SureDelivery Cryptographic System

City University of Hong Kong
Department of Computer Science

BSCCS/BSCS Final Year Project Report 2005-2006

05CS020

Project Title
SureDelivery Crypto System

(Volume 1 of 1)

Student Name: Liang Li
Student No.: 
Programme Code: BSCCS / BSCS2 **

Supervisor: Prof. Duncan Wong
1\textsuperscript{st} Reader: Prof. Ngo C W
2\textsuperscript{nd} Reader: Prof. Florence Mong
Student Final Year Project Declaration

I have read the project guidelines and I understand the meaning of academic dishonesty, in particular plagiarism and collusion. I hereby declare that the work I submitted for my final year project, entitled:

________________________________________________________________________

does not involve academic dishonesty. I give permission for my final year project work to be electronically scanned and if found to involve academic dishonesty, I am aware of the consequences as stated in the Project Guidelines.

Student Name: ___________________________   Signature: ___________________________

Student ID: ___________________________   Date: ___________________________
Abstract

Spam is becoming an increasing large problem nowadays. It is annoying and inconvenient or even worse in many cases. This project presents the design and implementation of a novel solution to the problem. Based on statistics, most spam is from those untrusting parties to legitimate email servers. To eliminate spam in this respect, the key idea for this project considers eliminating email spoofing. Spam can be eliminated significantly if all the emails are authenticated. My approach applies cryptographic tools: ID-Based Signature Scheme, to authenticate email before it reaches the user inbox. Moreover, in case, spammer is a legitimate user, Email Quota is proposed to prevent spammer sends email as legitimate user called inside attack. The proposed solution is designed on top of the existing anti-spam technologies that unauthorized messages are eliminated. Existing anti-spam technologies can still be applied to further improve the performance.

The implementation is a complete email system including: Email Client, Email Server and Key Management Server which ensures the security of the authentication process. Feasibility and practical issues are also demonstrated in the system architecture.
Acknowledgement

I would like to acknowledge the following people for their roles in helping and supporting me accomplish this project.

- Prof Deng XiaoTie
- Prof Duncan Wong
- Mr. Yang GuoMin
- Miss. Hu ChunYan

I am so grateful for your supervision, feedback and encouragement. Thanks for dedicating your precious time on my tough work.

Finally, I owe a great deal of gratitude to all my friends, who is supportive to me the whole year and had to put up with my complaints about my workload. Thanks to all of you who helped me get through a very long and tiring year.
1. INTRODUCTION

This project is considering a novel solution for the problem of the large amount of unwanted emails received, which lowers the aggregate value of emails as a communication medium. The problem is commonly known as “Spam”.

Current solutions to the Spam problem involve analytic algorithm, known as content filters, as well as some network control technologies, known as access filters. Some other technologies and policies are also proposed to reduce the damage of Spam as much as possible. However, none of the attempted solutions works perfectly.

Content filters eliminates spam according to whether there are some specific spam terms in the content. Those spam terms are called content filter rules. In content filter, the content filter rules dominates the result of the filters. A large number of content filter rules are needed to identify the characteristics of existing spam. However, spammer is still changing the characteristics of their spam email, which will progressively increase the number of content filter rules without an performance improvement. Even some smart machine learning methods is used, e.g. Bayesian learning filter, it still cannot precisely predict the spam characteristics. The reason is that using content filters results in an ideal definition (a set of rules) of spam to distinguish what email is a spam, which is unachievable. Content filter is good in the sense of specific kinds of spam, e.g. advertisement. However, it does not work perfectly and meanwhile a lot of computational effects are put into it.

Access filters is used by reject email connections by checking any faked information inside the network header. In this kind of filters, the fundamental technology used is based on network flow control, which is completely unrelated to email (TCP/IP level VS Application level). Good email might be filtered and spam might be delivered, which cannot even be traced. Furthermore, the access filter is applied to Internet Service Providers (servers) that all potential users are block, even though the majority are good guys. Access filters solves the common spam technique: Open Relay along with some side effects.

In this project, I start with the idea of eliminating email spoofing. Rather than the traditional anti-spam solution, I measure the authenticity of the email by adding and verifying a ID-based signature, which uses user identity (email address in this project) as the public key. A model is built and implemented supporting Offline Email Authentication and Outbound Email Control, which works on server side and client side respectively. On server’s side, Offline Email authentication is needed to ensure the legitimate source of the email. No email spoofing exists in
this model, which is the most popular techniques the spammer uses. The server’s side authentication is designed as offline to meet the nature of email communication (no real-time interaction). On client’s side, Outbound Email Control are used to restrict the certain amount of emails to be sent, Email Quota for simplicity. It tries to cut spam from their source. The server will re-issue the Quota back to the client when the emails is received and be regarded as valid. The purpose that the client side model is used is to prevent insider attacks in the model.

The central contribution of this project is the strategies and algorithms used to eliminate spam by eliminating email spoofing. This model is built on top of the traditional anti-spam solutions, that there is no collision applying both of them. Moreover, this model is light weighted compared to the traditional anti-spam email servers in the sense of less high computational filtering algorithms are replaced by light-weight cryptographic signature schemes. Good performance is expected inside the system construction.

My consideration also involve how to make a more holistic approach to the spam problem and how to make email more powerful not only in the sense of communication, such as E-Stamp\(^1\). Such an approach is based on the model I have built with some more attractive features and more security coordination with the existing email system. Some interesting and attractive improvement is also proposed in the later chapter in this report.

### 1.1 Report Structure

This report is divided into six chapters. The first chapter proceeds the complex nature of the spam problem and addresses the root of it. Based on this analysis, the idea, cutting spam off from the source, is proposed. I show how the idea works based on the analysis of Spam problem.

The second chapter reviews the existing anti-spam solutions and the cryptographic algorithms used in this project. For anti-spam solutions, it is divided into 2 parts: technical and non-technical. Details are given for technical solutions, however, I present the brief description for the non-technical solutions.

The third chapter concerns some security issues of the model. Dealing with cryptographic method, to ensure the security of this model, security analysis are discussed. It is the basic foundation that to build a secure system. Secondly, performance issues are also discussed since

---

\(^1\) E-Stamp: Electronic Stamp, is proposed recently as personal certificates. Different definition can also be found according to the usage. Here, we talks about use E-Stamp as credit.
email is a large scale system that how to adopt it into practice is one of the key factors needed to be concerned.

The fourth chapter introduces the system construction for this project. First of all, the system design overview presented by introducing different components in this model. Then a brief working flow is shown to give out a clear picture of the system run down and the collaborations between different components. Lastly, each component is presented in detail about its design and functionality.

The fifth chapter compares my solution with the access filter and the content filter anti-spam solution. Two aspects is compared: the factor influences the solution and the overall performance for these three solutions.

The final and sixth chapter discusses making use of the model in practical and some interesting further improvement. As I mentioned, the model is quite different from the traditional ones, I give out my suggestion of combining this model with the existing email infrastructure. Some interesting further improvements are introduced such as how to realize the E-Stamp ideas with small amount of modifications, some suggestion on improving the efficiency of the email system. Again, the intended central contribution of this project is the novel anti-spam idea using in this model.

1.2 Reading this report

This report is intended to write in a linear consequence. I recommend the reader first to get a brief idea of difficulties of solving Spam problems and the root of the problem I figured out. Secondly, the key idea is presented which is one of the most important parts of this project. The purpose of the third part is for the reader’s reference. For sophisticated reader in this field, you could skip the background literature of the existing anti-spam solutions and cryptographic primitives. Then the system construction is presented in sequence that a clear picture of the whole project is intended to present to the reader. The last chapter mainly focuses on the further improvement. All kinds of suggestion are warmly welcome and I am really appreciated to receive any comments.

1.3 Define Spam Problem

Before going through all kinds of anti-spam solutions, a definition of spam is needed for the purposes what kind of emails should be recognized as spam. However, the current situation is not so optimistic because: [5]
There are many varying and conflicting definitions of spam currently in use. Making sense of this reality can illuminate important aspects of spam.

Even though there are so many definitions for spam, no one could claim it as the most precise and accurate one. The only reason is that the definition of spam varies from people to people.

Our ideal definition of spam should be in a measurement of \textit{preciseness} and \textit{accuracy}. Such kinds of problems happen is due to the existing anti-spam solutions we are using. Generally, based on the current anti-spam solutions, four kinds of definitions are used:

- Spam as defined by individuals when dealing with the spam problem on a daily basis. Such definitions are personal and may turn on what email individuals consider unwanted and unsolicited. What is defined as spam for a particular individual depends on that individual’s preferences, and is codified into his or her mental processes.

- Spam as defined by spam content filters. Such definitions are used primarily by Internet Service Providers (ISPs) and Email Service Providers (ESPs) to filter spam messages for their customers. The definition in a particular instance is the algorithm used by a given spam content filter.

- Spam as defined by email sent that was illegal to send as determined by existing laws within the jurisdictions from where and to which it was sent. Such definitions are used by governments, ISPs and ESPs to prosecute spam senders. A definition in this context would be the language of laws currently on the books in particular jurisdictions.

- Spam as defined by email that is prohibited by private contract between email senders and their ISPs or ESPs. Such definitions are used by ISPs and ESPs to prohibit spam senders from using their resources to send spam. Definitions in this context would be the language of particular Acceptable Use Policies (AUPs) that were agreed to by particular users of ISP and ESP systems.

When an email comes in, based on the definition above, it could be regarded as spam by the first definition claimed with very high preciseness and accuracy. However, the same email looks fine to the second definition. E.g. ACM newsletter is quite informative and useful so that no content filter will discard such kind of email, however, a marketing professor might claim such kind of email is spam. Actually, the difficulties are different measurements are used for different emails at particular situations. In other words, \textit{what is spam to someone is not necessarily spam to someone else}.

The key reason for causing such problems is that all the definitions are based on the email content. However, it is quite sensitive to different people and very hard to measure since the content could vary according to your strategy. On the other hand, content is not the only measurement that we can access an email as a spam. Regardless what content inside a spam, the only reason that we
are receiving more and more spam is because over 99% spam are those unsolicited business emails which could benefit the spammer due to the cheap cost for sending emails.

In a spammer’s point of view, it is not that difficult to send a bunch of emails. However, the techniques the spammers use is mainly try to get themselves hidden well. The well known two are: Open Relays and Open Proxies.

- Open Relays: Spammers often try to obfuscate the point of origin of their mail. Sometimes their mail is fraudulent. Open relay is the technique spammers use for hiding their real address from being revealed. They try to send emails through some email server else.
- Open Proxies: This technique is quite similar with Open Relay. However, it use proxy server instead. For open relay, it is possible for a relay server detect the email is passing through it as well as the source of the email. However, proxy server only deals with Web Request. It is very hard to detect the real source of the email sender. [4]

A common characteristic for those spam techniques is that they all try to spoof the recipient email server meanwhile the email is untrusting. Based on this point, I suggest a new measurement: authenticity of emails. Only two values are in this measurement: true and false. All emails with false measurement will be regarded as untrusting emails, which means with high probability, it is a spam. This measurement is to check original source of the email. Any email using a faked address or a unknown address will have a false authenticity measurement and vice versa. The measurement is bounded with the email itself, no amendment is allowed. Again, as defined in the measurement, spoofing is no longer a possible way to send spam emails. However, for any inside attacker, some strategy is also designed. Details will be given later.

1.4 Solution - Making Email Spoofing Difficult

In this model, the authenticity measurement is used. The major objective of this model is to make spoofing as difficult as possible. To implement the authenticity measurement, I use identity-based signature built on emails (content, sender and meta data) . The signature ensures the authenticity and data integrity. Once the user signing key is legitimately issued. The signature should be considered as valid which implies true value in the trustiness measurement. In this model, a trusted key management server is needed. The key management server is a distinct server of all the email servers. The security of this server is ensured that makes the functionality of the whole model. In case, any abnormal email activities are found in the model, all kinds of activities are traceable to the key management server. Proper amendment will be done according.
Besides, insider attackers are also considered in this model. Once some legitimate users are compromised, a client side solution will work. This solution will actively control the number of outbound emails of each user under reasonable configuration, while legitimate users remain enjoying the convenience of email communication without noticing the outbound control behind the system. However, compromised users (spammers) are unable to send a large quantity of emails. You could also regards this client side solution simply as *Email Quota*. 

---

1. *Signing Key*: A key used to sign a message. Usually, it is only know to the user who signs the message
2. BACKGROUND LITERATURE

Based on the previous chapter, I characterize some of the nature of Spam problem. In the first part of this chapter, several popular existing anti-spam technologies are reviewed. What covered in this part is not an evaluation of all the technologies but a compendium of how it works and the special features he focus on the Spam problem.

The second part of this chapter is some cryptographic primitives. All the cryptographic algorithm I used are described, which includes Adi-Shamir Identity-Based Cryptosystems and Signature Scheme\(^1\) and AES Encryption Algorithm. The detail mathematical security proof are given as references in the Appendix chapter.

2.1 Current Anti-Spam Solution

Nowadays, the growing awareness of Spam damage results in different kinds of anti-spam solutions. Some of them are statistical algorithm and some are law enforcement. Generally, it could be grouped into 5 categories: Content Filtering, Access Filtering, Identifiability, Regulation and Enforcement Capacity. The previous three are built on some different kind of computer knowledge and the latter two try to solve the problem in the respect of law field. Detail examples are given on the first three categories and a brief description is given on the last two categories. Each of these solutions offers some relief to the spam problem, but some restriction also exists for most of them. [5]

2.1.1 Content Filtering

The most well known anti-spam solution category is content filtering. In this category, solution types examine and rate each message as to whether their content appears to resemble spam or not. If given a spam rating, then the message may be filtered in one way or another, which could range from immediate deletion to sequestration into another area or folder.

The most common content filtering type is rule-based content filtering. In this type, a set of rules or tests are run against the content of the message. Each test corresponding to some time in the message header or body that is associated with spam to some degree, e.g. exclamation points in the subject line. If a message meets the test, a value associated with the test is added to an overall spam score. After all the tests are run, a final spam score is achieved, which can then be used to filter messages as desired.
Now rule-based content filters are commonly associated with the client email software. Both Microsoft Outlook and Eudora has its own built-in content filters. Also, a lot of third party content filters are provided and open sourced. For particular notes, an open source filter called SpamAssassin is widely used by small ISPs and ESPs. A threshold score is set that all the emails have a higher score than the threshold one is considered as spam.

In addition to the static content filter, there is another kind of well known filter: Bayesian learning filter, which enables the rules-based technique to become dynamic. A model of machine learning\(^1\) is proposed to gradually update the filter for the assessment for a spam. For each message reported as spam, the characteristics associated with that message become more like a spam in the filter, namely increasing the probability that an email is a spam if it contains the similar characteristic. And conversely, for each message that is not a spam, the characteristics associated become less likely contained in a spam.

A final type in this category is the black list scheme that prevent certain suspicious senders from sending emails. This scheme works based on the email address the user uses. For those spammers repeat using the same address, abnormal activities will be blocked. It is easy to implement, however a large scale of database is needed.

To sum up, content filtering has the most academic activity as well as commercial activity. Certain percentage of spam are eliminated from this category. To date, it is one of the most favourite solutions to the industry.

**2.1.2 Access Filtering**

The second category of anti-spam solution is access filtering. In this category, solution types attempt to block spam senders from accessing email servers. This category is the realm of Internet Service Providers (ISPs) and Email Service Providers (ESPs). Differed from the black list solution in the previous category, it is most based on the network level rather than user level.

The first scheme came out is called *Best Practice*. Generally, it attempts to reduce the possibility of spammers using third party IP addresses to send spam. Server strategies are included such as shutting off consumer machines from sending bulk email in general or creating automated complaint systems to identify violators more quickly in order to shut them off. Estimated 35% spam are eliminated by this scheme. Also web-based email are recommended in this scheme that the spammer will not be able to send bunches of email through the web interface.
The second and also the most widely deployed access filters are those of server white list or server black list. This is analogous to the client black list mentioned in the previous category. But these list are on the server level. All those big ESPs have their own white lists and black lists such as Amazon, Microsoft, AOL and Yahoo. The two most well known lists are the Open Relay Database (ORDB) that tracks unprotected mail servers and the Realtime Blackhole List (RBL) that tracks known spam sources. When an email server uses a black list, it generally prevents email from originating right at the beginning of the SMTP exchange and refuse the connection.

The final type of access filtering is the reverse DNS test. DNS stands for Domain Name System that used to map the IP address to the domain names. E.g. www.yahoo.com will map to 128.0.0.1. This system is simply the reverse of DNS and then translate an IP address to a domain name. When a SMTP connection is established, the domain name received will be translated into IP address according to the Domain System. If the originating IP address does not match the IP translated, the email is regarded as suspicious.

2.1.3 Identifiability

The third category of anti-spam solution types is that of identifiability and also the origin of my idea. In this category, solution types attempt to more reliably identify the senders of email messages in general. The idea is that if senders can be reliably identified, then:

1. Spam senders can be more easily traced and thus dealt with
2. Content and access filtering can be improved because they can be based on more reliable sender information.

The first solution came out is the IP address authentication. It inherits the idea of Reverse DNS access filter that they publish some information in DNS that the email server or client can then use to authenticate the sender via their IP address. E.g. the sender’s email address can be used. In this case, the domain part of the address is first extracted. The sender would have previously published in their public DNS record which IP addresses are authorized to send email for the domain extracted. Those authorized IP addresses are then looked up through DNS. If there is a match, then the message passes; otherwise, the message is considered as suspicious.

In July 2005, Cisco Systems, Yahoo! and partners proposed an anti-spam technology, DomainKeys Identified Mail (DKIM), that focuses on identifying forged email addresses.

With DKIM, which relies on public key cryptography, a digital signature is attached to outgoing email so recipients can verify that the message comes from its claimed source. The
SureDelivery Cryptographic System

specification calls for email domain owners to create a pair of public and private cryptographic keys. The public key is published in the Domain Name System record, while the private key is stored on a DKIM-enabled mail server. Each outgoing message is then signed, with the signature stored in the email header. On the receiving end, a DKIM-enabled mail server extracts the signature and uses the public key to verify that the signature was generated by the sending domain.

2.1.4 Regulation & Enforcement Capacity

The last two categories are non-technical solutions. Public law as well as private contracts are announced to prohibit spam techniques. Those big ISPs and ESPs try to take their spam problems to court via civil suit. The Regulation solution is based on the existing lawsuits including consumer protection and computer fraud codes, which sending spam often violates. The Enforcement Capacity solution is the activities that push the execution of the Regulation. Most of the big ESPs have played an active role in these activities, however, still few cases were dealt with under such kind of lawsuits.

2.1.5 Existing anti-spam solutions summarized

To conclude the existing anti-spam solutions, there are a myriad of approaches including technical, legal and policy oriented approaches. Content filtering and access filtering have an attractive practical result and widely deployed. Identifiability presents a great idea on top of the previous two categories, but not widely implemented. Some other efforts are also made by people to reduce the damage of spam as much as possible. However, none of them really works perfectly and further improvement is still a hot research topic.

2.2 Cryptographic Primitives

This part gives a brief review of all the cryptographic algorithms I used: Adi Shamir ID-based signature scheme and AES symmetric Encryption/Decryption scheme.
2.2.1 Adi-Shamir ID-Based Signature Scheme

2.2.1.1 Overview

Adi-Shamir ID-Based Signature scheme is a public key cryptographic scheme, which differs from the traditional Public Key Infrastructure (PKI). No private and public key exchanging is needed, no public key directories is stored and no third party exists between the communication. Only one trusted key generation centre exists whose role is to issue the private key for a specific user the first time he join the system. No further communication between user and the key generation centre is needed. The centre is useless after all the users are joined. To sign a message, the issued private key is used to generate the signature. However, the verification procedure is totally independent from another other party. The public key used to verify the signature is just the identity of the user. [1]

The difference between ID-based signature scheme and traditional public key signature scheme is the public/private key pair generation. Instead of generating a random pair of public/private keys and publishing the public one, the user chooses his name and network address as his public key. In this model, it would be the email address. Such an identity uniquely identifies the user in a way he cannot later deny, and it is already known to other parties. The corresponding private key is computed by the key generation centre and issue to the user when he joins the system. To sign a message, the user uses the private key issued by the key generation centre to encrypted the message associated with his identity (public key) as the signature and pass it to the receiver. When the receiver get the message, first he extract the identity from the message. Then the user identity is used as the public key to verify the signature, which is his name or network ID.

The ID-based scheme resembles an ideal mail system. If you know somebody’s name and address you can send him messages that only he can read, and you can verify the signatures that only he could have produced. It makes the cryptographic aspects of the communication almost transparent to the user, and it can be used effectively even by laymen who know nothing about keys or protocols.

2.2.1.2 Scheme Implementation

There are generally three algorithms used: Key generation, Signing and Verification. The key generation algorithm is based on the RSA\(^1\) encryption/decryption system. The other two are built on top of the discrete logarithm problem. Here is some terminologies used in this implementation:
m is the message
s, t is the signature
i is the user's identity
n is the product of two large primes
e is a large prime which is relatively prime to $\phi(n)$
f is a one way function

The parameter $n$, $e$ and the function $f$ are chosen by the key generation centre, and all the users have the same $n$, $e$ and the same algorithmic description are specified. All these information can be public except the factorization of $n$. The factorization of $n$ is only known to the key generation centre which could also be regarded as the master private key. The key generation algorithm is to generate a pair of large primes which is roughly equal in length and key it as the master private key. The signing and verification algorithm is formed as follows:

Signing Algorithm:

$$s = g \ast r^{f(i,m)} (mod \ n)$$

Verification Algorithm:

$$s^e = i \ast t^{f(t,m)} (mod \ n)$$

The user private key $g$ can be easily computed by the key generation centre using the RSA decryption function:

$$g^e = i (mod \ n)$$
$$i^d = g (mod \ n)$$
$$e \times d \equiv 1 (mod \ \phi(n))$$

Since this signature scheme is a non-deterministic scheme that every time the same message comes in, a large number of possible signature could be generated. $t$ is generated by a random send $r$  such that:

$$r^e = t (mod \ n)$$

So the verification condition could be written as:
SureDelivery Cryptographic System

\[ s^e = (g \cdot r^{\phi(t,m)})^e = g^e \cdot r^{e \cdot \phi(t,m)} = i \cdot t^{f(t,m)} \pmod{n} \]

Only user public key and the signature is needed to verify the signature.

2.2.2 Advance Encryption Standard (AES)

Compact E-Cash presents efficient off-line anonymous e-cash schemes where a user can withdraw a wallet containing \(2^l\) coins each of which she can spend unlinkably. Our first result is a scheme, secure under the strong RSA and the \(\gamma\)-DDHI assumptions, where the complexity of the withdrawal and spend operations is \(O(l + k)\) and the user’s wallet can be stored using \(O(l + k)\) bits, where \(k\) is a security parameter. The best previously known schemes require at least one of these complexities to be \(O(2^l \cdot k)\). In fact, compared to previous e-cash schemes, our whole wallet of \(2^l\) coins has about the same size as one coin in these schemes. Our scheme also offers exculpability of users, that is, the bank can prove to third parties that a user has double-spent.

We then extend our scheme to our second result, the first e-cash scheme that provides traceable coins without a trusted third party. That is, once a user has double spent one of the \(2^l\) coins in her wallet, all her spendings of these coins can be traced. However, the price for this is that the complexity of the spending and of the withdrawal protocols becomes \(O(l \cdot k)\) and \(O(l \cdot k + k^2)\) bits, respectively, and wallets take \(O(l \cdot k)\) bits of storage. All our schemes are secure in the random oracle model.
3. Security and Performance Concern in System Design

In order to implement a cryptographic system, security is the most important issue we should concern. Otherwise, nothing is meaningful if the cryptographic basis is broken. Moreover, as we know, the huge scalability of email is one of major oracle considering a practical email system. Performance issue is also very important in our system. To design a secure and efficient anti-spam solution, there are several security and performance issues we concern:

3.1 Security Concerns

3.1.1 Unforgability of Signature

The first and foremost issue is that no faked signature can be verified as legitimate. The issue that we discuss here is an attacker that attempts to generate a faked signature to send email rather than the user itself is the attacker. The inside attack is considered later. In the system, only the sender who has got the private key and the trusted key generation server who has got the master private key can sign a legitimate signature. The key issues for this concern are

1. How to store the private keys securely
2. How to distribute the user private key in secret.

In order to ensure the security of the two issues above, there are two components that we already trust to perform the key storage and key delivery: email server and DNS server. The email server is the one who stores the private key. Since there is another strategy to prevent the insider attack, the private key is secure if the server perform legitimately. Another reason why it is put on the email server is that email signing and verification is performed on the email server, it is efficient to store the private key (signing/key) on the same server. No data communication is needed. As to the master private key, since it is only known to the key generation server and never be public, the key generation server is the best place to store the master private key.

To deliver the user private key securely, I suggest to deliver the key from the DNS server to the user mail inbox. Since the user is the one who should know the key, delivering the key to the mail inbox is reasonable. However, the reason we choose DNS server instead of the key generation server should never be public since the master private key is stored. So I choose another trusted component DNS server to be the one delivers the user private key. DNS server is already used to do some authentication job. One anti-spam solution called reverse DNS authenticate the email sender with the claimed domain name in the email header with the corresponding domain name of his IP address. So we can see that DNS server is one of the trusted party in the traditional email infrastructure. It is reasonable to deliver the user private key.
through the DNS server to the user mail inbox.

3.1.2 Double Spending

One of the important security feature that we should concern is double spending. It means that every email could be only verified as valid once, namely the signature is legitimate for only once. Any identical email could only be sent more than once is not a legitimate one. To prevent double spending in this system, the focus is the signature scheme we use. A random number is picked and associated with the email message every time it is sent. The random number is chosen widely that distinguish every email even all the other content is the same. Every envelope can only be attached to one email at the same time. Using one envelope more than once while the envelope is not released is regarded as double spending. Spammers can easily send spam by just repeating send one valid email many times if this security feature is not achieved. In our system, it is one of the most important security issues to be concerned.

3.1.3 Repudiability and Traceability

In traditional email, one who is a legitimate user of one email server could send an email claimed from another user who belongs to the same server, namely we called insider attack. By using the Email Quota, the inside attacker is not able to send spam efficiently. In case, some legitimate user still send spam by using his own quota, the one could report it as spam. When the system notices the spam email, the sender cannot deny that he sent it. This is called repudiability. In order to achieve the repudiability, there is a property needed in our signature scheme called traceability. For simplicity, this means when a signature is generated, the user private key can be revealed by using the master private key, which is associated with the user identity that is only known to the user himself. This property is used by the system administrator that users are not able to check the repudiability. Only if one email is reported as spam, the administrator will find the source and deal with this situation.

3.1.4 Data Integrity for Email Quota

As mentioned in the previous chapter, Email Quota is the oracle made that a legitimate user is not able to send large amount of emails efficiently. It should not be revealed to the sender. In this system, it is designed to stored at the windows registry encrypted. The client side software is considered as a black box that each time the sender sends an email, the black box is used. The decryption and re-encryption are all done by the black box. The assumption I made here is that
the user is not able to de-compile the client side software which has the decryption function.

3.2 Performance Issue

3.2.1 Performance Bottleneck at Email Server

As we know, email communication is so widely used that huge communication request are raised to the email server. Due to this reason, Email server is very likely to be the performance bottleneck since new computational effort are added. So the performance of the email server becomes a serious concern in the system design.

As to this concern, it could be divided into two parts: Signing message and Message Verification. First, for the verification, I trust that it is not able to cause serious performance problem or some attack: DOS attack\(^1\). The reason is that our signature scheme added on the email server is an offline solution. Offline means:

1. Each email may be operated independently
2. Operation is not necessary be conducted immediately on demand of the communication (SMTP connection)

Verification is conducted when the SMTP connection is finished. No email connection is needed to conduct the Message Verification. Furthermore, the parameters used in the verification are the user public key and the master public key; the former one is already contained in the email message, the latter one is stored on the email server, which means no additional communication is needed. From the above two points, Verification is totally independent and can be done after words. No side effects is on the email communication, thus there is no performance bottleneck for verification.

Signing is a little different from the Verification. Signing is also an offline operation in the respect of SMTP connection. It is conducted right before the SMTP connection is established and the user private key and message are already placed on the server. However, the problem happens that the signing operation is associated with the sending request from the email client. There is no intermediate interface to intercept the message and postpone the request. The worst case is this might cause a serious performance problem if numerous users send email

---

\(^1\) DOS attack: It is an attack through internet that floods it with so many additional requests that regular traffic is either slowed or completely interrupted simultaneously. However, the signing function that requires a huge computational resources. Even all legitimate users send email simultaneously, the resource used is still reasonable. But strategy should be considered to avoid anonymous DOS attack.
3.2.2 Performance Bottleneck at Key Management Server

Another potential busy traffic may happen at Key Management Server since all the user keys are under its management. The server is mainly responsible for two things: generation/issuing user private key and issuing Email Quota to the users. The first task cannot be the performance bottleneck. The reason is every user private key is only generated and issued to the user the first time he joins the system. As discussed above, no further communication is needed.

Unfortunately, the performance bottleneck may appear when the server issues the Email Quota to the users. This operation happens quite often because that every user needs to more quota to send email. Moreover, quota is issued more often than the number of the demand of lack quota in order not to affect the performance for sending email. Communication traffic grows at this point. The good thing for this problem is that issuing quota is not necessarily be a immediately process and email quota is not supposed to run out very soon. Batch process at non-busy traffic hour can be a good alternative to solve this problem. The server accumulates the quota issues for different users on the same server until the non-busy hour and issue them to their email server as a whole. It is fast since a number of issues are sent together rather than individually. Furthermore, non-busy traffic hour ensures there is no traffic collision during the transmission.
4. **SYSTEM ARCHITECTURE AND IMPLEMENTATION**

The system is the implementation of the model proposed in the introduction chapter. In this chapter, firstly, an overview is given on the system structure, functionality and the general relationship between different components. Secondly, a flow chart shows the complete run down of an email communication in this system as well as the distinct scenario. Finally, I introduce the responsibility and implementation detail for different components.

4.1 **Overview**

The complete system consists of three components:

- Email server with signing/verification functionality
- Key management server that is responsible for generating and issuing user private key and email quota
- Email client outbound email control is in charge of check and update the email quota a client used.

![Figure1. System Structure](image)
A detailed system architecture diagram is shown below. The email client and email server are built based on the traditional email architecture except adding two modules in between. The *email client outbound email control* performs right after the email client issues a sending request to the email server to send an email. Every time an email is sent, the quota is updated by the outbound email control and a warning is popped if there is not enough quota. In this system, Email Quota is stored in the form of a Windows registry value encrypted with the Advanced Encryption Standard (AES). It is considered as a black box oracle that the user has no idea about what is inside including the quota by assuming that nobody could decrypt the Email Quota message without the key management server.

The email server is used to do the signing/verification. Signing is done right before and SMTP connection is established. A signature is generated and added into the email in the form of a *Multipurpose Internet Mail Extensions* (MIME) header. The verification is done right after an incoming email is received (end of the SMTP connection). The specific MIME header is checked. If there is a signature exists in the header and is legitimate as well. The email is regarded as valid. Otherwise, it is discarded. The whole signing/verification process is dealt between email servers, no other entities are involved. A cryptographic primitives for Adi-Shamir signature scheme is also implemented on this server. It is regarded as a system library called by the email server.

The key management server is a distinct entity in the whole system. The major responsibilities are issuing and generating the user private key used for signing, delivering the master public key for verification, storing master private key and issuing email quota to the client outbound email control. It must be a trusted party in the whole system. In this system, the key management server is put on the same machine with the email server, the only difference between the code is that it is never public. It is because the limited resource that only one machine can be available to use, it is the remaining alternative to prototype the system functionality. The assumption made here is that the email server is never compromised. Further improvement is mentioned in the later chapter. No network communication is involved in key management server.

Moreover, it shares the cryptographic primitives with the email server implementation. Delivering the master public key in this system is actually not a real “delivering” process. The master public

---

1. SMTP: SMTP stands for Simple Mail Transfer Protocol. The standard e-mail protocol on the Internet and part of the TCP/IP protocol suite. SMTP defines the message format and the message transfer agent (MTA), which stores and forwards the mail.
key is generated is stored as a part of the email configuration file and the master private key is stored as a txt file distinctly. The master public key is loaded to the email server when it is started, but not the master private key which is assumed not known to the email server. There is a risk that if the email server is compromised, the whole system is also compromised.

Another role that the key management server is to issue the email quota. The Client Outbound Control is only responsible for reducing the email quota when email is sent. The key management server is the one who issues new email quota to the user used in the Client Outbound Control. The Email Quota is generated in form of encrypted message stored in form of window registry file and send to the user. When the user receives the file, the new registry value is added into the registry table that the Outbound Email Control can interact with.
4.2 Working Flow

4.2.1 Email Working Flow

In this part, a whole email life cycle is present. A detail data flow chart is showed below:

![Email Communication Flow Chart](image)

**Figure 2. Email Communication Flow Chart**

Step 1. Compose email – Email Client

When a user has composed an email, an email sending request is fired as an event and this event is caught by the Outbound Email Control and the Outbound Email Control is activated.

Step 2. Check Email Quota – Outbound Email Control

After the Outbound Email Control is notified by the sending request event, the Outbound Email server first retrieve the encrypted quota from the window registry table and decrypt it.
Secondly, the recipient list is extracted into distinct email address. For each address, the decrypted email quota is decreased by one. If email quota is less or equal to zero, the sending request to the email server is discarded by the Outbound Email Control that the current email is not sent. A warning message is informed to the user that “More Quota is needed”. Otherwise, the sending request and the composed email are forwarded to the email server, which is exactly the same as the traditional email system. The user is not aware of what happens inside.

Step 3. Signing the email – SMTP server

When the email server receives the new email message, the server will first sign the message and generate a signature based on the message content, sender email address (User Identity) as well as some meta data of the message by using the user private key stored in the server key directory. Then the generated signature is added into the message in the form of a MIME message header named: “SDSignature”. Lastly, an SMTP connection is established with the recipient email server that the message is delivered.

Step 4. Verify the email – POP3/IMAP server

When an email first reached the email server (incoming server), it is verified before put into the recipient mail inbox. Firstly, the server extracts the sender email address as the user identity and derives the user public key from it. The signature is extracted from the MIME header named with “SDSignature”. With the user public key and the master public key which is already stored at the server, the signature can be verified as legitimate or not. For those with illegitimate signatures, they are discarded as untrusting emails. If there is no signature inside, the email is also discarded as illegitimate. Only those emails with legitimate signatures are put into the corresponding recipient’s mail inbox.

4.2.2 Signing/Verification Data Flow

This part indicates a detail data flow how a signature is generated based on email and how the signature is verified. The following chart gives a general picture what data is involved how signing/verification works.
Signing

When a message comes in, the user identity is extracted from the message. According to the user identity, the corresponding user private key is got from the key directory which stores all the user private keys on that server. Then, the signature is supposed to be generated based on the key, message and some random factors.

Firstly, a large random number $r$ is generated. It is one of the secret information for that particular signature in order to prevent replay attack. The first part of the signature is generated based on the random number $r$ and the user public key $e$ by taking

$$r^e = t \pmod{n}$$

Secondly, the first part of the signature $t$, the message and its meta data is fed into a hash function $f$ to generate a hash value. To generate the second part of the signature $s$, calculate

$$s = g \ast r^{f(t,m)} \pmod{n}$$

Lastly, the whole signature is just the concatenation of the two signature generated with symbol “-”.

The signature is generated and added into the email as a MIME header, and passed to the recipient along with the message.

Verification

When a message is received, the first step to verify the message is to extract the sender’s email address as his user identity. Then signature is divided into two parts by the splitter “-”. Then the signature can be verified by

$$s^e = i \ast t^{f(i,m)} \pmod{n}$$
Where $i$ is the transferred number of sender’s email address, $f$ is the hash function used same as the signing procedure. If the equation established, it is a valid signature. Otherwise, it is a false one.

### 4.3 System Components

Generally, the system is divided into three distinct components: Email Client, Email Server and Key Management Server. In this part, component architecture and detail implementation is present.

#### 4.3.1 Email Client

Email Client is the entity who sends and receives emails. Differed from the traditional email client, the idea called Email Quota is used to control the number of emails user can send. One more module called Outbound Email Control is implemented to manage the Email Quota. *Microsoft Outlook 2003* is used as the Email Client software delivering and receiving emails. Outbound Email Control is implemented as a *toolbar* inside Outlook.

![Outlook Toolbar](image)

*Figure. Outlook Toolbar*

- **Functionalities:**

  1. Anti-Spam on/Anti-Spam off

     This function is in charge of the outbound email control – Email Quota Control. When the button is switched to “Anti-Spam on”, before every email is sent to the email server, Email Quota is checked. Only user with enough quota is able to send email to the email.

  2. White List

     This function is in charge of the outbound email control – Email Quota Control. When the button is switched to “Anti-Spam on”, before every email is sent to the email server, Email Quota is checked. Only user with enough quota is able to send email to the email.

  3. Check

     The major task of this function is to check the remaining quota the user has. It simply open the registry table and get the decrypted email quota. Then user will be informed about the quota he has.

  4. Options
The configuration of this toolbar is set by this function. The configuration includes Toolbar display, the cryptographic algorithm used to encrypt/decrypt the email quota.

- Code Architecture

![Toolbar Architecture Diagram]

**Figure. Toolbar Architecture**

### 4.3.2 Email Server

Generally, email server is a mail transfer agent (MTA) responsible for both sending and receiving emails. In this project, I use the Java Apache Mail Enterprise Server (a.k.a. Apache *James*) which supports SMTP and POP3 Mail server and NNTP News server. It is a complete and portable enterprise mail engine solution and is based on currently available open protocols. To authenticate emails, two new modules and one cryptographic library are added into the server. The following chart is the architecture of the email server.

- **Functionalities:**
  1. Mail Deliver Agent (MDA) – SMTP Server
This function is in charge of the outbound email control – Email Quota Control. When the button is switched to “Anti-Spam on”, before every email is sent to the email server, Email Quota is checked. Only user with enough quota is able to send email to the email.

2. Mail User Agent (MUA) – POP3 Server

This function is in charge of the outbound email control – Email Quota Control. When the button is switched to “Anti-Spam on”, before every email is sent to the email server, Email Quota is checked. Only user with enough quota is able to send email to the email.

3. Server Management Tool – Remote Manager

This function is in charge of the outbound email control – Email Quota Control. When the button is switched to “Anti-Spam on”, before every email is sent to the email server, Email Quota is checked. Only user with enough quota is able to send email to the email.

4. Sign email

The major task of this function is to check the remaining quota the user has. It simply open the registry table and get the decrypted email quota. Then user will be informed about the quota he has.

5. Verify email

The configuration of this toolbar is set by this function. The configuration includes Toolbar display, the cryptographic algorithm used to encrypt/decrypt the email quota.
4.3.3 Key Management Server

Key Management Server is the trusted entity to issue and manage user private key and email quota. It is implemented as a C# window application running on a distinct PC.
**Figure. Key Management Server**

- **Functionalities:**

1. **User Information Management**

   Personal user information including the secret key using to encrypt/decrypt the Email Quota can be updated by this function. A database containing all the information is associated with the server.

2. **Issue Email Quota**

   When editing the user information, there is a field named Quota containing the email quota the user has. It can be specifies and generated as a window registry file by clicking the button "Generate Key". The registry file is sent to the user through email and imported to the user local machine later.

3. **Issue User Private Key**

   The user private is generated by the Master Private Key. It is sent to the user through email when it is created. It will not be changed until the master public and private key pair changed.

4. **Generate Master Public and Private Key Pair**
This function enable the server to change their master public and private key pair. For security reason, it is not good to use one key pair all the time. This function is enabled to the Key management server.
5. Comparison on Anti-Spam Technologies

With the system construction, I will examine in more depth how good the solution is. The traditional solution: Access Filter and Content Filter is referred as comparisons. First of all, I will briefly reintroduce the Access Filter and Content Filter. Then two evaluations on solution related factors and performance are given.

Content filters eliminates spam according to whether there are some specific spam terms in the content. Those spam terms are called content filter rules. In content filter, the content filter rules dominates the result of the filters. Generally, there are two kinds of content filters in the point of rules used. The first one is Knowledge-Based. There is a large database storing the spam regulation. When an email comes in, there will be a weight how likely the email is a spam. If it is over a specific value, then it is regarded as a whole. The second one is based on machine learning, e.g. Bayesian Filter. The idea is similar except a machine learning system is used instead of a large database.

Access filters is used by reject email connections by checking any faked information inside the network header. In this kind of filters, the fundamental technology used is based on network flow control, which is completely unrelated to email (TCP/IP level VS Application level). Filters are usually set at ISPs or ESPs that emails are filtered when they just arrived. Access filters solves the common spam technique: Open Relay along with some side effects.

5.1 Solution Related Factors

<table>
<thead>
<tr>
<th></th>
<th>Access Filter</th>
<th>Content Filter</th>
<th>Email Authentication</th>
</tr>
</thead>
<tbody>
<tr>
<td>Content Sensitive</td>
<td>No</td>
<td>Strongly</td>
<td>Partial Reliance</td>
</tr>
<tr>
<td>Network Regulation Sensitive</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>Transparent to User</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Uniform Standard</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Deployment Cost</td>
<td>Low</td>
<td>Medium</td>
<td>High</td>
</tr>
</tbody>
</table>

Table 1 shows the factors that affects the performance for different anti-spam solution. Red represents bad influence. Blue represents good influence. Black represents acceptable influence.

➢ Content Sensitive

Email content is the most important factor that people distinguish whether an email is a spam or not. However, it might not be the best way to distinguish a spam. This measurement works on how important the email content affects the anti-spam solution. For access filter, there is nothing related
SureDelivery Cryptographic System

to the email content since everything is designed for network flow control. Email content is ignored that false judgement is easily made. For content filter, it totally depends on the content just like what people do. However, it is still hard to formalize some rules to distinguish what email is spam. Subjective options usually affects the decision. As a result, too much reliance will cause inaccurate judgment. For email authentication, the solution is designed against email spoofing while the signature is created based on the email content. It combines the idea of access filter and content filter which is supposed to be good.

- Network Regulation

Network Regulation is only used in access filters. It is supposed to be a good method against email spoofing. However, network regulation is not only design for email communication. Security collision may happen for some special regulation, e.g. Virtual Private Network. When there is an inside attack, the access filter is totally compromised.

- Transparent to User

In this respect of view, the first two is never revealed to the users since that it could be easily broken if the rules are published. Highly secrecy is required. However, in the email authentication, the whole scheme is public known to everyone. The only thing kept to be secret is the master private key. Whenever the master private key is save, the whole system is not compromised.

- Uniform Standard

Email is so widely used nowadays. Different anti-spam strategies for different email servers cause different performance, even if they are using the same type of filters. Broadcasting email is the direct victim under this point of view. As mentioned in the previous point, email authentication scheme is totally known to everybody that there is only one standard in the world. There is no such problem exists in the email authentication scheme.

- Deployment Cost

From practical experience, in order to have one standard for a large scale system. The deployment cost is usually huge. And this is the major disadvantage for email authentication scheme. However, if it is widely deployment, spam problem can be solve significantly or even eliminated.
5.2 Performance

<table>
<thead>
<tr>
<th></th>
<th>Access Filter</th>
<th>Content Filter</th>
<th>Email Authentication</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Computational Cost</strong></td>
<td>Negligible</td>
<td>High</td>
<td>Medium</td>
</tr>
<tr>
<td><strong>Result in more Spam sent</strong></td>
<td>No</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td><strong>Filter Existing Spam</strong></td>
<td>Acceptable</td>
<td>Good</td>
<td>Acceptable</td>
</tr>
<tr>
<td><strong>False Spam Judgment</strong></td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td><strong>Email Spoofing</strong></td>
<td>No</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td><strong>Regulation Management</strong></td>
<td>Hard</td>
<td>Hard/Medium</td>
<td>Easy</td>
</tr>
<tr>
<td><strong>Legitimate User Blocking</strong></td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
</tbody>
</table>

*Table 2.* Performance Comparison

Table 2 shows the performance for different anti-spam solution. Red represents bad performance. Blue represents good performance. Black represents acceptable performance.

- **Computational Cost**

Due to the heavy traffic on email server, computational cost is one of the most important performance issues considered here. For access filter, only the originated source is check with the claimed source. It is consider very fast that there is very little computation involved. For content filters, in order to match all the spam terms with the email content, a lot of computational resources is involved especially when an email is long. Besides, in order to adjust the spam terms used, additional computation is also needed. E.g. Bayesian Filter, which is also computational expensive. So for content filter, DOS attack is a serious thread. For email authentication, the computation is based on Number Theory that fixed computational resource is needed for every email. Stable performance is expected on the email server.

- **Result in more Spam Sent**

This problem only appears when the content filter is used. In the spammer’s point of view, in order to avoid the spam sent to be filtered by the content filter, the spam terms used should be changed that the content filter has no idea about the new knowledge. A loop may be formed by changing the spam content and updating the spam terms. However, more and more spam is sent.

- **Filter Existing Spam**

By contrast to the last point, content filter is good at distinguish email as spam if they are sent before. User no longer suffers spam from a fixed source, which is quite common.

- **False Spam Judgment**

False Spam Judgment is a common problem to all the filters. It is an especially serious problem for access filter. Since there is nothing related used in the content, email from relaying server is definitely eliminated. However, it is not the case that the email is always a spam. The service quality
of Email Communication is seriously affected. For content filter, the accuracy is totally depends on the content filter rules the filter has. False Spam Judgment is also made if the rules are not good.

- **Email Spoofing**

Email spoofing is the key problem that access filter and email authentication try to solve. There is no problem for these two methods in email spoofing. However, content filter does not provide any strategy to prevent spoofing. *Phishing attack* is dangerous under content filter.

- **Regulation Management**

Regulation management talks about the strategies used in the solution. As we talked in *Table 1*, access filter and content filter do not have a uniform strategy for their application. It is hard to agree on what kind of email are legitimate. Email authentication does not have such problem. Only one uniform scheme exists.

- **Legitimate User Blocking**

Similar to False Spam Judgment, Legitimate User Blocking means that some legitimate emails are regarded as spam. This is even worse than some spam is considered as legitimate. The problem usually appears in access filter since no related strategy works on email content. However, email authentication fixes the problem.
6. FURTHER IMPROVEMENT

6.1 Fast envelope processing

Cryptographic signature scheme is designed based on Number Theory. Moderate Computation resource is still needed in signing/verification process. The Adi-Shimar ID-Based Signature Scheme is still time consuming compared with the recent proposed signature scheme, in order to improve the system performance, signing and verification process is supposed to be as fast as possible. In order to reduce the computation, elliptic curve cryptosystems (ECC) is a good alternative instead of the traditional public key cryptosystems. The efficiency of ECC is due to the offering of equal security for a far smaller key size when compared with conventional public key cryptographic primitives such as RSA. This reduces processing overhead. Since the discovery of several new applications of pairings on elliptic curves, practical elliptic curve cryptosystems emerges and ECC is getting more and more popular.

Pairings also allow us to realize many new notions efficiently such as identity-based cryptography, short signature scheme and certificateless public key cryptography. These techniques can help us to implement our scheme in a large scale easily. Since in identity-based and certificateless public key cryptosystems, no certificate is required, and certificate management, which is more difficult than choosing or implementing a cryptographic algorithm, can be avoided.

Other techniques such as aggregation techniques and cryptographic hardware accelerators can also help to improve the performance of the system.

6.2 Token Envelope

Token Envelope is proposed that can combine the signature authentication scheme and Email Quota together. Client side software is no longer needed and no user secret key to control the Email Quota. Token envelope means each envelope is represented by one email quota which will play a role to be authenticated. The whole process is a variant of E-cash model. In E-Cash model, Token is the smallest unit a user can spend as his “cash”. Here we consider the number of envelopes the user can use as the number of “cash” he can spend, where all the “cash” are with the same value. Each time a user want to send an mail, a token will be attached to the email regarded as an envelope to ensure the validation of this email.

Token is generated by a delegator which is similar to our central. The account establishment process is regarded as the withdrawal protocol such that all the tokens are
authorized by the delegator and no sender can forge tokens on his own. The send encapsulated emails process is regarded as the payment protocol in e-cash model. The sender need to send his ‘token’ to the recipient to tell that not only the email is valid but he has spent one token as well. This process will ensure that the number of emails the user could send is limited so that all the outbound emails are under control. The Validate Incoming Emails process is the simple operation to check whether the token is valid or not. The email is trusted if the token is checked to be valid and the email will be put into the recipient inbox. The Envelope Release process is regarded as the deposit protocol in E-Cash model. When an email communication is completed, the recipient will give the token back to the central. Differing from E-Cash model, the token will be returned to the sender to authorize the sender a right to send one more email.

The whole model is also considered be have all the security issue as the E-Cash model has. As mentioned above, our approach is going to combine divisible E-Cash and compact E-Cash to maximize the efficiency of our whole process.

6.3 One-Time Signature

The reason why we use signature is that it is one of the best practical ways to achieve off-line feature with high security. The most important feature differs from the traditional signature is the signature can only used once. Generally speaking, the first time the user produces a signature along with a message with his private key should be regarded as valid. However, when the second time the user want to sign the same message with his private key, the signature he produced should be regarded as invalid. In our scheme, it would be a perfect tool to avoid any double-spending attack. When an email is received, the recipient only need to verify the signature associated with the email, if and only if it is valid, the email can be put into the mailbox. However, each signature can only be used once, that means no single email can be sent twice. In this sense, the only way that sender can broadcast emails is to create a number of emails associated with different signatures and no emails can be prone to misuse.

Non-Interactive-Zero-Knowledge (NIZK) would be suitable to use in this feature. We are going to raise some challenge to the sender so that he has to show that it is the message he has sent once without revealing his private information. However, the challenge could be computed based on the email address of both sender and receiver so that it could get a random bit that the sender can not control. Whenever the sender want to use this signature twice, another challenge should be raised. However, with the help of 2 different pieces of private information, the private information of the sender will be revealed so that the violated sender will be blocked.
6.4 E-Stamp

E-Stamp stands for electronic stamp. This idea comes from the traditional postage stamp we use in daily life. Since we already have Email, why not we can make use of E-Stamp. Postage stamp usually measures how expensive and important your mail is. For E-Stamp, similar usage can also be applied, for example, E-Stamp can be used as a measurement to rank how important an email is. This is a very exciting feature that email could have. Whenever we have an accurate measurement on how important an email is, we can simply rank all the emails according to their importance and find out which on are those I might have more interests to read.

Moreover, small scale of business can also fit into the email platform by making use of E-Stamp. For example, email can be used be a much more efficient to do questionnaire. The traditional way is quite time consuming and wasting resources. Email, however, will be quite a low cost medium. In order to attract receive to read the email, E-Stamp is put into the email as credit. Only those who have answers the questionnaire can get the credit in the E-Stamp. However, for those who are not interested can just ignore it. The more credit you put into the email, more chances that receiver sees the email at the top by simply ranking the emails.

The whole model is also based on the cryptographic tools and considered be have all the security issue as the E-Cash model has. Credit is used instead of money is because it is not that sensitive to the real money.
7. CONCLUSION

This project proposed and implemented a novel anti-spam solution which applies the key ideas: authenticating email and cutting spam from the source. It works effectively because most spam is from those untrusting parties to legitimate email servers. No more email spoofing exists in this email system. *ID-based Signature Scheme* is used to provide efficient and secure authentication between emails. The proposed system can be adopted to the existing email system that small modification needed on the email server. The strategy used is built on top of the existing anti-spam technologies that they can still be applied to improve the overall performance. More ideal result can be achieved if the system is widely deployed. Complete different signature scheme can also be applied to improve the system performance.

More uses can also be realized by applying different kinds of cryptographic tools. Email will work as an efficient transaction platform for small scale of business such as making use of E-Stamp. More innovative features might also appears to make email more popular than ever before.
8. REFERENCES


http://www.securityfocus.com/infocus/1763


http://www.cs.berkeley.edu/~bock/publications.html


http://www.silicon.com/research/specialreports/thespamreport/0,39025001,39150262,00.htm


# 9. Appendix

## 9.1 Monthly Log

<table>
<thead>
<tr>
<th>Month</th>
<th>Tasks</th>
</tr>
</thead>
</table>
| Oct   | Study Email Infrastructure  
       | Study Cryptographic Primitives |
| Nov   | System Architecture Design  
       | Class Design |
| Dec   | Microsoft Outlook Toolbar coding  
       | Email Server coding  
       | Key Management Server coding |
| Jan   | System Testing  
       | System Design Refinement |
| Feb   | System improvement  
       | System Testing |
| Mar   | Final Testing  
       | Demonstration  
       | Report writing |