

國立臺灣大學理學院數學系

碩士論文

Department of Mathematics

College of Science

National Taiwan University

Master Thesis



閾值密碼學的研究與分類

Threshold Cryptography: A Survey and Taxonomy

吳秉宸

Ping-Chen Wu

指導教授：陳君明 博士

Advisor: Jiun-Ming Chen, Ph.D.

中華民國 108 年 7 月

July 2019



致謝

漫長的五年，中間經歷了對求學的迷惘以及服兵役的一年，在指導教授陳君明教授耐心的指導並給予一次又一次的機會之下，終於將論文完成。在過去的幾年，教授總能憑藉著對密碼學的熟悉和對我的了解，給予我最恰到好處的任務，不會讓我無法完成，但也不會太過輕而易舉。在完成教授指派的各項任務同時，讓我逐漸熟悉密碼學的各個領域，並且在其中找到感興趣的主題開始鑽研。

另外我也要特別感謝蘇昱丞學長，以及唐爾晨、張凱傑、洪逸霖等學弟。感謝這些學長學弟常常陪我討論讀論文時遇到的各種問題。互相腦力激盪，天馬行空，不拘泥於舊有的限制。往往能讓我順利度過一個又一個研究時的瓶頸。



摘要

自區塊鏈問世以來，大眾對隱私的擔憂不斷提升。同時，密碼學的相關發展，如多方計算 (MPC)、零知識證明和同態加密等，為閾值密碼學發展奠定了穩固基礎。本研究深入討論兩種主要類型的閾值 ECDSA，並以演算法為例進行驗證。此外，根據回合數、傳輸量以及計算量，對兩種算法進行全面性比較。另一方面，本研究進一步探討閾值加密技術的各種應用，包括 TOPRF、TPPSS 以及雲端計算中的各種應用。

關鍵詞：閾值密碼學，多方計算，ECDSA，同態加密，雲端計算



Abstract

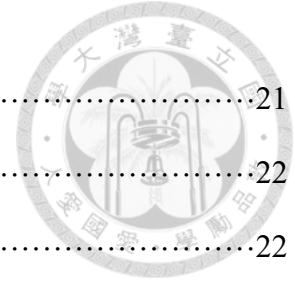
Ever since the emergence of blockchain, the concerns to privacy have been rising among the public. Meanwhile, the advancements of cryptography, such as MPC (Multi Party Computation), zero-knowledge proof, and homomorphic encryption, etc., pave a consolidated foundation for the threshold cryptography development. In this study, two major types of threshold ECDSA were discussed in depth, and each of them was testified via an algorithm as an example. In addition, the two algorithms were also compared comprehensively based on the number of rounds, the amount of transmission, and the amount of calculation. Furthermore, various applications of threshold cryptography, including TOPRFs, TPPSS, and a variety of applications in cloud computing, were also explored in this study.

Key words: threshold cryptography, MPC, ECDSA, homomorphic encryption, cloud computing

Table of Contents



致謝	ii
摘要	iii
Abstract	iv
Table of figures	viii
Chapter 1. Introduction	9
Chapter 2. Definition and Tools	12
2.1 Decisional Composite Residuosity Assumption (DCRA)	12
2.2 Paillier cryptosystem ¹¹	12
2.3 Oblivious Transfer (OT)	13
2.4 Multiplication into addition	14
2.5 Schnorr’s zero-knowledge proof with Fiat–Shamir heuristic	15
Chapter 3. Standardization, Applications and Challenges of Threshold Cryptography	17
3.1 Standardization and recommendations of threshold cryptography	17
3.2 Challenges and issues in standardization of threshold cryptography	18
3.3 Security of threshold cryptography	20
3.3.1 Threshold values	20
3.3.2 Concerning tradeoff among security properties	20
3.3.3 Confidentiality, integrity and availability	20



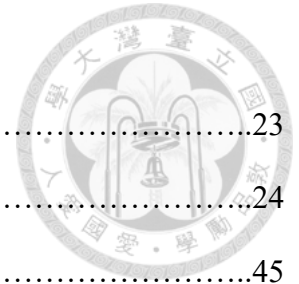
3.3.4 Defining f_x	21
3.4 Applications of threshold cryptography	22
3.4.1 Threshold Oblivious Pseudo-Random Functions (TOPRFs)	22
3.4.2 Threshold Oblivious Password Protected Secret Sharing (TOPPSS)	25
Chapter 4. Applications of threshold signature	27
4.1 Multi signature vs. Threshold signature	27
4.2 Threshold ECDSA	29
4.3 Securing Bitcoin wallets via a new DSA/ECDSA threshold signature scheme.	29
4.3.1 R Gennaro's algorithm	30
4.3.2 Efficiency analysis	32
4.4 Threshold ECDSA from ECDSA Assumptions ^{14, 15}	34
4.4.1 Doerner's algorithm	35
4.4.2 Efficiency analysis	37
4.5 Comparisons between the two studies	39
4.6 Other forms of threshold secret sharing	40
4.6.1 Weighted threshold ⁴⁷⁻⁵¹	40
Chapter 5. Threshold Cryptography on Cloud Computing	42
5.1 Introduction of cloud computing	42
5.2 The NIST definition of cloud computing	42
5.3 Threshold Cryptography on Cloud Computing	43



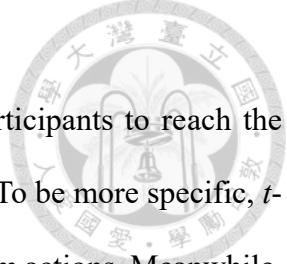
5.3.1 Threshold cryptography based on data security in cloud computing ⁵⁵	44
5.3.2 A secured key for cloud computing using threshold cryptography in Kerberos ⁵⁶	46
5.3.3 Searching for the optimal value for threshold on cloud computing ⁵⁷	48
Chapter 6. Conclusions	50
References	52

Table of Figures

Figure 1.....	23
Figure 2.....	24
Figure 3.....	45
Figure 4.....	47
Figure 5.....	47



Chapter 1. Introduction



Threshold cryptography is a technique that requires the number of participants to reach the threshold in order to sign or verify a message, and encrypt or decrypt data. To be more specific, t -out-of- n threshold represents any t (including above) participants can perform actions. Meanwhile, $t-1$ (or fewer) participants have little authorities, and they are not allowed to obtain any information about signing, verifying, encrypting or decrypting. The concept of threshold cryptography was first proposed by Adi Shamir in "How to share a secret" in 1979¹. He indicated that, in the traditional scenario, cryptography consisted of one sender, one receiver, and an active or passive eavesdropper who was an opponent. However, the situation may be complex when a scenario with multiple transmitters or multiple receivers is considered. For example, a procurement department of a multinational company requires two supervisors to sign and agree to proceed the purchasing process. In this case, two transmitters will be included in such scenario. In order to cope with this issue, the study of threshold cryptography has been increasingly thriving²⁻⁷. The National Institute of Standards and Technology (NIST) is also interested in the subfield of cryptography, and the institute released a series of reports to address its position on the threshold field⁸. NIST also held a threshold workshop in March 2019, which will be further described in a following chapter.

Among all the characteristics of threshold cryptography, the ability of ruling out single points of failure is worth noting. One can grasp the concept of single point of failure by borrowing the motto of the American Navy SEAL: 'Two is one, one is none.' In other words, two sets of equipment are usually prepared, while in fact only one set is accessible because one of them may malfunction at any time. If only one set is prepared, however, it means there is no equipment available at all. This motto best describes the concept of single point of failure. In the real world, a magnificent system can also be completely shut down due to a node failure. As what Yvo C.

Desmedt describe², there are only one transmitter and one receiver for traditional cryptography considerations. For instance, when it comes to the private key of Bitcoin, there is only one private key available. If the owner of the account loses the private key, the account can no longer be accessed. If threshold signature is applied, however, it is possible to allow the account to be accessed in cases where at most $(n - t)$ the private key is lost.

In the Chapter 2, various tools used in threshold cryptography will be discussed, including oblivious transfer (OT), a technology heavily used in multiparty computation (MPC). For instance, Yao's Millionaires' Problem⁹ is one of the classic OT applications. In addition, there is also a method of converting addition to multiplication, a technique proposed by Niv Gilboa¹⁰. Based on OT, it is possible to change the relationship of the part owned by both parties from multiplication to addition. Besides, the Paillier cryptosystem¹¹, a cryptosystem that implements additive homomorphic encryption, will also be introduced. Finally, the Schnorr protocol¹², a fairly efficient zero-knowledge proof, will be discussed as well.

In Chapter 3, NIST's recommendations and comments of threshold cryptography will be primarily discussed, followed by the review of 'National IRTIR 8214 Threshold Schemes for Cryptographic Primitives' published by NIST in 2018⁸. Besides, several fundamental features of threshold will also be introduced. Finally, the NIST office's threshold workshop and various applications about threshold, including OPRF, TOPRF, PPSS, etc., will be addressed.

In Chapter 4, at first, the differences between multi signature and threshold signature will be discussed by comparing the advantages and disadvantages of the two signatures, followed by the introduction of the threshold ECDSA, a threshold version of the most popular digital signatures used on cryptocurrency. In addition, considering the strong motivation for threshold ECDSA, an in-depth analysis of the current two major types of threshold ECDSA was conducted in this study.

The amount of computation required to perform Paillier's encryption, decryption and homomorphic addition was firstly analyzed, followed by the introduction of Gennaro's algorithm¹³. Meanwhile, the algorithm of Doerner^{14,15}, a threshold ECDSA algorithm that does not require additional hardness assumptions, was also explored in the discussion. Subsequently the comparisons between these two algorithms, including the number of rounds, the amount of information transmitted, the amount of calculation and the t -of- n model, were performed in the study as well. Ultimately, the part of the two algorithms that may be optimized in the future were also discussed.

Lastly, in Chapter 5, cloud computing will be focused and discussed. Similar to the threshold cryptography, it is also a field greatly thriving. Due to the significant demands for MPC, cloud computing and threshold can be granted as advantageous and promising fields. Hence, NIST also anticipated to develop standards for cloud computing. At the beginning of the chapter, the various definitions of cloud computing proposed by NIST¹⁷ will be discussed, followed by the challenges encountered during the course of developing the standards. Afterwards, the chapter will be concluded by introducing several studies of cloud computing and threshold cryptography.

Chapter 2. Definition and Tools



2.1 Decisional Composite Residuosity Assumption (DCRA)

- **Definition.** A number z is said to be a n -th residue modulo n^2 if there exist a number $y \in Z_{n^2}^*$ such that

$$z = y^n \text{ mod } n^2.$$

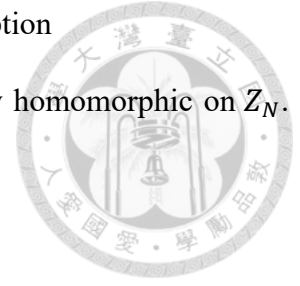
- **Conjecture.** There exists no polynomial time distinguisher for n -th residues modulo n^2 .

This intractability hypothesis will be referred to as the Decisional Composite Residuosity Assumption (DCRA).

2.2 Paillier cryptosystem¹¹

Key Generation	Generate two large prime P, Q of equal length, and set $N=PQ$. Let $\lambda(N)=\text{lcm}(P-1, Q-1)$ be the Carmichael function of N . Finally choose $\Gamma \in Z_{N^2}^*$ such that its order is a multiple of N .	
	Public key	(N, Γ)
	Secret key	$\lambda(N)$.
Encryption	To encrypt a message $m \in Z_N$, choose $x \in_R Z_N^*$.	
	Compute	$c = \Gamma^m x^N \text{ mod } N^2$
	Send	c
Decryption	To decrypt a ciphertext $c \in Z_{N^2}$, let L be a function defined over the set $\{u \in Z_{N^2} : u \equiv \text{mod } N\}$ computed as $L(u) = \frac{(u-1)}{N}$	
	Compute	$m = \frac{L(c^{\lambda(N)})}{L(\Gamma^{\lambda(N)})} \text{ mod } N$

Choose $x \in_R Z_N^$ means randomly choose $x \in Z_N^*$.



● **Additive Homomorphic Properties.** As already seen, the two encryption functions $m \mapsto \Gamma^m r^N \bmod N^2$ and $m \mapsto \Gamma^{(m+Nr)} \bmod N^2$ are additively homomorphic on Z_N .

Practically, this leads to the following identities:

$\forall m_1, m_2 \in Z_N$ and $k \in \mathbb{N}$

◇ $e(m_1) +_E e(m_2) \equiv e(m_1)e(m_2) \equiv e(m_1 + m_2) \bmod N^2$.

◇ $k \times_E e(m) \equiv (e(m))^k \equiv e(km) \bmod N^2$.

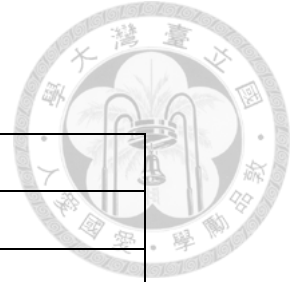
2.3 Oblivious Transfer (OT)

OT is a promising cryptographic primitive, which is a fairly important and basic technology in the field of secure multiparty computation. OT was originally invented by Michael O. Rabin in 1981¹⁷. The original version of OT is a message m sent by sender to the receiver. Receiver has a 1/2 probability to receive. Sender knows nothing about whether the receiver has received the message. If there is only such an effect, there is actually useless. Later, the better-used OT was invented by Shimon Even, Oded Goldreich, and Abraham Lempel¹⁸. This technique was invented to perform secure multiparty computation. After that, people continued to invent 1-of- n OT¹⁹⁻²² and t -of- n OT²³⁻²⁵. Furthermore, there is also an OT Extension available to improve OT efficiency²⁶.

In the study, the 1-of-2 OT system is applied. 1-of-2 OT is performed by two parties (sender and receiver). Sender masters two messages m_0 and m_1 . Receiver masters 1 bit c ($c = 0$ or 1). Sender will send two packages to the receiver, which contain m_0 and m_1 respectively. Although the receiver receives two packages, he or she can only open one of the packages (m_c), and the other package will only see a series of meaningless information if it is turned on. After the end, the sender will not know which message the receiver receives. The receiver only sees the target message (m_c) and knows nothing about the other message.

The following is the ‘The Simplest Protocol for Oblivious Transfer’ invented by Tung Chou

and Claudio Orlandi²⁷ based on the Diffie-Hellman key Exchange²⁸.



	Sender	Receiver
Input	m_0, m_1	c
Output	none	m_c

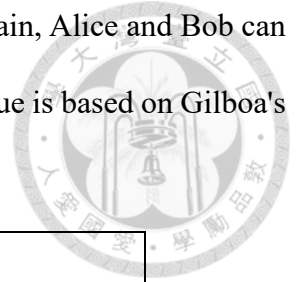
	Sender		Receiver	
1.	Choose $a \in_R Z_p$		Choose $b \in_R Z_p$	
	Compute	$A = g^a$		
	Send	A		
2.			Compute	if $c = 0, B = g^b$. if $c = 1, B = Ag^b$ $k_r = H(A^b)$
			Send	B
3.	Compute	$k_0 = H(B^a)$ $k_1 = H\left(\left(\frac{B}{A}\right)^a\right)$ $e_0 = E_{k_0}(m_0)$ $e_1 = E_{k_1}(m_1)$		
	Send	(e_0, e_1)		
4.			Compute	$m_c = D_{k_r}(e_c)$

*H: hash function. E: symmetric encryption system. D: decryption of E.

2.4 Multiplication into addition

Multiplication into addition is used in the F_{mul} part of Doerner's paper¹⁴. Multiplication into addition is a fairly critical technique in the paper. Because the final signature of ECDSA is in the form of $(H(m) + s_k \cdot r)/k$. The part of k in the denominator causes ECDSA to be very unsuitable for performing threshold. So Doerner moved their attention to the $1/k$ and s_k/k parts of the signature. Perform F_{mul} by $1/k_a$ of Alice and $1/k_b$ of Bob, it can let Alice and Bob get t_a^1 and t_b^1

respectively. Where $t_a^1 + t_b^1 = (1/k_a) \cdot (1/k_b)$. Similarly, execute F_{mul} again, Alice and Bob can also get t_a^2 and t_b^2 whose sum is equal to $(sk_a/k_a) \cdot (sk_b/k_b)$. This technique is based on Gilboa's paper¹⁰. See the following protocol for details.



		Alice	Bob
1.	Choose $s_0, \dots, s_{p-1} \in_R Z_q$		
	Let $t_i^0 = s_i$ for all i.		
	Compute	$t_i^1 = 2^i a + s_i$, for all i.	
	Send	$(t_0^0, t_0^1), \dots, (t_{p-1}^0, t_{p-1}^1)$	
2.	Let the binary representation of b be b_{p-1}, \dots, b_0 . Alice and Bob execute p times 1-of-2 OTs. In the i-th invocation Bob chooses $t_i^{b_i}$ from the pair (t_i^0, t_i^1) .		
3.	Compute	$x = - \sum_{i=0}^{p-1} s_i$	Compute $y = \sum_{i=0}^{p-1} t_i^{b_i}$

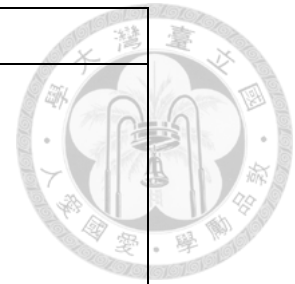
2.5 Schnorr's zero-knowledge proof with Fiat-Shamir heuristic

Schnorr's zero-knowledge proof¹² is one of the simplest and frequently used proofs of knowledge. Fiat-Shamir heuristics²⁹ are seen as converting public-coin interactive knowledge proofs into non-interactive knowledge proofs. The following is the working process of the two together.

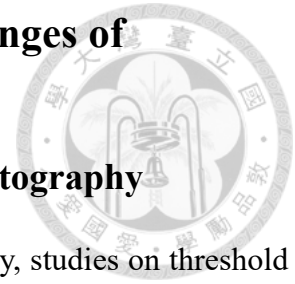
- The protocol is defined for a cyclic group G_q of order q with generator g.
- Prover wants to prove that he knows x of $y = g^x \text{ mod } q$.

	Prover		Verifier	
1.	Choose $r \in_R Z_q$			
	Compute	$t = g^r$ $c = H(g, y, t)$ $s = r + c \cdot x$		
	Send	(s,c,t)		
2.			Compute	$v_0 = g^s$ $v_1 = t \cdot y^c$
			Output	Accept, if $v_0 = v_1$. Reject, if $v_0 \neq v_1$.

* H: hash function.



Chapter 3. Standardization, Applications and Challenges of Threshold Cryptography

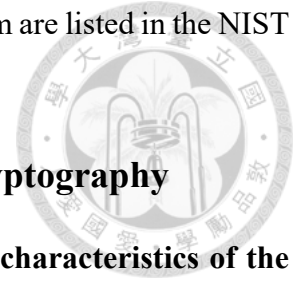


3.1 Standardization and recommendations of threshold cryptography

Due to the advanced development of threshold cryptography technology, studies on threshold cryptography have been increasingly dominant and thriving. Acting as one of the leading roles in cryptography, it is inevitable for National Institute of Standards and Technology (NIST) to establish its sophisticated guidelines and recommendations. NIST demonstrated its determination to standardize threshold encryption schemes in the report conducted by Brandão et al in 2019⁸, and the institute also revealed the benefits of standardized threshold encryption schemes. Developing cryptographic primitives with a well-characterized threshold scheme provides significant advantages of security, intriguing the majority to be involved in the threshold scheme of the NIST-approved encryption primitives.

However, there are still concerns remained in terms of application and flexibility. For instance, what conditions should everyone adopt the guidelines as a standard for selecting standard threshold encryption schemes? What parameters and functional flexibility can the threshold encryption scheme standard tolerate? Should additional standardization and verification be set independently for some basic primitives? These issues are not directly resolved in the 2019 report. Instead, the report suggests people of interests to solve these problems and develop an objective basis at the user end. Meanwhile, the report also addresses a variety of representative issues that need to be considered, namely safety certification assessment, operational efficiency and implementation applicability. Nevertheless, the report also presents promising applications related to the standardization of threshold encryption schemes. Hence, the process of solving these issues may bring a significant leap forward to the security of the cryptographic primitives. Prior to the standard

establishment, there is still presence of various concerns, and several of them are listed in the NIST Internal Report (NISTIR) 8214⁸ and will be discussed in Chapter 3.2.



3.2 Challenges and issues in standardization of threshold cryptography

Question 1: Are the designated criteria able to properly describe the characteristics of the threshold scheme?

When it comes to such an issue, one is suggested to first clearly define the threshold scheme even under the circumstances where the representations such as f -out-of- n threshold are not unified. In addition, although threshold cryptography is not considered as a state-of-the-art technology, it should be acknowledged that such advancement is considerably distinct from other cryptographic primitives. Besides ensuring cryptography to yield desirable characteristics of the threshold, one should also take security and efficiency into consideration.

Question 2: What is the efficiency and performance of the operation as a function of the threshold parameter?

Similar to the approaches to defining the optimal value (*i.e.* k) for threshold cryptography on cloud computing addressed in the NIST report, one is suggested to firstly consider the uses of security properties, including confidentiality, integrity and availability, and subsequently discuss the tradeoff of the security property and operational efficiency.

Question 3: Can the complexity of implementation possibly lead to new errors or misconfigurations?

Although threshold cryptography technology has the chances to tolerate a range of side channel attacks, such as differential power analysis (DPA) and differential fault analysis (DFA), its novelty may lead to unfamiliar challenges. After the standard guidelines and recommendations have been proposed, there may be possibilities where new attacks occur, or new errors appear due to the

complexity of the actual operation.

Question 4: Is the security verification reliable?

Despite the fact that the security verification of threshold cryptography was carefully implemented, it still takes time for a new cryptosystem to consolidate its reliability and authenticity. The threshold cryptography scheme adopting the NIST standards is considered as a brand-new cryptosystem, where complete and rigorous proof is still desired to justify its safety assessment.

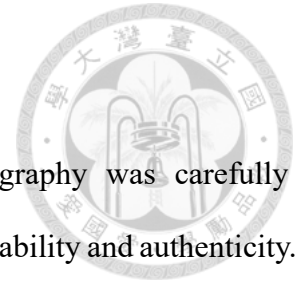
Question 5: How is its reliability compared to the traditional implementations?

The reliability of traditional implementations has been well understood and recognized. When it comes to that of a new system, however, one may be easily influenced and show moderate to low tolerance of corrupted nodes.

Question 6: Is the scheme applicable to NIST-approved cryptographic primitives?

According to the Federal Information Processing Standards Publication 186-4 (FIPS-186-4)³⁰, three NIST-approved digital signature algorithms have been specified, including: digital signature algorithm (DSA), algorithm developed by Rivest, Shamir and Adleman (RSA)³¹ and elliptic curve digital signature algorithm (ECDSA). Therefore, when one is anticipated to establish new standards, these digital signature algorithms should be aligned with the current modernization process and incorporate the structure of the testing methodology derived by the NIST cryptographic validation programs. This approach may greatly save time and efforts in updating resources.

In addition to the challenges described above, there are still many issues to be considered. Overall, NIST expects its institute to drive an open and transparent process towards standardization of threshold schemes for cryptographic primitives. During the course of publishing official guidelines, the institute promised to consult cryptography research community as well as



stakeholders in the government and industry to deliver sophisticated outcomes fulfilling the needs of all aspects.



3.3 Security of threshold cryptography

3.3.1 Threshold values

Overall, the total number n of components, or the ‘thresholds,’ can be expressed in two ways: a decent component (i.e., undamaged) of the minimum required number k and an undesirable component of a maximally allowable number f . It is worth noting that $f + k$ does not necessarily equal n .

3.3.2 Concerning tradeoff among security properties

The question of whether threshold cryptography is more secured than traditional cryptography still remains unknown. If one performs analyses of various security properties on study cases, it will be found that while threshold cryptography may enhance the security nature of certain aspects, it can also reveal its additional weaknesses in security. Therefore, the tradeoff of various security properties among each other should also be considered when applying threshold cryptography. Under such circumstances, the use of threshold cryptography should be carefully evaluated in real-world scenarios. The following is a tradeoff among various security features based on the three properties of information security: confidentiality, integrity and availability (CIA). Next, threshold values corresponding to the nature of CIA will be discussed.

3.3.3 Confidentiality, integrity and availability

Confidentiality, integrity and availability, or CIA, are the security triangle portfolio of information security. Any violation of the incident or behavior of the triangle will reduce the protection strength of the security mechanism and threaten the company's important assets or confidential information. Therefore, CIA is the core judgment criterion of information security.

Every employee of an entity is able to identify the events or behaviors requiring regulation and management based on CIA principles.

Confidentiality serves its main purpose to maintain the confidentiality of information, meaning any confidential information cannot be disclosed under unauthorized entities, including a person, a group or a system. In generally, the definition of f derived by threshold cryptography indicates the of property confidentiality is satisfied.

Integrity, on the other hand, implies its purpose to maintain the veracity, consistency and completeness of the information, meaning the modification of any confidential information must be authorized and not tampering.

Additional, availability presents its purpose of maintaining the smoothness of the work activities, meaning the authorized entity can obtain or use the information ‘opportunely’ and ‘without interruption.’

3.3.4 Defining f_x

Threshold value f_x represents the maximum number of incorrectly operated members that can be tolerated under the property of maintaining x .

Example 1. Threshold signature agreement

Any kind of n -out-of- n threshold signing agreement shall be considered. According to the definition, whenever a signature is to be completed, n members must work together. If there are only $n-1$ members, the signature fails. In this case, if one would like to maintain the confidentiality (on the basis where the adversaries are able to complete the signature), the confidentiality can be managed by the system by failing the signature as long as one follows the specifications. Thus, $f_c = n - 1$. On the other hand, concerning the integrity, if one does not use the original components for calculation deliberately, indicating that the signature was tampered with and will be identified

immediately the verifier. Hence, $f_i = 0$. Finally, when it comes to the availability, the signature cannot be completed as long as one refuses to perform the calculation. Therefore, $f_a = 0$. According to the example described earlier, the agreement can obtain an optimal threshold for confidentiality ($f_c = n - 1$) while yielding a pessimal threshold for availability and integrity ($f_a = 0$ and $f_i = 0$).

Example 2. Threshold random number generator

A threshold random number generator is granted as an approach to outputting uniformly random bit-strings. The production can be achieved by using three different random number generators, where S is defined as the XOR value of the three generator bit streams S_i ($i = 1, 2, 3$). In this case, integrity, or i , means whether S is truly random, whereas availability, or a , indicates whether S can be successfully generated. A genuine random S can be achieved if one of the S_i is random and independent of other S_i . Hence, based on the assumption of independence, $f_i = 2$. Meanwhile, since S fails to be defined when one of S_i is missing, therefore, $f_a = 0$.

To summarize the two examples described earlier, under a similar scenario, it is revealed that the threshold value will change accordingly in the circumstances where the designated security properties have been altered. In respect of applications, it is necessary to analyze and identify the properties required in order to meet the needs. Such challenge is something practical and inevitable to confront when establishing new threshold standards.

3.4 Applications of threshold cryptography

3.4.1 Threshold Oblivious Pseudo-Random Functions (TOPRFs)

Procedure 1. Pseudo-Random Functions (PRFs)^{32, 33}

Definition 1:

A function $F : \{0, 1\}^n \rightarrow \{0, 1\}^n$ is a random function if it is constructed as follows: For each $x \in \{0, 1\}^n$ pick a random $y \in \{0, 1\}^n$, and let $F(x) = y$.



Definition 2:

$F = \{f_s : \{0, 1\}^{|s|} \rightarrow \{0, 1\}^n \mid s \in \{0, 1\}^*\}$ is a family of PRFs if:

- [Easy to compute]
given $s \in \{0, 1\}^n$ and $x \in \{0, 1\}^n$, can efficiently compute $f_s(x)$.
- [Pseudo-randomness]

for all non-uniform PPT “oracle machines” D , there exists a negligible function $q(k)$ such that $|\Pr[s \leftarrow \{0, 1\}^k : (D^{f_s})(1^k) = 1] - \Pr[F \leftarrow RF_k : (D^f)(1^k) = 1]| \leq q(k) \square$.

In short, the purpose of PRF is to make it challenging to distinguish the differences between a PRF function and a truly random function, where the PRF simulates a genuine random function for ambiguity. An illustrated description is shown in **Figure 1**.

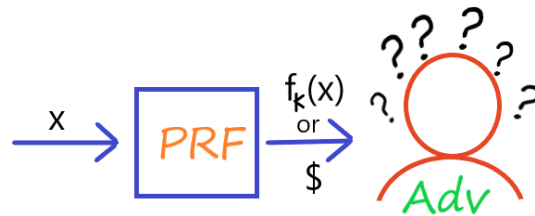


Figure.1

Procedure 2. Oblivious PRFs (OPRFs)³⁴

In terms of pseudo-random function, information is concealed from the two parties (server and user) that are involved in a PRF. Server holds the key (k) of the PRF and provides the service of the PRF. However, the server will not learn any information (including input and output) during the course of service. As for the user, one can get the output after the input is transferred and interacted with the server. Similarly, during the course of acquiring the output, the user will not receive any information about the function. An illustrated description is shown in **Figure 2**.

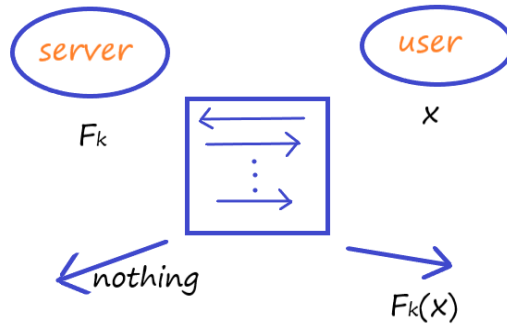


Figure.2

The extensive uses of OPRFs have been addressed in two studies conducted by Stanisław Jarecki, et al.^{35, 36}. Both studies discussed a variety of applications, including: private information retrieval (PIR), password protected secret sharing (PPSS), searchable encryption, file de-duplication, etc. OPRF can also be applied as the basis of PIR and oblivious transfer (OT).

Procedure 3. Diffie-Hellman based Oblivious Pseudo-Random Functions (DH- OPRFs)

Among the various OPRFs, there is a particularly common type called DH-OPRF. As its name suggests, it is an OPRF based on Diffie-Hellman assumption²⁸. The implementation method is shown as follows.

	User		Sever	
Input	x		F_k	
output	$F_k(x)$		none	
1.	Choose $r \in_R Z_q$			
	C	$a = (H'(x))^r$		
	S	a		
2.			C	$b = a^k$
			S	b
3.	C	$F_k(x) = H(x, b^{1/r})$		

Procedure 4. Threshold Oblivious PRFs (TO-PRFs)³⁶

Concerning that a user may correspond to multiple servers, threshold oblivious-PRFs, or TO-PRFs, are developed subsequently. As for a TO-PRF, the key (k) used by the PRF is composed of servers. These servers are combined via t -out-of- n threshold, finishing the threshold by using Shamir Secret Sharing. First, the user transfers the same a as the DH-OPRF to the t servers, which are selected by the users from the set $[n]$. Each server uses k_i to calculate b_i and send it to the user. The user subsequently applies Shamir Secret Sharing¹ to reassemble these b_i into a real output.

Similar to OPRF, the user is able to obtain the output at the end of the entire operation process accompanying with the condition where the server will not know the information about the input and the output. As for TOPRF, each server will not learn about the complete k as long as the corrupt servers do not exceed t .

3.4.2 Threshold Oblivious Password Protected Secret Sharing (TOPPSS)

Procedure 1. Password Protected Secret Sharing (PPSS)^{37,38}

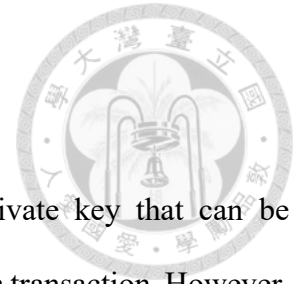
Password Protected Secret Sharing, which is a protocol that allows a user to share his or her data secretly among n trustees. Imagine a scene where Alice wants to trade with bitcoin, and she stores the private key on a server because the key is too difficult to remember due to the length of the private key. Whenever a trade is needed, she can retrieve the private key by entering the correct password. However, there is an underlying downside of such approach. The password stored at the server's end is vulnerable to a single point of failure. In this case, to recover the private key clandestinely, all it takes for the attacker is to break into the server and then install an offline dictionary attack to project the password created by the user. In order to cope with this flaw, one of the solutions is to store the secrets on multiple servers and have these servers share the private key. This mechanism is to ensure the attacker will need to break into multiple servers if one is

trying to fetch the private key. Nevertheless, there is another critical issue to be managed when taking this route. The situation can be complex when each server uses an individual password, while using a unified password is not desirable either because it is vulnerable to the attacks. Fortunately, the use of PPSS can solve the issues addressed earlier. The PPSS scheme allows users to secretly share secrets between n servers (with threshold t) and only a single password needs to be memorized for authentication. In this case, even if the attacker cracks any of the t (including the following) servers, one still fails to get the access to any information related to the private key.

Procedure 2. TOPPSS: PPSS via Threshold OPRF[40]

In a scheme where each server holds a random key k_i for an OPRF f independently, the designated secret is processed with a (t, n) secret sharing scheme at initiation. Each share is stored at one of the n servers, whereas server S_i stores the i -th shares encrypted under f_{k_i} (password). During the course of reconstruction, the user receives the encrypted shares from $t + 1$ servers and subsequently decrypts them by using the f_{k_i} (password) values run by the OPRF of each server.

Chapter 4. Applications of threshold signature



4.1 Multi signature vs. Threshold signature

When using virtual currency, such as Bitcoin, one will obtain a private key that can be considered as a bank account, and uses it to execute the signature and make a transaction. However, there may be underlying problems when using such approach. For instance, a user fails to get the access to the account of the private key is lost or stolen. In order to deal with this issues, multi signature is proposed as the solution^{40,41}. In a nutshell, multi signature requires more than one key to execute the signature, and 2-of-2, 2-of-3 or 3-of-3 are commonly seen. Surely the use of t -of- n can also be applied.

2-of-2 can be applied to a joint account such as a husband and wife. Any transaction must be permitted by both husband and wife to be performed. The 2-of-3 scene can be a person with three keys. One of the keys is carried with an individual, another key is hidden in a safe and offline place (such as a safe), and the other one is handed over to the company that provides the service. When the individual performs a transaction, he or she uses both his or her own key and the key kept by the company to sign it. If the key is lost, one can retrieve the key hidden in the safe and the trading activities will not be impeded. This mechanism can also prevent the company from hacking the individual's assets because there is only one key available.

However, there are also disadvantages of multi signature. First of all, the privacy is not sufficiently secured. All private keys required must be presented each time when individuals sign. Similarly, the efficiency can also be insufficient. For instance, the multi signature of 2-of-3 represents two private keys must be provided in each signature. The verifier also needs to verify the two private keys each time. Moreover, there is lack of flexibility. If an asset owner is not satisfied with the current t -of- n multi signature and would like to use the a -of- b scheme instead,

new private keys will be generated and all the inheritance will need to be transferred to the new account.

Therefore, threshold signature⁴²⁻⁴⁵ is considered as an ideal approach to solve the problems. Threshold signature does not generate a new private key. Instead, it distributes new parts to n users. When a t -name user signs and the threshold signature algorithm is operated, one will receive a signature signed with the original private key (or to be clearer, the signatures are indistinguishable). There are a number of advantages of threshold signature. First, the privacy is ensured and secured because the private keys are not required to be presented during the signing process, and no one else can identify who is involved in the transaction. Second, compared to multi signature, the use of threshold signature is more cost-effective because each transaction will only produce one signature. Thus, such application can save more resources than multi signature when transferring and verifying. Third, the use of threshold signature yields fair flexibility. If a user is not satisfied with the current t -of- n threshold signature and would like to switch to the a -of- b scheme, repeating the set-up steps alone is sufficient and no new private keys required.

Despite all the benefits discussed earlier, there are still disadvantages in regards to the use of threshold signature. First, the person who signs must be online. Performing the threshold signature algorithm requires the users to interact with each other. On the contrary, the multi signature users can send their signatures to the network in advance and then go offline. When other users come online, they can simply send their signatures along with the signatures of the offline signer. However, this can also be considered as one of the disadvantages when using multi signature. Once a transaction is signed, the signature cannot be recovered. Thus, the last party signed has a slight advantage over the other parties. Second, due to the limitation of current technology, the efficiency of threshold signature is promisingly satisfying. In the real world, only a handful of practical

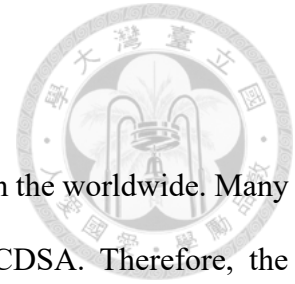
applications have been implemented.

4.2 Threshold ECDSA

ECDSA is one of the most commonly used digital signature algorithms in the worldwide. Many mainstream virtual currencies, including Bitcoin and Ethereum, use ECDSA. Therefore, the research on threshold ECDSA has been increasing, and the threshold signature introduced in this chapter will focus on threshold ECDSA. Currently, the Paillier cryptosystem has been widely adopted to implement threshold ECDSA. Paillier cryptosystem¹¹, invented by Pascal Paillier in 1999, addresses its sophisticated assumption based on the decisional composite residuosity assumption. Paillier cryptosystem is able to implement homomorphic encryption of additions and is therefore well utilized in the ECDSA threshold. However, the cryptosystem itself is extremely inefficient, so there are barely practical study cases in the real-world setting. On the other hand, Doerner proposed the threshold ECDSA algorithm in 2018¹⁴, using the technology of OT. One of the main outcomes discussed in the study is that a more promising efficiency could be achieved without the use of supernumerary assumptions than those using the Paillier cryptosystem.

4.3 Securing Bitcoin wallets via a new DSA/ECDSA threshold signature scheme.

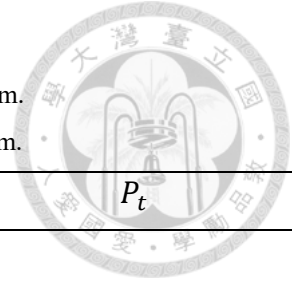
In the study conducted by R Gennaro et al., a set of threshold ECDSA was designed based on the Paillier cryptosystem¹³. This paper claims that they presented the first threshold signature scheme compatible with Bitcoin's ECDSA signatures. Meanwhile, the version of DSA was also introduced in the study. Therefore, the ECDSA version and the analysis of the efficiency R Gennaro's algorithm will be discussed in this section. (this algorithm is called as R algorithm for convenience)




4.3.1 R Gennaro's algorithm

*M: message. C: compute. S: send. H: hash function. E: encryption of Paillier cryptosystem.

D: decryption of Paillier cryptosystem. $+_E$: additive Homomorphic of Paillier cryptosystem.



	P_1	$P_2 \sim P_{t-1}$	P_t
setup	Participant P_i choose $x_i \in_R Z_q$		
	Secret key	$x = \prod_i x_i \text{ mod } q$	
	Public key	$y = x \cdot G \text{ in } \mathcal{G}$	
1.	Choose $k_1 \in_R Z_q$		
	C	$z_1 = k_1^{-1} \text{ mod } q$ $a_1 = E(z_1)$ $b_1 = E(x_1 z_1 \text{ mod } q)$ Set $a_1^* = b_1^* = \perp$	
	S	$M, a_1, b_1, a_1^*, b_1^*$ to P_2	
2 to t-1.			At round i , participant P_i . Abort if any input $\notin G_E$. Choose $k_i \in_R Z_q$.
	C	$z_i = k_i^{-1} \text{ mod } q$ $a_i = z_i \times_E a_{i-1}$ $b_i = E(x_i z_i \text{ mod } q) \times_E b_{i-1}$ $a_i^* = E(z_i), b_i^* = E(x_i z_i \text{ mod } q)$	
	S	$M, a_1, \dots, a_i, b_1, \dots, b_i, a_1^*, \dots, a_i^*, b_1^*, \dots, b_i^*$ to P_{i+1}	
t.			Abort if any input $\notin G_E$. Choose $k_t \in_R Z_q$.
	C	$z_t = k_t^{-1} \text{ mod } q$ $R_t = k_t G \text{ in } \mathcal{G}$	
	S	R_t to P_{t-1}	

	P_1	$P_2 \sim P_{t-1}$	P_t
t+1 to 2t-2.			
	At round t+i, participant P_{t-i} .		
	C	$R_{t-i} = k_{t-i} R_{t-i+1}$ in \mathcal{G}	
	S	R_t, \dots, R_{t-i} to P_{t-i-1}	
2t-1.	C	$R_1 = k_1 R_2$ in \mathcal{G} ZK proof Π_1	
	S	R_1, Π_1 to P_2	
2t to 3t-3			
	At round 2t+i-2, participant P_i .		
	C	ZK proof Π_i	
	S	$R_1, \dots, R_i, \Pi_1, \dots, \Pi_i$ to P_{i+1}	
3t-2			Choose $c \in_R Z_q$
	C		$m = H(M)$. $r = H'(R_1) \in Z_q$. $u^* = E(z_t)$. $u = [(mz_t \bmod q) \times_E a_{t-1}] +_E$ $[(rx_t z_t \bmod q) \times_E b_{t-1}] +_E$ $E(cq)$. ZK proof Π_t
	S		$u, u^*, \Pi_1, \dots, \Pi_t$ to all the other participants.
final	Let $s = D(u) \bmod q$. The participants output (r,s) as the signature for M .		

- ZK proof Π_i which states
 - ✧ $\exists \eta_1, \eta_2 \in [-q^3, q^3]$ such that
 - ✧ $\eta_1 R_i = R_{i+1}$ and $\frac{\eta_2}{\eta_1} G = y_i$
 - ✧ $D(a_i) = \eta_1 D(a_{i-1})$ and $D(b_i) = \eta_2 D(b_{i-1})$
 - ✧ $D(a_i^*) = \eta_1$ and $D(b_i^*) = \eta_2$

4.3.2 Efficiency analysis

Analysis 1. Hardness assumption

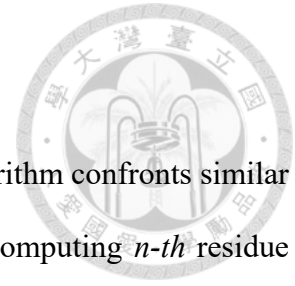
Due to the basis of the Paillier cryptosystem, it is certainly that the algorithm confronts similar difficult assumption as the Paillier cryptosystem, where the problem of computing n -th residue classes is computationally difficult. Currently it is not directly proved that this assumption is more difficult than the assumption of ECDSA. Therefore, in theory, the R Gennaro's algorithm may be likely less secured than ECDSA.

Analysis 2. Number of rounds

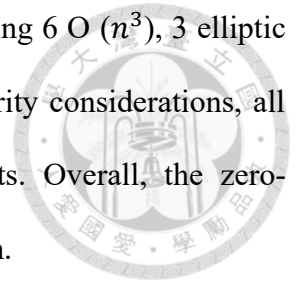
According to the study, the number of rounds required for the R algorithm to perform t -of- n threshold is $3t-2$. However, it is also observed that the message delivered by the participants actually indicates no causal relationship in the 1 to $t-1$ round and t to $2t-2$ round. The reason why such scheme designed in this study is that the zero-knowledge proof will come in handy if the previous $t-1$ round passes the commits for the following message. With the implementations of additional methods and restrictions, perhaps the reduction in the number of rounds can be broken from this place, reducing the number of rounds needed to $2t-1$. In fact, R Gennaro et al. conducted the study of 'Threshold-optimal DSA/ECDSA signatures and an application to Bitcoin wallet security' in 2016⁴⁶. In this new algorithm, only 6 rounds are required.

Analysis 3. Zero-Knowledge proof

Among the uses of R algorithm, there have been many projects implementing zero-knowledge proof as an approach to confirmation. Therefore, the amount of calculations spent on zero-knowledge proof is considerable, and it is proved that a total of 20 values need to be calculated each time. Most of the calculations are $O(n^3)$ level operations or elliptic curve multiplications. In terms of transmission, these 20 values need to be transmitted, and many of the messages are N or



N^2 bits. As for the verifier, one is required to verify 10 equations, containing 6 $O(n^3)$, 3 elliptic curve multiplications, and 1 hash calculation. Moreover, because of security considerations, all participants must verify the zero-knowledge proof of other participants. Overall, the zero-knowledge proof consumes a lot of computing resources in the R algorithm.



Analysis 4. Send messages

In many situations where the R algorithm is being applied, the message just received must be sent to the next participant. Therefore, the total amount of messages to be transmitted in the entire algorithm is considerably large ($23t^2-21t-2$), and most of them are N or N^2 bits. It is worth noting that P_t can practically make a full signature at the end of the algorithm. However, in R Gennaro’s study, P_t was designed to pass u, u^* and all zero-knowledge proofs to all participants. Although each participant has already obtained some information, this work still requires a great amount of traffic ($10t(t-1)$ messages).

Analysis 5. The amount of calculation

Due to the fact that this algorithm is based on the Paillier cryptosystem, there will be a number of operations appear in the process, such as $g^ax^b \bmod N^2$. The complexity of the operation is $O(n^3)$, which is about 4.68RSA*. In addition, a lot of computation tasks required because all the participants must verify the zero-knowledge proof of all the other participants. The calculations required for different participants in a t -of- t thresholds ECDSA are shown as follows:

	P_1	$P_2 \sim P_{t-1}$	P_t	average
RSA	50	66	58	$66-24/t$
(\times)	5	5	5	5
($+$)	1	1	1	1
Hash	0	0	2	1



RSA: Perform a calculation of RSA encryption.

(\times): Perform a multiplication of points on elliptic curve.

($+$): Perform an addition on elliptic curve.

Hash: Perform a hash operation.

* Definition: 1RSA is the amount of computation required to perform an RSA encryption, which is the amount of computation required to execute $g^a \bmod N$.

4.68RSA: Suppose a has as many bits as b . The i -th bit of a is called a_i . Calculating $g^a x^b$ is similar to the square and multiplication of $(gx)^c$. The number of squares is the same, there are three cases when multiplying: $(a_i, b_i) = (1,1), (1,0), (0,1)$. So, the calculation of $g^a x^b \bmod N$ is about 1.17RSA. However, the calculation here is $\bmod N^2$, so the total is 4.68RSA

Analysis 6. t -of- n threshold

In terms of the R algorithm, the method of executing the t -of- n threshold is to perform the t -of- n threshold $\binom{n}{t}$ times. In other words, a desirable way to perform t -of- n threshold was not developed in the study. In addition, as for the t -of- t case, R Gennaro also stated in the study that the proposed algorithm was only suitable for situations where t was not too large. Fortunately, thanks to statistics, the most commonly used multi signature forms in Bitcoin transactions are 2-of-2, 2-of-3 and 3-of-3.

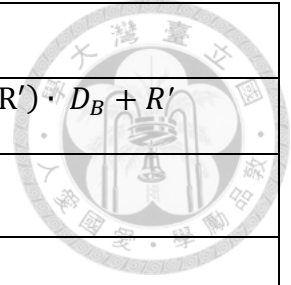
4.4 Threshold ECDSA from ECDSA Assumptions^{14, 15}

The study conducted by Doerner et al in 2018 proposed that 2-of-2 and 2-of- n threshold signatures could be achieved only on the assumption of ECDSA. Soon, in the study published in 2019, it was indicated that t -of- n threshold signature could be achieved when being on the basis of ECDSA assumption. The core technology of these different threshold protocols is the 2-of-2 version. The researchers used the OT technology to perform multiplication and addition to achieve

a combination of different signatures. The 2-of-n version is a setup of Lagrange Interpolation before signing. After the selection of any two participants, the two designated individuals perform the same as the 2-of-2 version. Similarly, the t -of- n version is also based on t -of- t . As for t -of- t , the main concept is to perform the multiplication of the t participants by performing the F_{mul} in the 2-of-2 version repeatedly through some combination. The following focuses on the introduction of the 2-of-2 version of the threshold signature and the t -of- t F_{mul} combination method. (this algorithm is called as D algorithm for convenience)

4.4.1 Doerner's algorithm

		Alice		Bob		
1.	Choose instance key seed $k'_A \in_R Z_q$				Choose instance key $k_B \in_R Z_q$	
		C	$D_B = k_B \cdot G$			
		S	D_B			
2.	C	$R' = k'_A \cdot D_B$ $k_A = H(R') + k'_A$ $R = k_A \cdot D_B$				
	S	R				
3-1.	Choose a pad $\varphi \in_R Z_q$					
F_{mul}	input	$\varphi + 1/k_A$		input	$1/k_B$	
	output	t_A^1		output	t_B^1	
3-2.	input	$1/sk_A$		input	$1/sk_B$	
F_{mul}	output	t_A^2		output	t_B^2	



	Alice		Bob	
4.			C	$R = H(R') \cdot D_B + R'$
	For both Alice and Bob let $(r_x, r_y) = R$.			
5.	Submit	(prove, k_A, D_B)	Submit	(prove, R, D_B)
F_{ZK}^{RDL}			Bob receives a bit indicating whether the proof was sound. If it was not, he aborts.	
6.	C	$m' = H(m)$	C	$m' = H(m)$
7.	C	$\Gamma^1 = G + \varphi \cdot k_A \cdot G - t_A^1 \cdot R$ $\eta^\varphi = H(\Gamma^1) + \varphi$ $sig_A = (m' \cdot t_A^1) + (r_x \cdot t_A^2)$ $\Gamma^2 = (t_A^1 \cdot pk) - (t_A^2 \cdot G)$ $\eta^{sig} = H(\Gamma^2) + sig_A$		
	S	η^φ and η^{sig}		
8.			C	$\Gamma^1 = t_B^1 \cdot R$ $\varphi = \eta^\varphi - H(\Gamma^1)$ $\theta = t_B^1 - \varphi/k_B$ $sig_B = (m' \cdot \theta) + (r_x \cdot t_B^2)$ $\Gamma^2 = (t_B^2 \cdot G) - (\theta \cdot pk)$ $sig = sig_B + \eta^{sig} - H(\Gamma^2)$
Final	Bob uses the public key pk to verify that $\sigma = (sig, r_x)$ is a valid signature on message m. If the verification fails, Bob aborts. If it succeeds, he outputs σ .			

4.4.2 Efficiency analysis

Analysis 1. Difficult assumption

As what addressed in the study, the algorithm (or D algorithm) did not apply additional obscure assumptions. Instead, the use of OT technology was implemented to complete the threshold mechanism. However, it is difficult for the applied OT technology to discrete logarithm problem above the elliptic curve. In other words, using the D algorithm does not reduce the security of the signature.

Analysis 2. Number of rounds

According to the study, the number of rounds required for 2-of-2 and 2-of- n versions is 2. However, in order to compare with the R algorithm, the round calculation of the two should adopt the same definition: If the i round has not been completed, the $i + 1$ round cannot be completed (for all i). In this way, the number of rounds of the D algorithm is 4.

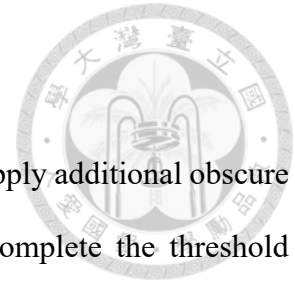
Analysis 3. Zero-Knowledge proof

The zero-knowledge proof used in the D algorithm is based on the Schnorr protocol¹². This zero-knowledge proof, which is a fairly efficient zero-knowledge proof algorithm, was invented by Claus Schnorr in 1991. By incorporating the use Fiat–Shamir heuristic²⁹, Doerner et al made zero-knowledge proof converted to a non-interactive version. One can refer to Chapter 2.4 for the brief introduction.

Using the Schnorr protocol to make the zero-knowledge of the D algorithm saves a lot of computing resources. The prover's calculation consists of 1RSA and 1 hash, while the verifier's calculation is about 2RSA.

Analysis 4. Send messages

The number of messages that the D algorithm needs to transmit is mostly in F_{mul} , meaning the



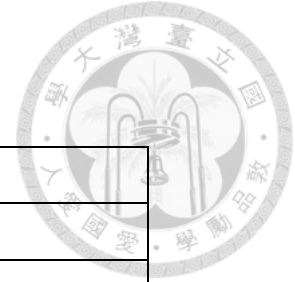
technique of multiplication and addition will be required and the intensive use of OT is also needed. In order to minimize the number of messages that need to be transmitted, the researchers also applied the OT extension²⁶, where the OT that has been process many times to k times (k is the security parameter of OT) can be reduced. In fact, in Gilboa's multiplication addition¹⁰, it is also possible to further reduce the number of messages to be transmitted by changing the 2-bit to 4-bit.

Analysis 5. t -of- n

Strictly speaking, Doerner et al did not propose a new t -of- n threshold scheme in the study. Instead, they did some sorting combinations based on the 2-of-2 version. In terms of the version proposed in the study, t is preferably the power of 2. However, such arrangement is not an arbitrary choice. On the contrary, this method is designed to allow each participant to operate as much as possible. Although the number of times F_{mul} needs to be performed remains unchanged, the number of rounds that need to be executed by t -of- n threshold to $\log_2 n$ can be reduced. The algorithm is symmetrical for each participant. We show what participants 1 needs to do in the process.

Round1.	F_{mul}	1
	With	P_2
Round2.	F_{mul}	2
	With	P_3, P_4
Round3.	F_{mul}	4
	With	P_5, P_6, P_7, P_8
Round $\lceil \log_2 t \rceil$.	F_{mul}	$2^{(\lceil \log_2 t \rceil - 1)}$
	With	$P_{2^{(\lceil \log_2 t \rceil - 1) + 1}}, \dots, P_t$

4.5 Comparisons between the two studies



Rounds (signing):

	R	D
2-of-2	4	4
2-of-n	4	4
t-of-t	3t-2	$5+2\log_2 t$
t-of-n	3t-2	$5+2\log_2 t$

* The t-of-t case and the t-of-n case of these two algorithms are almost the same, the main difference is in the setup part.

Communication (Bits):

	R	D
2-of-2	$343\kappa^{**}$	$\kappa(\kappa^{OT} + 16\kappa + 14s + 10) + 3$
2-of-n	343κ	$\kappa(\kappa^{OT} + 22\kappa + 20s + 11) + 3$
t-of-t	$\kappa(155t^2 - 122t - 33)$	$\frac{t(t-1)}{2}(7\kappa^2 + 12\kappa \cdot s + \kappa \cdot \kappa^{OT} + 28\kappa + 10)$
t-of-n	$\kappa(155t^2 - 122t - 33)$	$\frac{t(t-1)}{2}(9\kappa^2 + 18\kappa \cdot s + \kappa \cdot \kappa^{OT} + 30\kappa + 10)$

* κ, κ^{OT}, s are all security parameters.

** According to the paper¹³, use $N > \kappa^8$ in Paillier cryptosystem

Operation:

	R		D	
2-of-2	(X)	10	(X)	8
	(+)	4	(+)	2
	Hash	2	Hash	$2.5\kappa^{OT} + 22\kappa + 17s + 9$
	RSA	58.97	RSA	2
2-of-n	(X)	10	(X)	8
	(+)	4	(+)	2
	Hash	2	Hash	$2.5\kappa^{OT} + 28\kappa + 23s + 9$
	RSA	58.97	RSA	2

t-of-t	(X)	5t	(X)	5
	(+)	3t-2	(+)	t-1
	Hash	t+1	Hash	$\frac{t(t-1)}{2}(2.5\kappa^{0T} + 11\kappa + 16S + 26)$
	RSA	25.61t+15.75-10/t	RSA	1.5t
t-of-n	(X)	5t	(X)	5
	(+)	3t-2	(+)	t-1
	Hash	t+1	Hash	$\frac{t(t-1)}{2}(2.5\kappa^{0T} + 15\kappa + 21S + 26)$
	RSA	25.61t+15.75-10/t	RSA	1.5t

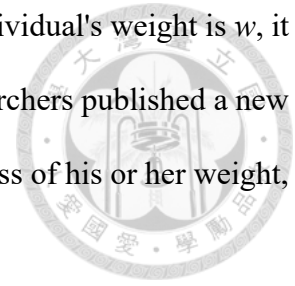
4.6 Other forms of threshold secret sharing

Although each algorithm discussed earlier shows its own advantages, cryptography can't be confined to la tour d'ivoire, after all. There are always a variety of challenging conditions in the real world. A more targeted algorithm is needed in order to solve the problems. A number of different threshold secret sharing will be introduced in the following section.

4.6.1 Weighted threshold⁴⁷⁻⁵¹

When using multi signature or threshold signature, there may be situations where each individual possesses different levels of authority in the same group. For example, the signature of an account is 4-of-10 threshold signature is implemented in a company, and the company consists of ten members, including two supervisors and eight employees. The supervisors think that the two of them can use the account as long as they reach a consensus, and permission from the other staff members required. In this case, it is possible to assign 2 weights to both supervisors, and the weights of other staff members is considered as 1. It is something intuitive to adopt such approach. When using the threshold signature scheme mentioned earlier, people with more weights will be allocate with more secret shares. Dikshit and Singh⁴⁷ also performed their studies in a similar manner, however, they later realized there was a fatal flaw embedding in this algorithm. For

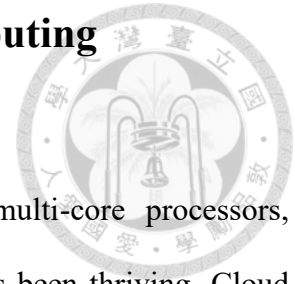
instance, in this scenario, everyone needs to manage a lot of keys. If an individual's weight is w , it is likely for one to reach all the keys. In order to cope with this issue, researchers published a new study in 2017⁴⁸. In the study, it was proposed that each individual, regardless of his or her weight, would only get the access to a single key.



4.6.2 Hierarchical threshold

Let's discuss about another scenario. In a school, according to the school regulations, it is stipulated that a group of fifteen people per application must be met, and at least one teacher must be included in each group. There may be people commenting that an adjustment of right distribution may solve this problem. However, such approach does not necessarily be considered as a solution. If the number of students is extremely large, one can simply use a large number of students to replace the teacher's weight, regardless of the weight of the students. In addition, if the problem gets more complicated, for instance, where at least two directors, four teachers, and seven students should be included among the fifteen people, some additional schemes may be required. Tamir Tassa et al proposed an approach of Hierarchical Threshold Secret Sharing to solve such problems⁵². The Birkhoff interpolation was applied in the study.

Chapter 5. Threshold Cryptography on Cloud Computing



5.1 Introduction of cloud computing

With the advancement of technology, such as grid computing, multi-core processors, virtualization, the Internet, etc., the development of cloud computing has been thriving. Cloud computing can save a lot of redundant resources for enterprises. In fact, many companies spend a lot of money on building the IT infrastructure, but the average usage rate is less than 20%. In other words, nearly 80% of the resources are wasted. Imagining a scenario where space and services rental are available based on one's needs. Tasks such the equipment room maintenance, network management and software upgrades can simply be assigned to a dedicated individual. Such scheme can be much more time-efficient and cost-effective. After all, people don't need to raise a cow by themselves to drink milk. Similarly, cloud computing is gradually sprouting under such thoughts. The vision of cloud computing is to make information services, such as public services (including water and electricity), available at any time. This example also carries one of the dominant properties of cloud computing: instant flexibility. That is to say, turn on the faucet when one is in need, and vice versa.

5.2 The NIST definition of cloud computing

In November 2010, the National Institute of Standards and Technology (NIST) launched the cloud computing technology projects. It is mainly to assist the federal government to adopt cloud computing instead, and to strengthen the traditional information system and the new cloud application model. Meanwhile, the US federal government also supports and assists the NIST to develop relevant standards, as well as the deployment of cloud security architecture and cloud computing. A series of documents and reports were also drafted during the course of projects, including: (1) NIST SP800-145 Cloud Computing Definition¹⁶, (2) NIST SP800-146 Cloud

Computing Recommendations⁵³, and (3) NIST SP500-292 Cloud Computing System Architecture⁵⁴. A wide range aspects of cloud security were focused in the projects, including cloud computing security barriers, cloud computing security measures list, etc. The NIST guidelines and recommendations can be considered as the essential guidelines for the development of cloud security in the future.

Specifically, NIST further explains its definition by outlining the five essential characteristics, three service models, and four deployment models. The five essential characteristics include on-demand self-service, broad network access, resource pooling, rapid elasticity, and measured service. The three services models discussed in the reports are Software as a Service (SaaS), Platform as a Service (PaaS), and Infrastructure as a Service (IaaS), while the four deployment models consist of private cloud, community cloud, public cloud, and hybrid cloud.

5.3 Threshold Cryptography on Cloud Computing

Although cloud computing can help companies save a broad range of resources, including hardware (such as equipment and maintenance costs), storage space and computing, security is a critical concern when it comes to the application of cloud computing. The security issues may be discussed from several aspects. First, the security of the server supplier is the primary and crucial concern. It is questionable whether the data can be restored when the server entity storing the data is damaged. In addition, it is greatly concerning whether the stored data is prone to be leaked or destroyed. Moreover, the highly concentrated information will bring convenience to the hackers (the eggs should not be placed in the same basket).

Take some real-world cases as examples. Business.com surveyed through LinkedIn, 75% of the 65 respondents believe that "security" is the first consideration of cloud computing. In addition, Gartner surveyed 169 data center managers, and 85% thought that the 'securit' of cloud computing

might be the reason for them to retreat. Similarly, according to the CNN's article, 'How safe is cloud computing?', hackers can capture a lot of customer information in clouds. In a nutshell, when one puts more eggs in a single basket, the prize is much bigger for the others.

Some studies have tried to use the threshold cryptography to solve security problems. Those researchers anticipate to be able provide more security to the key used by the secret share schemes.

5.3.1 Threshold cryptography based on data security in cloud computing⁵⁵

Acronyms and definition

- a. DO (Data Owner): the data owner will upload the data to the CSP, so that DO can save the cost of storing and managing the data.
- b. CSP (Cloud Service Provider): servers that provide cloud services, such as Amazon's Simple Storage Service (S3) and Elastic Block Store (EBS)

Procedure

Sushil Kr Saroj, Sanjeev Kr Chauhan, Aravendra Kr Sharma, Sundaram Vats, based on the NDAC model, coupled with the technique of threshold cryptography, enabled themselves not only provide the strong data confidentiality but also reduces the number of keys. And because of threshold cryptography, one can avoid collusion attack of malicious users and cloud service provider and heavy computation (due to large number of keys).

In short, as the illustration shown in **Figure 3**, User requests registration from DO. After DO confirms the identity of User, DO will divide users in groups, and DO will provide encryption keys, tokens, algorithm (MD5) and other necessary things for secure communication to user groups in response of registration. Then DO will transmit the updated Capability List and the file encrypted with k_T (the symmetric key held by DO and User) to the CSP (this can avoid the CSP from knowing). When User needs the data, he or she will apply to the CSP. After the CSP confirms the

identity of the User based on the Capability List, it will perform a modified Diffie-Hellman exchange with the User. People call it modified DH algorithm as they encrypt the DH parameters using the public key of one side while using nonce in each direction during session key (k_s) generation and data transfer. This helps to resist any negative attacks during the process. After User receives the file, one will complete the desire decrypt with other users in the group (in addition to the file is encrypted, there is actually the hash value of the file so User can verify the integrity of the file).

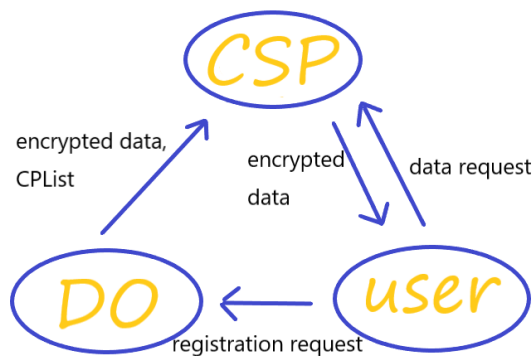


Figure 3

In this study, the authors divided all the steps into four algorithms, where the part related to the threshold is in algorithm 4. Algorithm 4, the algorithm for decryption of a file for User 1, is similar to threshold encryption technology, describing the procedure how a file is decrypting for User 1 after acquiring the encrypted message. Because one can't decrypt key alone, he or she first updates the PKS Vector (initially, all its bits are zero) with the key component, and then sends the PKS Vector and the encrypted message to the next user in the same group. Next user decrypts the message and updates the PKS Vector with his key component. This process continues until all bits of PKS Vector are one. It is clear to see that the application is not used by all key components (threshold for critical components only). Afterwards, the data is sent back to the originator user, who then decrypts the message and gets it. Initially, User 1 will not decrypt the message (M).

Instead, User 1 will only update the PKS Vector.

5.3.2 A secured key for cloud computing using threshold cryptography in Kerberos⁵⁶



Acronyms and definition

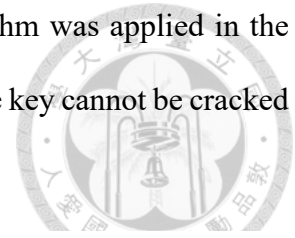
- a. Kerberos: is a network authentication protocol. It is designed to provide strong authentication for client/server applications by using Symmetric-key algorithm. A free implementation of this protocol is available from the Massachusetts Institute of Technology.
- b. Authentication Sever (AS): is responsible for providing Ticket Granting Ticket (TGT) to the client. After AS confirms the identity of the client, AS encrypts TGT with the client's private key and sends it to the user.
- c. Ticket Granting Sever (TGS): is responsible for providing Service Ticket (ST) to client. Client will hand over the TGT that he got from the AS to the TGS. After the TGT is verified, the ST will be handed over to client.

Procedure

This paper proposed a cloud computing authentication model based on Kerberos protocol, which uses threshold cryptography technology to provide higher security and improve key availability. This model can also benefit by filtering unauthenticated access and reducing the computational and storage resources that cloud providers spend on each customer's identity authentication. It acts as a third party between the client and the cloud server to allow authorized and secure access to the cloud service. In Kerberos authentication, the main issue is how to prove an individual's true identity. For example, when a client accesses a service on a server, how does the server determine if the client has access to the service on its own host.

Due to the fact that all the authentication processes are controlled by a centralized key distribution center, once the system is compromised, it is easy for an attacker to gain information

about the user's key. To avoid such situation, threshold encryption algorithm was applied in the study to divide the ticket into n shares and delivers it to the server, so that the key cannot be cracked until the number of votes for the threshold value is obtained.



Traditional Kerberos:

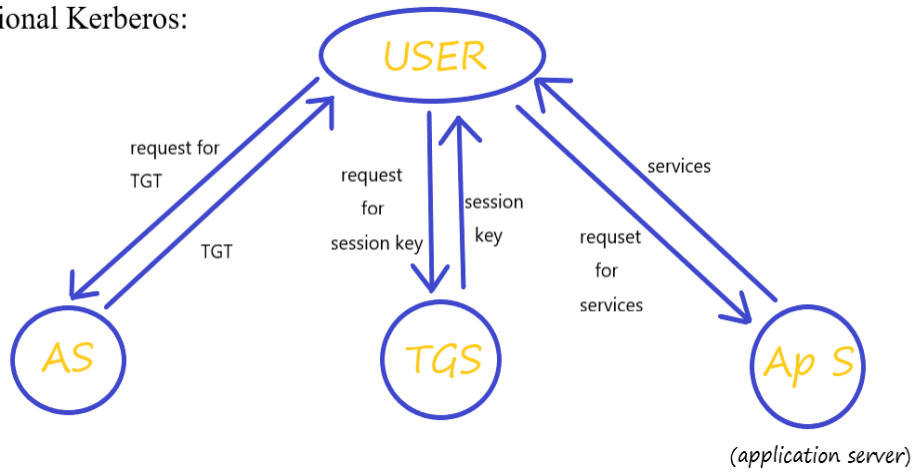


Figure 4

Kerberos with threshold:

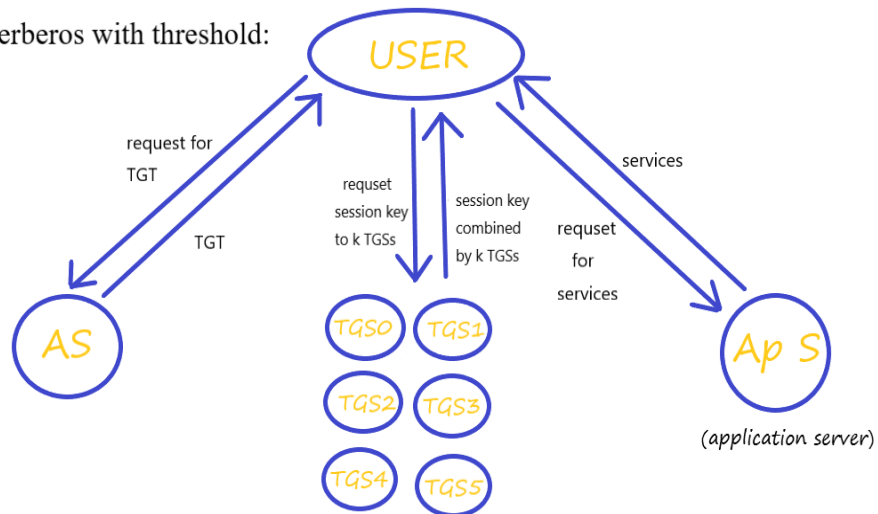


Figure 5

5.3.3 Searching for the optimal value for threshold on cloud computing⁵⁷

In the scope of cloud computing, threshold cryptography can help to increase security by protecting the key. Dividing the key into n copies can reduce the risk of the key being lost cracked by the attacker. Moreover, n people (every combination will only use k , $k < n$) use the assigned key component to combine only one key, which greatly reduces the burden of keys storage. However, when it comes to the research on threshold cryptography, there is barely discussion related to the question of 'If n is fixed, how much k should be selected'. Therefore, Weena Janratchakool, Sirapat Boonkrong and Sucha Smachat used CloudSim to simulate cloud environment and collect time consumed in key distribution and key reconstruction process to achieve the optimal threshold value. The key size used in the experiment will be only 2048-bit private keys according to the NIST recommendation.

First, $n = 256$, $n = 224$, and $n = 196$ were fixed respectively. When the number of threshold (that is, k) is gradually increased, the Key Distribution Time will increase. According to experiment conducted in the study, however, the results of Key Distribution Time showed little discrepancies, which were linearly increased and the gaps among the three results were small. Therefore, only the Average of Key Distribution Time was considered in the following analysis in the study.

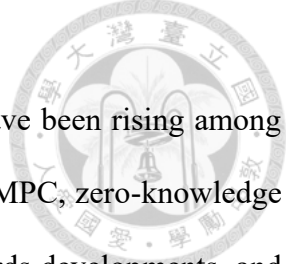
Next, concerning what happens to the key reconstruction time when k is gradually increased under different n , the curves obtained from the two experiments were considered as intersections, and the researchers indicated that the part of the intersection was the best k . According to the experimental results, the best k was nearly 31% of n .

Although the authors were not convinced by the route of how to get the best k , the research on searching for the best k is expected in the future since it is increasing promising to protect sensitive data in terms of cloud computing (especially the private key stored via cloud computing).

Concerning security issues, using threshold cryptography to improve the security of private keys is one of the convincing approaches adopted by many people. Therefore, the importance of searching for the best k is correspondingly increasing.



Chapter 6. Conclusions

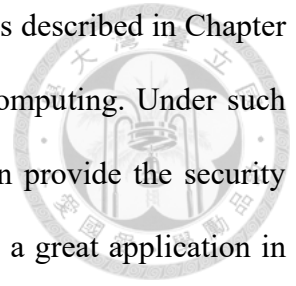


Ever since the emergence of the blockchain, the concerns to privacy have been rising among the public. Factors, such as the advancements of cryptography (including MPC, zero-knowledge proof, and homomorphic encryption, etc.), NIST's proactive attitude towards developments, and the public's anticipation for avoiding single point of failures in the system, stimulate an inevitable trend in the threshold cryptography development. Early in 1979, cryptographers have successively published pertinent studies since Adi Shamir proposed the concept of 'How to share a secret.' Thanks to the advanced technology, the development of threshold cryptography has entered its golden era. People are convinced that threshold cryptography will play a role worldwide in the near future.

In a short period of time, due to the impact of virtual currency, the development of ECDSA threshold is in a compelling need. In this study, two major types of ECDSA thresholds were discussed, and particularly the Rosario Gennaro the algorithm optimized in 2016. There have been teams, who devote to developing smart contracts, adopting the optimized algorithm and apply it in a real-world setting. However, according to the analyses of the two algorithms discussed earlier, Gennaro's algorithm yields an additional hardness assumption compared to ECDSA. In additional, Paillier cryptosystem will consume significant computing resources when performing encryption, decryption, and homomorphic addition. Concerning the efficiency and security, therefore, we concluded that Doerner's algorithm may be considered as a more optimal ECDSA threshold than the others.

As what were discussed earlier, the evolutionary conditions of threshold cryptography act as a propeller in the development of cloud computing. While NIST has been showing a proactive attitude towards the developments, a number of multinational enterprises are also targeting on the

cloud computing markets. According to the comments made by third parties described in Chapter 5, the majority are still hesitant when it comes to the security of cloud computing. Under such circumstances, the use of threshold cryptography plays a role since it can provide the security required in cloud computing. Kerbero, proposed by MIT's research lab, is a great application in cloud computing. On top of that, it is promising that threshold cryptography will be able to provide a more optimal and desirable security in the near future. We are also expecting to see more combined applications of threshold encryption and cloud computing, together creating synergistic benefits in the years ahead.



References

1. Shamir, A.: How to share a secret. *Communications of the ACM* 22, 612-613 (1979)
2. Desmedt, Y.: Threshold Cryptography. *European Transactions on Telecommunications* 5, 307-315 (1994)
3. Feldman, P.: A Practical Scheme for Non-interactive Verifiable Secret Sharing. 28th Annual Symposium on Foundations of Computer Science (1987)
4. Benny Chor, S.G., Silvio Micali, Baruch Awerbuch: Verifiable secret sharing and achieving simultaneity in the presence of faults. *SFCS '85 Proceedings of the 26th Annual Symposium on Foundations of Computer Science* 383-395 (1985)
5. Alfredo De Santis, Y.D., Yair Frankel, Moti Yung: How to share a function securely. *STOC '94 Proceedings of the twenty-sixth annual ACM symposium on Theory of Computing* 522-533 (1994)
6. Yvo Desmedt, Y.F.: Shared generation of authenticators and signatures. *Annual International Cryptology Conference* 457-469 (1991)
7. Pedersen, T.P.: A Threshold Cryptosystem without a Trusted Party. *Workshop on the Theory and Application of Cryptographic Techniques* 522-526 (1991)
8. Brandão, L.T.A.N., Mouha, N., Vassilev, A.: Threshold Schemes for Cryptographic Primitives: Challenges and Opportunities in Standardization and Validation of Threshold Cryptography. *NIST Internal Report (NISTIR) 8214* (2019)
9. Yao, A.C.: Protocols for secure computations. *SFCS '82 Proceedings of the 23rd Annual Symposium on Foundations of Computer Science* 160-164 (1982)
10. Gilboa, N.: Two Party RSA Key Generation. *Annual International Cryptology Conference* 116-129 (1999)
11. Paillier, P.: Public-Key Cryptosystems Based on Composite Degree Residuosity Classes.



International Conference on the Theory and Applications of Cryptographic Techniques 223-238 (1999)



12. Schnorr, C.P.: Efficient Identification and Signatures for Smart Cards. Conference on the Theory and Application of Cryptology 239-252 (1989)

13. Steven Goldfeder, A.H.N., Rosario Gennaro, Harry Kalodner, Joseph Bonneau, Joshua A. Kroll, Edward W. Felten: Securing Bitcoin wallets via a new DSA/ECDSA threshold signature scheme. (2015)

14. Doerner, J., Kondi, Y., Lee, E., Shelat, A.: Secure Two-party Threshold ECDSA from ECDSA Assumptions. 2018 IEEE Symposium on Security and Privacy (SP), pp. 980-997 (2018)

15. Jack Doerner, Y.K., Eysa Lee, Abhi Shelat: Threshold ECDSA from ECDSA Assumptions: The Multiparty Case. IACR Cryptology ePrint Archive 2019 (2019)

16. Peter Mell, T.G.: The NIST Definition of Cloud Computing. NIST Special Publication 800-145 (2011)

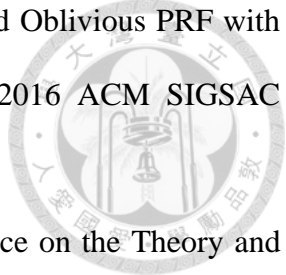
17. Rabin, M.O.: How to Exchange Secrets with Oblivious Transfer. Harvard University Technical Report 81 (1981)

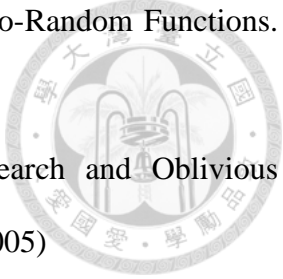
18. Lempel, S.E.G.: A Randomized Protocol for Signing Contracts. Advances in Cryptology 205-210 (1983)


19. Pinkas, M.N.a.B.: Oblivious Polynomial Evaluation. SIAM J. Comput. 35, 1254-1281 (2006)

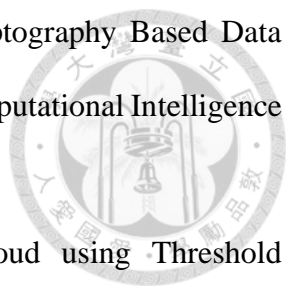
20. Bill Aiello, Y.I., and Omer Reingold: Priced Oblivious Transfer: How to Sell Digital Goods. International Conference on the Theory and Applications of Cryptographic Techniques 119-135 (2001)

21. Sven Laur, H.L.: A New Protocol for Conditional Disclosure of Secrets And Its Applications. International Conference on Applied Cryptography and Network Security 207-225 (2007)

- 
22. Vladimir Kolesnikov, R.K., Mike Rosulek, Ni Trieu: Efficient Batched Oblivious PRF with Applications to Private Set Intersection. CCS '16 Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security 818-829 (2016)
23. CREPEAU, G.B.a.C.: All-or Nothing Disclosure of Secrets. Conference on the Theory and Application of Cryptographic Techniques 234-238 (1986)
24. Y. Ishai, E.K.: Private simultaneous messages protocols with applications. Proceedings of the Fifth Israeli Symposium on Theory of Computing and Systems (1997)
25. Hui-Feng Huang, C.C.C.: A New t-out-n Oblivious Transfer with Low Bandwidth. Applied Mathematical Sciences 1, 311-320 (2007)
26. Yuval Ishai, J.K., Kobbi Nissim, and Erez Petrank: Extending Oblivious Transfers Efficiently. Annual International Cryptology Conference 145-161 (2003)
27. Tung Chou, C.O.: The Simplest Protocol for Oblivious Transfer. International Conference on Cryptology and Information Security in Latin America 40-58 (2015)
28. W. Diffie, M.H.: New Directions in Cryptography. IEEE Transactions on Information Theory 22, 644-654 (1976)
29. Amos Fiat, A.S.: How To Prove Yourself: Practical Solutions to Identification and Signature Problems. Conference on the Theory and Application of Cryptographic Techniques 186-194 (1986)
30. Technology, N.I.o.S.a.: Digital Signature Standard (DSS). FIPS PUB 186-4 (2013)
31. R.L. Rivest, A.S., and L. Adleman: A Method for Obtaining Digital Signatures and Public-Key Cryptosystems. Communications of the ACM 21, 120-126 (1978)
32. Oded Goldreich, S.G., Silvio Micali, Kazuo Ohta, Leonid Reyzin: How to construct random functions. Journal of the ACM 33, 792-807 (1986)

- 
33. Moni Naor, O.R.: Number-Theoretic Constructions of Efficient Pseudo-Random Functions. *Journal of the ACM* 51, 231-262 (2004)
34. Michael J. Freedman, Y.I., Pinkas, Omer Reingold: Keyword Search and Oblivious Pseudorandom Functions. *Theory of Cryptography Conference* 303-324 (2005)
35. Stanisław Jarecki, X.L.: Efficient Oblivious Pseudorandom Function with Applications to Adaptive OT and Secure Computation of Set Intersection. *Theory of Cryptography Conference* 577-594 (2009)
36. Stanislaw Jarecki, H.K., Jason Resch: Threshold Partially-Oblivious PRFs with Applications to Key Management. *Cryptology ePrint Archive: Report 2018/733* (2018)
37. Ali Bagherzandi, S.J., Yanbin Lu, Nitesh Saxena: Password-Protected Secret Sharing. *Bibliometrics* 433-444 (2011)
38. Michel Abdalla, M.C., Anca Nitulescu, and David Pointcheval: Robust Password-Protected Secret Sharing. *ESORICS 2016* 61-79 (2016)
39. Stanisl: TOPPSS: Cost-minimal Password-Protected Secret Sharing based on Threshold OPRF. *International Conference on Applied Cryptography and Network Security* 39-58 (2017)
40. Silvio Micali, K.O., Leonid Reyzin: Accountable-Subgroup Multisignatures. *CCS '01 Proceedings of the 8th ACM conference on Computer and Communications Security* 245-254 (2001)
41. Raju GANGISHETTI, M.C.G., Manik Lal DAS, Ashutosh SAXENA: Identity Based Multisignatures. *INFORMATICA* 17, 177-186 (2006)
42. Harn, L.: Group-oriented (t,n) threshold digital signature scheme and digital multisignature. *IEE Proceedings - Computers and Digital Techniques* 141, 307-313 (1994)
43. Choonsik Park, K.K.: New ElGamal Type Threshold Digital Signature Scheme. (1996)

- 
44. Chuan-Ming Li, T.H., Narn-Yih Lee: Remark on the Threshold RSA Signature Scheme. CRYPTO '93 Proceedings of the 13th Annual International Cryptology Conference on Advances in Cryptology 413-420 (1993)
45. Shoup, V.: Practical Threshold Signatures. EUROCRYPT'00 Proceedings of the 19th international conference on Theory and application of cryptographic techniques 207-220 (2000)
46. Rosario Gennaro, S.G., Arvind Narayanan: Threshold-optimal DSA/ECDSA signatures and an application to Bitcoin wallet security. International Conference on Applied Cryptography and Network Security 156-174 (2016)
47. Dikshit, P., Singh, K.: Weighted threshold ECDSA for securing bitcoin wallet. ACCENTS Transactions on Information Security 2, 43-51 (2016)
48. Pratyush Dikshit, K.S.: Efficient Weighted Threshold ECDSA for Securing Bitcoin Wallet. 2017 ISEA Asia Security and Privacy (ISEASP) (2017)
49. P.Morillo, C.P., G.Sáez, J.L.Villar: Weighted threshold secret sharing schemes. Information Processing letters 70, 211-216 (1999)
50. Drăgan, C.C., Țiplea, F.L.: Distributive weighted threshold secret sharing schemes. Information Sciences 339, 85-97 (2016)
51. Beimel, A., Tassa, T., Weinreb, E.: Characterizing Ideal Weighted Threshold Secret Sharing. SIAM Journal on Discrete Mathematics 22, 360-397 (2008)
52. Tassa, T.: Hierarchical Threshold Secret Sharing. Journal of Cryptology 20, 237-264 (2007)
53. Mark L. Badger, T.G., Robert Patt-Corner, Jeffrey M. Voas: Cloud Computing Synopsis and Recommendations. NIST Special Publication 800-146 (2012)
54. Fang Liu, J.T., Jian Mao, Robert B. Bohn, John V. Messina, Mark L. Badger, Dawn M. Leaf: NIST Cloud Computing Reference Architecture. Special Publication (NIST SP) - 500-292 (2011)

- 
55. Saroj, S.K., Chauhan, S.K., Sharma, A.K., Vats, S.: Threshold Cryptography Based Data Security in Cloud Computing. 2015 IEEE International Conference on Computational Intelligence & Communication Technology, pp. 202-207 (2015)
56. Shubha Bharill, T.H., Praveen Lalwani: A Secure Key for Cloud using Threshold Cryptography in Kerberos. International Journal of Computer Applications 79, 35-41 (2013)
57. Janratchakool, W., Boonkrong, S., Smanchat, S.: Finding the Optimal Value for Threshold Cryptography on Cloud Computing. International Journal of Electrical and Computer Engineering (IJECE) 6, (2016)