

Protocol Foundations 001: Cryptography

Mario Havel and Tim Beiko

Throughout history, humans have used *cryptography* to protect sensitive messages from eavesdropping. Famously, to avoid trusting messengers, Julius Caesar would encode messages he sent to generals at war. He specifically replaced each letter in his original message by one that was a fixed distance away from it in the alphabet. For example, the word *attack* could be modified to *dwwdfn* by using a distance of 3: *d* is three letters past *a*, *w* is three from *t*, and so on. This technique is called a **Caesar cipher**. As long as Caesar communicated the offset to a general in advance, he could send seemingly illegible messages to them which they could then decipher.¹

Ciphers that were harder to reverse-engineer than the Caesar cipher naturally developed over time. One example is the **Vigenère cipher**. Instead of using a constant value to swap each letter in a message, this cipher varies the offset for each character based on the alphabetical order of letters in a secret code word, referred to as a *key*. For example, the key *abc* would shift the first letter of an encoded word by one letter alphabetically, the second letter by two, and the third letter by three, following from the respective positions of *a*, *b*, and *c* in the alphabet. If the input message is longer than the key, we cycle over it again.²

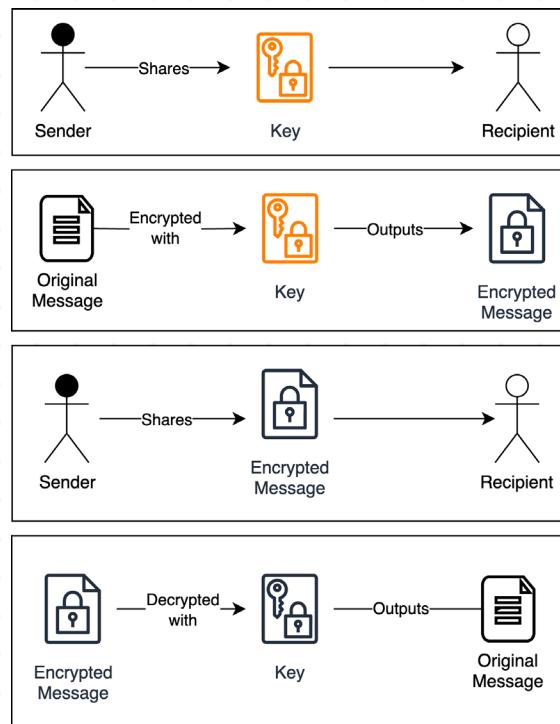
The evolution of mathematics led to more complex encryption techniques. With the advent of algebra, methods were developed using principles like modular arithmetic and exponentiation to increase the difficulty of deciphering messages. The shift towards mathematical cryptography laid the groundwork for the

development of ever more sophisticated encryption techniques. Eventually, machines were used to increase the sophistication of ciphers beyond what humans could calculate directly. One such example is the (in)famous **Enigma machine** Germany relied on during World War II.³

While the specifics of how the messages were encrypted varied and became more complex, conceptually all these techniques shared the same design:

an input message is encrypted using a key, the key is shared with the recipient, and the encrypted message is decrypted by the recipient using the key.

This design, where the same key is used by the sender and recipient, is called **symmetric-key encryption**.⁴



1. en.wikipedia.org/wiki/Caesar_cipher

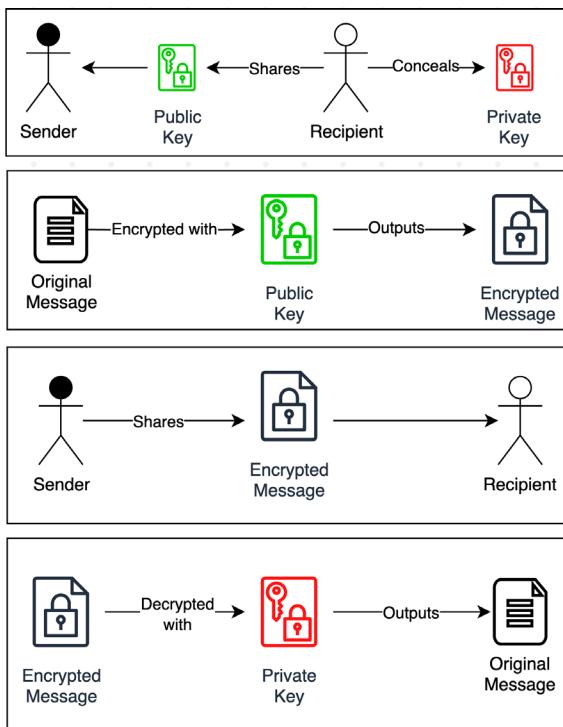
2. en.wikipedia.org/wiki/Vigen%C3%A8re_cipher

3. en.wikipedia.org/wiki/Enigma_machine

4. en.wikipedia.org/wiki/Symmetric-key_algorithm

The biggest flaw of symmetric encryption is the assumption that the key can be communicated to the intended recipient without exposing it to an unwanted party. In other words, whether it took the form of a single digit, a secret word, or a machine with a specific configuration, the key had to be passed in a trusted setting from the sending to the receiving party. The security of all subsequent messages depended on maintaining the safe transmission of the key.

In 1976, *New Directions in Cryptography*⁵ introduced a solution to this problem that would lay the foundation for what is now considered modern cryptography: **asymmetric encryption**. The conceptual breakthrough was moving from a single, shared key (symmetric-key encryption), to a pair of keys, or a **keypair**. The pair consists of a *private key* and *public key*.



To encrypt a message, the recipient first generates a private key, which is a *large, random number*. The number has to be large and random enough that it cannot

be guessed, anecdotally larger than the number of atoms in the universe. The public key is then generated from the private key in a deterministic way. The public key is also a large number, but does not reveal information about the private key. Looking at a public key, one can not guess the private key from which it was generated.

Because the private key was used to derive the public key, and due to the mathematical properties of encryption algorithms, a message encrypted with the public key can always, and *only*, be decrypted using its associated private key. The public key can safely be advertised broadly by the recipient and used by anyone to encrypt messages. These can then only be deciphered using the corresponding private key, which never needs to be shared.

This is a subtle but powerful shift in the evolution of cryptography:

asymmetric cryptography allows anyone to advertise a public key which can be used by anyone else to communicate messages only they can decipher.

A physical metaphor for public/private keys is a personal locking mailbox. Anyone can deposit a letter in it, however only the owner of the mailbox can open it and read the messages it contains.

Secure encryption using public keys is only one of the affordances enabled by asymmetric cryptography. A private key can also be used to create a mathematical proof linking arbitrary data to a public key. This type of proof, called a *digital signature*, can be verified by anyone with access to the data and the associated public key.⁶ These allow both *individuals* and *institutions* to have a common standard to assess the authenticity of data online.

5. www-ee.stanford.edu/~hellman/publications/24.pdf

6. en.wikipedia.org/wiki/Digital_signature

Encryption and digital signatures have become the backbone of modern communication. Whenever you connect to a website, your browser uses the *https* protocol to encrypt all communication with the website server.⁷ This ensures no other party (e.g., ISPs) can read it and allows for the safe transmission of sensitive information such as passwords or payment data. The secure connection is established by verifying a *signed certificate* from a trusted source proving the server truly belongs to the domain you are connecting to.

However, not all secure and encrypted communication schemes required trusted third parties. The **Pretty Good Privacy (PGP)** protocol⁸ is one of the earliest and most adopted examples in the context of *email encryption*. A user shares their public key in an unencrypted email message which their recipient can then use to send back an encrypted response. In this case, no central authority verifies the validity of the key: the sender trusts that the public key they have for the recipient is correct.

Users can also proactively broadcast their public keys, such as on a personal website or business card. Communities can further establish a *web of trust* by having members use their own key to create 0 attesting to the authenticity of other members' keys. Of course, users still need to assess how trustworthy each web of trust is.

A more recent example of practical secure communication in today's chat era is the **Signal** protocol. Implemented in the homonymous application, the Signal protocol uses a form of public keys that allow a seamless user experience for both one-to-one and many-to-many (group chat) conversations. The use of the Signal protocol became the golden standard of encrypted low-latency communication, gaining adoption from other major applications such as WhatsApp.

Although they implement the same protocol, Signal and WhatsApp do this in a significantly different way. Signal is a *free and open source software* product, meaning that anyone can audit its code to verify the implementation accuracy. On the other hand, WhatsApp's codebase is completely proprietary; users cannot rely on independent public auditors to review it. They have to trust the application developers that the code is free of both unintentional bugs and malicious hidden functionality like *backdoors*, which refer to encryption implementations that can be purposefully sidestepped by an external party to access decrypted (plain text) messages. The risk of users' most sensitive information being exposed to external parties without their consent is why the cryptography community has such a strong bias towards openness.

The average user may never read the source code of an application they use, but security researchers being able to do so, and having others evaluate their claims, reduces the risks of intentional or unintentional bugs. The more people look at the code, the lower the odds that a bug exists, undiscovered.

With its origins in ancient wartime strategies, cryptography has now become the cornerstone of digital security. It's a science that harnesses the power of mathematics to create systems bound by unyielding laws of our universe, ensuring a secure, interconnected world.

MARIO HAVEL & TIM BEIKO are part of the Ethereum Foundation's Protocol Support team which helps facilitate Ethereum network upgrades as well as other protocol-related initiatives, including Summer of Protocols.

MARIO | github.com/taxmeifyoucan
TIM | warpcast.com/tim

7. en.wikipedia.org/wiki/HTTPS

8. en.wikipedia.org/wiki/Pretty_Good_Privacy

ProtocolKit

summerofprotocols.com
hello@summerofprotocols.com

© 2023 Ethereum Foundation. All contributions are the property of their respective authors and are used by Ethereum Foundation under license. All contributions are licensed by their respective authors under CC BY-NC 4.0. After 2026-12-13, all contributions will be licensed by their respective authors under CC BY 4.0.
Learn more at: summerofprotocols.com/ccplus-license-2023

ISBN-13: 978-1-962872-53-9

Printed in the United States of America

Printing history: December 2023

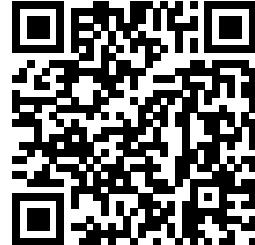
1 3 5 7 9 10 8 6 4 2

ISBN 978-1-962872-53-9



9 781962 872539

ProtocolKit



RETROSPECTUS



NEWSLETTER

