

Adjusting Three Schemes of Anonymous User Identification and Key Distribution

Peter Adamson Ndem

Submitted to the
Institute of Graduate Studies and Research
in partial fulfillment of the requirements for the Degree of

Master of Science
in
Computer Engineering

Eastern Mediterranean University
February 2015
Gazimağusa, North Cyprus

Approval of the Institute of Graduate Studies and Research

Prof. Dr. Serhan Çiftçiođlu
Acting Director

I certify that this thesis satisfies the requirements as a thesis for the degree of Master of Science in Computer Engineering.

Prof. Dr. Işık Aybay
Chair, Department of Computer Engineering

We certify that we have read this thesis and that in our opinion it is fully adequate in scope and quality as a thesis for the degree of Master of Science in Computer Engineering.

Assoc. Prof. Dr. Alexander Chefranov
Supervisor

Examining Committee

1. Assoc. Prof. Dr. Alexander Chefranov

2. Asst. Prof. Dr. Gürcü Öz

3. Asst. Prof. Dr. Önsen Toygar

ABSTRACT

User identification, user anonymity, mutual authentication and key distribution are desirable features for securing communication in a distributed network. Yang et al, Mangipudi-Katti and Hsu-Chuang in a new efficient user identification and key distribution scheme providing enhanced security, a secure identification and key agreement protocol with user anonymity (SIKA) and a novel user identification scheme with key distribution preserving user anonymity respectively were schemes proposed for these security features. They all have a problem related with not existence of inverses of identifiers used in them. Identifiers and signatures were selected and not all the identifiers have inverse identifiers modulus N . The inverse of identifiers selected for use will not exist in all cases as the probability of selecting of non-invertible identifiers modulus N goes to one leading to the failure of the schemes that is demonstrated in this present research.

In this thesis, we propose a way to improve the problem by coming up with a two theorems. The first theorem is a procedure which will ensure selection of identifiers that will always have inverse identifiers under condition of usage of RSA – likekey settings which are kept secret (prime factorization of $N = p * q$). This procedure has one input $N = p * q$ which is the public parameter received from a Smart Card producing Center, a trusted third party, where $p, q > 2$ are unknown primes, and one output identifier which is any integer number between 2 and $N-1$, invertible modulo N . We also show that the output identifier will always have invertible mod N and invertible signatures mod N . The second theorem guarantees that the output identifier selected in the first theorem will also have invertible identifier modulo N

and its encrypted value will also be invertible modulo N . The proposed improvement ensures the schemes preserve their features and will be efficient with no computational complexities and increased communication cost.

Keywords: Identifier, inverse, authentication, prime number, factorization

ÖZ

Kullanıcı kimliğinin belirlenmesi, kullanıcı anonimliği, ortak kimlik doğrulama ve anahtar dağıtımı, dağıtık bir ağdaki iletişimin güvenli olması için aranan özelliklerdir. Güvenlik özelliklerine ilişkin olarak geliştirilmiş güvenlik sağlayan yeni ve etkili bir kullanıcı kimliği belirleme ve anahtar dağıtımı şemasında, Yang ve diğerleri, Mangipudi-Katti ve Hsu-Chuang tarafından kullanıcı anonimliğine sahip olan güvenli bir kimlik belirleme ve anahtar anlaşması protokolü şemasının yanı sıra, kullanıcı anonimliğini koruyan ve anahtar dağıtım özelliğine sahip olan yeni bir kullanıcı kimliği belirleme şeması önerilmiştir. Bu şemalarda tanıtıcıya ait terslerin mevcut olmaması bir sorun oluşturmaktadır. Tanıtıcılar ve imzalar seçilmiş olsa da tanıtıcıların tümü modulus N tanıtıcı terslerine sahip değildir. Kullanılmak üzere belirlenen tanıtıcı terslerinin seçilmesinde ters çevrilemeyen modulus N tanıtıcılarının seçilmiş olma olasılığı, bu çalışmada incelenen şemalarda başarısız sonuçlara neden olmaktadır.

Bu tezde, bu sorunun, geliştirdiğimiz iki teori ile çözülmesi amaçlanmaktadır. İlk teori, gizli RSA benzeri anahtar ayarlarının kullanılması koşuluyla seçim sırasında belirlenen tanıtıcıların her zaman ters tanıtıcılara sahip olmasını sağlayan bir prosedürdür ($N = p * q$ değerinin asal çarpanlara ayrılması). Bu prosedür, güvenilir bir üçüncü taraf olan Smart kart üretici Merkezinden alınan ortak bir parametre olan $N = p * q$ girdisine sahiptir. $p, q > 2$ bilinmeyen asal sayıların olduğu bu değerde, tam sayı 2 ve ters çevrilebilir modulo N değeri arasındaki bir sayıdır. Ek olarak, çıktı tanıtıcısında her zaman ters çevrilebilir mod N değerinin ve ters çevrilebilir mod N imzalarının bulunacağı gösterilmektedir. İkinci teori ise birinci teoride seçilen çıktı

tanıtıcısının bir modulo N ters çevrilebilir tanıtıcısına sahip olacağını ve şifrelenen değerinin de ters çevrilebilir modulo N olacağını garanti eder. Önerilen geliştirmede, şemaların, özelliklerini bilgi işlem karmaşıklıkları ile karşılaşılmadan iyileştirilmiş iletişim maliyetleri ile korunması sağlanır.

Anahtar Sözcükler: Tanıtıcı, ters, kimlik doğrulama, asal sayı, faktörizasyon

This research is dedicated first of all to God Almighty and
my savior Jesus Christ.

To Engr. And Mrs. Adamson Ndem and family,

my friends

and all who have always been there.

ACKNOWLEDGMENT

I will love to appreciate all those who gave me their support to complete my study.

A special thanks to my thesis supervisor, Assoc. Prof. Dr. Alexander Chefranov, whose valuable information, guidance and support, helped me to coordinate this study.

I also acknowledge with much appreciation the crucial role of the staffs in Computer Engineering Department, for their provision of the necessary materials to complete this study.

Lastly, I thank the almighty God, my parent, brother, sisters and friends for their constant encouragement, love, care and prayers without which this study would not be possible.

I love you all.

TABLE OF CONTENTS

ABSTRACT.....	iii
ÖZ	v
DEDICATION	vii
ACKNOWLEDGMENT.....	viii
LIST OF TABLES	xii
LIST OF FIGURES	xiii
LIST OF ABBREVIATIONS	xiv
1 INTRODUCTION	1
1.1 Background of the study.....	1
1.1.1 What is security?.....	1
1.2 Purpose of the thesis	7
1.3 Outline for this thesis	9
2 LITERATURE REVIEW	10
2.1 Concepts of securing a networks.....	10
2.1.1 Pretty Good Privacy (PGP).....	11
2.1.2 Kerberos protocol	11
2.2 Public key cryptography.....	13
2.2.1 RSA Algorithm	14
2.2.1.1 RSA Key generation	14
2.2.1.2 RSA Encryption	14
2.2.1.3 RSA Decryption.....	15
2.2.2 Diffie and Hellman key exchange	15
2.3 Identity-inclined cryptography	16

2.3 Single sign-on.....	16
3 REVIEW OF YANG ET AL, MANGIPUDI - KATTI AND HSU – CHUANG SCHEMES	18
3.1 Overview of the schemes	18
3.2 System initialization stage.....	18
3.3 Registration stage	19
3.4 User identification in Yang et al, Mangipudi - Katti and Hsu - Chuang schemes	19
3.4.1 User identification in the Yang at al. scheme	19
3.4.2 User identification in the Mangipudi–Katti scheme	20
3.4.3 User identification in the Hsu–Chuang scheme.....	22
4 EXAMPLES OF FAILURES OF YANG ET AL, MANGIPUDI - KATTI AND HSU - CHUANG SCHEMES.....	25
4.1 Introduction	25
4.2 Counter-example for Yang et al, Mangipudi - Katti and Hsu - Chuang Schemes.....	25
4.2.1 Example of failure for the Yang et al. and Mangipudi–Katti schemes....	25
4.2.2 Example of failure for the Hsu-Chuang scheme.....	26
5 IMPROVING YANG ET AL, MANGIPUDI - KATTI AND HSU - CHUANG SCHEMES	28
5.1 Finding invertible identifiers from Z_N^*	28
5.2 Theorem 1.....	31
5.3 Theorem 2.....	33
6 CONCLUSION.....	35
REFERENCES	36

APPENDICES	39
Appendix A: Java Code for generating invertible numbers	40
Appendix B: Output of the code.....	41

LIST OF TABLES

Table 1. Statistics of invertible numbers	29
---	----

LIST OF FIGURES

Figure 1. User identification in Yang et al. scheme	19
Figure 2. User identification in Mangipudi–Kattis scheme	21
Figure 3. User identification in Hsu–Chuang scheme	23
Figure 4. The probability of selecting of non-invertible identifiers.....	30
Figure 5. A graph showing growth of invertible identifiers	30

LIST OF ABBREVIATIONS

SCPC	Smart card producing center is a trusted authority
U_j	The user
P_j	The service provider
ID_i	The identity of the entity i
S_i	The secret token of the entity i
e_i	Public key for encryption by identity i
d_i	Private key for encryption by identity i
$E_k(M)$	Encryption of a plaintext M using a key k
$D_k(C)$	Decryption of encrypted text C using a key k

Chapter 1

INTRODUCTION

1.1 Background of the study

1.1.1 What is security?

Security is the ability to be protected against being in danger or loss [1]. It is synonymous with safety and is to be secured. Security is being totally free from danger and loss. Data security, information security, application security, computer security and network security are the various areas that need to be protected against danger and loss as data, information, application, computer and network stores things that are valuable and important. A secured system should provide a control system for the user and should deploy security audit system. Data should be sent by an authorized user in an authorized format to an authorized recipient. The trinity of data security is confidentiality, integrity and availability.

- Data confidentiality

This refers to the access to information by authorized users. If unauthorized access to information occurs, it is said that the information is no longer confidential. This requires that data can only be seen by authorized users only.

- Data integrity

This refers to the completeness and accuracy of information stored. If a name is missing an alphabet, it is said that the name is incorrect and is incomplete and inaccurate. This requires that data can only be modified or updated by authorized users.

- Data availability

This refers to the ease by which data can be accessed. If a data is corrupted due to storage failure, the data has availability issues. This requires that data can only be made available to authorized users. An organization's security strength is based on maturity, effectiveness and completeness on security controls implemented. These measures are implemented in one or more layers from the physical infrastructure (buildings, facility), network infrastructure (network security), IT systems (system security) and applications and information (application security). To ensure security, certain steps need to be taken such as;

1. Data Encryption

Data encryption refers to encoding information in such a way that only the person with the key can decode it. Traditional mechanisms can to some extent protect data privacy but is not sufficient. Encryption depends on the reliability of the strength of the decryption. There are three basic encryption methods: hashing, symmetric and asymmetric cryptography.

2. Hashing

Is a technique that creates a unique, fixed-length signature for a dataset. A hash is unique to a specific message. Therefore, any minor change results to an entirely different hash that alerts the user to potential tampering. In hashing, once the data is encrypted, the operation cannot be reversed or deciphered. This means that even if the hash is obtained by an attacker, it is impossible to use a decryption method to retrieve the original contents of that information.

3. Symmetric cryptography

Also called private key is requires that the key used to encrypt and decrypt data must remain secure because anyone who has the key can decode the information. This

encryption method has two ways of application. Either as a stream cipher which encrypts one character at a time or a block cipher which encrypts a fixed-length of data chunks.

4. Asymmetric cryptography

Asymmetric cryptography or public key is likely more secure than symmetric cryptography. It uses two types of cryptography; public key and private key. The public key is available to everyone and is used to encrypt information before sending. The private key remains with the receiver of the message who uses the private key to decrypt the coded message. The use of two keys overcomes a major weakness symmetric cryptography. Encrypting a file before uploading it to your system storage is a good way to add an extra layer of protection for your documents.

- Security Authentication and Authorization

Security authentication is verifying whether who users claim to be or sometimes machine. Authorization is verifying the rights a user has to access that information. These two processes work hand in hand with each other, first is authentication and then authorization.

- Password

In any computing environment, a password is used as authentication method to grant authorization to users. This is no longer sufficient because as systems get better, hackers become smarter. If anyone has the correct username and password, the user is confirmed to be authentic. The drawback in this system is that passwords can be stolen or accidentally revealed. Because of this, stronger authentication methods need to be applied such as biometrics, tokenization and certification authorities.

- Biometrics

This authentication method that uses facial scans, voice and or iris recognition, fingerprints that is converted to digital information that can be interpreted by a computer. This method will consider only facial and voice recognition. Due of the difficulty to obtain a user's biometric data, it is very unlikely that a biometric data can be accidentally obtained. Facial and voice recognition are the favorites to use because of the ease of use. Facial recognition uses digital image or video frame to automatically identify and verify a person. This is done by selecting facial features from the image or video and comparing them to the ones in a database. Extra features such as blinking and tilting has been implemented to ensure that still images and photographs cannot be used. Voice recognition uses a voice signature (tone, pitch, and accent) to identify and verify a user. Same as facial recognition, the voice signature is transformed to digital data that the computer can interpret and compare to a recorded voice. A randomizer can generate different set of words every time to ensure that even if a malicious user has a recorded voice, he or she cannot guess what words would be prompted for authentication.

- Tokenization

This simply refers to replacing sensitive data with unique identification symbols that keep all essential information in order without risking its security. This means replacing useful information that makes sense to data element that is non-sensitive. Tokenization which is used in e-commerce to enhance credit card and online banking security is being used to boost system computing security. Vendors like cipher system support tokenization. Without a tokenization system, the value cannot be reversed and is hereby useless to a malicious user. The tokenization system provides data processing applications

with the authority and interfaces to request tokens, or de-tokenize back to sensitive data. Only authorized users should be granted access to tokenization system and processes. Much consideration should be given to these authentication items when considering tokenization solution; Identification – requires a level of believe that applications, users and processes that request access is identified. Enrollment – requires the uniqueness of identity during account provisioning. Authentication – validates the user identity at the time of a request. Authorization – requires the verification of the authenticated user. Termination – requires revoking or denying the user authentication. Maintenance – requires the management (modification or termination) of accounts.

- Certification Authorities

A certificate authority (CA) is a control in a network that gives and maintains security credentials and public keys for message encryption. CA checks with registration authority (RA) to confirm that the information given by the solicitor of a digital certificate. If the information is verified by the RA, then the certificate is issued by the CA. Depending on the public key infrastructure (PKI) implemented, the certificate includes the owner's public key, the expiration date of the certificate, the owner's name, and other information about the public key owner. The PKI enables the sharing and identification of public encryption keys which aids users to safely exchange data over networks and internet.

- Access Control Policy

The access control policy depends greatly on the administration of the organization (structure). A lot of time and resources is being spent on

protection from network hacks and attacks. However, it is viewed that one of the biggest threats to corporate data is insider threat. To effectively eliminate insider threat and mitigate attacks on a computer system, a network must be set on a need-to-know and need-to-use basis. System vendors must ensure that every user or employee can only log in according to pre-authorization that has been issued. Access rights should be granted cautiously based on the role and security level of the user in that organization. Access control policy defines the rules on how access control should be regulated and enforced. Access control policy should include administration control (policies and procedures, personnel control, duty rotation and separation).

A distributed network is made up of clients and servers connected together which can potentially communicate with each other sharing data and resources. The distributed systems in recent years has been an environment that aids the linking of different workgroups together such as departments, branches, and divisions of an organization. In a distributed environment, data is usually kept in servers, usually on many servers and accessed by many users simultaneously. These various users and servers might be located in different geographic locations and connected by WAN links. In the 20th century, the distributed systems in the distributed network consisted of large numbers of desktop computers connected together. Today, the internet and web technologies have greatly expanded the concept of distributed systems. The internet is a now a great distributed collection of systems that spans various content distributed servers. Data security and communication security is now a very critical issue on today's networks as lot of internet theft such as eavesdroppers, internet hackers, intruders, forgers, and other attackers are on the increase. User

identification, user anonymity, mutual authentication, and key distribution are desirable features for securing communications in a distributed network.

- 1) User identification is an assurance a communicating party gets during a communication that the other party is a legitimate or registered user with evidence. Mechanisms for identifying users are specifically designed to ensure access control to data or resources with privileges are assigned to a particular identity such as Kerberos [2].
- 2) User anonymity ensures that the identities of the communicating parties are kept hidden throughout an entire communication [3, 4] as an intruder cannot obtain personal information about any of the communicating parties by masquerading or wire tapping.
- 3) For mutual authentication, both communicating parties get assured they are both communicating to a legitimate party with different evidence. The user's password must be kept private and never to be sent over the network, especially as text where eavesdroppers could easily capture the information and use it to access secure systems by masquerading as the legitimate user. Instead, different unique handshake schemes are used to authenticate users in a secure way as and sharing secret keys.
- 4) Key distribution establishes a secret key which is shared in common by the communicating parties for the symmetric encryption of the rest of their communication [5-8].

1.2 Purpose of the thesis

User identification, user anonymity, mutual authentication, and key distribution are desirable features for securing communications in a distributed network. Yang et al [9] came up with new efficient user identification and key distribution scheme

providing enhanced security. Mangipudi-Katti [11] proposed a secure identification and key agreement protocol with user anonymity (SIKA). Hsu-Chuang [12] demonstrated that the user anonymity requirement proposed in [9] and [11] was not met and came up with a novel user identification scheme with key distribution preserving user anonymity to ensure that both the user and the service provider identifies themselves during a handshake. Despite all the their claim of efficient, secure and novel they have problem related with the use of inverses of identifiers but the inverses may not exist which leads to the failure of the schemes. The schemes did not meet claim of efficiency.

Based on the schemes analysis, we construct numerical counter - examples demonstrating failures of the three schemes, and propose a deterministic algorithm allowing getting invertible identifiers under condition of usage of RSA – likekey settings which are kept secret (factorization of $N = p * q$, p and q being secret primes). Invertibility exist when two numbers are relatively prime to each other. Relatively prime numbers is a set of numbers which have no other common factor than one. In other words the highest common factor is one. Numbers that divides N are not invertible modulus N as the numbers share a common factor with N .

The problem of generating such identifiers is as follows: The identifiers are generated by communicating parties not knowing p , neither q . p and q are secret primes that is generated by the *SCPC* (Smart card producing center) a trusted third party that act as a middle man between the communicating parties. Hence, in the random choice, numbers selected from Z_N may be multiples of p or q and thus be not invertible modulo N that is actually the problem of the schemes [9, 11, and 12].

1.3 Outline for this thesis

This study is made up of six chapters. Chapter one discusses the purpose of the study. Chapter 2 contains the review of literature where past security problems and schemes are discussed and in Chapter 3; the three schemes are reviewed and analyzed. In Chapter 4, numerical counter-examples are given showing failures of the three protocols. Chapter 5 presents our deterministic algorithm for invertible identifiers generation that fixes the problem. And finally, the conclusion to the research comes up in Chapter 6.

Chapter 2

LITERATURE REVIEW

2.1 Concepts of securing a networks

Following up the previous works of other researchers in the field of security will also provide a good ground for the reviewing this literature. Internet services have been on the increase, accessing secured resources in a distributed [13] environment needs to be studied and given proper attention. Schemes [8-12] that have been proposed in recent years can be divided into two categories, which are secret key cryptography (SKC), hash-based authentication [13] and public-key cryptography [15]. Secret key cryptography (SKC) which both decryption and encryption is done with a single key. Public key cryptography (PKC) does encryption with one key and decryption with another. Hash function makes use of mathematical transformations to encrypt data that are irreversible. Trust is the requirement of cryptography. SKC ensures that a message is confidential, hashed codes ensure a message is confidential and PKC gives a solution of distributing messages secretly. In order for the communicating parties to be such they are communicating legitimately, they need to be a trusted third party. There are different types of trusted models that are used in cryptographic schemes. There are, Pretty Good Privacy (PGP) users which decide to keep their set of trusted public keys, Kerberos: a secret key distribution scheme using a trusted third party and Certificates: which allows various sets of trusted third parties to authenticate each other. These trusted models differ in complexity, general applicability, scope, and scalability.

2.1.1 Pretty Good Privacy (PGP)

Pretty Good Privacy is a long standing scheme used widely for private e-mail based on public key methods. A PGP user keeps and maintains a local key ring of all their known and trusted public keys. Web of trust is what each users uses to ensure a key is trustworthy. In PGP, no statement or protocol is made on how to determine a trusted user. So at all times, encryption and signatures are based on public keys which can only be used when the appropriate public key is on the user's key ring. This model, however, has limitations as many due to the fast growing applications, many public keys would be required for single user to store and maintain. In case a legitimate user uses another computer when to send a message and can't access his key ring finally PGP also may not proffer fast and secure communication.

2.1.2 Kerberos protocol

Kerberos [2] makes use of a client and server architecture. Kerberos provides authentication for a user-to-server instead of host-to-host authentication. Secret key technology is what is applied to the security and authentication model. The network host has its own secret key. A Key Distribution Center (KDC) is a trusted host that maintains the keys of all other hosts on the network and knows the keys of all the hosts or some of the host involved in what we call a realm. The new node and a KDC are configured anytime there is a new node online with node's key then the keys are sent round through a secured channel. It consists of the three participants. They include the client, the server and the key distribution server. Unlike the KDC which consists of authentication server (AS) and ticket granting server (TGS). A client sends a request to the authentication server with certain login information. The authentication server checks to ensure the user is legitimate. If the user exists on the registered list, the ticket granting server sends a reply to the client with the secret

key. It is based on the trusted third party and symmetric cryptography protocols. It shares different secret key with different users on the network. Having that secret key ensures legitimacy. Below is the explanation of how the protocol works.

1. Client requests for a Ticket Granting Ticket from the AS and awaits response.

The AS first ensures the client is legitimate and then provides the client with a secret key for this login session for authentication which is the TGS session key and a ticket-granting ticket (TGT). These give the client permission to communicate directly with the AS. The lifetime on a ticket is finite, which needs to be gotten when timed out.

2. The client uses the application server's key to establish a connection to the service it needs since he can obtain the application server's key from the TGS.

The client gives the TGS session key and TGT to the TGS; the TGS responds with an application session key (ASK) and an encrypted form of the Application Server's secret key; this secret key is not sent on the network in any other form.

3. The client is now legitimate and proves its identity to the server by supplying the Kerberos ticket, ASK, and encrypted Application Server secret key. Application Server in turn authenticates itself to the client. This can now aid communication.

Two types of keys are used for generating both tickets and authenticators. When a client wants to communicate with the server, it sends a name to the TGS and awaits a response from the TGS. Kerberos authentication server checks client legitimacy by

checking his registered list, and issues a session key which is been used between both communicating parties which is known as the Ticket Granting Ticket (TGT). Kerberos now encrypts the session key with the client's secret key. TGT and session keys are been saved by the client for subsequent communication and the password is been erased by the one way hash. Kerberos gives a solution to lots of problems of PGP's web of trust, it has a larger scope and can be scalable. The Kerberos server needs to know all client systems before any transactions.

2.2 Public key cryptography

Public key cryptography operations are based on finite fields since finite fields are exact and thus are easily reversible using a trapdoor function [16]. A trapdoor function with a public parameter is used for encryption which is reversible if additional secret information is available. The two common examples of public key cryptography are the RSA [15] and Diffie-Hellman [16] encryption schemes. Parameters in a public key cryptography are constructed for encryption and decryption and not as part of the message to the intended communicating party. In order to ensure parameter verification of a service request, directories are looked up using constructed public key infrastructures (PKI) [15] which makes up the user's public key. The information stored in the directories is called the certificate. The certificate is usually signed by a trusted third party authority. Due to security leakages, many keys get compromised and becomes invalid, most certificate lifetime expires and are being sent to a certificate revocation lists. Instead of parameters for encryption and decryption to be random, identity-inclined cryptography computes its keys from the attributes of a recipient that is publicly known. However, the private key of the users should at all times be kept anonymous to the trusted party in a PKI based settings. By contrasting, in an identity-inclined cryptography, impersonation

can be done by the trusted third party since he sets up encryption parameters and generates user's private key.

2.2.1 RSA Algorithm

RSA algorithm developed by Rivest, Adi Shamir and Len Adleman is a public key algorithm that is used to encrypt, decrypt, sign signatures and agree on shared keys for communication. RSA algorithm uses keys sizes from 1024 to 2048. It is based on difficulty of prime factorization of the data as data should be encrypted and decrypted automatically by the receiver. RSA algorithm ensures the public key of the receiver is used for messages encryption, and the receiver can decrypts messages encrypted by the sender with its own private key. RSA-based signature algorithm ensures a receiver verifies the message signed by the sender's private key with sender's public key; this guarantees the authentication of an RSA signature algorithm. Although the RSA algorithm is much secured, computing power improvement has made a great growth in the key length. Now RSA is characterized and observed by implementing the algorithms for computation which increases its performance. Overview of RSA algorithms is given below.

2.2.1.1 RSA Key generation

Select two large primes p and q and computes $N = p * q$, selects e where the $gcd(e, \varphi(N)) = 1$, and calculate $d = e^{-1} \text{ mod } \varphi(N)$, where $\varphi(N) = (p-1)(q-1)$ is an Euler's totient function. (e, N) are public system parameters and (d, p, q) are the private key.

2.2.1.2 RSA Encryption

Step 1. User A sends his public key (e, N) to user B , keeping his private key (d, p, q) secret.

Step 2. Anytime user B wishes to send message M to user A , he first gets M converted to an integer number such that it satisfies $0 < m < n$ using a padding scheme that has been agreed upon by both users.

Step 3. User B computes c using user's A public key parameter as $c \equiv m^e \pmod{n}$.

Step 4. C is sent to user A .

2.2.1.3 RSA Decryption

Step 1: User A computes m from c by using d as $m \equiv c^d \pmod{n}$.

Step 2: Given m , user A retrieves M by reversing the padding scheme.

2.2.2 Diffie and Hellman key exchange

Diffie and Hellman were the ones to bring about the concept of public-key cryptography. It very easy to compute exponents and computing discrete logarithms become so difficult. Diffie and Hellman allowed two users; user A and user B which generate a secret key and makes a mathematical calculation with the need to exchange some information over an unsecure channel so that eavesdropper cannot determine the shared secret key based upon this information he get through either wire tapping.

User A and User B agrees on a large prime number, P , choose some number G so that $G < P$. G must be primitive with respect to P . G is primitive to P if we can find integers j so that $G^j \pmod{P} = i$ for all values of i from 1 to $P-1$. User A or user B selects P and G ; and follow the steps below

User A

Selects a large random number,
 $X_A < P$.

X_A = User A 's private key

User B

Selects a large random number,
 $X_B < P$.

X_B = User B 's private key

$Y_A = \text{User } A \text{'s public key}$

$Y_B = \text{User } B \text{'s public key}$

1. Computes $Y_A = G^{X_A} \text{ mod } P$.

1. Computes $Y_B = G^{X_B} \text{ mod } N$.

2. Exchange public key with User B.

2. Exchange public key with User A.

3. Computes $K_A = Y_B^{X_A} \text{ mod } P$

3. Computes $K_B = Y_A^{X_B} \text{ mod } P$

X_A and X_B are private parameters which are kept secret while Y_A and Y_B are public parameters respectively. With their private key and the public key, user A and user B computes their secret keys, K_A and K_B , respectively, which is also $G^{X_A X_B} \text{ mod } N$.

2.3 Identity-inclined cryptography

Identity-inclined cryptography has been used in different research areas including networks security, organizations, rights protection, process automation and interoperability improvement among key generation centers. A user friendly key which meets user anonymity, mutual authentication, transparency and efficiency for identity-inclined cryptography is currently not available. User identification, user anonymity, mutual authentication, and key distribution are aspects that needs special attention if identity-inclined cryptography shall be applied to large organizations and the world at large. Identity-inclined cryptography does really give solutions to these issues; instead propose ways to implement identity-inclined cryptography to meet some of these requirements [9, 11,12].

2.3 Single sign-on

Single sign-on technology is a current technology which improves identification and password authentication [17,18]. SSO as it is called are systems designed to ensure one time login of a user into a network and allows permission in the network

without need to re-enter authentication information for each application in the network. Research studies on SSO mention that it relieves users of having to remember lots of identifiers and password reducing computational cost and improving security.

Chapter 3

REVIEW OF YANG ET AL, MANGIPUDI - KATTI AND HSU – CHUANG SCHEMES

3.1 Overview of the schemes

There are three participants involved in these schemes: the *SCPC* which is smart card producing center, the user (U_i), and the service provider (P_j). The schemes have three stages which are the system initialization, the registration, and the user identification.

- 1) The system initialization stage is where the *SCPC* generates the system parameters used by all the communicating parties for communication.
- 2) In the registration stage, encrypted identifiers are obtained during the registration stage after the user or the service provider registers with the *SCPC* by sending their selected identifier.
- 3) Finally, the service provider checks the user's legitimacy in the user identification stage.

The system initialization and registration stages are the same for all the schemes; they differ in the user identification stage only as shown below.

3.2 System initialization stage

In this stage, two large primes p and q are generated by the *SCPC*, then the *SCPC* computes $N = p * q$, selects e where the $\text{gcd}(e, \varphi(N)) = 1$, and performs the calculation $d = e^{-1} \text{ mod } \varphi(N)$, where $\varphi(N) = (p - 1)(q - 1)$ is an Euler's totient function. The element $g \in Z_N^*$ is been chosen by the *SCPC, which is the generator of Z_p^* and Z_q^* .*

The *SCPC* publishes (e, N, g) as public system parameters and keeps (d, p, q) as the private key.

3.3 Registration stage

The *SCPC* receives identities, ID_i and ID_j , from the registered user U_i and registered service provider P_j , and uses RSA to encrypt identifiers $S_i = (ID_i)^d \bmod N$ and $S_j = (ID_j)^d \bmod N$ respectively with its private key d and sends S_i and S_j via a secure channel to U_i and P_j .

3.4 User identification in Yang et al, Mangipudi - Katti and Hsu - Chuang schemes

3.4.1 User identification in the Yang at al. scheme

User identification stage in scheme [9] is illustrated by Figure 1 and is represented by the following steps Y1-Y4.

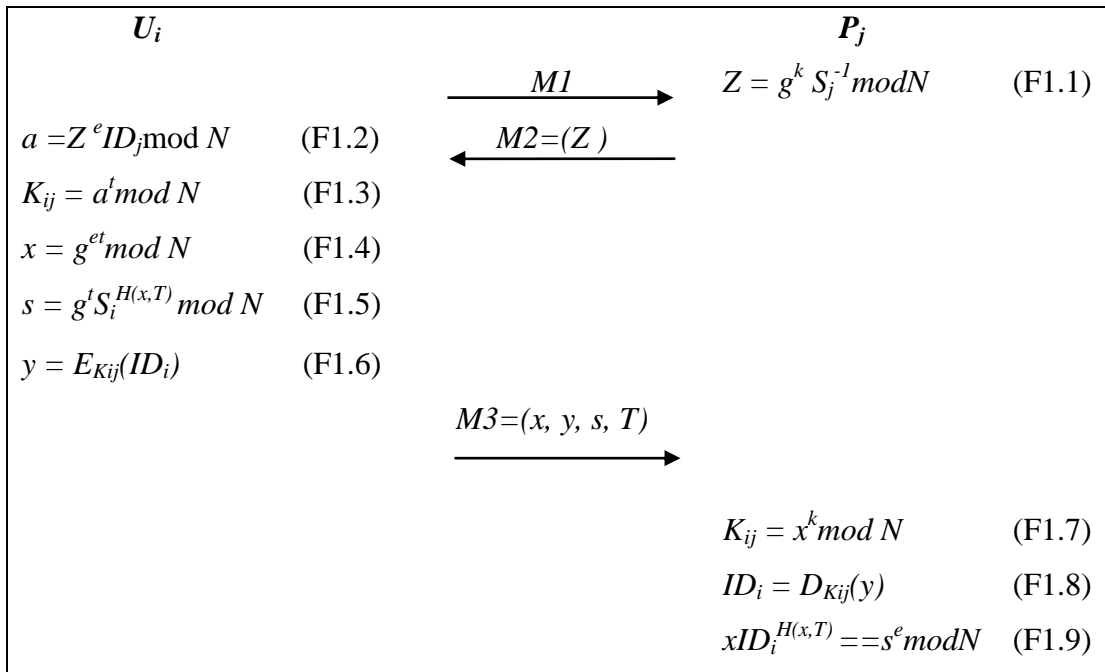


Figure 1. User identification in Yang at al. scheme

STEP Y1: $M1$ is a service request submitted to the service provider P_j by the user U_i .

STEP Y2: Service provider P_j on receiving $M1$, selects a random integer number k , gets $Z = g^k S_j^{-1} \bmod N$ computed as shown in (F1.1), and sends $M2 = (Z)$ to U_i . Note that $S_j^{-1} \bmod N$ is used in (F1.1) but its existence is not guaranteed since no conditions were imposed on the choice of ID_j in Section 4.2.

STEP Y3: After user U_i receives $M2$, U_i selects a random integer number t , and, in (F1.2)-(F1.6), computes $a = Z^e ID_j \bmod N$, $K_{ij} = a^t \bmod N$, $x = g^{et} \bmod N$, $s = g^t S_i^{H(x,T)} \bmod N$, $y = E_{K_{ij}}(ID_i)$, where T ensures the current timestamp. The key K_{ij} is now the shared session key between U_i and P_j , to ensure user's identifier, ID_i , is encrypted so that it is hidden while travelling to P_j thus supporting anonymity of the user. U_i then sends $M3 = (x, y, s, T)$ to P_j .

STEP Y4: Server P_j on getting $M3$, first checks validness of T . If $M3$ is invalid, P_j terminates the communication; or else, P_j , in (F1.7), (F1.8), computes $K_{ij} = x^k \bmod N$ which is the session key and uses it to get the identifier $ID_i = D_{K_{ij}}(y)$. Then P_j checks his maintained list of all registered users if ID_i is valid. If ID_i is not valid, P_j terminates the communication; or else, P_j goes ahead to verify in (F1.9) if $x ID_i^{H(x,T)} = s^e \bmod N$. Once the verification is successful, P_j is sure that U_i is a legitimate user.

3.4.2 User identification in the Mangipudi–Katti scheme

User identification stage in Mangipudi–Katti scheme [11] is as shown in Figure 2 and represented by steps MK1-MK4 below similar to steps Y1-Y4 (see Section 3.4.1). Unlike [9], each service provider, P_j , sets an additional RSA-like key pair as in the key generation stage. It is used together with a hash function for authentication of the service provider by a user (see (F2.1)-(F2.5)) which is not made in [9]. Two large primes p_j and q_j are generated by P_j . Then P_j computes $N_j = p_j * q_j$, selects e_j where the $\gcd(e_j, \varphi(N_j)) = 1$, and performs the calculation $d_j = e_j^{-1} \bmod \varphi(N_j)$. The

element $g_j \in Z_{N_j}^*$ is been chosen by the service provider P_j , which is the generator of both $Z_{p_j}^*$ and $Z_{q_j}^*$. Then P_j publishes (ID_j, e_j, N_j, g_j) as public system parameters and keeps (S_j, d_j, p_j, q_j) as the private key.

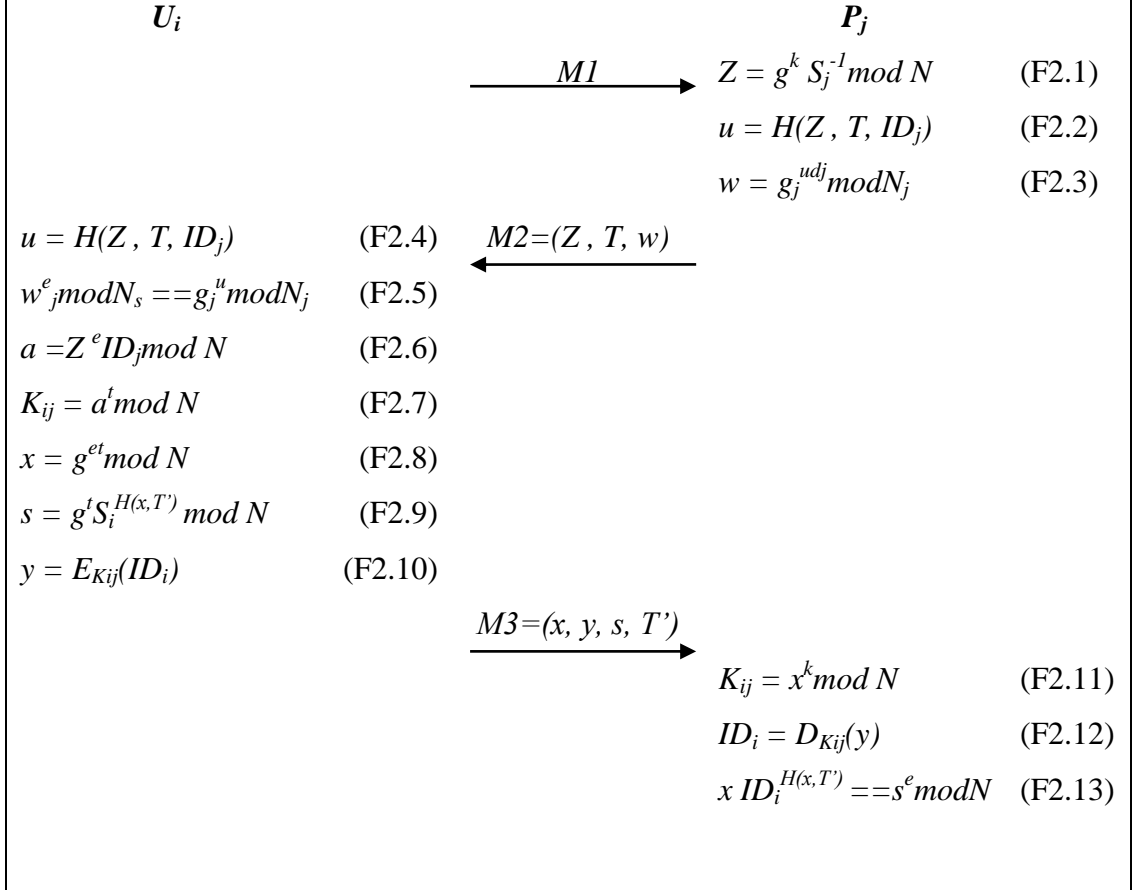


Figure 2. User identification in Mangipudi–Katti scheme

STEP MK1: $M1$ is a service request submitted to the service provider P_j by the user U_i .

STEP MK2: Service provider P_j on receiving $M1$, selects a random integer number k , in (F2.1), gets $Z = g^k S_j^{-1} \text{mod } N$ computed. Then, P_j computes in (F2.2), (F2.3) a signature w for (Z, T, ID_j) as $u = H(Z, T, ID_j)$, $w = g_j^{ud_j} \text{mod } N_j$ and T ensures the current timestamp to guide against replay attacks. P_j sends $M2=(Z, T, w)$ to U_i as shown in (F2.1). As seen, $S_j^{-1} \text{mod}$

N is also used here, but no conditions were imposed on ID_j , therefore $S_j^{-1} \bmod N$ may not exist.

STEP MK3: After receiving $M2$, user U_i checks if T is valid, calculates $u=H(Z,T,ID_j)$ in (F2.4), and then verifies the integrity of $M2$ by checking if $w^e \bmod N_j = g_j^u \bmod N_j$ in (F2.5). If it is invalid, U_i terminates the communication or else U_i selects a random integer number t and, in (F2.6)-(F2.10), computes $a = Z^e ID_j \bmod N$, $K_{ij} = a^t \bmod N$, $x = g^{et} \bmod N$, $s = g^t S_i^{H(x,T)} \bmod N$, $y = E_{K_{ij}}(ID_i)$, where T' ensures the current timestamp. The key K_{ij} is now the shared session key shared between U_i and P_j . Then U_i sends $M3=(x,y,s,T')$ to P_j .

STEP MK4: P_j on receiving $M3$, first checks if T' is valid. If it is not valid, P_j terminates the communication; or else, P_j in F(2.11), computes $K_{ij} = x^k \bmod N$ which is the session key and gets $ID_i = D_{K_{ij}}(y)$ in (F2.12). P_j further checks for ID_i in his maintained list of all registered users. If ID_i is not valid, P_j terminates the communication; or else, P_j goes ahead to verify in (F2.13) if $x ID_i^{H(x,T')} = s^e \bmod N$. Once the verification is successful, P_j is sure that U_i is a legitimate user.

3.4.3 User identification in the Hsu–Chuang scheme

User identification stage in Hsu–Chuang scheme [12] is as shown in Figure 3 and represented by the following steps HC1-HC7. The scheme, similar to [11], uses service provider authentication by a user but with the help of hash function only (see (F3.8)-(F3.12)).



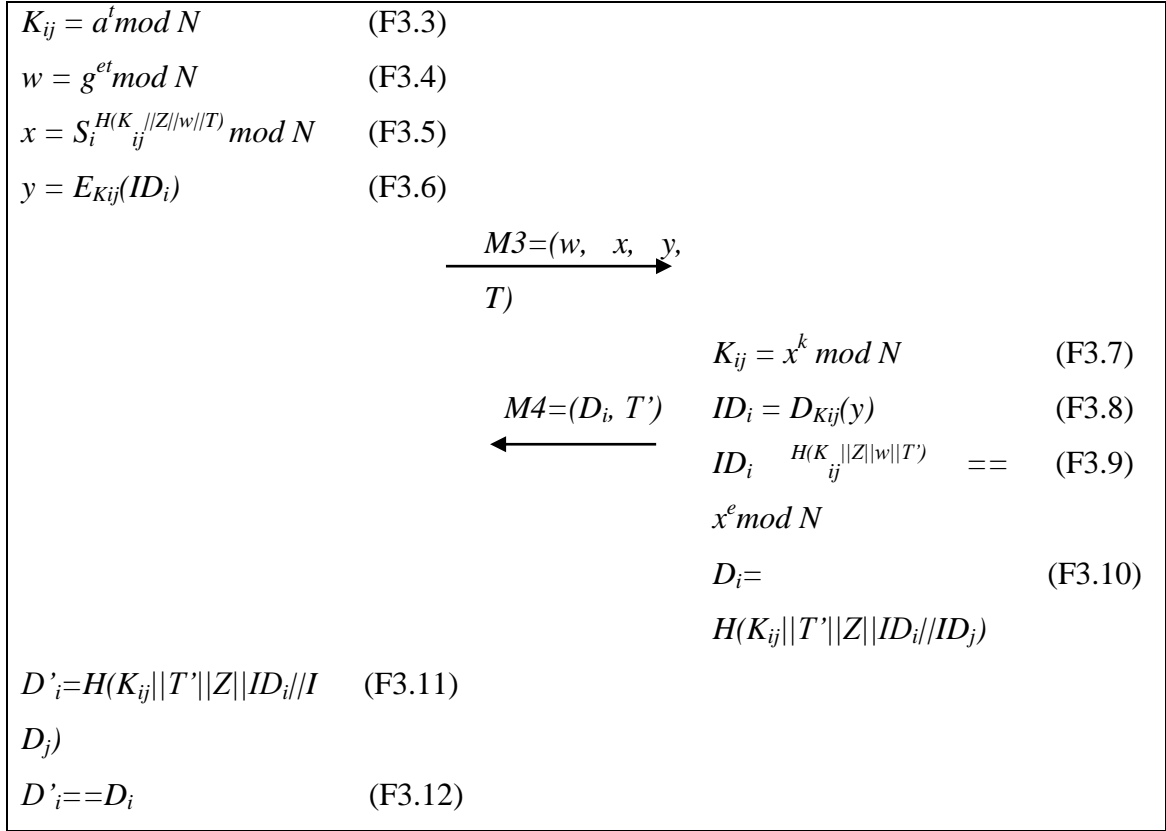


Figure 3. User identification in Hsu–Chuang scheme

STEP HC1: Message $M1$ is a service request submitted to the service provider P_j by the user U_i .

STEP HC2: Service provider P_j selects a random integer number k , on getting $M1$, gets $Z = g^k S_j \text{mod } N$ in (F3.1) computed, and sends $M2 = (Z)$ to U_i .

STEP HC3: After user U_i receives $M2$, U_i chooses a random integer number t and, in (F3.2)-(F3.6), gets $a = Z^e ID_j^{-1} \text{mod } N$, $K_{ij} = a^t \text{mod } N$, $w = g^{et} \text{mod } N$, $x = S_i^{H(K_{ij} || Z || w || T)} \text{mod } N$, $y = E_{K_{ij}}(ID_i)$ computed, where T ensures the current timestamp. The key K_{ij} is now the shared session key shared between U_i and P_j . Then U_i sends $M3 = (w, x, y, T)$ to P_j . As seen in (F3.2), $ID_j^{-1} \text{mod } N$ is used here and no conditions were imposed on ID_j in Section 2.2, therefore $ID_j^{-1} \text{mod } N$ may not exist.

STEP HC4: P_j on receiving $M3$, first checks if T is valid. If it is not valid, P_j terminates the communication; or else, P_j gets $K_{ij} = x^k \bmod N$ computed in (F3.7) which is the session key.

STEP HC5: P_j gets in (F3.8) the identifier $ID_i = D_{K_{ij}}(y)$. P_j further checks for ID_i in his maintained list of all registered users. If ID_i is not valid, P_j terminates the communication; or else, P_j goes ahead to verify in (F3.9) if $ID_i^{H(K_{ij}||Z||w||T')} = x^e \bmod N$. Once the verification is successful, P_j now gets sure that U_i is a legitimate user.

STEP HC6: P_j now gets $D_i = H(K_{ij}||T'||Z||ID_i||ID_j)$ also computed in (F3.10) and forwards $M4 = (D_i, T')$ to U_i , where T' ensures the current timestamp.

STEP HC7: U_i on getting $M4$ from P_j , ensures T' is valid. If yes, in (F3.11), gets $D'_i = H(K_{ij}||T'||Z||ID_i||ID_j)$ computed, then checks in (F3.12) if $D_i = D'_i$. Once the verification is successful, U_i is sure that P_j is a legitimate service provider.

Chapter 4

EXAMPLES OF FAILURES OF YANG ET AL, MANGIPUDI - KATTI AND HSU - CHUANG SCHEMES

4.1 Introduction

In this section, examples are constructed to show failures of the three schemes [9, 11,12]. In the system initialization stage (see Section 3.2) of the schemes, the *SCPC* chooses two prime numbers, say, $p = 13$ and $q = 5$, computes $N = p \cdot q = 13 \cdot 5 = 65$; $\varphi(N) = (p - 1)(q - 1) = (13 - 1)(5 - 1) = 12 \cdot 4 = 48$. *SCPC* selects $e = 5$ such that the $\gcd(5,48) = 1$ and from $5 \cdot d = 1 \pmod{48}$ computes $d = e^{-1} \pmod{48} = 29$. The *SCPC* also chooses $g = 2$ which generates elements of both Z_{13}^* and Z_5^* . *SCPC* publishes $(e, N, g) = (5, 65, 2)$ and keeps $(d, p, q) = (29, 13, 5)$ secret. In the Registration stage (see Section 3.3) service provider P_j chooses an $ID_j = 15$ and sends it to the *SCPC*; *SCPC* computes $S_j = 15^{29} \pmod{65} = 45$ and sends it via a secure channel to the service provider P_j . User U_i also chooses $ID_i = 24$ and sends it to the *SCPC*; *SCPC* computes $S_i = 24^{29} \pmod{65} = 59$ and sends it via a secure channel to user U_i .

4.2 Counter-example for Yang et al, Mangipudi - Katti and Hsu - Chuang Schemes

4.2.1 Example of failure for the Yang et al. and Mangipudi–Katti schemes

After the user U_i requests a service in $M1$ from the service provider P_j , as both Figure 1, for Yang et al. [9], and Figure 2, for Mangipudi-Katti [11] schemes, show

respectively (see Sections 3.4.1, 3.4.2) the service provider P_j computes Z in (F1.1) and (F2.1) respectively in order to send it in $M2$. User U_i , upon receiving $M1$, will find out that $S_j^{-1} \text{ mod } N = 45^{-1} \text{ mod } 65$ does not exist [1] since $\text{gcd}(45,65)=5 \neq 1$, and 45 and 65 are not relatively prime.

4.2.2 Example of failure for the Hsu-Chuang scheme

In Hsu-Chuang scheme [12] (see Section 3.4.3), after the user U_i requests a service $M1$ from the service provider P_j , as shown in Figure 3, the service provider P_j computes $Z = 2^6 \cdot (15)^{29} \text{ Mod } 65 = 20$ where $ID = 15$ for $M2$ and sends it to user U_i . On the other hand, the user upon receiving Z in $M2$ from the service provider P_j selects a random integer number t and in an attempt to compute a , K_{ij} , w , x , and y finds out that in (F3.2) $ID_j^{-1} \text{ mod } N = 15^{-1} \text{ mod } 65$ does not exist since again $\text{gcd}(15, 65)=5 \neq 1$, and the numbers 15 and 65 are not relatively prime that is required for the multiplicative inverse existence [1]. Due to not existence of the inverses required in the schemes [9,11,12], the rest of the schemes fail.

Thus, we have shown that the protocols may fail because of non-invertibility modulo N of the identifiers of the users and service providers encrypted by *SCPC* (in schemes [9, 11]) or not encrypted (in scheme [12]). Non-invertibility might happen because the identifiers are not relatively prime to N which is published by *SCPC*. The identifiers may be selected from Z_N but as far as $N = p * q$, and p , q are kept secret by *SCPC* and not known to the both the service provider and the user, randomly selected from Z_N identifiers may be multiples of p or q , thus not being relatively prime to N : as we have seen in the Counter-examples 1 and 2, the identifiers, 45 and 15, are multiples of $q=5$. Hence, there is a problem of selecting identifiers not being multiples of p or q not knowing both p and q . Such a procedure of finding numbers

from Z_N^* relatively prime to N fixing the problem of the three protocols is proposed in the chapter 4.

Chapter 5

IMPROVING YANG ET AL, MANGIPUDI - KATTI AND HSU - CHUANG SCHEMES

5.1 Finding invertible identifiers from Z_N^*

Number of numbers in Z_N^* is given by the Euler's totient function

$\varphi(N) = (p-1)(q-1)$, and $\lim_{N \rightarrow \infty} \frac{\varphi(N)}{N} = 1$. Hence, for large numbers, identifiers

randomly selected from Z_N with probability close to 1 will be invertible modulo N .

That is why, in spite of the schemes [9,11,12] were published in 2004-09, still the problem considered in the present paper was not revealed. For example, in [17,18], scheme [12] is reviewed and analyzed by [17] but the problem of non-invertibility of identifiers discussed here is not revealed there. Nevertheless, the probability of selecting of non-invertible identifiers leading to the failure of all three schemes is positive, and it is important to be able selecting invertible identifiers to guarantee correct work of the schemes. Table 1 provides some statistics on the number of invertible numbers in Z_N with growth of N . The code used for generating the number of invertible numbers excluding 1 is found in Appendix A where parameters p and q are both supplied and N is computed and the number of invertible numbers excluding one is computed. The probability is gotten by dividing the total number N by the number of the number of invertibility number computed by the code in Appendix A. The code runs through all the numbers in N and checks if all the

number from 2 up until $N-1$ is relatively prime or co-prime to N . ie if the highest common factor all numbers between 2 and $N-1$ is 1.

Table 1. Statistics of invertible numbers.

P	Q	N	Number of invertible numbers excluding 1	Probability of invertibility
5	3	15	7	0.466667
7	5	35	23	0.657143
13	5	65	47	0.723077
11	7	77	59	0.766234
13	11	143	119	0.832168
17	13	221	191	0.864253
19	17	323	287	0.888545
23	19	437	395	0.90389
29	23	667	615	0.922039
31	29	899	839	0.933259
37	31	1147	1079	0.940715
41	37	1517	1439	0.948583

The chart in Figure 4 shows that the probability of selecting of non-invertible identifiers from N will always rise to one but not one which entails that failure is eminent. The larger the number of primes the higher the probability. It can not reach maximum efficiency since it the probability never get to one. The is very risky for security systems. It is better it is bad meaning not in good conditions or it is good, functioning with maximum efficiency.

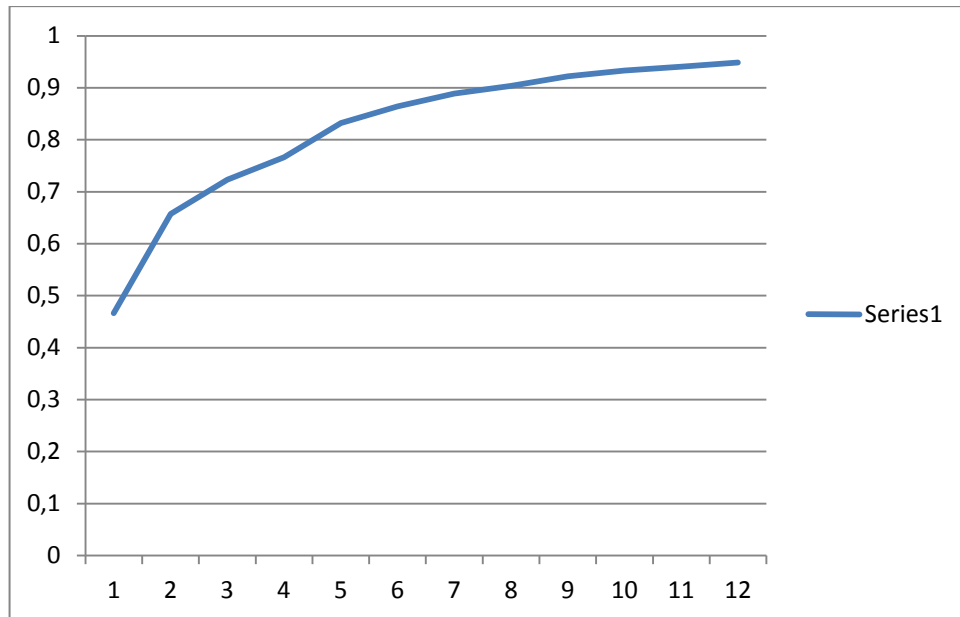


Figure 4. The probability of selecting of non-invertible identifiers

Figure 5 shows that as N increases, number invertible identifiers also increases and increases the number of non - invertible identifiers.

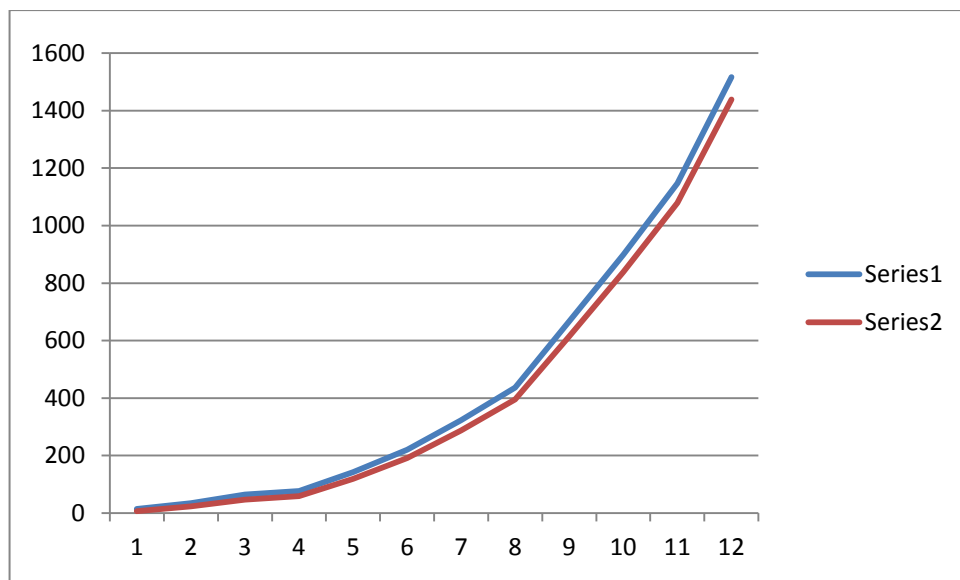


Figure 5. A graph showing growth of invertible identifiers

Next, we propose a procedure, $InvertID()$, of a random numerical identifier generation such that its inverse exists. This procedure has one input $N=p*q$ which is

the public parameter received from the *SCPC*, where $p, q > 2$ are unknown primes, and one output ID which is any integer number between 2 and $N-1$, invertible modulo N , i.e. $gcd(ID, N) = 1$. The procedure will output an identifier in line 2 if the imputed identifier is invertible otherwise it runs an increment and returns the incremented identifier in line 4 and in case the selected identifier was not outputted in line 4 it will always output an invertible identifier in line 6. The full functioning of the procedure is explained below in cases 1 and 2.

```

Procedure InvertID(input N; output ID){ //begin procedure
    Generate randomly ID from 2..N-1; //line 1
    If ( $ID^{-1} \bmod N$  exists) return ID; //line 2
    ID++; //line 3
    If ( $ID^{-1} \bmod N$  exists) return ID; //line 4
    ID++; //line 5
    Return ID; //line 6
} //end procedure

```

5.2 Theorem 1

Procedure $InvertID(N, ID)$ above returns ID such that $ID^{-1} \bmod N$ exists, $N = p \cdot q$, $p, q > 2$ - distinct primes.

Proof

Let first randomly generated ID (in line 1) be called ID_1 .

Case 1: If ID_1 has inverse modulo N , then it is returned (in line 2) and the theorem is true in that case.

Case 2: ID_1 is not invertible as a multiple of either p or q , taking p for instance. As far as condition in line 2 is false, new value is obtained in line 3: $ID_2 = ID_1 + 1$. ID_2 can't be a multiple of p because $p > 2$, ID_1 is a multiple of p , and, hence, $ID_2 = ID_1 + 1 = kp + 1$.

Case 2.1: ID_2 is not a multiple of q . Then ID_2 is invertible modulo N and is returned in line 4; thus, the theorem is true.

Case 2.2: ID_2 is a multiple of q . Then in line 5, $ID_3 = ID_2 + 1$, and ID_3 can't be a multiple of q because ID_2 is a multiple of q and $q > 2$. ID_3 can't be a multiple of p because $ID_3 = ID_1 + 2$, ID_1 is a multiple of p , and $p > 2$. Hence, ID_3 is not a multiple of q , neither of p . Hence, $\gcd(ID_3, N) = 1$, and ID_3 is invertible modulo N ; ID_3 is returned in line 6. Thus, in that case, the theorem is also true.

As far as no other cases are possible, and in all possible cases it is true, and numbers p and q in N are symmetric, the theorem is fully proved.

Let us consider an example of applying $InvertID()$ with $N = 65 = 5 \cdot 13$. Let in line 1, $ID = 25$ which is not invertible. Then, in line 3, $ID = 25 + 1 = 26$, which is also not invertible. Then, in line 5, $ID = 26 + 1 = 27$, which is invertible modulo 65 and returned in line 6 since $\gcd(27, 13) = 1$.

Use of the procedure $InvertID()$ allows in $O(1)$ time getting invertible identifiers which may be used for fixing scheme [12] since it uses in (F3.2) inverse of the service provider identifier. But schemes [9, 11] use (see (F1.1), (F2.1)) inverses of the service provider identifier encrypted by the private key d of $SCPC$ (see Section

3.3). Hence, we need this encrypted identifier, S_j , to be invertible modulo N . For that, we have the next

5.3 Theorem 2

$$\text{If } C = M^d \text{ mod } N, \quad (1)$$

And

$$\gcd(M, N) = 1, \quad (2)$$

$N = p \cdot q$, where $p, q > 1$ are two different primes, then

$$\gcd(C, N) = 1, \quad (3)$$

Proof

Let $\gcd(C, N) \neq 1$. Then $\gcd(C, N) \in \{p, q\}$. Let

$$\gcd(C, N) = p. \quad (4)$$

Hence $C = a \cdot p$, where $a \in \mathbb{Z}_N$. From (1),

$$C = M^d + kN = M^d + kpq. \quad (5)$$

As far as p divides C according to (4), from (5) it follows that p divides also M^d . As far as M is not divisible by p , according to (2), it's any power contains only divisors of M inside which there is no p . Since p is prime, M^d is not divisible by p and we arrive at the contradiction: M^d is divisible by p (see (4), (5)), and, from the other side, is not divisible by p , as we get from (2). Hence, our assumption of divisibility of C by p is false. Similar, non-divisibility of C by q may be shown. As far as N has only two divisors, p, q , (3) and Theorem 2 are proved.

So, Theorem 2 guarantees that if an identifier, ID_j , is selected as invertible modulo N , it's encrypted by $SCPC$ value, S_j , is also invertible modulo N , and may be safely used

in the schemes [9, 11]. Thus, the use of the proposed procedure, *InvertID()*, allows safe use of the three schemes [9,11,12].

Chapter 6

CONCLUSION

In this research, for three important schemes [9,11,12] intended for user anonymity, key distribution, mutual authentication and user identification were reviewed. There are a new efficient user identification and key distribution scheme providing enhanced security, a secure identification and key agreement protocol with user anonymity (SIKA) and a novel user identification scheme with key distribution preserving user anonymity.

All used inverse of identifiers, we show that they might fail due to the identifiers non-invertibility modulo $N=p*q$, where p and q are secret primes not known to the users and service providers. The problem is that due to the secrecy of p and q , generating invertible modulo N identifiers using random choice leads to indeterministic algorithms (number of trials to get invertible identifier is not limited in the worst case). We propose a procedure, *InvertID()*, combining one random choice and it is at most two increments that guarantees invertible identifiers generation in $O(1)$ time complexity. The procedure guarantees selection of identifiers that will be invertible at all time. Use of this procedure for the choice of identifiers allows safe work of the schemes [9,11,12] preserving all their useful features.

REFERENCES

- [1] W. Stallings, *Cryptography and Network Security. Principles and Practices*. 3rd Edition, Pearson Education International, Upper Saddle River, 2003.
- [2] J. Kohl, C. Neuman, The Kerberos authentication service (v5), Internet RFC 1510, 1993.
- [3] Y. Cai, Y. Wang, Identity-based conference key distribution protocol with user anonymity, *Chinese Journal of Electronics* 16 (1) (2008) 179–181.
- [4] T. J. Cao, H. Lei, Privacy-enhancing authenticated key agreement protocols based on elliptic curve cryptosystem, *Acta Electronica Sinica* 36 (2) (2008) 397–401.
- [5] J. W. Byun, D. H. Lee, J. I. Lim, EC2C-PAKA: An efficient client-to-client password-authenticated key agreement, *Information Sciences* 177 (19) (2007)3995–4013.
- [6] L. Chen, Z. Cheng, N. P. Smart, Identity-based key agreement protocols from pairings, *International Journal of Information Security* 6 (4) (2007) 213–241
- [7] A. O. Freier, P. Karlton, P. C. Kocher, Secure Socket Layer 3.0, Internet Draft, 1996.

- [8] T. C. Wu, T. T. Huang, C. L. Hsu, K. Y. Tsai, Recursive protocol for group-oriented authentication with key distribution, *Journal of Systems and Software* 81(7) (2008) 1227–1239
- [9] Y. Yang, S. Wang, F. Bao, J. Wang, R. H. Deng, New efficient user identification and key distribution scheme providing enhanced security, *Computer & Security* 23(8) 2004 697–704, 2004.
- [10] T. S. Wu, C. L. Hsu, Efficient user identification scheme with key distribution preserving anonymity for distributed computer networks, *Computers and Security* 23 (2) (2004) 120–125.
- [11] K. V. Mangipudi, R. S. Katti, A secure identification and key agreement protocol with user anonymity (SIKA), *Computer & Security* 25(6) 2006, 420–425.
- [12] C.-L. Hsu, Y.-H. Chuang, A novel user identification scheme with key distribution preserving user anonymity for distributed computer networks, *Information Sciences* 179(4) 2009, 422–429.
- [13] Y. Xu, R. Song, L. Korba, L. Wang, W. Shen, and S. Y. T. Lang, “Distributed device networks with security constraints,” *IEEE Trans. Ind. Inf.*, vol. 1, no. 4, pp. 217–225, Nov. 2005.
- [14] L. Lamport, “Secret word authentication with insecure communication,” *Commun. ACM*, vol. 24, no. 11, pp. 770–772, Nov. 1981.

- [15] R. Rivest, A. Shamir, and L. Adleman, "A method for obtaining digital signatures and public key cryptosystems," *Commun. ACM*, vol. 21, no. 2, pp. 120–126, Feb. 1978
- [16] Diffie W. and Hellman M. New directions in cryptography. *IEEE Trans. Inform. Theory* IT-22, (Nov. 1976), 644-654.
- [17] C.-C. Chang, C.-Y. Lee, A secure single sign-on mechanism for distributed computer networks, *IEEE Trans. Ind. Electron.* 59(1) 2012, 629–637. Jan 2012
- [18] Guilin Wang, Jiangshan Yu, and Qui Xie, "Security Analysis of a SSO mechanism for Distributed systems", *IEEE Trans. In industrial informatics*, vol.9, no.1, Feb.2013

APPENDICES

Appendix A: Java Code for generating invertible numbers

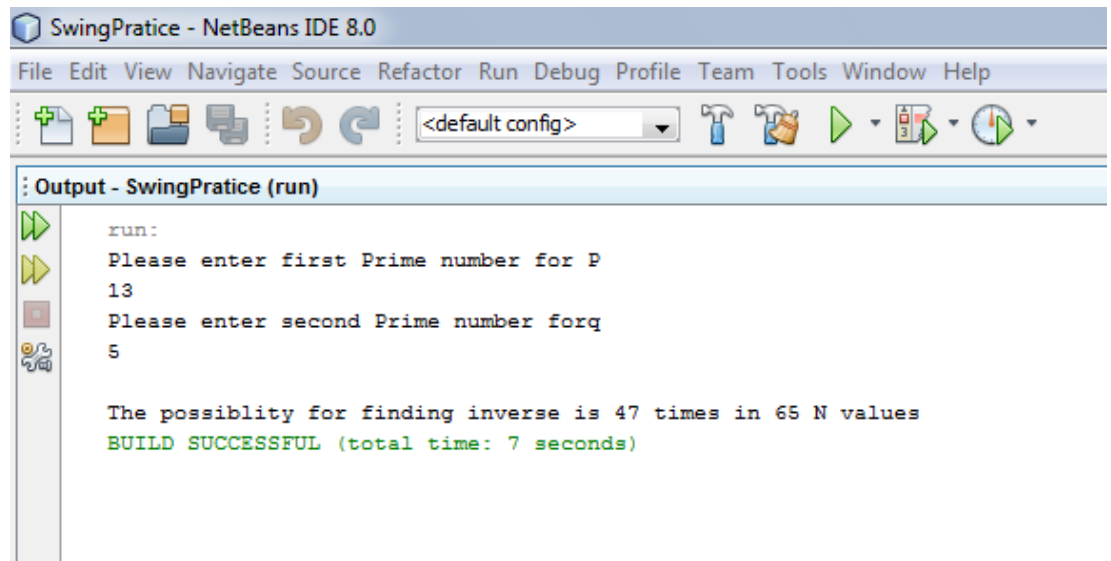
```
/**
 *
 * @author piroadams
 */
public class Num_inverses {
public static int find_GCD(int num_a, int num_b) {
int num_temp;
while (num_b > 0) {
num_temp = num_a % num_b;
num_a = num_b;
num_b = num_temp;
}
return num_a;
}
public static void main(String[] args) {
int id, prime_p, prime_q, cnt = 0, n;
Scanner in = new Scanner(System.in);
System.out.println("Please enter first Prime number
for P");
prime_p = in.nextInt();
System.out.println("Please enter second Prime number for
q");
prime_q = in.nextInt();

n = prime_p * prime_q;
for (id = 2; id < n; id++) {
if (Num_inverses.find_GCD(id, n) == 1)
cnt++;
}

System.out.println();
System.out.println("The possibility for finding inverse
is " + cnt + " times" + " in " + n + " N values");

}
}
```

Appendix B: Output of the code



SwingPratice - NetBeans IDE 8.0

File Edit View Navigate Source Refactor Run Debug Profile Team Tools Window Help

<default config>

Output - SwingPratice (run)

```
run:
Please enter first Prime number for P
13
Please enter second Prime number forq
5

The possiblity for finding inverse is 47 times in 65 N values
BUILD SUCCESSFUL (total time: 7 seconds)
```