An Introduction to Cryptographic Security Methods

Moving from 20th Century Legacy Methods to the Next Generation of Quantum Resistant Public Key Cryptography via Group Theoretic Cryptography

Veridify Security
100 Beard Sawmill Road
Suite 350
Shelton, CT 06484

(203)227-3151
info@Veridify.com
www.Veridify.com

1. Introduction:

The need for secure communications dates to antiquity. Whether we are looking at the 5000 year old cylinder seals of ancient Mesopotamia, which were used for authentication, or the Caesar cipher, that was used to protect military communications, a theme quickly emerges: information needs to be secured.

There are two distinct paradigms for approaching security. One is the symmetric or private key path, that can be viewed as a descendant of the very early attempts at security. Any such system assumes that, if two individuals wish to communicate securely, they must both possess a unique common secret key, which is used for both encryption and decryption. While many such systems have been developed over time, it is this ubiquitous assumption that leads to vulnerabilities and compromised security. Secret keys need to be securely generated and distributed to all the users of the system, and if any user is compromised, the security of the entire system is obliterated. The need for a new paradigm was first addressed in the 1970's with the advent of a radically different mindset about how security can be achieved. The emergence of public-key cryptography has proved to be pivotal for all technology moving forward. In a public-key system, every user has their own private key and, additionally, every user has a public key which is available to any other user. Each user can engage in secure communications using the private and public keys, and if one user is compromised, no harm is done to the remainder of the system.

Public-key cryptosystems themselves fall into two categories: the Diffie-Hellman type protocol (Whitfield Diffie and Martin Hellman in 1976), and the RSA type protocol (Ron Rivest, Adi Shamir, and Leonard Adleman 1978). Before exploring the structures of these distinct approaches, it is worth taking a moment to consider the broader impact of public-key methods. As ubiquitous as the Internet is today, it was not until the 1990's that the necessary security protocols were put in place that enabled users to function securely while online. The online revolution in the financial sector, e-commerce, and medical records all rely on the security breakthrough provided by public-key cryptography. In many respects, we are at a similar evolutionary point with embedded systems, wireless sensor networks, and the Internet of Things (IoT). The solutions that enabled the Internet revolution are not suitable to these low-resource environments. Furthermore, these legacy systems will not be quantum resistant (i.e., secure against the quantum computers being developed today). A new generation of public-key infrastructure needs to emerge. The basic tenet persists: information needs to be secured.

2. A Heuristic View of Cryptographic Structures.

When viewed diagrammatically at a high level, private key methods of encryption all take the following form. Two users, Alice and Bob, each possess the same secret key, κ, that is used for both encryption and decryption (hence the term *symmetric*). When Alice wishes to communicate securely with Bob, she inputs her message (often termed the plaintext), along with the secret key κ, into the encryption protocol, which then outputs the encrypted form of the message, referred to as the Ciphertext, as shown in Figure 1.

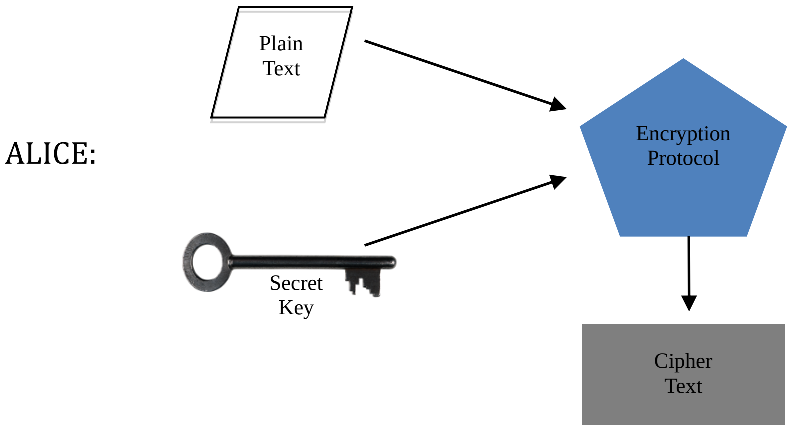

Figure 1: Alice creates an encoded message with a private key she shares with Bob.

Upon receiving the Ciphertext from Alice, Bob can obtain her original message by inputting their shared key κ, along with Ciphertext into the decryption protocol, as shown in Figure 2.

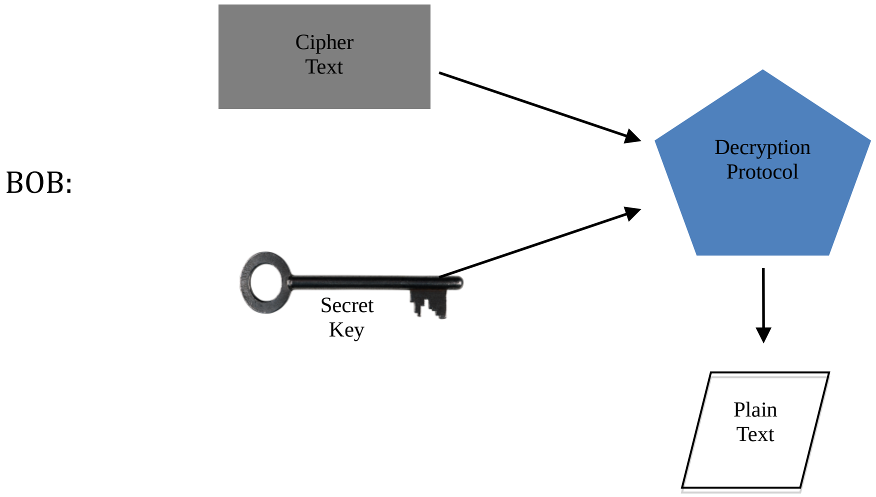

Figure 2: Bob converts encoded message from Alice into plain text using shared private key.

As intuitive as such a private-key method is, the vulnerabilities quickly emerge. The security of the system relies on the secret key κ remaining uncompromised at all times. That means all attempts at searching for the secret key must fail. Given our increasing computational power, the search for the secret key will take less and less time, and a key can only remain secure for so long. This leads to the need to securely distribute new keys. Further, with a private-key solution, if one user's private key is compromised, then all users who utilize that key for encryption will be impacted: they would all need to be alerted and would all need a securely distributed new key.

As the number of users of a private key system grows (in the case of the IoT - into the billions!), the need to confidentially distribute new keys on a regular basis leads to the progressively unwieldy problem of key management. While it is generally true that private-key systems have a fast running time, which perpetuates their use in many commercial systems today, their intrinsic weaknesses are ubiquitous.

With public-key methods, often termed asymmetric, the cryptography frees itself from most of the above discussed difficulties by having each user utilize a private key of their own, that only that user knows, and a mathematically related public key that is openly shared. There are two general methods for using public-key cryptography, RSA and Diffie-Hellman. In both cases, the systems are designed so that the public keys can be distributed to all parties prior to communicating and the security of the system is not compromised by the publication of the public keys when a session begins.

In the case of an RSA-type system, if Alice wants to send Bob a message, she uses Bob's public key to encrypt her plaintext, as shown in Figure 3.

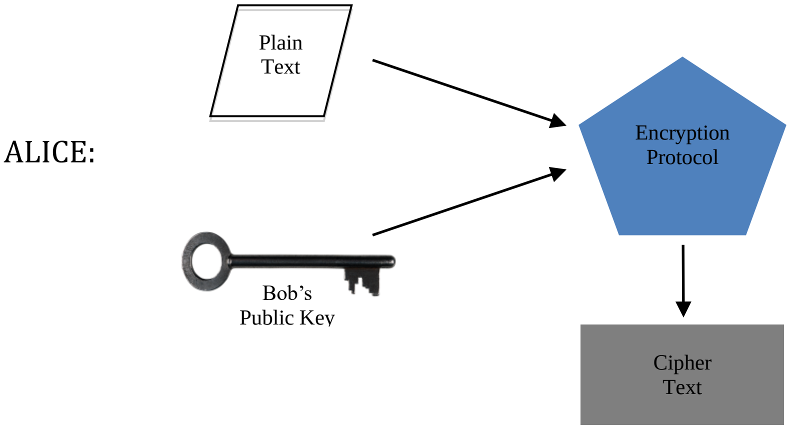

Figure 3: RSA type system: Alice uses Bob's Public Key to encrypt her message to Bob.

Bob, upon receiving the Ciphertext, can obtain Alice's original message by inputting his private key κ, along with Ciphertext into his decryption protocol, as shown in Figure 4.

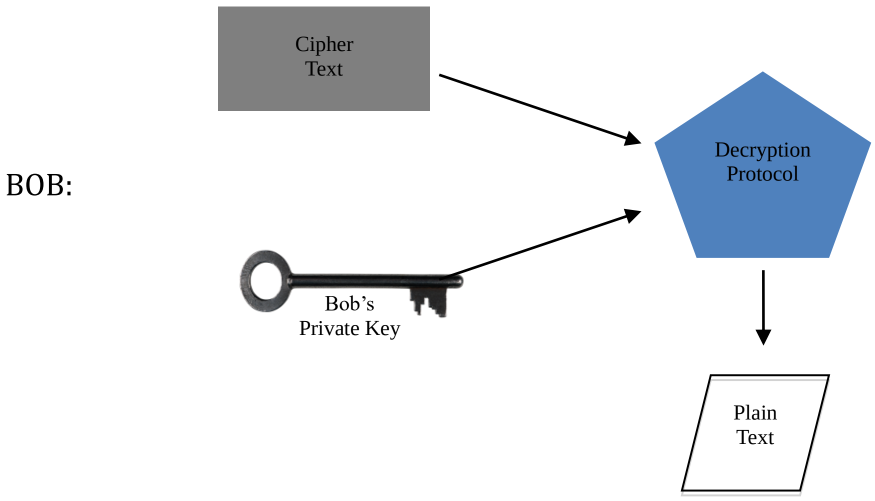

Figure 4: RSA type system: Bob uses his own private key to decrypt the encoded message.

With an RSA-type system, only Bob is in possession of his private key, which is requisite for decryption. Anyone intercepting the encrypted message from Alice will be unable to decipher it.

In the case of a Diffie-Hellman-type system, Alice and Bob use each other's public keys, together with their own respective private keys, to establish a shared secret key, which can be used for encryption and decryption of their message. Only Alice and Bob, who are participating in this session, will have access to the necessary shared secret key to encrypt or decrypt a message. The Diffie-Hellman type systems are structurally distinct, as is shown in Figure 5.

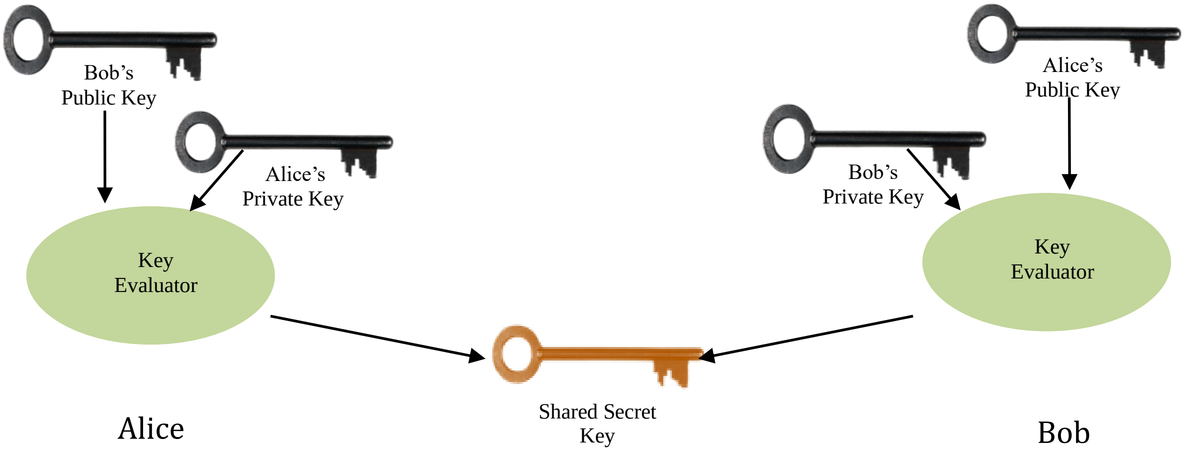

Figure 5: Diffie-Hellman type system

The output of the Diffie-Hellman type system is a shared secret cryptographic key (known only to Alice and Bob), the value of which is the same for both Alice and Bob. This shared key can then be used for encryption and decryption of the message passed between them, similar to a private key system. In comparison, the output of an RSA type system is the encrypted text itself. In both of these asymmetric systems, each user's private key is unique to that user.

A succinct view of various public-key and private-key systems, including Group Theoretic Cryptography (GTC), which will be discussed in the next section, is in the concept tree shown in Figure 6.

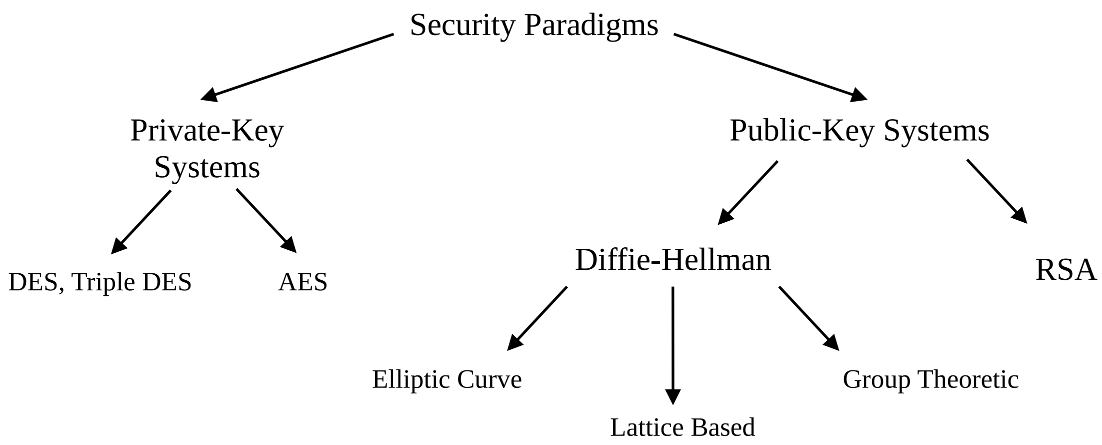

Figure 6: Categorization of private key and public key security paradigms

In a private-key situation, the security of the system relies on the secret key κ remaining uncompromised at all times. In contrast, in a public-key system, if any one user's private key becomes known, the other users are insulated from having their communications compromised: the private key that is used for encryption is not common across the entire system, as found in a symmetric system, but rather each user has its own private key. In a public key scenario, the mathematical relationship between a user's private and public key is designed so that it is infeasible to derive the private key from the public key (such a relationship is referred to as a one-way function). To be able to effectively reverse engineer a public key would be as difficult as a mathematically intractable problem. That is, mathematical problems that may be solvable but are so difficult or time-intensive to complete that the solution will provide no benefit once derived. The security of these systems lies in this intractability.

Ensuring that the distributed public keys are authentic is essential to security. One approach to this involves Public Key Certificates, using a Digital Signature Algorithm to bind a public key to an identity. Using a series of certificates, one can validate any public key back to a trusted "root" key. Additionally, protocols to prevent malicious use of public data also need to be put in place. While technical in nature, these matters can be mitigated with available tools.

A more profound issue with today’s commonly implemented public-key protocols, including Elliptic Curve Cryptography (ECC), concerns the computational footprint they entail. While memory and energy usage are not a primary concern for most environments requiring cryptographic security, these issues, along with runtime, lie at the heart of any small computing device security discussion. Every one of these cryptographic systems at its core utilizes the multiplication of large numbers. As a result, the computing resources required to achieve security grow rapidly as the level of security is increased.

As the market for smaller computing devices and IoT infrastructure continues to grow, so does the need for stronger and more efficient security solutions to address these markets and the need for the next generation of public-key cryptography becomes evident.

3. The Veridify Approach.

With the goal of providing the rapid security solution discussed above, Veridify has introduced Group-Theoretic Cryptography (GTC) based protocols and public-key cryptosystems, whose characteristics are specifically suited to these low-resource environments.

The foundation for the security of Group-Theoretic public-key cryptosystems and protocols, including a digital signature, are three distinct areas of mathematics: the theory of braids, the theory of matrices with polynomial entries (expressions of finite length constructed from variables), and modular arithmetic. At its core is a highly specialized function (replacing the standard system’s operations), known as E-Multiplication™, which brings together these mathematical tools and enables the system to provide high-speed security without overwhelming the memory and power available. This core function is highly resistant to reverse engineering due to its connections with mathematically intractable problems.

In January 2010, the United States Patent and Trademark Office granted Veridify Security U.S. Patent 7,649,999, entitled “Method and apparatus for establishing a key agreement protocol,” for its technology invention in the field of cryptography. The reader may learn more about the company’s methods from reviewing this patent. The method is summarized in its abstract, stating:

> *“A system and method for generating a secret key to facilitate secure communications between users. Public keys are exchanged between first and second users. Each user's private key may be iteratively multiplied by the other user's public key to produce a secret key. Secure communication may then occur between the first and second user using the secret key.”*

Structurally, Veridify’s public-key cryptosystem is of the Diffie-Hellman type. Alice and Bob each use their own private key and the other person’s public key to generate a shared secret key, via the E-multiplication function. The output of the system is also a common secret cryptographic key that can then be used as a private key system for encryption and decryption of the message passed between Alice and Bob.

Some of the significant features of Veridify's approach include:

- Low power consumption for the processor, high-speed implementation for real-time processing, and a small computational footprint.
- Key management is mitigated because secure disposable keys can be generated for each communication session.
- These protocols are secure against replay attacks and man-in-the-middle attacks using, for example, the Walnut™ digital signature algorithm (described in detail below).
- All GTC protocols run in linear time in the key size, while all other systems, including RSA and Diffie-Hellman protocols scale quadratically. When viewed from a user resource perspective, the graph in Figure 7 gives a high-level indication of the intrinsic benefits of GTC. Group Theoretic Cryptography is much less resource-intensive and can run on devices that have limited computing power.

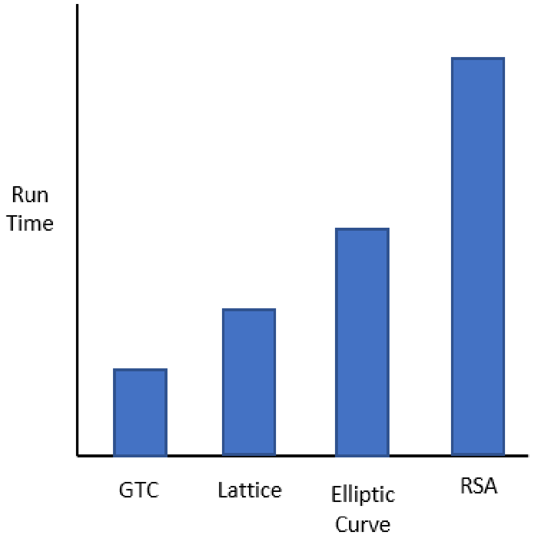

Figure 7: Public-key crypto systems vary in the amount of computing resources they require.

Security protocols which are based on secret methods are often insecure because they have not had the benefit of widespread testing and analysis. Cryptographic security emerges when methods are published for peer review and presented at conferences. Details about how and why the group-theoretic based key agreement protocol is suitable and impactful for low resource devices have appeared in the American Mathematical Society's peer-reviewed book "Algebraic Methods in Cryptography,"[1] and in Mathematics of Cryptography and Coding in the Quantum Era, "WalnutDSA™: a group theoretic digital signature algorithm"[2]. When viewed at a high level, the Braid group facilitates an irreversible one-way function whose output consists of both specialized matrices and permutations. To execute the public-key protocol, the users exchange

[1] https://bookstore.ams.org/conm-418
[2] https://doi.org/10.1080/23799927.2020.1831613

respective public keys (over what may be an open and insecure channel) and combines their own private keys with the received public ones via the one-way function to produce the unique common secret key.

The GTC digital signature algorithm, WalnutDSA, is a fast method of authenticating a user of a protocol. The security of WalnutDSA rests on the infeasibility of reversing E-multiplication in the protocol together with the difficulty of searching for specialized elements in the braid group. At a high level, every user has a public key and a private key. The user (signer) can generate a digital signature of a message using their private key, and a recipient of the signed message can rapidly verify the authenticity of the signature (and hence the signer) using E-multiplication. The process, which outputs YES (signer is authenticated) or NO (signer is not authentic) is depicted in Figure 8.

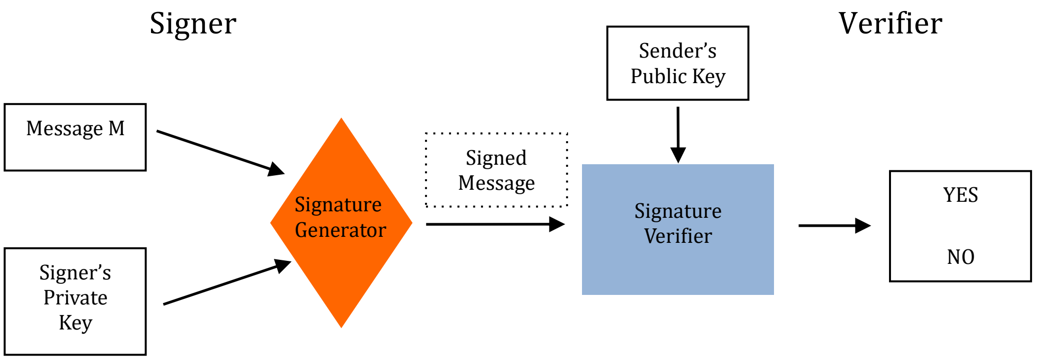

Figure 8: Signature Generation and Verification

4. Veridify's Mathematician/Cryptographers

The inventors of Veridify's public-key cryptosystems are co-founders Dr. Michael Anshel, Dr. Dorian Goldfeld, and Dr. Iris Anshel.

Dr. Michael Anshel is a security thought-leader and world-class mathematician with expertise in the field of cryptography. Dr. Anshel has authored and co-authored numerous papers in the area of public-key cryptography, is the co-inventor of several patents in the area of cryptography, zeta-one-way functions, and braid group and has received numerous fellowships and honors. He is a Professor Emeritus in the Department of Computer Science at The City College of New York.

Dr. Goldfeld is a world-class mathematician who has published over 50 papers and lectured internationally on a wide range of cryptographic topics and methods including applications of elliptic curves, quadratic fields, zeta functions, public-key cryptography, and group theoretic approaches to public-key cryptography. In 2009 he was inducted as a Fellow of the prestigious American Academy of Arts & Sciences. He is the co-inventor of several patents in the areas of multistream encryption systems, high-speed cryptography, and cryptographically secure algebraic key establishment protocols based on monoids. He has been a professor in the Faculty of Mathematics at Columbia University since 1985.

Dr. Iris Anshel, Veridify's Chief Scientist, is an accomplished mathematician and cryptographer. In addition to extensive research and publications and several patents, Dr. Anshel has experience in the commercialization of security technology. As a co-founder of Arithmetica, she was responsible for documenting methods for commercial deployment of new cryptography protocols including the AAG Braid Group Cryptosystem and supported sales and business development activity.

5. About Veridify Security

Veridify Security delivers fast, small footprint, ultra-low-energy, and quantum-resistant public-key security tools for low-resource processors powering the Internet of Things (IoT). Veridify's DOME™ Device Ownership Management and Enrollment™ solution provides a zero-touch onboarding and ownership management platform for the smallest IoT devices in the field without requiring a pervasive cloud or network connection. The company's Key Agreement Protocols and Digital Signature Algorithms are used for secure device-to-device communications, as well as secure boot and secure software updates for automotive, consumer, healthcare, industrial, and smart home applications. Veridify is partners with leading semiconductor manufacturers including Intel, Renesas, ON Semiconductor, and STMicroelectronics, and offers Software Development Kits and security tools for a wide range of environments.

More information about how GTC can be used for embedded systems and other platforms with low computing power can be found in Veridify's white paper – *Security in Low Resource Devices*, available at http://www.veridify.com/resources/documentation/.

For more information on our cybersecurity tools or securing the Internet of Things, please contact us at info@veridify.com. More information about Veridify can be found on its web site at http://www.veridify.com. Veridify's insights on security can be found on its blog at https://www.veridify.com/iot-security-blog/ and on Twitter at https://twitter.com/Veridify.