# CASE STUDY: CRYPTOGRAPHIC ATTACKS ON SYMMETRIC CIPHERS, ASYMMETRIC CIPHERS, AND HASH FUNCTIONS

AVISHEK DHAKAL
220064

Avishek Dhakal (CU ID:12981148 | Student ID: 220064)

**ST6051CEM Practical Cryptography**

Santosh Bhandari

August 17, 2024

# Abstract

This case study is a journey of the vulnerabilities inherent in cryptographic systems, focusing on the three core pillars: symmetric ciphers, asymmetric ciphers, and hash functions. The report will delve into the diverse attack methodologies employed against these cryptographic primitives, dissect their potential impact, and examine the countermeasures designed to fortify their defenses. It utilize the CrypTool to visualize various things during study. By understanding the intricacies of these attacks and the strategies to mitigate them, we can enhance the resilience of our cryptographic systems and ensure the continued protection of our digital assets in the face of evolving threats.

# Table of Contents

*Introduction* ........................................................ 7

*Symmetric Cipher* ........................................................ 8

1. Brute force Attack: The Relentless Pursuit ........................................................ 9

2. Side-Channel Attacks: Exploiting the Unintended ........................................................ 10

3. Padding Oracle Attacks: Exploiting Implementation Flaws ........................................................ 11

Countermeasures: The Ongoing Arms Race ........................................................ 12

1. White-Box Cryptography ........................................................ 12

1. Secure Cipher Implementation ........................................................ 12

2. Authenticated Encryption ........................................................ 12

*Asymmetric Ciphers* ........................................................ 13

1. Mathematical Attacks: Targeting the Underlying Problem ........................................................ 14

1.1. Factorization Attacks on RSA ........................................................ 14

1.2. Common Factor Attack ........................................................ 14

2. Chosen Ciphertext Attack Exploiting Decryption Oracles ........................................................ 15

3. The Logjam Attack ........................................................ 16

Countermeasures: The Ongoing Arms Race ........................................................ 16

1. Strengthening the Mathematical Foundations ........................................................ 16

2. Secure Padding Schemes ........................................................ 17

3. ssHybrid Encryption: The Best of Both Worlds ........ 17

*Hash Functions* ........ *17*

1. Collision Attacks on MD5 and SHA-1 ........ 18

1.1. Birthday Attack ........ 19

1.2. Pre-image and Second Pre-image Attacks: The search for inputs ........ 19

2. Length Extension Attack ........ 20

*Quantum Computing: A new era of Cryptography* ........ *21*

*Conclusion* ........ *24*

*References* ........ *25*

# Table of Figure

**Figure** **page no.**

*Figure1: How symmetric cryptography works (Gupta, 2023)*........................................................ 8

*Figur2: Onetime Pad encryption decryption*.................................................................... 9

*Figure3: DES brute force attack*.......................................................................................... 10

*Figure4: Padding Oracle Attack on AES*........................................................................... 11

*Figure5: How asymmetric Encryption works*.................................................................. 13

*Figure6: Working of Logjam attack* ................................................................................... 16

*Figure7: Properties of a good hash function*.................................................................... 18

*Figure8: Colliding hash in md5*.......................................................................................... 19

*Figure9: Difference in quantum computing and classical computing (Woolnough et al., 2023)* 22

# Introduction

From the very simplest method of hiding the information to Julius Ceaser shifting each letter and then Vigenère introducing a key to lock the information we have a history of trying to protect information. And when we look at this day and age where the information people want to protect is everywhere and the modern mathematician is giving their everything to develop new algorithms every day. Cryptography is what we use to denote the art and science of secure communication that provides essential tools to protect information from who we don't want to see. The journey of cryptography has been one of relentless innovation and adaptation.

The evolution of cryptography has mirrored the advancements in comunications and technology from simple substitution ciphers to mechanical encryption and then ultimately to digital cryptographhy in this new era. With digitization the reliance on cryptography to secure information has increased a lot. Cryptography provides the bedrock upon which the security of our digital infrastructure rests, ensuring the confidentiality, integrity, and authenticity of information in an increasingly interconnected world.

However, the effectiveness of cryptography has always been challenged by people trying to break it to gain unauthorized access. This case study embarks on the journey of Cryptoanalysis which involves studying ciphers(focusing on three main pillars of modern cryptography): symmetric ciphers, asymmetric ciphers, and hash functions to uncover weakness or vulnerabilities in them. Furthermore, we  will dive into the attack methods used by attackers, dissect their potential impact and examine the countermeasure designed to defend againsts them.

## Symmetric Cipher

Symmetric ciphers is the most fundamental form of encryption, and its history goes back thousands of years back. Symmetric cryptography relies on a unique secrete key used for both encryption and decryption. In modern, Times Data Encrption Standard (DES) was the first standardized symmetric cipher (Grassi et al., n.d.). Other popular symetric alorithms include Advances Encryption Standard (AES) and Triple DES(3DES).

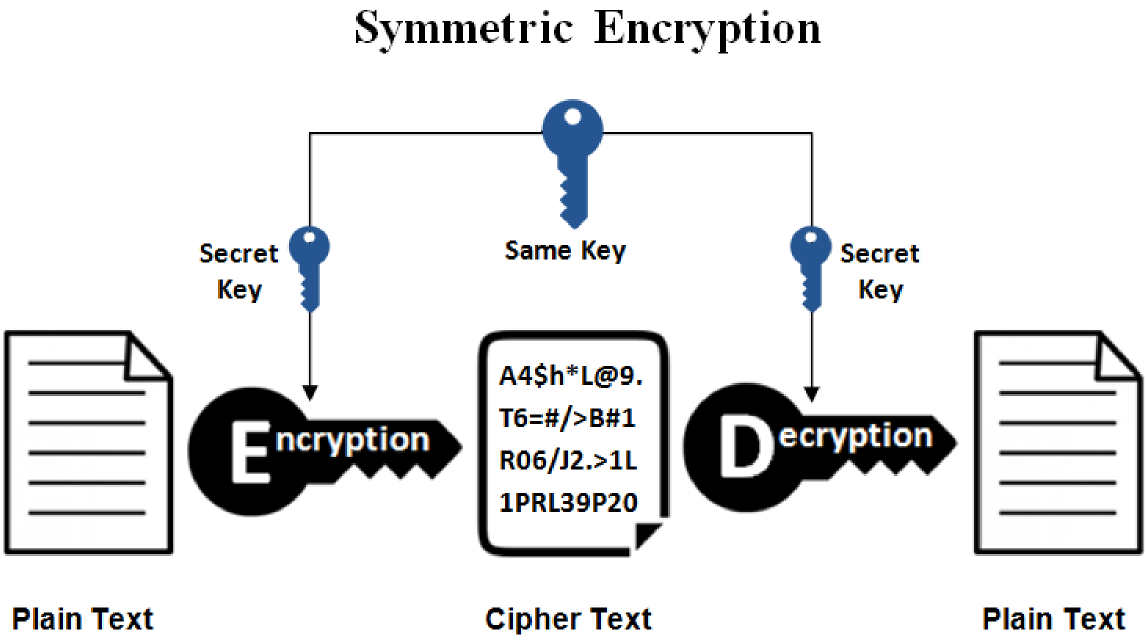
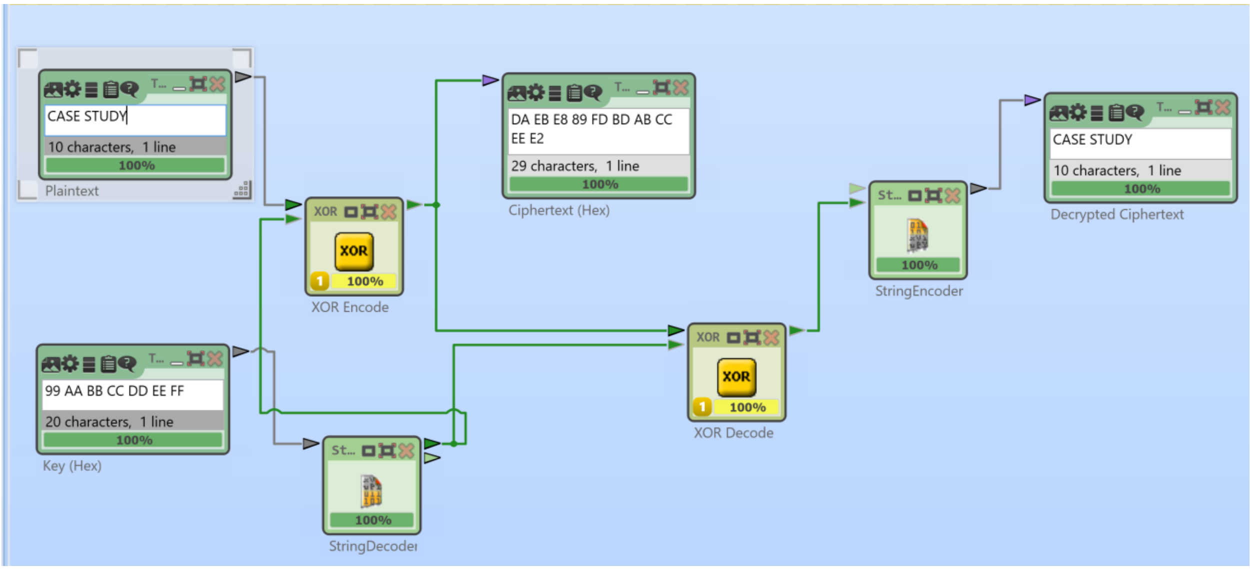


*Figure1: How symmetric cryptography works (Gupta, 2023)*

Symmetric encryption is simple. You have one key that encrypts your plain text to cipher text and the same key will decrypt it for you. Look at example below:



Plaintext: CASE STUDY
10 characters, 1 line
100%

Key (Hex): 99 AA BB CC DD EE FF
20 characters, 1 line
100%

Ciphertext (Hex): DA EB E8 89 FD BD AB CC EE E2
29 characters, 1 line
100%

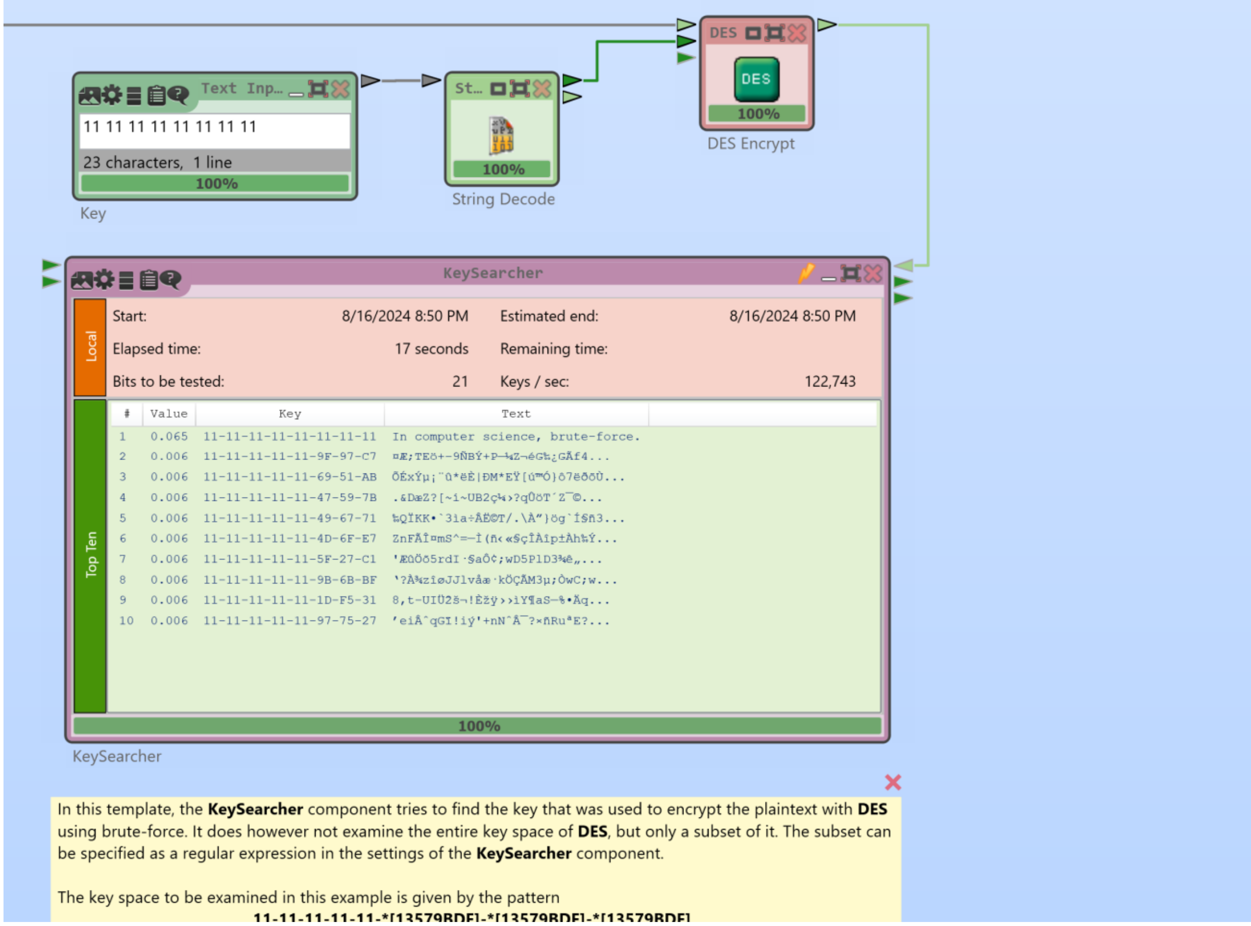
Decrypted Ciphertext: CASE STUDY
10 characters, 1 line
100%

*Figur2: Onetime Pad encryption decryption*

Since encryption is simple to implement so is to break it. The security of symmetric cipher is continuously challenged by an array of attacks.

### 1. Brute force Attack: The Relentless Pursuit

The brute force attack is simple and relentless strategy: trying every possible key until the correct one is discovered. The number of keys possible is theoretically infinite because it can be of 2^n long. For example, a 64-bit key has a 2^64 possible keys (Approximately 18 quintillion).Due to the very attack the widely used DES algorithm is now considered insecure as it used 56-bit-key which Deep Crack in 1998 cracked a DES- encrypted message in only 56 hours. (to, 2024).



11 11 11 11 11 11 11 11

23 characters, 1 line

| Start: | 8/16/2024 8:50 PM | Estimated end: | 8/16/2024 8:50 PM |
|---|---|---|---|
| Elapsed time: | 17 seconds | Remaining time: | |
| Bits to be tested: | 21 | Keys / sec: | 122,743 |

| # | Value | Key | Text |
|---|---|---|---|
| 1 | 0.065 | 11-11-11-11-11-11-11-11 | In computer science, brute-force. |
| 2 | 0.006 | 11-11-11-11-11-9F-97-C7 | ¤Æ;TEö+-9ÑBÝ+P–¼Z¬éG‰¿GÃf4... |
| 3 | 0.006 | 11-11-11-11-11-69-51-AB | ÕÉxÝµ¡¨û*ëÈ\|ÐM*EŸ[ú™Ó}ô7ëðõÙ... |
| 4 | 0.006 | 11-11-11-11-11-47-59-7B | .&DæZ?[~í~UB2ç¼›?qÛöT´Z¯©... |
| 5 | 0.006 | 11-11-11-11-11-49-67-71 | ‰QÏKK•`3ìa÷ÂË©T/.\À”}ög`Í§ñ3... |
| 6 | 0.006 | 11-11-11-11-11-4D-6F-E7 | ZnFÅÎ¤mS^=–Ì(ñ‹«§çÎÀîp±Àh‰Ý... |
| 7 | 0.006 | 11-11-11-11-11-5F-27-C1 | 'ÆûÖõ5rdI·§aÔ¢;wD5PlD3¾ê„... |
| 8 | 0.006 | 11-11-11-11-11-9B-6B-BF | '?Â¾zîøJJlvåæ·kÖÇÃM3µ;ÒwC;w... |
| 9 | 0.006 | 11-11-11-11-11-1D-F5-31 | 8,t-UIÜ2š¬!Èžÿ››ìY¶aS–%•Ãq... |
| 10 | 0.006 | 11-11-11-11-11-97-75-27 | ′eiÂˆqGI!iý'+nNˆÃ¯?×ñRuªE?... |

100%

In this template, the **KeySearcher** component tries to find the key that was used to encrypt the plaintext with **DES** using brute-force. It does however not examine the entire key space of **DES**, but only a subset of it. The subset can be specified as a regular expression in the settings of the **KeySearcher** component.

The key space to be examined in this example is given by the pattern

**11-11-11-11-11-*[13579BDF]-*[13579BDF]-*[13579BDF]**

*Figure3: DES brute force attack*

## 2. Side-Channel Attacks: Exploiting the Unintended

A side channel attack is not aa single attack but a family of attacks. These attack don’t directly aim to find weakness in the algorithms but rather observe the cipher’s behavior during encryption and decryption. These observations reveal valuable clues about the secret key or encrypted data. This attack is based on the principle that all algorithms leave a trace whether it may be power consumption, electromagnetic emissions or timing differences.

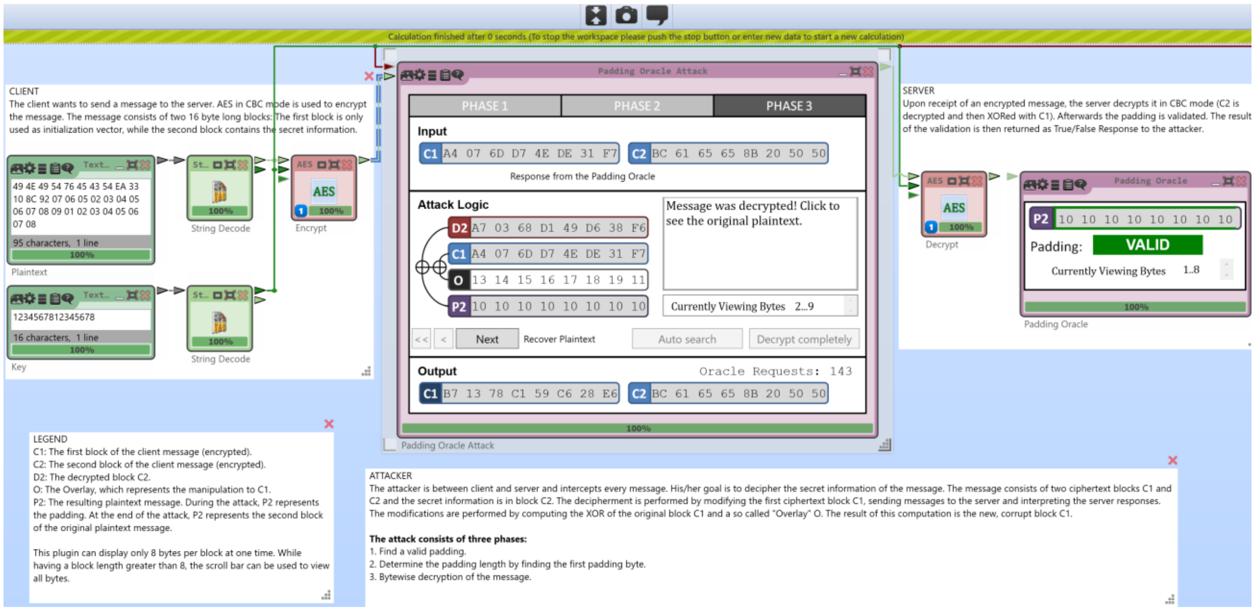
This attack has been successfully implemented in what we consider a secure algorithm which is AES-256. A power analysis attack on a bootloader which used 256 bit was successfully carried out by measuring the power consumption of device during encryption and decryption .(Flynn &

Chen, n.d.). It may need a few extra devices to measure various data but it is a very effective attack type on symmetric cipher.

### 3. Padding Oracle Attacks: Exploiting Implementation Flaws

This attack is very sophisticated and targets the implementation of block cipher modes that requrie padding such as CBC(Cipher Block Chaining). The attack exploits vulnerabilities in how the system handles padding errors to decrypt messages without knowing the secret key.

The attack leverages an "oracle," which is a system or component that inadvertently reveals information about the validity of padding in a ciphertext. The attacker sends carefully crafted ciphertexts to the oracle, observing its responses to different padding modifications. By analyzing these responses, the attacker can systematically deduce information about the plaintext, eventually decrypting the entire message.



Calculation finished after 0 seconds (To stop the workspace please push the stop button or enter new data to start a new calculation)

CLIENT
The client wants to send a message to the server. AES in CBC mode is used to encrypt the message. The message consists of two 16 byte long blocks: The first block is only used as initialization vector, while the second block contains the secret information.

Plaintext:
```
49 4E 49 54 76 45 43 54 EA 33
10 8C 92 07 06 05 02 03 04 05
06 07 08 09 01 02 03 04 05 06
07 08
```
95 characters, 1 line

Key:
```
1234567812345678
```
16 characters, 1 line

Padding Oracle Attack

Input
```
C1 A4 07 6D D7 4E DE 31 F7    C2 BC 61 65 65 8B 20 50 50
```
Response from the Padding Oracle

Attack Logic
```
D2 A7 03 68 D1 49 D6 38 F6
C1 A4 07 6D D7 4E DE 31 F7
O  13 14 15 16 17 18 19 11
P2 10 10 10 10 10 10 10 10
```
Message was decrypted! Click to see the original plaintext.

Currently Viewing Bytes 2...9

Output                                    Oracle Requests: 143
```
C1 B7 13 78 C1 59 C6 28 E6    C2 BC 61 65 65 8B 20 50 50
```

SERVER
Upon receipt of an encrypted message, the server decrypts it in CBC mode (C2 is decrypted and then XORed with C1). Afterwards the padding is validated. The result of the validation is then returned as True/False Response to the attacker.

Padding Oracle
```
P2 10 10 10 10 10 10 10 10
```
Padding: VALID

Currently Viewing Bytes 1..8

LEGEND
C1: The first block of the client message (encrypted).
C2: The second block of the client message (encrypted).
D2: The decrypted block C2.
O: The Overlay, which represents the manipulation to C1.
P2: The resulting plaintext message. During the attack, P2 represents the padding. At the end of the attack, P2 represents the second block of the original plaintext message.

This plugin can display only 8 bytes per block at one time. While having a block length greater than 8, the scroll bar can be used to view all bytes.

ATTACKER
The attacker is between client and server and intercepts every message. His/her goal is to decipher the secret information of the message. The message consists of two ciphertext blocks C1 and C2 and the secret information is in block C2. The decipherment is performed by modifying the first ciphertext block C1, sending messages to the server and interpreting the server responses. The modifications are performed by computing the XOR of the original block C1 and a so called "Overlay" O. The result of this computation is the new, corrupt block C1.

**The attack consists of three phases:**
1. Find a valid padding.
2. Determine the padding length by finding the first padding byte.
3. Bytewise decryption of the message.

*Figure4: Padding Oracle Attack on AES*

Th attack has been successfully exploited in real world scenarios notably against SSL3.0. This POODLE attack allowed attackers to decrypt sensitive information, such as session cookies, transmitted over supposedly secure connections (Tomasz Andrzej Nidecki, 2020). The discovery of this vulnerability led to the deprecation of SSL 3.0 and a widespread shift towards more secure protocols like TLS.

### Countermeasures: The Ongoing Arms Race

Symmetric Encryption carries a historic value as well as modern mathematicians has implmeneted it using complex algorithms in modern time yet the use of one secret key will always be considered not so secure. The fast encryption performed by symmetric ciphers still makes it relevant and many important protocols like SSL and TLS utilize it. So inorder to counter the attacks research have introduced new techniques to counter these attacks:

1. **White-Box Cryptography**

   White-box cryptography seeks to protect cryptographic keys even in environments where the attacker has full visibility and control over the implementation. This approach involves obfuscating the key and the cryptographic algorithm's operations, making it extremely challenging for attackers to extract the key even with complete access to the code and execution environment.

1. **Secure Cipher Implementation**

   Numerous techniques like masking, hiding and randomizing operation have been introduced in ciphers to mitigate side-channel information reading.

2. **Authenticated Encryption**

   The combination of encryption and authentication provides an additional layer of security. Authenticated encryption modes, such as Galois/Counter Mode (GCM) or

Counter with Cipher Block Chaining Message Authentication Code (CCM), ensure both the confidentiality and integrity of data.

## Asymmetric Ciphers

Asymmetric Ciphers, also known as public-key cryptography introduced a never heard of concept in cryptography by employing a pair of keys: a public key for encryption and a private key for decryption. This enables secure key communication without the need for secret key exchange like in symmetric ciphers making it far secure. The most widely used asymmetric cipher is Rivest-Shamir-Adleman (RSA) (Mirza et al., 2002) which relies on factoring large composite numbers. Other asymmetric Ciphers include Diffie-Hellman key exchange and Digital Signature Algorithms.

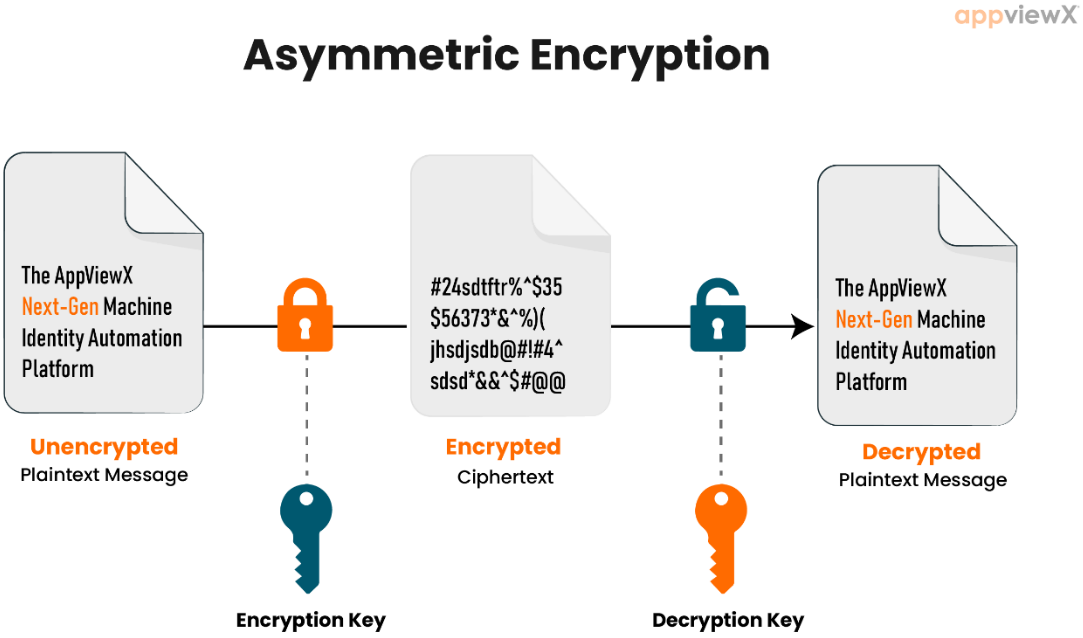


*Figure5: How asymmetric Encryption works*

## 1. Mathematical Attacks: Targeting the Underlying Problem

The security of asymmetric ciphers is connected to the hard mathematical problem, so the attackers focus on targeting them rather than directly targeting algorithms. The security of RSA hinges on the difficulty of factoring large composite numbers into their prime factors whereas Elliptic Curve Cryptography (ECC) relies on the difficulty of solving the discrete logarithm problem on elliptic curves.

### 1.1. Factorization Attacks on RSA

The security of RSA relies on the difficulty of factoring the modulus 'N', which is the product of two large prime numbers 'p' and 'q'.

If an attacker can factor 'N', they can easily compute the private key 'd'.  When the gap between ‘p’ and ‘q’ are very small this attack can be executed.

A team of researchers in the late 90s successfully factored a 512-bit RSA modulus (Cavallar et al., n.d.). Back then it took them a few days but now with the hardware we have its matter of small time. SO today even 1024-bit RSA are considered vulnerable.

### 1.2. Common Factor Attack

This attack exploits the scenario where multiple RSA keys share one of the prime factors which may happen due to poor random number generation or some sort of implementation error. When two or more RSA moduli share a prime factor, an attacker can efficiently discover this shared factor using the greatest common divisor (GCD) algorithm. Once they have the shared prime, they can divide each modulus by it to obtain the other prime factor. With both primes in hand, they can break each affected RSA key.

Let’s suppose a scenario:

Shared prime: `p = 17`

Ram’s other prime: `q1 = 23`

Sita’s other prime: `q2 = 29`

Ram’s modulus: `n1 = p * q1 = 17 * 23 = 391`

Sita’s modulus: `n2 = p * q2 = 17 * 29 = 493`

Now what the attacker will do is obtain their public keys which have their moduli: n1 = 391 and n2 = 492. The attacker calculates the Greatest common divisor(GCD) of n1 and n2

Gcd(391,493) = 17

The attacker with this discovered share prime factor will now easily calculate the private keys and decrypt both the user’s message.

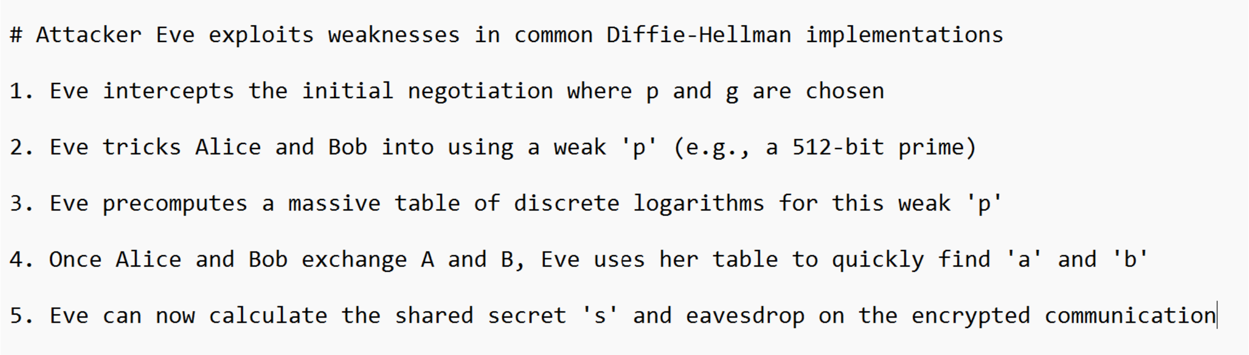
**2. Chosen Ciphertext Attack Exploiting Decryption Oracles**

Here the attacker gathers information by obtaining decryption of chosen ciphertexts and from this information the attacker will try to recover secret key used for decrypting. It has a category which is Adaptive Chosen Ciphertext Attacks where the attacker adaptively chooses ciphertexts to be decrypted by the oracle, using the information gained from previous decryptions to inform the selection of subsequent ciphertext.

The Bleichenbacher's Attack successfully compromised PCKS #1 sending carefully crafted error ciphertexts to decryption oracle and then analyzing the error message. This highlighted the importance of secure padding and to avoid revealing information about decryption to potential attackers.

### 3. The Logjam Attack

The logjam attack demonstrated that many Diffie Hellman Implementation were vulnerable due to the use to weak parameters (Cunningham, 2018). It highlighted the importance of using strong prime numbers (at least 2048 bits) and regularly updating cryptographic libraries.



```
# Attacker Eve exploits weaknesses in common Diffie-Hellman implementations

1. Eve intercepts the initial negotiation where p and g are chosen

2. Eve tricks Alice and Bob into using a weak 'p' (e.g., a 512-bit prime)

3. Eve precomputes a massive table of discrete logarithms for this weak 'p'

4. Once Alice and Bob exchange A and B, Eve uses her table to quickly find 'a' and 'b'

5. Eve can now calculate the shared secret 's' and eavesdrop on the encrypted communication
```

*Figure6: Working of Logjam attack*

### Countermeasures: The Ongoing Arms Race

While the asymmetric ciphers were revolutionary and strong eventually followed by attackers trying to break them and they have been very successful in this pursuit too. But the cryptographers has also been able to introduce counter measures against it.

### 1. Strengthening the Mathematical Foundations

The primary defense against mathematical attacks lies in increasing the key sizes of asymmetric ciphers. The current recommendation for RSA is to use key sizes of at least 2048 bits, and even larger key sizes may be necessary in the future as computing power continues to advance. The selection of strong prime numbers during key generation is also crucial.

**2. Secure Padding Schemes**

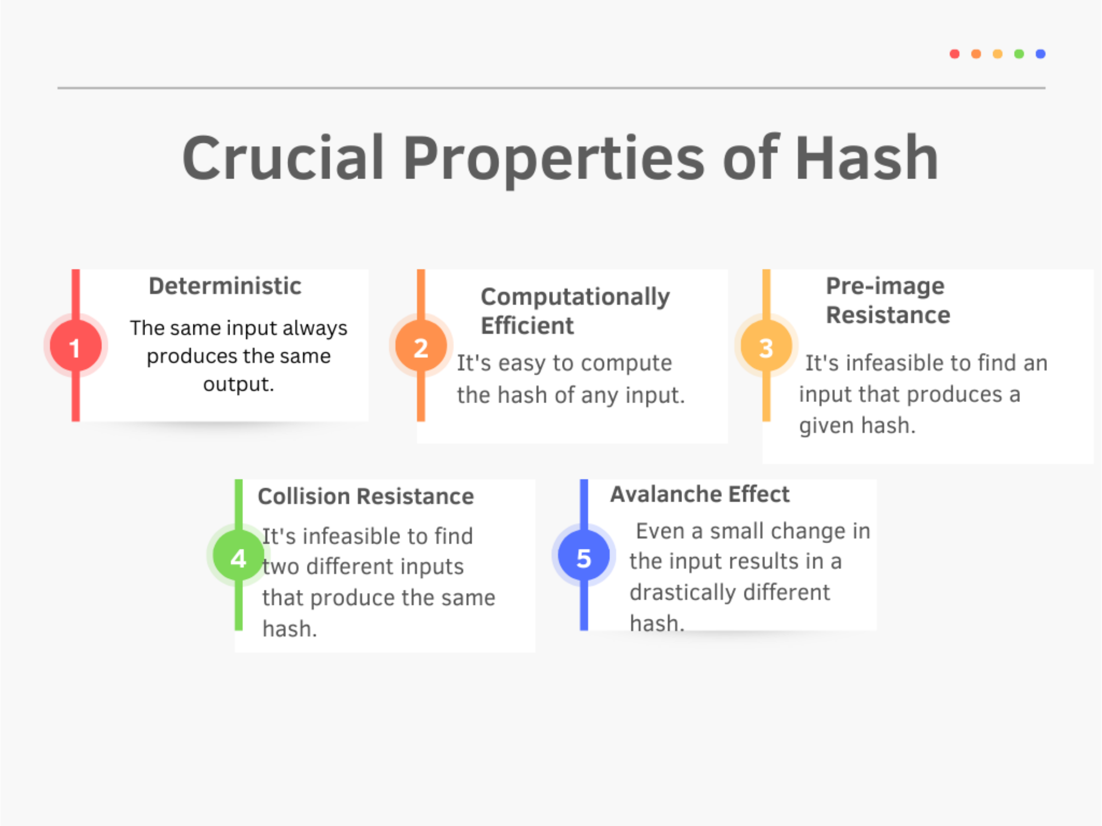
The padding schemes like Optimal Asymmetric Encryption Padding (OAEP) is introduced to prevent attackers from exploiting decryption oracles and manipulating ciphertext.

**3. ssHybrid Encryption: The Best of Both Worlds**

Hybrid encryption combines the strengths of symmetric and asymmetric ciphers to provide a balanced approach to security and efficiency. In this scheme, asymmetric encryption is used to securely exchange a symmetric key, which is then used for bulk data encryption. This approach leverages the speed of symmetric ciphers for data encryption while relying on the security of asymmetric ciphers for key exchange.

## Hash Functions

Hash functions—such as MD5, SHA-1,SHA-256, SHA-3, and BLAKE2—comprise the cryptographer's Swiss Army Knife: they are used in digital signatures, public-key encryption, integrity verification, message authentication, password protection, key agreement protocols, and many other cryptographic protocols(Aumasson & Green, n.d, 2018.)Unlike the ciphers we have discussed caretes long outputs hash function takes any length of inputs and produce a fix short value known as hash value or digest.



## Crucial Properties of Hash

1 **Deterministic**
The same input always produces the same output.

2 **Computationally Efficient**
It's easy to compute the hash of any input.

3 **Pre-image Resistance**
It's infeasible to find an input that produces a given hash.

4 **Collision Resistance**
It's infeasible to find two different inputs that produce the same hash.

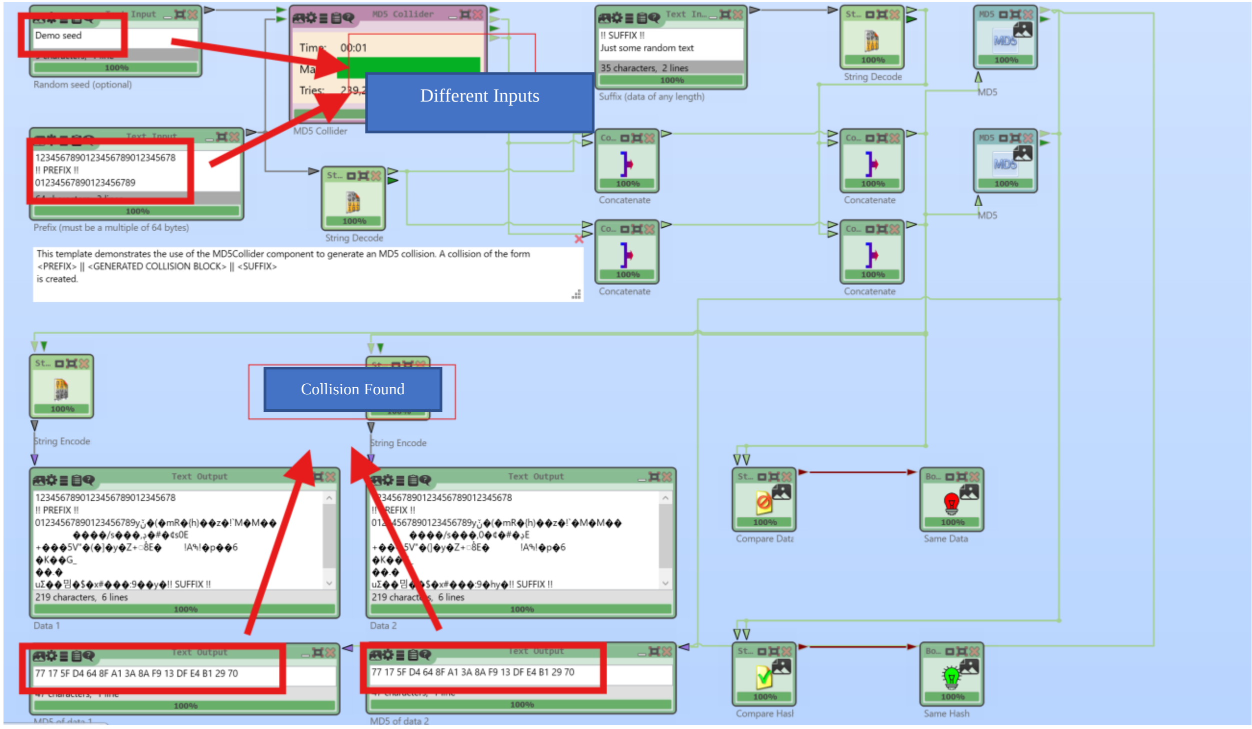
5 **Avalanche Effect**
Even a small change in the input results in a drastically different hash.

*Figure7: Properties of a good hash function*

The application of hash function is very huge from every small ascpect of computer security to large ones. So for attackers attacking hash function is very worthy vector for attack. There are few attacks proved to have compromised few of the widely used hash function:

### 1. Collision Attacks on MD5 and SHA-1

A collision attack is performed by giving two inputs and trying to produce the same hash value. In figure7 we talked about how collision resistance is one of the key properties of a Hash function.



```
Demo seed
```

```
12345678901234567890123456789
!! PREFIX !!
01234567890123456789
```

```
!! SUFFIX !!
Just some random text
35 characters,  2 lines
```

This template demonstrates the use of the MD5Collider component to generate an MD5 collision. A collision of the form
<PREFIX> || <GENERATED COLLISION BLOCK> || <SUFFIX>
is created.

```
12345678901234567890123456789
!! PREFIX !!
01234567890123456789y�(�mR�(h)��z�!`M�M��
        ����/s���,�#�¢s0E
+���5V"�(�]�y�Z+8E�      !A%!�p��6
�K��G_
��.�
uΣ��$�x#���:9��y�!! SUFFIX !!
219 characters,  6 lines
```

```
77 17 5F D4 64 8F A1 3A 8A F9 13 DF E4 B1 29 70
```

```
12345678901234567890123456789
!! PREFIX !!
01234567890123456789y�(�mR�(h)��z�!`�M�M��
        ����/s���,0�¢�#�E
+���5V"�()�y�Z+8E�      !A%!�p�6
�K��
��.�
uΣ���$�x#���:9�hy�!! SUFFIX !!
219 characters,  6 lines
```

```
77 17 5F D4 64 8F A1 3A 8A F9 13 DF E4 B1 29 70
```

*Figure8: Colliding hash in md5*

## 1.1.Birthday Attack

It is a brute force collision attack that exploits mathematics behind birthday problem in probability theory which states that i a group of 23 people, there's a greater than 50% chance of two people sharing same birthday (GeeksforGeeks, 2018). So, by hashing many random inputs the attacker increases the chance of finding two input with same hash. For example, finding collision of hash function with a 32-bit output requires 2^16 hash computation.

## 1.2.Pre-image and Second Pre-image Attacks: The search for inputs

Pre-image attacks and second pre-image attacks target the one-wayness property of hash functions. In a pre-image attack, the attacker is given a hash value and tries to find an input that produces that hash. In a second pre-image attack, the attacker is given an input and its corresponding hash value and tries to find a different input that produces the same hash.

Both attacks are computationally challenging for secure hash functions. However, weaknesses in the design or implementation can sometimes be exploited to make these attacks more feasible. Successful pre-image attacks can compromise password security, as attackers can potentially recover passwords from their stored hash values. Second pre-image attacks can be used to forge digital signatures or tamper with data integrity checks.

### 2. Length Extension Attack

These attacks exploit the structure of certain hash functions, such as those based on the Merkle-Damgård construction, to append data to a message without knowing the original key. The attacker can compute the new hash value based on the original hash, the appended data, and the length of the original message. This attack can be particularly dangerous in scenarios where the hash function is used for message authentication, as it allows an attacker to modify a message and its corresponding authentication tag without detection.

The use of hash function is very wide, so it has been a challenge for cryptographers to keep it safe while the technology around us is become powerful day by day. Yet efforts have been constantly put to improve it.

### Hash Function Design: Building Resilience

Modern hash functions like SHA-3 and BLAKE2 incorporate design principles that address the weaknesses of older hash functions. They employ techniques like sponge constructions (in, 2020), which offer greater resistance to collision and pre-image attacks. Additionally, they incorporate features like length hiding and salting to further enhance their security.

Length extension attacks, which exploit the structure of Merkle-Damgård based hash functions, are mitigated in modern designs. SHA-3, for example, uses a sponge construction that inherently prevents length extension attacks. Salting, the process of adding random data to the input before hashing, is commonly used in password storage to protect against pre-computed rainbow table attacks.

The ongoing research and development in hash function design reflect the continuous effort to stay ahead of evolving attack techniques. By incorporating robust design principles and addressing known vulnerabilities, modern hash functions provide a strong foundation for secure cryptographic applications in the face of ever-increasing computational power and sophisticated attack methods.

## Quantum Computing: A new era of Cryptography

The current cryptography is heavily guarded by the limitations of classical computers. Symmetric ciphers like AES, asymmetric ciphers like RSA and ECC, and hash functions like SHA-256 are considered secure due to the immense computational power required to break them. However, the growing improvements in making a of quantum computing threatens to disrupt this status.

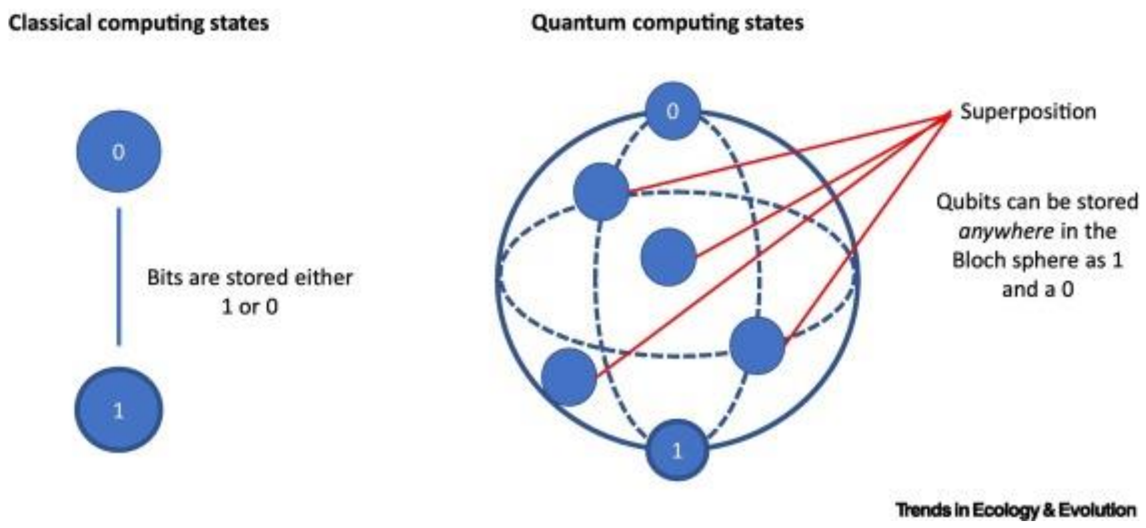

Classical computing states
Quantum computing states
0
1
Bits are stored either 1 or 0
0
1
Superposition
Qubits can be stored *anywhere* in the Bloch sphere as 1 and a 0
Trends in Ecology & Evolution

*Figure9: Difference in quantum computing and classical computing (Woolnough et al., 2023)*

Quantum computers use the principle of quantum mechanics to perform computations exponentially faster than modern computers. This advantage poses a significant risk to all the current cryptographic ciphers we consider secure now. Shor’s algorithm for instance can theoretically break RSA and ECC in polynomial time, rendering them unusable now. (What Is Shor’s Algorithm, 2023). Similarly, Grover's algorithm can speed up brute-force attacks on symmetric ciphers, reducing their effective key sizes. (“Quantum Computing,” 2019).

It's not like the algorithms of quantum computers are good for breaking stuff only rather they are also very safe in protecting our information. The researchers of cryptography community are actively developing quantum resistant algorithms, also know as post-quantum cryptography. These algorithms are designed to withstand attacks from both classical and quantum computers. (and, 2024)

The transition to post-quantum cryptography will be a complex undertaking. It will require updating existing infrastructure, developing new protocols, and ensuring backward compatibility.

However, it is crucial to start this process now to avoid a potential cryptographic apocalypse when quantum computers become powerful enough to break current encryption standards. The future of secure communication and data protection hinges on our ability to adapt to the quantum revolution.

# Conclusion

Cryptography they say is the strongest link in the chain for computer security yet so much time is invested in improving it daily. The never ending pursuit of cracking encrypted messages and forging digital signatures continues, driving the development of new cryptographic techniques and the ongoing battle to safeguard sensitive information in the digital age. The attacks and countermeasures explored in this case study underscore the dynamic nature of cryptography and the importance of continuous research and vigilance. The new era about to come will bring so many changes with it. So, By understanding the vulnerabilities of cryptographic systems and adopting a multi-layered approach to security, we can improve our defenses and protect the confidentiality, integrity, and authenticity of our digital assets.

# References

- Flynn, C., & Chen, D. (n.d.). Side Channel Power Analysis of an AES-256 Bootloader. Retrieved August 16, 2024, from https://eprint.iacr.org/2014/899.pdf
- to, C. (2024). EFF DES cracker. Crypto Wiki; Fandom, Inc. https://cryptography.fandom.com/wiki/EFF_DES_cracker
- Chaigneau, C. (2018). Cryptanalysis of symmetric encryption algorithms. Hal.science. https://theses.hal.science/tel-02012149
- Grassi, L., Manterola Ayala, I., Hovd, M., Øygarden, M., Raddum, H., & Wang, Q. (n.d.). Cryptanalysis of Symmetric Primitives over Rings and a Key Recovery Attack on Rubato. Retrieved August 16, 2024, from https://eprint.iacr.org/2023/822.pdf
- Tomasz Andrzej Nidecki. (2020, June). What Is the POODLE Attack? | Acunetix. Acunetix. https://www.acunetix.com/blog/web-security-zone/what-is-poodle-attack
- Mirza, D. R., Ido Dubrawsky, Flynn, H., Joe Kingpin Grand, Graham, R., Johnson, N. L., Dan Effugas Kaminsky, F. William Lynch, Manzuik, S. W., Permeh, R., Pfeil, K., & Russell, R. (2002). Cryptography. Elsevier EBooks, 165–203. https://doi.org/10.1016/b978-192899470-1/50009-4
- Cavallar, S., Dodson, B., Lenstra, A., Lioen, W., Montgomery, P., Murphy, B., Te Riele, H., Aardal, K., Gilchrist, J., Guillerm, G., Leyland, P., Marchand, J., Morain, F., Muffett, A., Putnam, C., & Zimmermann, P. (n.d.). Factorization of a 512-bit RSA Modulus. https://www.iacr.org/archive/eurocrypt2000/1807/18070001-new.pdf
- c0D3M. (2019, October 10). *Bleichenbacher Attack Explained - c0D3M - Medium*. Medium; Medium. https://medium.com/@c0D3M/bleichenbacher-attack-explained-bc630f88ff25
- Cunningham, K. (2018, November 8). *Logjam is a security vulnerability against a Diffie–Hellman key exchange ranging from 512-bit (US export-grade) to 1024-bit keys. It was*

*discovered by a group of computer scientists and publicly reported on May 20, 2015*. Linkedin.com. https://www.linkedin.com/pulse/logjam-2015-kurtis-cunningham

- GeeksforGeeks. (2018, September 10). Birthday attack in Cryptography. GeeksforGeeks; GeeksforGeeks. https://www.geeksforgeeks.org/birthday-attack-in-cryptography/
- in. (2020, August 6). *What is the sponge construction in simple terms?* Cryptography Stack Exchange. https://crypto.stackexchange.com/questions/83258/what-is-the-sponge-construction-in-simple-terms
- Quantum Computing. (2019). In *National Academies Press eBooks*. https://doi.org/10.17226/25196
- and, S. (2024). *Post-Quantum Cryptography | CSRC*. Nist.gov. https://csrc.nist.gov/projects/post-quantum-cryptography
- *What is Shor's Algorithm*. (2023). Quera.com. https://www.quera.com/glossary/shors-algorithm#:~:text=Shor's%20Algorithm%2C%20named%20after%20mathematician,known%20classical%20algorithms%20for%20factoring.
- Woolnough, A. P., Lloyd C.L. Hollenberg, Cassey, P., & Thomas A.A. Prowse. (2023). Quantum computing: a new paradigm for ecology. *Trends in Ecology & Evolution*, *38*(8), 727–735. https://doi.org/10.1016/j.tree.2023.04.001