyzing the security of the proposed systems;

(iv) implementing the such systems to obtain the real (or simulation) results;

(v) showing new questions as the future interesting direction by interpreting the existing results.

There is a very effective way to carry through this process is through collaboration and research meeting with scholars and students in both cryptography and information security, which is extremely useful for stimulating, exchanging and distilling new ideas. The blueprint of our general research methods is four-folds:

- **Research collaborations.** I will have many collaborations for different topics. With the such collaborations, our group can share research resource to the other group, and also can discuss research problems together. In particular, group members can enjoy the collaborative procedures and results, and then create a pretty nice environment for research.

- **Research visits.** Correspondingly, I plan to arrange short-term visit to collaborators in academics or industry. I will also demonstrate the research results during our visit. I will send students for short-term research visit as well.

- **Advising students via close research meetings.** I am aware of that close research meetings are very effective to help students to start research and enlighten their passions, since I can demonstrate the right direction and attitude to think about research through the meetings. As mentioned, I also encourage students to talk to anyone to learn different research perspectives and organize some study groups by themselves. Our group, cryptography and information security lab (CIS lab @YZU), was initiated in fall 2017. Now we have six graduates and more than ten undergraduates. The scale of the group is not large but in my opinion it suffices to take care of the works of this proposal. The other group, information security lab (IS lab @NCCU) is also the same scale, and it will act as a supporter if CIS lab needs some assistance, or as a checker to verify the research results.

- **Leading projects in courses.** According to the two lectures (Cryptography and Network Security @YZU) I taught before, I required students to complete projects. I believe that it is a nice opportunity to do research, work together, and enjoy the results with students. As course arrangements, I would like to address theoretical investigations with graduates in Cryptography, and practical applications with undergraduates in Network Security. I will list some relevant topics with the focus of the proposal, and guide them as much as possible.

2.2 Yearly Research Plans

As a concrete research plan, it must succeed our previous research and connect to the near future direction, and then further look for new goals. Now we elaborate our research plans on secure delegated computation in distributed services. There are many exciting security issues we will tackle in the following five years, including three directions (i) Secure Delegated Computation in Cloud Services, (ii) Preserving Privacy in Distributed Services, and (iii) Secure Delegated Computation in Distributed Services. In addition of the schedule of this proposal, we also keep opening our field of view and investigate relevant topics on the above-mentioned directions. Figure 2 shows a schedule in details in which we describe the tasks for each time slot and its relevant research topics.

2.2.1 The Years 1 and 2

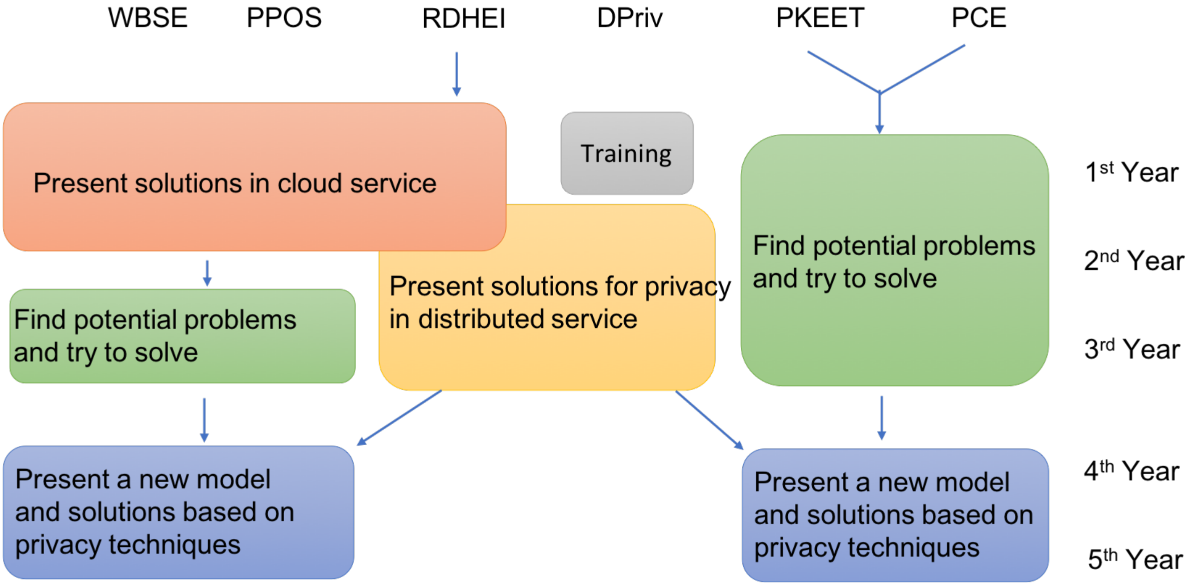
Privacy-preserving outsourced similarity test (PPOS). We would like to get rid of the use of HE, since garbled circuits can also be used to take summation computation. We propose a new scheme based on CP-ABE and garbled circuit. In the following, we highlight our techniques in our scheme.

- $\mathbb{A}$ generates a specific circuit that hardcodes $\mathbb{A}$'s input (feature value) and computes summation of $XOR(x_A, \cdot)$ for each feature dimension; the input of the specific circuit is exactly the search request of $\mathbb{B}$. This suffices to combine the two individual computations of Zhang et al. [91, 92]'s scheme.

- $\mathbb{A}$ uploads the garbled circuit of the specific circuit and the CP-ABE ciphertext of the garbling key to the searching cloud.

- $\mathbb{B}$ can decrypt the CP-ABE ciphertext to obtain the garbling key, and then uses it to generate the garbled input for his search request and send the garbled input to the searching cloud.

- Finally, the searching cloud runs evaluation of the garbled circuit and input, and then returns the "plain" result of the evaluation to $\mathbb{B}$.


WBSE
PPOS
RDHEI
DPriv
PKEET
PCE
Present solutions in cloud service
Training
Find potential problems and try to solve
Present solutions for privacy in distributed service
Find potential problems and try to solve
Present a new model and solutions based on privacy techniques
Present a new model and solutions based on privacy techniques
1st Year
2nd Year
3rd Year
4th Year
5th Year

WBSE: Witness searchable encryption
PPOS: Privacy-preserving outsourced similarity test
RDHEI: Reversible data hiding in encrypted image
Dpriv: Decentralize privacy
PKEET: Public key encryption with equality test
PCE: Plaintext checkable encryption

Figure 2: Schedule of this proposal

Except for the proposed scheme, we pay more attention on the security. We firstly introduce the definition of simulation security of PPOS with semi-honest adversaries. For proving the security of our scheme, the security of garbled circuits cannot be directly applied. Finally, we plan to decouple the garbled circuit into many components of chosen double encryption, and then complete the security proof. However, if the proof is done, we must require a special security property (non-standard, but reasonable) of CP-ABE. This leads us to consider using standard properties of CP-ABE to construct the PPOS.

Witness-based searchable encryption (WBSE). Many PESK systems depend on pairing [9, 35, 42, 63, 87, 97] because it has some excellent properties. We use bilinear map as the building block and construct a new WBSE scheme. Our main technique is to pack n different and individual trapdoors together, and thus create a degree $n-1$ polynomial f where every point x_i induces $f(x_i) = s$ and x_i is a Diffie-Hellman session key produced from a witness and receiver's public key. At a high level, the trapdoor is composed of n encodings of n coefficients of f and one encoding of a keyword[7], and in particular the keyword ciphertext is of encodings of a point x_i and its powers $x_i^2, ..., x_i^n$. Finally, bilinear map can support evaluation of Test over the encodings of f and x_i. Summarizing the result, the trapdoor contains f, and the ciphertext does x_i. This suffices to achieve the size of trapdoor is exactly n with constant overheads. We also plan to formalize an abstracted notation, witness-based searchable encryption with

[7] Keep *encoding* implicitly. Intuitively, we say that encoding converts an input x to a group element with some additional randomness.

trapdoor aggregation. In addition, we would like to propose an efficient breakthrough to cross the barrier.

For more details, we have some initial idea to construct the nearly optimal scheme under the barrier. There are three groups $\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T$ of prime order q. A bilinear map is denoted by $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$ and it is a general type 3 bilinear pairing. $H(\cdot)$ be a cryptographic hash function defined by $\{0,1\}^* \to \mathbb{Z}_q^*$. We choose a NP-language $\mathcal{L}$ for the witness relation and $\mathrm{w} \in \mathcal{WT}$, where $\mathcal{WT}$ denotes the witness space, $\mathcal{WT} = \mathbb{Z}_q^*$ and the instance t is defined as follows:

$$R(\mathrm{w}, t) = 1 \land \mathrm{w} \in \mathcal{WT} \land t = g^w.$$

- $\mathsf{KeyGen}(1^\lambda)$: It takes the security parameter λ as input (μ and α are uniformly chosen), and then returns

$$sk = (\mu, \alpha)$$
$$pk = (q, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e, g, A = g^\alpha, U = g^\mu, H(\cdot)).$$

- $\mathsf{WBSE}(pk, m; w)$: Choose these parameter parameters, a witness $w \in \mathcal{WT}$, $r \in \mathbb{Z}_q^*$ and a keyword m, then compute:

$$x = H(A^w)$$
$$L_u = [U^r, U^{r\cdot x}, U^{r\cdot x^2}, ..., U^{r\cdot x^n}]$$
$$\nu = A^r \cdot g^{H(m)\cdot r}$$
$$C = (t, \nu, L_u).$$

- $\mathsf{Trapdoor}(sk, m, t_1, t_2, ..., t_n)$: It firstly fetches all the instance t_i and computes L_x as follows

$$x'_i = H((g^{w_i})^\alpha)$$
$$L_t = [t_1, t_2, ..., t_k] = [g^{w_1}, g^{w_2}, ..., g^{w_k}].$$
$$L_x = [x'_1, x'_2, ..., x'_k],$$

Choose a_k, $s \in \mathbb{Z}_q^*$ uniformly at random and construct k-degree F function:

$$F(x) = \prod_{i=1}^{k}(x - x'_i) + s = a_k \cdot x^k + a_{k-1} \cdot x^{k-1} + ... + a_0.$$

Choose a generator $h \in \mathbb{G}_2$ and return a trapdoor $T = (h_k, t_d)$ by setting

$$h_k = (h^{a_k}, h^{a_{k-1}}, ..., h^{a_0}).$$
$$t_d = h^{\frac{u\cdot s}{\alpha + H(m)}}.$$

- $\mathsf{Test}(C, T)$: Check if the searchable ciphertext and trapdoor enjoy the same keyword m by computing the following equation:

$$\prod_{i=0}^{k} e(U^{r\cdot x^i}, h^{a_i}) = e(\nu, t_d).$$

If yes, outputs 1 denoting that C and T contain the same keyword, and 0 otherwise. We ignore to verify the correctness.

Now we already have a solution of WBSE. However, we still need to complete the security proof. Except that, our goal is to attempt to achieve trapdoor size of polylog or constant.

Reversible data hiding in encrypted image by using secret sharing. Our starting point is to formalize the new notion of key setting and care about the efficiency. To achieve better efficiency, we must avoid using public key encryption. However, if we do not use any public key encryption scheme, it is impossible to protect image privacy (the original purposes of RDHEI) without the shared key between $\mathcal{P}$ and $\mathcal{R}$. Thus, for preserving privacy, the ideal class is that **receiver shares "only one" secret key with the image provider** (SOK, for short). In particular, there is no shared key between $\mathcal{H}$ and $\mathcal{R}$, which precisely implies that the embedding procedure does not take any shared key as input.

The proposed SOK schemes are inspired from some existing SNK schemes (i.e., [76, 96]). We found that these SNK schemes work with Paillier encryption (or other addition homomorphic encryption) to preserve image privacy, and the property of homomorphic evaluation is used to embed the message. For achieving our above-mentioned requirements, we replace the parts of Paillier encryption with secret sharing that also enjoys homomorphic evaluation in some ways[8]. We show an abstraction of those SNK schemes, and then under the abstraction, introduce our method. The high level idea of our method is composed of the following two steps.

- **Encryption.** Secret sharing acts as a symmetric encryption to encrypt the cover-image, so our method use one shared key between $\mathcal{P}$ and $\mathcal{R}$. However, ours does not construct shares for each pixel like Wu et al.'s method. For preserving the total size, we pack t pixels and t random factors together to generate only t shares, and put the shares back as encrypted pixels and set random factors as the key. It suffices to avoid the size blow-up, and also keeps correctness of decryption by using t random factors and t shares. The technique of our method is inspired by the multi-secret sharing, but we slightly modify it for security and framework of SOK.

- **Data embedding.** We divide the secret message into several units. Then, for embedding a unit, we generate another t shares without needing any key (as known as the above-mentioned random factors), and then use homomorphic evaluation and embedding procedure to embed message into the t encrypted pixels.

[8] Here homomorphic evaluation is to perform homomorphic operation over encrypted data and further obtain the encrypted result.

The proposed method strictly relies on the properties of secret sharing. Summarizing the main techniques, secret sharing serves as the underlying primitive offering security, multiple secret preserves size complexity, and inherently additive homomorphism realizes the data embedding.

For generalization, if SNK schemes satisfy *some properties*, they can be converted to SOK. Hence, our method can be generalized as a compiler. As a concrete instantiation, given Shiu et al.'s SNK scheme based on difference expansion, we show the SOK-type RDHEI by slight modification. The scheme overview is described as follows. $\mathcal{P}$ will pre-process the cover-image (including double the differences between two adjacent pixels[9]) and generate a new cover-image, referred to as the processed image, and then send $\mathcal{H}$ the encrypted image by using polynomial interpolation. $\mathcal{H}$ will obtain a new polynomial which carries a secret message in the released LSB plane, and then use addition homomorphism to generate the encrypted image with embedded message. Finally, by decryption $\mathcal{R}$ is able to obtain the stego-image (a.k.a image with embedding message), and then recover the cover-image and secret message. We remark that in this scheme, the embedding of the original method of [79] is decoupled into two steps: (1) double the differences as the pre-processing run by $\mathcal{P}$ and (2) embed a secret message run by $\mathcal{H}$. Follow by the same technique, we also can provide another SOK scheme from [96] (SNK).

2.2.2 The Years 2 and 3

Decentralized privacy with distributed services. In this project, we aim for presenting a decentralized platform system based on smart contracts [17, 36, 77] that is suitable for third-party applications [39] and support decentralized privacy. To achieve this issue of privacy above [55, 57], the smart contract-based system should have the following attributes:

- Management: The system which identifies the users as the owner of the data and services with delegated authority can add and manage users. In addition,the users in this system can also add third-party services.

- Authorization: Users can grant a set of permissions to the services in the system and change the permission set at any time.

- Revocation: Users have the right to revoke the services or the permissions of the services in the system. Moreover, the user can apply to be revoked from this system.

- Transparency: All actions executed in the smart contract are completely recorded and any entity can review and check.Each user has transparency over the collection and use of his personal data.

- The entity can only execute specific operations and cannot execute operations on behalf of other entities in the system or totally as other entities.

[9] In fact, the pre-processing also deals with location map which is used to record the unembeddable pixel pairs.

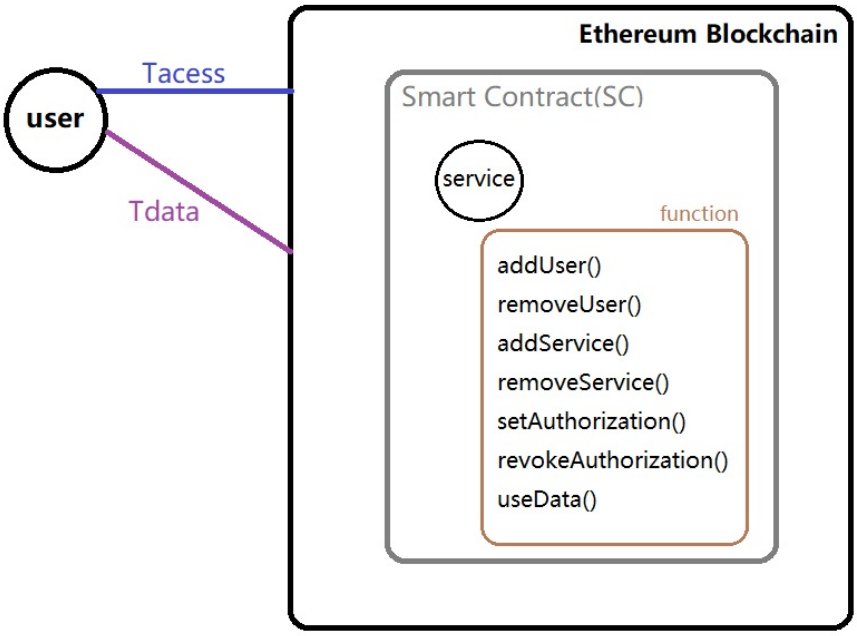


Figure 3: Overview of the smart contract-based system with decentralized privacy

The overall structure of the system is shown in Fig. 3. The system is mainly composed of two entities: user and smart contract. Users are the one who wants application services and provides various information and data. Smart contract (SC) is the provider of these applications and uses personal data to implement the corresponding business, and als used to enable interaction between users, services and blockchain platforms in ethereum and finally achieve decentralized privacy.

User, service and blockchain platforms mainly execute two types of transactions: T_{access}, used for access management;and T_{data}, used for data transmission and uses. Both of the two transactions are implemented through smart contracts (SC) in the system. The SC provides different functions for efficient and secure implementation of decentralized privacy.The use of SC can guarantee the transparency of the process about using data and authorization, while maintaining the anonymity of users.The SC is an efficient and secure programmable agreement that runs exactly the same way as programming. All operations performed by the SC are also posted on the blockchain of ethereum. The SC requires the following characteristics: 1) allow the system to add users (and revoke with user permission), 2) allow users to add services (and revoke services), 3) allow users to manage and modify information in a transparent manner,and 4) allow users to grant service access to data when needed (and change or revoke at any time) [38].

Smart contracts can be implemented in a variety of programming languages, but the most widely used is Solidity [13]. We plan to use Solidity to achieve smart contracts based on the Ethereum platform [6, 70, 83].The code inside the smart contract provides the following functions:

- addUser(u.address, u.data): This function can only be executed by the owner or creator of the smart contract(the restriction on which entity executes the function will be achieved through the modifiers in the code). It adds the user into the system and performs a series of initialization operations. It takes the user's address (u.address) and information (u.data) as input, and generates the user's signature key pair and encryption key.This function outputs the inputs and the time stamp when the function is executed, and writes the output into the SC, then the SC is updated.

- removeUser(u.address): This funtion that removes the user from the system can only be executed by the owner or creator of the SC. It takes the user's address (u.address) as input and successfully deletes the user, then the SC is updated finally.

- addService(s.address, s.notes): This funtion that adds the service into the SC can only be executed by the user in this system.It takes as input the address of the service (s.address) and service's notes (s.notes) including the required permissions. This function outputs the inputs and the time stamp when the function is executed, and writes the output into the SC, then the SC is updated.

- removeService(s.address): This function that removes the service from the SC can only be executed by the user in this system. It takes the service's address (s.address) as input and successfully deletes the service, then the SC is updated finally.

- setAuthorization(s.address, license): This function can only be executed by the user in the SC to grant or modify the permissions of data which the service need.I t takes as input the address of the service (s.address) and the service's permissions given by the user (s.license). This function outputs the inputs and the time stamp when the function is executed, then the SC is updated.

- revokeAuthorization(s.address): This function can only be executed by the user in the SC to revoke the permissions be granted to the service. It takes the service's address (s.address) as input and outputs the inputs, then the SC is updated finally.

This project is our first step to work on a well-packaged platform of distributed services. Starting from deploying the such basic distributed services, our next step will touch and convert some secure delegated computation in cloud services into that in distributed services across smart contracts (connecting to Years 4 and 5 plans).

Reversible data hiding in encrypted image with distributed services. Our motivation is that $\mathbb{F} = 251$ induces compression. Intuitively, we need a secret sharing over 256, XOR secret sharing is a candidate. However, XOR secret sharing only can work on n-out-of-n, but it suffices to construct our solution. Let us describe (n, n) secret sharing scheme. A secret message will be split into n shares (also called shadows) (share$_1$,share$_2$,...,share$_n$). Where collecting n shares together can recover the secret, but any less than n shares cannot get any

information about the secret. Let us briefly describe the (n, n) XOR secret sharing scheme.
Share Generation. Taking secret $\in \{0,1\}^\ell$ as input, this algorithm chooses uniform $\text{share}_i \in \{0,1\}^\ell$ or all $i, 1 \leq i \leq n-1$. Set the last share as share_n = $\bigoplus_{i=1}^{n-1}\text{share}_i \oplus$ secret where $\oplus$ is the bit-wise exclusive OR operation.
Secret recovery. This algorithm recover secret by computing secret = $\bigoplus_{i=1}^{n}\text{share}_i$.

For clearly describing our idea, we directly instantiate a pixel pair (x,y) along with the following steps:

- ImageEnc: $\mathcal{P}$ pre-processes (x, y) to compute $l = \lfloor\frac{x+y}{2}\rfloor$ and $d = x - y$, and then computes $x' = l + d$ and $y' = l - d$. Note that d can be positive or negative, but does not influence on the results. By using share generation, for a pair (x', y'), $\mathcal{P}$ generates $\{\{x'_1, y'_1\}, \{x'_2, y'_2\}, ..., \{x'_n, y'_n\}\}$ where x'_j, y'_j for all j, $1 \leq j \leq n-1$, are chosen uniformly, and $x'_n = x' \bigoplus_{i=1}^{n-1}$ and $y'_n = y' \bigoplus_{i=1}^{n-1}$. $\mathcal{P}$ then sends (x'_i, y'_i) to $\mathcal{H}_i$.

- Embedding: For embedding a bit 1, we consider two cases:

  - If n is odd, for all i, $1 \leq i \leq n$, $\mathcal{H}_i$ sets $x''_i = x'_i \oplus 0x01$ and $y''_i = y'_i$ to get the result $EIwS_i$.
  - If n is even, for all i, $1 \leq i \leq n-1$, $\mathcal{H}_i$ sets $x''_i = x'_i \oplus 0x01$ and $y''_i = y'_i$. $\mathcal{H}_n$ does not perform any processing, and directly sets $x''_n = x'_n$ and $y''_n = y'_n$ to get the result $EIwS_i$.

  For embedding a bit 0, $\mathcal{H}_i$ does not perform any processing, and directly sets $EIwS_i$ ($x''_i = x'_i$ and $y''_i = y'_i$).

- ImageDec: $\mathcal{R}$ use $\bigoplus_{i=1}^{n} EIws_i$ to get secret. $\mathcal{R}$ recovers x'' and y'' by computing $x'' = \bigoplus_{i=1}^{n} x''_i$ and $y'' = \bigoplus_{i=1}^{n} y''_i$.

- Extracting: If x'' and y'' are both odd or both even, $\mathcal{R}$ extracts $b = 0$ and recover $x' = x''$ and $y' = y''$; if not, extracts $b = 1$ and recovers $x' = x'' - 1$ and $y' = y''$ since b is only embedded into the first pixel. $\mathcal{R}$ can easily compute $l = \lfloor\frac{x'+y'}{2}\rfloor$ and $d = \frac{x'-y'}{2}$ and obtain the original pixel pair of CI by computing $x = l + \lfloor\frac{d+1}{2}\rfloor$ and $y = l - \lfloor\frac{d}{2}\rfloor$. At the end, this step will outputs SM and recovers CI.

In this project, we will complete this initial idea and relevant experiments.

2.2.3 The Years 4 and 5

In the last two years, we plan to tackle more challenging questions that we currently do not have ideas to solve them. We hope that new ideas will emerge in the following years for us to tackle these questions. We will also be open to explore new questions that arise along the way of our research. Some concrete questions in mind are PPOS, WBSE, PKEET, and PCE over distributed services.

PPOS, WBSE, PKEET, and PCE with concrete distributed service implementations. As we knew, these topics are in cloud services. To lift them to be run in distributed services is a challenging question. The main key point is that they are secure delegated computation, and thus privacy should be the most significant concern. Very straightly, a simple idea is to be use decentralized privacy implemented by smart contracts. This does not work because smart contracts over blockchain are totally public, and decentralized privacy only keeps user privacy, not for the whole computation. As a result, in general we can view a computation as an input and a function. Keeping input privacy may be able to use SC-based decentralized privacy techniques (but not clear yet), but function privacy is a main challenge. To tackle it, we list a few directions and strategies.

1. Garbled circuit can be a building block to delegate computation. It is usually constructed from symmetric encryption. In a sense, presenting a smart contract to implement the garbled circuit should not be difficult.

2. PPOS, WBSE, PKEET, and PCE involve much complicated tools (i.e., public key encryption, bilinear map, ...). Now we do not have concrete ideas to solve these issues.

3. Smart contract is one of distributed services. Ideally, we may be able to create another platform that can realize the secure delegation.

New distributed service models of PPOS, WBSE, PKEET, and PCE. Follow the above methods. The last step is to formalize the distributed service platforms. This is quite interesting, since for different styles of services, we will have distinct models to capture the functionality and security. Thus, this open area is worth to study in this project, including the real-life scenarios and theoretical models, and potential solutions.

3 Expected Results and Impact

3.1 Expected Results

We expect to solve some of the following problems and publish several papers in the journals such as IEEE Transactions on Information Forensic and Security or some conferences. Our goal is to publish one journal paper and one conference per year. Hope we can achieve this.

The Years 1 and 2. For *witness based searchable encryption*, we already have some observation in an ongoing work. We expect that packing the trapdoors together with some techniques of secret sharing can help to provide an efficient solution (i.e., trapdoor size is down to n only.) Also, we expect to tackle to break the barrier n down to a constant.

For *privacy preserving outsourced similarity test*, we use garbled circuit to replace using homomorphic encryption. The realistic security model will be formalized. However, the challenge is the proof, since the security of garbled circuit cannot directly be applied. We expect to non-black-box use of security of garbled circuit to complete the proof. This induces an issue that we use non-standard assumption of attributed-based encryption (even the assumption is still reasonable). We plan to solve the issue as well.

For *reversible data hiding in encrypted image using secret sharing*, we expect to state a new class, referred to as shared-one-key (SOK). In this class, only the image provider has a shared secret key with the receiver, and in particular, anyone who knows the embedding procedure can hide. For flexibility, SOK is much weaker than SNK. However, the existing SNK schemes rely on additive homomorphic encryption. We use secret sharing as the underlying ingredient to construct our SOK scheme to achieve better efficiency and preserve the total size. Then, we will present a compiler to convert a SNK scheme with some properties to a SOK version.

The Years 2 and 3. For *decentralized privacy in distributed services*, we will design a smart contract-based mechanism used in a distributed platform system to address the privacy issue. Users own and control their personal data and do not need to trust any third-parties. As a result, smart contract will fully act as the service provider. This work is sort of tiny, but will influence on the progress of years 4 and 5.

For *reversible data hiding in encrypted image in distributed services*, we expect to obtain a more effective solution (with respect to capacity and image quality). Our plan is to rely on the other type of secret sharing working with XOR operations. Finally, we claim our method is efficient since only simple operations will be used.

The Years 4 and 5. The expected results for the years 4 and 5 would be more open ended. Our ambitious goal is to be familiar with the techniques used or proposed in the first three years, and then go one small step in distributed services. The questions listed here are more challenging, and we may not be able to solve all of them. Nevertheless, even if we cannot solve the problems, we hope that along the way of studying these questions, new ideas and questions will emerge and may lead to new findings. In the best case, we will have plenty of secure delegation in distributed services, and those corresponding version in cloud have been already well-studied.

3.2 Potential societal impact.

For more than a decade, IT companies such as Google and Facebook crucially rely on distributed network to replicate their database and computing infrastructure. The subjects of this proposal are in the frontier of security research on delegating computation in distributed services. In a sense, when the service provided by IT companies achieve enough security level, consumers must have strong confidence to use. We hope this project would publicize security research in Taiwan and increase the interaction with worldwide research and industry communities. Additionally, as it is often the case for basic research with potential applications, we hope that the ideas, concepts, and techniques we develop throughout the research can be used in the real-life. In fact, we keep our faith for the value of this project, since we already have a very strong distributed service platform, as known as Ethereum smart contract.

Students participating this project would be able to interact with me and researchers in the relevant fields and gain research experience in frontier security research on the academia side. Also, on IT industrial side, they will need to implement lots of systems supporting security in distributed service, and thus the programming skill will be increased without doubt. However, we hope (through the project) complete training could nurture their ability of security and mathematical thinking, which will either benefit their future career or lead them to mature researchers or IT engineers.

One significant result is to increase students' senses of security. I expect students can learn this from taking the courses. Going one step further, students who have more interests can obtain the research experience and training in the fields of information security and cryptography. These experiences will be helpful and valuable. Moreover, for research I expect to complete the scheduled plan in five years, have novel results in theory and practice, and find new future directions. Finally, I will not only aim for publishing the results in top journals and conferences, but also contribute myself more to *education* and *industry* (elaborate the special plans in the next subsection).

References

[1] Climateprediction.net. (http://www.climateprediction.net/).

[2] Folding@home . (http://folding.stanford.edu/).

[3] Quake-Catcher Network. (http://quakecatcher.net/).

[4] Prabhanjan Ananth, Yu-Chi Chen, Kai-Min Chung, Huijia Lin, and Wei-Kai Lin. Dele-

gating ram computations with adaptive soundness and privacy. In *Theory of Cryptography Conference*, pages 3–30. Springer, 2016.

[5] Dana Angluin and David Lichtenstein. Provable security of cryptosysterns: a survey. Technical report, TR-288, Yale University, 1983.

[6] Nicola Atzei, Massimo Bartoletti, and Tiziana Cimoli. A survey of attacks on ethereum smart contracts (sok). In *Principles of Security and Trust*, pages 164–186. Springer, 2017.

[7] Jean Bacon and Ken Moody. Toward open, secure, widely distributed services. *Communications of the ACM*, 45(6):59–64, 2002.

[8] J. Baek, R. Safavi-Naini, and W. Susilo. Public key encryption with keyword search revisited. In *ICCSA 2008*, pages 1249–1259. Lecture Notes in Computer Science 5072, Springer, 2008.

[9] Joonsang Baek, Reihaneh Safavi-Naini, and Willy Susilo. Public key encryption with keyword search revisited. In *International conference on Computational Science and Its Applications*, pages 1249–1259. Springer, 2008.

[10] Mihir Bellare, Viet Tung Hoang, and Phillip Rogaway. Foundations of garbled circuits. In *Proceedings of the 2012 ACM conference on Computer and communications security*, pages 784–796. ACM, 2012.

[11] Philip A Bernstein. Middleware: a model for distributed system services. *Communications of the ACM*, 39(2):86–98, 1996.

[12] John Bethencourt, Amit Sahai, and Brent Waters. Ciphertext-policy attribute-based encryption. In *Security and Privacy, 2007. SP'07. IEEE Symposium on*, pages 321–334. IEEE, 2007.

[13] Karthikeyan Bhargavan, Antoine Delignat-Lavaud, Cédric Fournet, Anitha Gollamudi, Georges Gonthier, Nadim Kobeissi, Natalia Kulatova, Aseem Rastogi, Thomas Sibut-Pinote, Nikhil Swamy, et al. Formal verification of smart contracts: Short paper. In *Proceedings of the 2016 ACM Workshop on Programming Languages and Analysis for Security*, pages 91–96. ACM, 2016.

[14] Dan Boneh, Giovanni Di Crescenzo, Rafail Ostrovsky, and Giuseppe Persiano. Public key encryption with keyword search. In *International conference on the theory and applications of cryptographic techniques*, pages 506–522. Springer, 2004.

[15] Dan Boneh, Craig Gentry, Sergey Gorbunov, Shai Halevi, Valeria Nikolaenko, Gil Segev, Vinod Vaikuntanathan, and Dhinakaran Vinayagamurthy. Fully key-homomorphic encryption, arithmetic circuit abe and compact garbled circuits. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 533–556. Springer, 2014.

[16] Zvika Brakerski and Vinod Vaikuntanathan. Efficient fully homomorphic encryption from (standard) lwe. *SIAM Journal on Computing*, 43(2):831–871, 2014.

[17] Vitalik Buterin et al. A next-generation smart contract and decentralized application platform. *white paper*, 2014.

[18] J. W. Byun, H. S. Rhee, H. A. Park, and D. H. Lee. Off-line keyword guessing attacks on recent keyword search schemes over encrypted data. In *SDM'06*, pages 75–83. Lecture Notes in Computer Science 4165, Springer, 2006.

[19] Sébastien Canard, Georg Fuchsbauer, Aline Gouget, and Fabien Laguillaumie. Plaintext-checkable encryption. In *Cryptographers' Track at the RSA Conference*, pages 332–348. Springer, 2012.

[20] Yu-Chi Chen. Speks: Secure server-designation public key encryption with keyword search against keyword guessing attacks. *The Computer Journal*, 58(4):922–933, 2014.

[21] Yu-Chi Chen. Fully incrementing visual cryptography from a succinct non-monotonic structure. *IEEE Transactions on Information Forensics and Security*, 12(5):1082–1091, 2017.

[22] Yu-Chi Chen, Sherman SM Chow, Kai-Min Chung, Russell WF Lai, Wei-Kai Lin, and Hong-Sheng Zhou. Cryptography for parallel ram from indistinguishability obfuscation. In *Proceedings of the 2016 ACM Conference on Innovations in Theoretical Computer Science*, pages 179–190. ACM, 2016.

[23] Yu-Chi Chen, Gwoboa Horng, and Chang-Chin Huang. Privacy protection in on-line shopping for electronic documents. In *Information Assurance and Security, 2009. IAS'09. Fifth International Conference on*, volume 2, pages 105–108. IEEE, 2009.

[24] Yu-Chi Chen, Gwoboa Horng, and Chang-Chin Huang. Privacy protection in on-line shopping for electronic documents. *Information Sciences*, 277:321–326, 2014.

[25] Yu-Chi Chen, Gwoboa Horng, and Chao-Liang Liu. Strong non-repudiation based on certificateless short signatures. *IET Information Security*, 7(3):253–263, 2013.

[26] Yu-Chi Chen, Gwoboa Horng, and Du-Shiau Tsai. Comment on "Cheating prevention in visual cryptography". *IEEE Transactions on Image Processing*, 21(7):3319–3323, 2012.

[27] Yu-Chi Chen, Chao-Liang Liu, and Gwoboa Horng. Cryptanalysis of some user identification schemes for distributed computer networks. *International Journal of Communication Systems*, 27(11):2909–2917, 2014.

[28] Yu-Chi Chen, Kunhan Lu, Raylin Tso, and Mu-En Wu. An improved visual cryptography with cheating prevention. In *International Workshop on Digital Watermarking*, pages 444–454. Springer, 2014.

[29] Yu-Chi Chen, Chih-Wei Shiu, and Gwoboa Horng. Encrypted signal-based reversible data hiding with public key cryptosystem. *Journal of Visual Communication and Image Representation*, 25(5):1164–1170, 2014.

[30] Yu-Chi Chen, Du-Shiau Tsai, and Gwoboa Horng. A new authentication based cheating prevention scheme in naor–shamir's visual cryptography. *Journal of Visual Communication and Image Representation*, 23(8):1225–1233, 2012.

[31] Yu-Chi Chen, Du-Shiau Tsai, and Gwoboa Horng. Visual secret sharing with cheating prevention revisited. *Digital Signal Processing*, 23(5):1496–1504, 2013.

[32] Yu-Chi Chen and Raylin Tso. A survey on security of certificateless signature schemes. *IETE Technical Review*, 33(2):115–121, 2016.

[33] Yu-Chi Chen, Raylin Tso, Gwoboa Horng, Chun-I Fan, and Ruei-Hau Hsu. Strongly secure certificateless signature: Cryptanalysis and improvement of two schemes. *J. Inf. Sci. Eng.*, 31(1):297–314, 2015.

[34] Yu-Chi Chen, Raylin Tso, Masahiro Mambo, Kaibin Huang, and Gwoboa Horng. Certificateless aggregate signature with efficient verification. *Security and Communication Networks*, 8(13):2232–2243, 2015.

[35] Lin Cheng, Zhengping Jin, Oiaovan Wen, and Hua Zhang. A novel privacy preserving keyword searching for cloud storage. In *Privacy, Security and Trust (PST), 2013 Eleventh Annual International Conference on*, pages 77–81. IEEE, 2013.

[36] Christopher D Clack, Vikram A Bakshi, and Lee Braine. Smart contract templates: foundations, design landscape and research directions. *arXiv preprint arXiv:1608.00771*, 2016.

[37] George Coulouris, Jean Dollimore, Tim Kindberg, and Gordon Blair. *Distributed Systems: Concepts and Design*. Addison-Wesley Publishing Company, USA, 5th edition, 2011.

[38] Jason Paul Cruz, Yuichi Kaji, and Naoto Yanai. Rbac-sc: Role-based access control using smart contract. *IEEE Access*, 6:12240–12251, 2018.

[39] Patrick Dai, Neil Mahi, Jordan Earls, and Alex Norta. Smart-contract value-transfer protocols on a distributed mobile application platform. *URL: https://qtum. org/uploads/files/cf6d69348ca50dd985b60425ccf282f3. pdf*, 2017.

[40] Angsuman Das, Avishek Adhikari, and Kouichi Sakurai. Plaintext checkable encryption with designated checker. *Adv. in Math. of Comm.*, 9(1):37–53, 2015.

[41] Keita Emura, Atsuko Miyaji, Akito Nomura, Kazumasa Omote, and Masakazu Soshi. A ciphertext-policy attribute-based encryption scheme with constant ciphertext length. In *International Conference on Information Security Practice and Experience*, pages 13–23. Springer, 2009.

[42] Liming Fang, Willy Susilo, Chunpeng Ge, and Jiandong Wang. Public key encryption with keyword search secure against keyword guessing attacks without random oracle. *Information Sciences*, 238:221–241, 2013.

[43] Ian Foster, Carl Kesselman, Jeffrey M Nick, and Steven Tuecke. Grid services for distributed system integration. *Computer*, (6):37–46, 2002.

[44] Sanjam Garg, Steve Lu, Rafail Ostrovsky, and Alessandra Scafuro. Garbled ram from one-way functions. In *Proceedings of the forty-seventh annual ACM symposium on Theory of computing*, pages 449–458. ACM, 2015.

[45] Craig Gentry. Fully homomorphic encryption using ideal lattices. In *Proceedings of the 41st annual ACM symposium on Symposium on theory of computing-STOC\'09*, pages 169–169. ACM Press, 2009.

[46] Craig Gentry and Dan Boneh. *A fully homomorphic encryption scheme*, volume 20. Stanford University Stanford, 2009.

[47] Craig Gentry, Shai Halevi, Steve Lu, Rafail Ostrovsky, Mariana Raykova, and Daniel Wichs. Garbled ram revisited. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 405–422. Springer, 2014.

[48] Shafi Goldwasser, Yael Kalai, Raluca Ada Popa, Vinod Vaikuntanathan, and Nickolai Zeldovich. Reusable garbled circuits and succinct functional encryption. In *Proceedings of the forty-fifth annual ACM symposium on Theory of computing*, pages 555–564. ACM, 2013.

[49] Vipul Goyal, Abhishek Jain, Omkant Pandey, and Amit Sahai. Bounded ciphertext policy attribute based encryption. In *International Colloquium on Automata, Languages, and Programming*, pages 579–591. Springer, 2008.

[50] Wien Hong, Tung-Shou Chen, and Han-Yan Wu. An improved reversible data hiding in encrypted images using side match. *IEEE Signal Processing Letters*, 19(4):199–202, 2012.

[51] Kaibin Huang, Raylin Tso, and Yu-Chi Chen. Somewhat semantic secure public key encryption with filtered-equality-test in the standard model and its extension to searchable encryption. *Journal of Computer and System Sciences*, 2017.

[52] Kaibin Huang, Raylin Tso, Yu-Chi Chen, Sk Md Mizanur Rahman, Ahmad Almogren, and Atif Alamri. Pke-aet: public key encryption with authorized equality test. *The Computer Journal*, 58(10):2686–2697, 2015.

[53] Yan Huang, David Evans, Jonathan Katz, and Lior Malka. Faster secure two-party computation using garbled circuits. In *USENIX Security Symposium*, volume 201, pages 331–335, 2011.

[54] Y. H. Hwang and P. J. Lee. Public key encryption with conjunctive keyword search and its extension to a multi-user system. In *Proceedings of Pairing 2007*, pages 2–22. Lecture Notes in Computer Science 4575, Springer, 2007.

[55] Ari Juels, Ahmed Kosba, and Elaine Shi. The ring of gyges: Investigating the future of criminal smart contracts. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, pages 283–295. ACM, 2016.

[56] Mustafa S Abdul Karim and Koksheik Wong. Universal data embedding in encrypted domain. *Signal Processing*, 94:174–182, 2014.

[57] Ahmed Kosba, Andrew Miller, Elaine Shi, Zikai Wen, and Charalampos Papamanthou. Hawk: The blockchain model of cryptography and privacy-preserving smart contracts. In *2016 IEEE symposium on security and privacy (SP)*, pages 839–858. IEEE, 2016.

[58] Shyam Nandan Kumar. Cryptography during data sharing and accessing over cloud. *International Transaction of Electrical and Computer Engineers System*, 3(1):12–18, 2015.

[59] Junzuo Lai, Robert H Deng, and Yingjiu Li. Fully secure cipertext-policy hiding cp-abe. In *International Conference on Information Security Practice and Experience*, pages 24–39. Springer, 2011.

[60] Ming Li and Yang Li. Histogram shifting in encrypted images with public key cryptosystem for reversible data hiding. *Signal Processing*, 130:190–196, 2017.

[61] Ninghui Li, Tiancheng Li, and Suresh Venkatasubramanian. t-closeness: Privacy beyond k-anonymity and l-diversity. In *Data Engineering, 2007. ICDE 2007. IEEE 23rd International Conference on*, pages 106–115. IEEE, 2007.

[62] Yehuda Lindell and Benny Pinkas. An efficient protocol for secure two-party computation in the presence of malicious adversaries. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 52–78. Springer, 2007.

[63] Qin Liu, Guojun Wang, and Jie Wu. An efficient privacy preserving keyword search scheme in cloud computing. In *Computational Science and Engineering, 2009. CSE'09. International Conference on*, volume 2, pages 715–720. IEEE, 2009.

[64] Kede Ma, Weiming Zhang, Xianfeng Zhao, Nenghai Yu, and Fenghua Li. Reversible data hiding in encrypted images by reserving room before encryption. *IEEE Transactions on information forensics and security*, 8(3):553–562, 2013.

[65] Sha Ma, Qiong Huang, Mingwu Zhang, and Bo Yang. Efficient public key encryption with equality test supporting flexible authorization. *IEEE Transactions on Information Forensics and Security*, 10(3):458–470, 2015.

[66] Sha Ma, Yi Mu, and Willy Susilo. A generic scheme of plaintext-checkable database encryption. *Information Sciences*, 429:88–101, 2018.

[67] Sha Ma, Yi Mu, Willy Susilo, and Bo Yang. Witness-based searchable encryption. *Information Sciences*, 453:364–378, 2018.

[68] Sha Ma, Mingwu Zhang, Qiong Huang, and Bo Yang. Public key encryption with delegated equality test in a multi-user setting. *The Computer Journal*, 58(4):986–1002, 2015.

[69] Ashwin Machanavajjhala, Johannes Gehrke, Daniel Kifer, and Muthuramakrishnan Venkitasubramaniam. \ell-diversity: Privacy beyond\kappa-anonymity. In *null*, page 24. IEEE, 2006.

[70] Satoshi Nakamoto. Bitcoin: A peer-to-peer electronic cash system. 2008.

[71] Zhicheng Ni, Yun-Qing Shi, Nirwan Ansari, and Wei Su. Reversible data hiding. *IEEE Transactions on circuits and systems for video technology*, 16(3):354–362, 2006.

[72] Pascal Paillier. Public-key cryptosystems based on composite degree residuosity classes. In *International Conference on the Theory and Applications of Cryptographic Techniques*, pages 223–238. Springer, 1999.

[73] Pascal Paillier. Public-key cryptosystems based on composite degree residuosity classes. In *International Conference on the Theory and Applications of Cryptographic Techniques*, pages 223–238. Springer, 1999.

[74] D. J. Park, K. Kim, and P. J. Lee. Public key encryption with conjunctive field keyword search. In *Fifth International Workshop WISA'04*, pages 73–86. Lecture Notes in Computer Science 3325, Springer, 2004.

[75] H. S. Rhee, J. H. Park, W. Susilo, and D. H. Lee. Trapdoor security in a searchable public-key encryption scheme with a designated tester. *Journal of Systems and Software*, 83(5):763–771, 2010.

[76] Chih-Wei Shiu, Yu-Chi Chen, and Wien Hong. Encrypted image-based reversible data hiding with public key cryptography from difference expansion. *Signal Processing: Image Communication*, 39:226–233, 2015.

[77] Nick Szabo. Smart contracts. *Unpublished manuscript*, 1994.

[78] Qiang Tang. Public key encryption schemes supporting equality test with authorisation of different granularity. *International journal of applied cryptography*, 2(4):304–321, 2012.

[79] Jun Tian. Reversible data embedding using a difference expansion. *IEEE transactions on circuits and systems for video technology*, 13(8):890–896, 2003.

[80] Marten Van Dijk, Craig Gentry, Shai Halevi, and Vinod Vaikuntanathan. Fully homomorphic encryption over the integers. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 24–43. Springer, 2010.

[81] H. Y. Wu W. Hong, T. S. Chen. An improved reversible data hiding in encrypted images using side match. *IEEE Signal Processing Letters*, 19:199–202, 2011.

[82] Brent Waters. Ciphertext-policy attribute-based encryption: An expressive, efficient, and provably secure realization. In *International Workshop on Public Key Cryptography*, pages 53–70. Springer, 2011.

[83] Gavin Wood. Ethereum: A secure decentralised generalised transaction ledger. *Ethereum project yellow paper*, 151:1–32, 2014.

[84] DN Wu, QQ Gan, and XM Wang. Verifiable public key encryption with keyword search based on homomorphic encryption in multi-user setting. *IEEE Access*, 2018.

[85] Peng Xu, Hai Jin, Qianhong Wu, and Wei Wang. Public-key encryption with fuzzy keyword search: A provably secure scheme under keyword guessing attack. *IEEE Transactions on computers*, 62(11):2266–2277, 2013.

[86] Guomin Yang, Chik How Tan, Qiong Huang, and Duncan S Wong. Probabilistic public key encryption with equality test. In *Cryptographers' Track at the RSA Conference*, pages 119–131. Springer, 2010.

[87] Yang Yang, Ximeng Liu, Xianghan Zheng, Chunming Rong, and Wenzhong Guo. Efficient traceable authorization search system for secure cloud storage. *IEEE Transactions on Cloud Computing*, 2018.

[88] Andrew Chi-Chih Yao. How to generate and exchange secrets. In *Foundations of Computer Science, 1986., 27th Annual Symposium on*, pages 162–167. IEEE, 1986.

[89] W. C. Yau, R. C. Phan, S. H. Heng, and B. M. Goi. Keyword guessing attacks on secure searchable public key encryption schemes with a designated tester. *International Journal of Computer Mathematics*, 90(12):2581–2587, 2013.

[90] Chia-Mu Yu, Chi-Yuan Chen, and Han-Chieh Chao. Privacy-preserving multikeyword similarity search over outsourced cloud data. *IEEE Systems Journal*, 11(2):385–394, 2017.

[91] Lan Zhang. *Privacy-preserving Computing and Applications*. PhD thesis, Tsinghua University, 2014.

[92] Lan Zhang, Taeho Jung, Cihang Liu, Xuan Ding, Xiang-Yang Li, and Yunhao Liu. Pop: Privacy-preserving outsourced photo sharing and searching for mobile devices. In *Distributed Computing Systems (ICDCS), 2015 IEEE 35th International Conference on*, pages 308–317. IEEE, 2015.

[93] Weiming Zhang, Kede Ma, and Nenghai Yu. Reversibility improved data hiding in encrypted images. *Signal Processing*, 94:118–127, 2014.

[94] X. Zhang. Reversible data hiding in encrypted image. *IEEE Signal Processing Letters*, 18:255–258, 2011.

[95] Xinpeng Zhang. Separable reversible data hiding in encrypted image. *IEEE transactions on information forensics and security*, 7(2):826–832, 2012.

[96] Xinpeng Zhang, Jing Long, Zichi Wang, and Hang Cheng. Lossless and reversible data hiding in encrypted images with public-key cryptography. *IEEE Transactions on Circuits and Systems for Video Technology*, 26(9):1622–1631, 2016.

[97] Yousheng Zhou, Xiaofeng Zhao, Siling Liu, Xingwang Long, and Wenjun Luo. A time-aware searchable encryption scheme for ehrs. *Digital Communications and Networks*, 2018.

[98] Guy Zyskind, Oz Nathan, et al. Decentralizing privacy: Using blockchain to protect personal data. In *Security and Privacy Workshops (SPW), 2015 IEEE*, pages 180–184. IEEE, 2015.