RSA Encryption

Since ancient times, it has been very valuable to coordinate military attacks, especially amongst comrades who attack from different directions. Sending messages without the enemy intercepting or understanding the message if it is intercepted was and still is a crucial part of planning synchronized attacks. Because of this, many ciphers and codes were invented to secure messages such that only allies could understand and decode these messages. Encoding messages is just as valuable today for that same purpose, but arguably more import for the internet and all of its functions today. The internet is an integral part of life of almost every person. Most people cannot avoid using the internet for at least one aspect of their life, from paying bills, to communicating with others, to entertaining themselves. All of the internet relies on encryption to send secure data such as passwords from one place to another. Encryption relies on the RSA algorithm, which uses number theory to ensure that this data is practically impossible to crack.

Early computers used a single number that had to be known and kept secret to encrypt information. The National Security Agency ran the encryption systems at the time and could crack all encrypted messages if they wanted to or needed to do so.[1] Rivest, Shamir, and Adleman figured out a way to send encrypted messages without needing as much information to be able to encrypt and decrypt messages that was also faster to encrypt than a previous method discovered by Diffie, Hellman, and Merkle. Rivest, Shamir, and Adleman had read the Diffie, Hellman, and Merkle method and soon came up with a faster way to encrypt information that was just as secure, if not more secure. [1] Three English scientists also claimed to have discovered the same RSA algorithm before Rivest, Shamir, and Adleman but it was not published because the British Government wanted it to stay secret. [5] The RSA is considered a ‘trapdoor function’ that is easy to compute but hard to reverse engineer or decrypt without

knowing the decryption exponent. [2] Once this is known, it is relatively easy to break the encryption. [2]

The RSA algorithm only requires that both parties know and keep secret two large prime numbers, **p** and **q**, and the decryption exponent, **d**. Both parties need to know **n** and **e** the encryption exponent, but this information can be public knowledge without affecting the strength of the encryption. [3] If M is the message that you want to be encrypted, and C is the encrypted message, then the coded message $C \equiv M^{\mathbf{e}} (\text{mod } \mathbf{n})$. When the other party receives C, all they have to do is take $C^{\mathbf{d}} (\text{mod } \mathbf{n})$ and that will be the message decoded.

Proof:

Let M be the message, C be the encrypted message, **p** and **q** be large prime numbers such that $\mathbf{n}=\mathbf{p}\cdot\mathbf{q}$, $\mathbf{e} \in \mathbb{N}$ that is the encryption exponent, and **d** be the secret decryption key that only the sender and receiver know such that $\mathbf{e}\cdot\mathbf{d} \equiv 1 (\text{mod } \mathbf{n})$. Let

$$C \equiv M^{\mathbf{e}} (\text{mod } \mathbf{n})$$

Then to decrypt C,

$$C^{\mathbf{d}} \equiv (M^{\mathbf{e}})^{\mathbf{d}} (\text{mod } \mathbf{n}) \equiv M^{(\mathbf{e}\cdot\mathbf{d})} (\text{mod } \mathbf{n}).$$

By Lemma 9.2.7 in [1],

$$\varphi(\mathbf{n})=\varphi(\mathbf{p}\cdot\mathbf{q})=(\mathbf{p}-1)\cdot(\mathbf{q}-1)$$

Choose **e**,**d** such that

$$\mathbf{e}\cdot\mathbf{d} \equiv 1 (\text{mod } \varphi(\mathbf{n}))$$

By definition of modular arithmetic,

$$\mathbf{e}\cdot\mathbf{d}=1+\varphi(\mathbf{n})\cdot\mathbf{k}$$

for $\mathbf{k} \in \mathbb{Z}$. It follows that

$$M^{(\mathbf{e}\cdot\mathbf{d})}=M^{(1+\varphi(\mathbf{n})\cdot k)}=M\cdot M^{(\mathbf{k}\cdot\varphi(\mathbf{n}))}=M\cdot(M^{\varphi(\mathbf{n})})^{\mathbf{k}}$$

By Euler's Theorem,

$$M^{\varphi(\mathbf{n})}\equiv 1 \pmod{n}$$

By substitution, the multiplication property of modular arithmetic, and exponent laws,

$$M\cdot(M^{\varphi(\mathbf{n})})^{\mathbf{k}}(\text{mod } \mathbf{n})\equiv M\cdot 1^{\mathbf{k}} \text{ (mod } \mathbf{n})\equiv M \text{ (mod } \mathbf{n})$$

The RSA encryption means that it is very easy to send encrypted messages and information if you know the public key. This means you can send an encrypted message to not worry about someone intercepting and cracking the message without having **d**.

There is also a method to increase security by verifying that the sender is correct sender and not an identity thief trying to send false information. This is done by using the decryption exponent **d** as the encryption exponent and sending that coded message to the person meant to decrypt the message. The receiver can then take the message and manually encrypt it themselves to ensure that the received message is the same as what they encoded. For this to work, both parties have to agree on the message or the method to compute the message to be encoded with the decryption exponent. [4] This method is called signing. Without signing, anyone could encrypt a message and send it via the normal communication channels, and the receiver would not be able to distinguish who sent the message.

Despite the security features of RSA encryption, the encryption can be broken quite easily if **e** is small. [4] The encryption and decryption exponents generally are 1024-bit number that are roughly 300 digits long in base 10. [4] However, if **e** is too large, then the time to encrypt will very long. [5] There is also a weakness in this encryption if the same **n** is used for many encrypted messages with different encryption and decryption exponents. [4] In this case it is necessary to have a variety of large primes that can compose **n** if one is sending encrypted

messages to many parties with a different key each time. Another way to increase security is add at least 64-bits of random junk data onto the encrypted message. However, in general the best way to break RSA encryption is to factor **n**, or less efficiently find **d**, however there are many other attacks that can work depending on the situation, including using the Chinese Remainder Theorem. [5] To see why factoring **n** will break the encryption, consider that once **n** is factored into **p** and **q**, it is trivial to find φ(**n**) and thus **d** will be the inverse to **e** in $\mathbb{Z}_{\varphi(\mathbf{n})}$, which then means one has the decryption key. One final note on the RSA Algorithm is that if, or more likely when, quantum computers become viable and reliable, RSA encryption can be broken quite efficiently with more than one algorithm that factors **n**, because these algorithms run very quickly on quantum computers but are not efficient on current non-quantum computers. [5] At this point, a new encryption-decryption system will need to be implemented immediately across the internet.

Sources:

1. James Pommersheim, Tim K Marks, and Erica Flapan, Number Theory: A Lively Introduction with Proofs, Applications, and Stories, Wiley, 2010.
2. Dan Boneh, *Twenty Years of Attacks on the RSA Cryptosystem*, https://www.ams.org/notices/199902/boneh.pdf, 1999
3. Evgeny Milanov, *The RSA Algorithm*, https://sites.math.washington.edu/~morrow/336_09/papers/Yevgeny.pdf, 2009
4. DI Management, *RSA Algorithm*, https://www.di-mgt.com.au/rsa_alg.html, 2018, Accessed 11-28-2018
5. Song Y. Yan, *Cryptanalytic Attacks on RSA*, https://link.springer.com/content/pdf/10.1007%2F978-0-387-48742-7.pdf, 2008