Multi Asymmetric Cryptographic RSA Scheme

Andrew Davis and Matthew Wagner

Advisors: Dr. Fu and Ahmad Mansour

National Science Foundation

Research Experience for Undergraduates

Objectives

To develop a one to many asymmetric encryption scheme that requires minimum collusion between receivers.

Introduction

In recent years there has been an increased need to be able to send a single message to multiple receivers while also keeping the contents of the message secure. An example of this is a multicast encryption scheme which is an encryption scheme designed to be encrypted by one sender and decrypted by a group of receivers. A specific form of a multicast system is TCP/IP multicast systems where all receivers are located at the same IP address. In this system a sender would want to make sure that not only would an attacker on the IP address not be able to learn the contents of the message but also each individual not be able to learn the contents of others' messages.

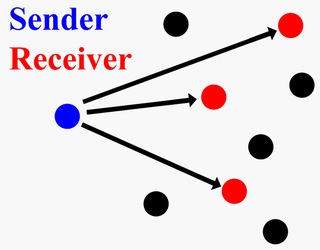


Fig. 1 (Above) A picture example of a multicast scenario.

Currently there are some cryptographic schemes that try to solve this problem inefficiently. We wanted to create a scheme that is scalable to large systems of receivers. Thus we decided to focus on creating an asymmetric system because they only use one key per person. This means that it is much easier to scale.

Methods

Chinese Remainder Theorem

We use the Chinese Remainder Theorem as one of the bases for a one to many encryption scheme. This theorem works by creating a number. From this number we can modulate it by a set of co-prime integers and receive multiple different numbers.

$$X \equiv a \pmod{M_1}$$
$$\vdots$$
$$X \equiv b \pmod{M_i}$$
$$\vdots$$
$$X \equiv c \pmod{M_n}$$
$$Gcd(M_1, \cdots, M_i, \cdots, M_n) = 1$$

Fig. 2 (Above) A picture of the Chinese Remainder Theorem.

$$E_1(m_1, K_1^+, N_1) = C_1$$
$$\vdots$$
$$E_i(m_i, K_i^+, N_i) = C_i$$
$$\vdots$$
$$E_n(m_n, K_n^+, N_n) = C_n$$
$$\sum C_i = C$$
$$D_1(C, K_1^-, N_1) = m_1$$
$$\vdots$$
$$D_i(C, K_i^-, N_i) = m_i$$
$$\vdots$$
$$D_n(C, K_n^-, N_n) = m_n$$

Methods (Cont.)

RSA

RSA is a standard in asymmetric cryptosystems. We use this as another base for our cryptosystem. From RSA we use the same key generation and decryption. The following equation is the equation for RSA decryption.

$$m = C^{K^-} \; mod \; N$$

Our Proposed Method

Our proposed method includes three phases: initialization, encryption, and decryption. Receivers must generate their key's the same way one would in RSA.

Table of Symbols

Symbol	Description
C	Cipher-text
K^+	Public Key created during Key Generation.
K^-	Private Key created during Key Generation.
m	Plain-text
N	Modulus created during Key Generation

During our initialization phase the sender must calculate X and A_i using the following equations:

$$X = \prod N_i$$

$$A_i = S_i \frac{X}{N_i} \quad \text{where} \quad S_i \frac{X}{N_i} = 1 \; mod \; N_i$$

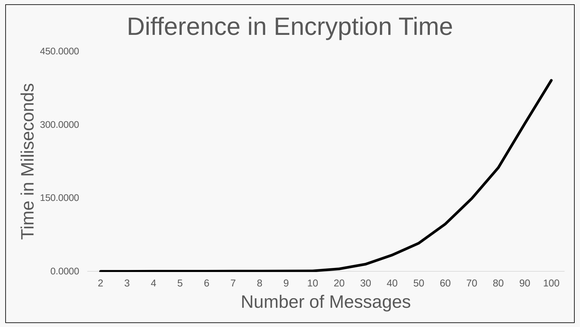
Next the sender will proceed to the encryption phase. During this phase the receiver will calculate the cipher-text using a set of messages he wanted to send, the values calculated in the initialization phase and the set of the receiver's public keys.

$$C = \left(\sum m_i^{K_i^+} * A_i \right) mod \; X$$

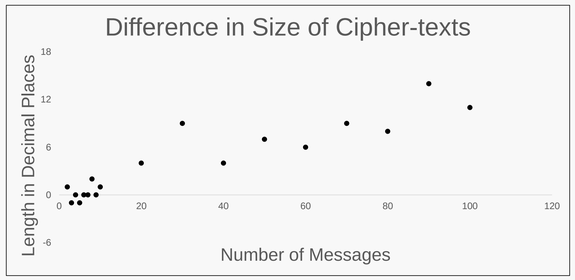
After creating the cipher-text C the sender will send this same cipher-text to all receivers. Each receiver will decrypt this cipher-text and will retrieve their own message. The receiver's will use the same decryption method as text-book RSA. The decryption equation is shown above in the RSA section.

Results

We ran tests the measure the time of our algorithm and the size of the generated cipher-text compared to that of RSA and a concatenated string of RSA encryption. We varied the number of receivers for each test. Each of these tests were done on a Toshiba Click 2 laptop which had 4 GB of RAM and a Quad-core process clocked at 2.2Ghz per core, roughly. The test code was written in C# 4.5 in visual studio. Each test was run 100 times and the times were averaged to remove any outliers in time. The keys were generated from Ps and Qs taken from a list of 330 primes between 61,547 and 65,267, avoiding recurrence of any prime number. This means that P and Q were both 16 bits and the K+ and K- were also roughly 16 bits, but N was closer to 32 bits. K+ was randomly generated.



Graph 1 (Above) This graph shows the length of our method's encryption time minus the correlating RSA's encryption length for a given number of receivers.



Graph 2 (Above) This graph shows the length of the RSA cipher-text length subtracted from our correlating cipher-text for a given number of receivers.

Analysis and Future Recommendations

When we examine our result graph one will notice some interesting trends.

- First, our encryption time increases exponentially compared to RSA. We suspect this is happening because when starting with a relatively small number, for example a 16 bit numbers, our algorithm will start to deal with much larger numbers than RSA. So, it takes more time.
- Second, the size of our cipher-text compared to the same number of RSA cipher-text's concatenated together get's smaller as we include more receivers.

From this it is relatively easy to see the trade-off of our algorithm. We trade time for size. Additionally our algorithm is more secure, we show this in our paper but not on this poster.

In the future we could compare the results from our software implementation to a hardware implementation. This would increase the accuracy of our results because software cannot handle very large keys, so we could see how our system reacts to a much more realistic key size. We also would like to test information for a realistic scenario rather than computer simulation.

Conclusion

We propose a one to many encryption scheme that is asymmetric in design and requires no collusion between the sender and receiver as well as between receivers. The algorithm is based on the standard encryption scheme of RSA and the Chinese Remainder theorem. From testing, we found our proposed scheme trades time for security and size of messages. We hope to further find out how our algorithm compares when tested in hardware and how it would compare in a real world scenario. This algorithm is not perfect but can be used as a basis for new, better one to many encryption schemes.

Acknowledgement

This research work was conducted at Oakland University in the UnCoRe program - REU site supported by the National Science Foundation under Grant No. CNS-1460897. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation.