RSA: The Algorithm, Common Modulus Attack, & Legal Implications

By: Ivan Novasak

Southern New Hampshire University

MAT 260: Cryptology

For: Toke Knudsen

19 February 2023

Introduction

In this paper the RSA (Rivest–Shamir–Adleman) cipher and an attack mechanism known as the Common Modulus is described. Also covered are legal implications in the world of encryption. RSA is an public-key encryption method developed in the 1970s by Ron Rivest, Adi Shamir and Leonard Adleman which is used in a variety of places, including PGP, authentication on the Web, e-mail messages, and other digital transactions like financial/credit cards. RSA uses both a public and private key for encryption and decryption. The user would distribute their public key out to people who they want to be able to decrypt their message and would use a private key to encrypt said messages. The encryption/decryption algorithm is described as follows.

Algorithm

There are 4 major steps involved in implementing RSA:

A. Key Generation

B. Key Distribution

C. Message Encryption

D. Message Decryption

A. The encryption key is generated as follows:

First [step 1], choose 2 large prime numbers at random that are denoted as p and q that are at least 308 decimal digits long. This will allow the modulus of the product to be at least 2048 bits which is the minimum standard according to the US government as of 2015. (Simmons, 2009) Software developers implementing RSA should check to see if the standard was updated more recently, as these numbers may have changed since 2015. The next step [2] in the algorithm is computing $n = pq$ where n is the modulus mentioned earlier. This value n shall be part of the public key to be released. Next [step 3], a number called $\lambda(n)$, is calculated. $\lambda(n) = lcm(\lambda(p), \lambda(q)) = lcm(p - 1, q - 1)$. That LCM can be calculated via the Euclidean algorithm because $lcm(p - 1, q - 1) = \frac{|(p-1)(q-1)|}{gcd(p-1,q-1)}$. That value computed, $\lambda(n)$, must be kept secret. Next [step 4] one must choose an integer denoted E that has the following attributes:

- $2 < E < \lambda(n)$
- $gcd(E, \lambda(n)) = 1$

Note that in this paper, capital E is used in this paper for this number to not confuse it with the well known transcendental number e used in calculus and other mathematical fields. The smallest possible compatible value is 3, but typically $2^{16} + 1 = 65537$ is used. Low values for this number E are insecure, so it is best to a much higher value. In the real world a number that is high but not 65537 should be used as 65537 is one of the most common values. E is to be part of the **public key**. Finally, [step 5], calculate the number d which shall be the modular multiplicative inverse of $E \bmod \lambda(n)$. The specific equation to be solved is $dE \equiv 1 \ (mod\, \lambda(n))$. This number d needs to be kept secret; it's part of the **private key**. Once d is calculated, it is safe to throw away p, q, and $\lambda(n)$. (Wikipedia contributors, 2023)

B. Distributing The Public Key:

For one person to send a message to another that was encrypted with RSA, the sender has to have the receiver's **public key** to encrypt the message. The receiver will use their private key to decrypt the message on their end. The receiver gives the sender their public key ahead of time, so the sender can encrypt the message. (Wikipedia contributors, 2023) An easy example of this playing out online are PGP keys. Some people use PGP to encrypt e-mails and they leave their public key accessible on their website.

C. Encryption Process

The sender of the message carries out the encryption via the following formula: $c \equiv m^{E} (mod\ n)$ where c is the ciphertext and m is an integer that was computed from the message along with a padding scheme that the recipient and sender mutually agreed upon in advance. Once *c* is computed, that is what is sent out. An optional extra step that can be carried out at this point is called signing, which is used when a receiver wants to ensure that the sender really is the one encrypting the message. Signing is done via a hashing algorithm with the equation $(h^{E})^{d} = h^{Ed} = h^{dE} = (h^{d})^{e} \equiv h\ (mod\ n)$ where *h* is the hash of m. (Wikipedia contributors, 2023)

D. Decryption Process

The recipient can decrypt the message by carrying out the calculation $c^d \equiv (m^E)^d \equiv m\ (mod\ n)$ then they can reverse that aforementioned padding scheme to display their message. The message may still be an integer at this point, so it will need to be converted back to English or whatever type of text was used. If signing was done, comparison of the hash values will be carried out at the time of decryption to verify that the message was sent by the intended sender.

Common Modulus Attack

The scenario is laid out such that a person wants to send a message out using the public key defined here as **(n, E$_1$)**. A third person other than the sender and recipient is intercepting the encrypted messages. An unexpected event happens that the public key is changed to a different key that is defined as **(n, E$_2$)**. The interceptor has the same message *m* encrypted to 2 different ciphertexts c_1 and c_2 that have the same modulus but different exponents. So mathematically we have $c_1 \equiv m^{E_1}(mod\ n)$ and $c_2 \equiv m^{E_2}(mod\ n)$. If $gcd(E_1, E_2) = 1 = gcd(c_2, n)$ the ciphertext can be found. The attack starts by calculating integers *x* and *y* where $xE_1 + yE_2 = 1$ then using the Extended Euclidean Algorithm like:

$c_1^x + c_2^y = (m^{E_1 x})(m^{E_2 y}) = m^{E_1 x + E_2 y} = m^1 = m.$ Typically, *y* is a number less than zero, so evaluating c_2^y takes some care:

Let $y = \quad - j$. Now substitute:

$c_2^y = c_2^{-j} = (c_2^{-1})^j = (c_2^{-1})^{-y}$. Keep in mind that these calculations are all benign carried out assuming mod n where n is known and c_2 **has a modular inverse in that base**. This can be checked via $gcd(c_2, n) = 1$ - if that equation is true, c_2 has a modular inverse. Otherwise the calculation will not work. A Python script that performs this algorithm is shown on the referenced web page. (Pogiatzis, 2018)

Legal Implications

The laws surrounding encryption systems can vary greatly depending on country and jurisdiction. One example being in the US, the government may require encryption providers to decrypt data only with a warrant, whereas in China the government has powers to require encryption providers to give encrypted data key access to the government. Sometimes the governments can require the encryption provider to provide a backdoor so law enforcement can gain access but most other users can not. There are ongoing legal battles around the world where encryption providers have provided end-to-end encryption for messaging services and were not designed to have any backdoor access. (Bischoff & Bischoff, 2020) The aforementioned information was mainly about companies or other providers of encryption products/protocols. One example that is still playing out involves whether governments should or should not have access to decrypt end-to-end encrypted messaging protocols. The high-profile situation in the 2000s/2010s involves BlackBerry Messenger - an instant messaging system that was available exclusively on BlackBerry smartphones and utilised a type of tunnel separate from the general cellular data protocols that was not crackable by any law enforcement. It was the first end-to-end encrypted message system that was widely available to consumers and professionals alike. During 2010, China, India, and UAE viewed the BlackBerry Messenger as a security threat, with India reacting to at the time a recent terrorist attack. (Whittaker, 2010) More recently in 2022, governments have expressed desire to have access to users' WhatsApps messages. Meta, the owners of WhatsApp will not lower the security and potentially compromise billions of users of their service for the benefit of a small set of users (certain governments). (McCallum, 2022) Ultimately this debate across consumers, governments, and the tech industry, which is still ongoing, is the **balance between security and privacy**.

References

Bischoff, P., & Bischoff, P. (2020, October 20). *Encryption laws: Which governments place the heaviest restrictions on encryption*? Comparitech. https://www.comparitech.com/blog/vpn-privacy/encryption-laws/

McCallum, B. S. (2022, July 30). *WhatsApp: We won't lower security for any government*. BBC News. https://www.bbc.com/news/technology-62291328

Pogiatzis, A. (2018, December 5). *RSA Attacks: Common Modulus - InfoSec Write-ups*. Medium. https://infosecwriteups.com/rsa-attacks-common-modulus-7bdb34f331a5

Simmons, G. J. (2009, July 22). *RSA encryption | Definition, Example, & Facts*. Encyclopedia Britannica. https://www.britannica.com/topic/RSA-encryptionReferences

Whittaker, Z. (2010, August 1). *BlackBerry encryption "too secure": National security vs. consumer privacy*. ZDNET. https://www.zdnet.com/article/blackberry-encryption-too-secure-national-security-vs-consumer-privacy/

Wikipedia contributors. (2023, January 20). *RSA (cryptosystem)*. Wikipedia. https://en.wikipedia.org/wiki/RSA_(cryptosystem)