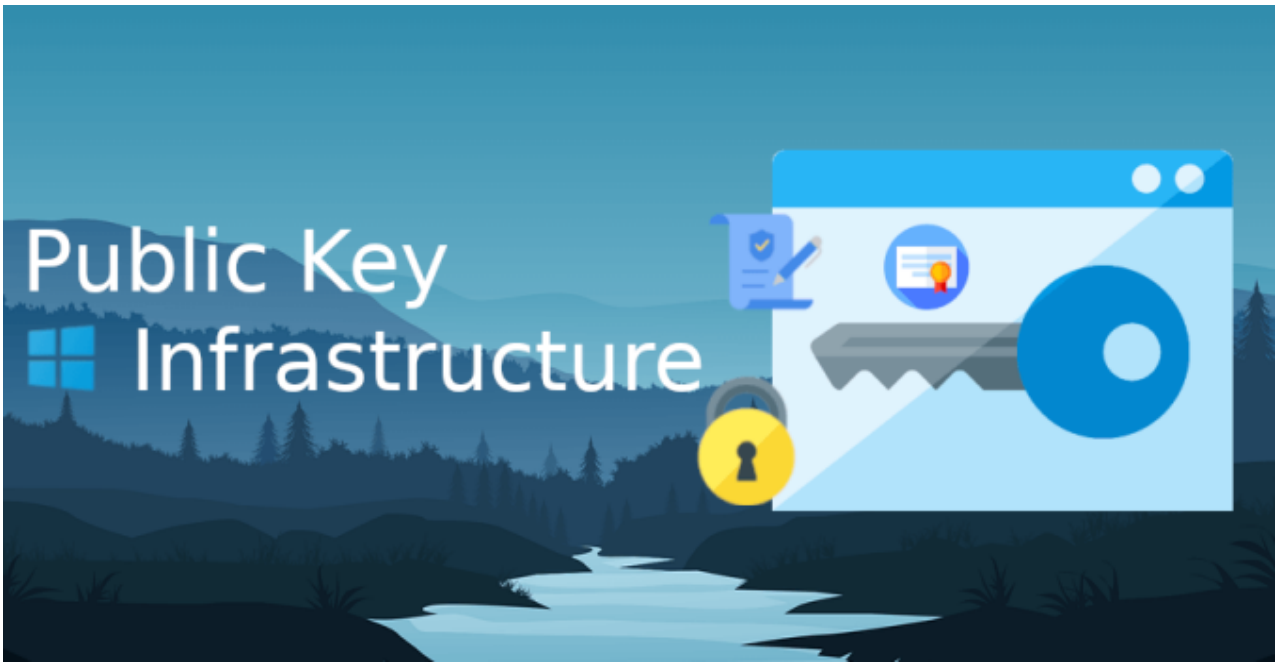
# PKI – Part 2: Choosing the key length and algorithm

michaelwaterman.nl/2023/09/08/pki-part-2-choosing-the-key-length-and-algorithm

Michael Waterman September 8, 2023




As I advance into the heart of PKI in this second installment, the focal point shifts towards two important decisions that anyone involved in setting up a PKI must deal with, choosing the appropriate key length and the cryptographic algorithm. These choices are far from arbitrary, for they are the foundation of a secure and efficient PKI. The selection at this stage will determine the resilience against cyber treats for the foreseeable future.

(*updated on 05-05-2025*)

But why is the choice of key length and algorithm so important? In the simplest terms, the key length and the algorithm act as both the shield and the sword , protecting sensitive data from unauthorized access and ensuring the authenticity and integrity of the digital communications.

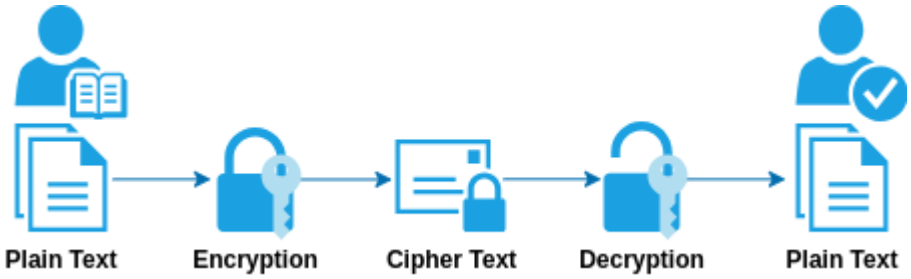
The key length dictates the cryptographic strength, a longer key offering a better defense against brute force attacks. Meanwhile, the choice of algorithm influences not just the security but also the speed of encryption and decryption processes. It’s a delicate balancing act between security and performance.

## Understanding cryptography basics

### Text to Ciphertext: An overview

Before I delve into the more intricate aspects of PKI, it’s essential to start with the basics of cryptography. At its core, cryptography is the art of securing communication. When I encrypt plaintext (readable data) into ciphertext (encoded data), I shield the information,

making it inaccessible to unauthorized parties. This conversion is executed through a systematic algorithmic process where the plaintext is transformed using a key, a secret parameter known only to the communicating parties.




This secure method of transmission ensures that your confidential data remains just that, confidential.

### Simple encryption example: Caesar Cipher

To understand encryption better, let's look at one of the simplest encryption techniques – the Caesar cipher, named after Julius Caesar, who reportedly used it to communicate confidentially with his generals.

The Caesar cipher is a type of substitution cipher where each character in the plaintext is 'shifted' a certain number of places down or up the alphabet. For instance, with a shift of 3, 'A' would be replaced by 'D', 'B' with 'E', and so on.

Let's take a plain text: "HELLO". With a shift of 3 places, it would be encrypted to "KHOOR".

This method, although straightforward and easy to decipher in the modern age, lays the foundation of understanding more complex encryption methods.

### Defining and understanding key length

It's important to understand the concept of 'key length.' The 'key' is a piece of information utilized by the encryption algorithm to transform plaintext into ciphertext and vice versa during decryption. The 'key length' refers to the number of bits in the key. Typically, a longer key length denotes a higher level of security, as it creates a broader range of potential keys, making it exponentially harder for unauthorized parties to crack the code.

However, a longer key length also demands more computational power, creating a need to balance between security and efficiency. In the coming sections, I will analyze the different key lengths and how to choose the one that suits your requirements best.

## RSA and ECC – A comparative analysis

### Overview of RSA and its mathematical foundation

**RSA**, named after its inventors Rivest-Shamir-Adleman, stands as one of the first and most widely used public-key cryptosystems. It operates based on the mathematical properties of prime numbers (*a number that can only be divided by itself and 1 without remainders*) and their usage in factorization problems.

At its core, RSA involves three primary steps: key generation, encryption, and decryption. The security of RSA is rooted in the difficulty of factorizing the product of two large prime numbers, a problem known to be hard to solve, hence providing a secure foundation for encrypting information.

A vital concept here is the key length, which essentially refers to the size of the prime numbers used. Longer key lengths imply a higher security level but come at the cost of computational efficiency.

## Overview of ECC and its mathematical foundation

**Elliptic Curve Cryptography (ECC)**, on the other hand, is a newer approach leveraging the algebraic structure of elliptic curves over finite fields. This method promises the same level of security as RSA but requires a much smaller key size, hence resulting in faster computations and reduced storage and memory usage.

The security of ECC is grounded in the elliptic curve discrete logarithm problem, a complex problem to solve, providing a robust defense against potential attacks.

## Key lengths: RSA vs. ECC

When it comes to choosing between RSA and ECC, one critical aspect to consider is the key length. Generally, ECC offers a higher security level per bit compared to RSA, meaning you need a much smaller key size with ECC to achieve the same security level you'd get with a much larger RSA key.

For instance, a 256-bit key in ECC is generally considered to be as secure as a 3072-bit key in RSA. This discrepancy in key length allows ECC to facilitate secure communications with less computational power, making it a more efficient option, especially for systems with limited resources (mobile, IOT, etc).

## Computational effort required for RSA and ECC

While determining the cryptographic system to use, understanding the computational effort required for each is paramount. RSA, with its longer key lengths, generally demands more computational power, leading to slower encryption and decryption processes compared to ECC.

ECC, with its shorter key lengths, facilitates faster encryption and decryption processes, offering a more streamlined and efficient solution, albeit with a different foundational mathematical approach.

In understanding the computational aspects of both RSA and ECC, we need to discuss the computational effort in terms of both key length and CPU cycles. Let's delve into two tables that detail these aspects, helping us draw a comparative analysis:

**Table 1: Key length comparison**

In this table, I will illustrate the comparative key lengths between RSA, ECC, and their symmetric key counterparts. It provides a direct comparison, helping in understanding the kind of security level you'd be getting with different key lengths.

In the table:

- **RSA key length**: These are common key lengths used in RSA encryption, with longer key lengths generally offering stronger security but at the cost of increased computational resources.
- **ECC key length**: These ranges represent the approximate ECC key lengths that offer security comparable to the RSA key lengths in the adjacent cell.
- **Symmetric key equivalent**: This column provides the symmetric key lengths (like those used in AES encryption) that offer security roughly equivalent to the RSA and ECC key lengths in the same row.

This table gives a ballpark comparison, aiding in understanding the relative strengths of different key lengths across these cryptographic systems. It's based on the general consensus among cryptographic communities, and exact equivalences might vary based on specific implementations and updates in cryptographic research.

**Table 2: CPU cycles comparison**

Following the key length comparison, the second table will focus on the approximate number of CPU cycles required to perform cryptographic operations with RSA and ECC at various key lengths. It is important to note that these values can be somewhat speculative and vary greatly depending on the specific circumstances, including the hardware and software involved.

*The "Relative computational effort" column shows how much more processing power RSA requires compared to ECC for equivalent security — higher numbers indicate RSA is less efficient.*

However, these values are rough estimates. They're based on general benchmarks and available information and can vary depending on specific implementations, software, hardware, and other factors.

In comparison between the computational effort required for both ECC and RSA, it is clear that ECC provides a more efficient performance, requiring fewer computational cycles for encryption processes with comparable security levels offered by RSA. Despite

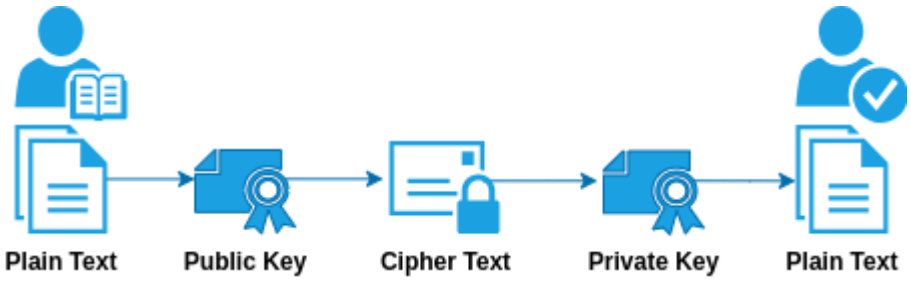
its computational efficiency, ECC's adoption rate is still tempered by compatibility and integration complexities in existing systems. Therefore, while ECC stands as a powerful, efficient alternative to RSA.

## The world of algorithms

Algorithms play a crucial role in ensuring the security and confidentiality of information. The two primary types of algorithms are symmetric and asymmetric.

**Symmetric algorithms** use the same key for both encryption and decryption. This type of encryption is faster and requires less computational power. However, the primary challenge here is the secure distribution of the key; since it is the same for both processes, it must be kept highly secure to prevent unauthorized access. I wrote a blog on it, if you want to know how a secure exchange works.

**Asymmetric algorithms**, on the other hand, employ a pair of keys: a public key, which is shared openly, and a private key, which remains confidential. The public key encrypts data, while the private key decrypts it. Despite being more secure due to the two-key system, asymmetric encryption demands more computational resources, making it slower compared to symmetric encryption.




Understanding the differences and unique benefits of these algorithms is fundamental in the secure and efficient transmission of data in various systems.

### Introducing Suite B and its successor, the Commercial National Security Algorithm suite

**Suite B**, a set of cryptographic algorithms published by the National Institute of Standards and Technology (NIST). Suite B, introduced in 2005, included specifications for encryption, hashing, digital signatures, and key exchange.

Following Suite B, the **Commercial National Security Algorithm Suite (CNSA)** took the reins as of 2015, superseding Suite B and introducing updated standards to ensure heightened security in cryptographic communications. This suite guides the secure communications within the (US) government and between government entities and their partners in the commercial sector. It presents a more robust set of guidelines, incorporating advancements and lessons learned since the days of Suite B. A few key improvements:

**Stronger cryptographic algorithms**:

- **Encryption**: Suite B used AES with key sizes of 128 and 256 bits for top secret information. CNSA continued using AES but emphasized on 256-bit keys for a higher security margin.
- **Digital Signatures**: While Suite B employed ECDSA, CNSA transitioned to using Digital Signature Algorithm (DSA) alongside ECDSA to offer more secure options.
- **Key Exchange**: Suite B used ECDH, a trend that CNSA continued but with more secure parameter sets to enhance security.

**Enhanced hash functions**:

> **Hash Functions**: SHA-256 and SHA-384 were the hash functions advocated by Suite B. CNSA elevated this recommendation to SHA-384 to ensure a higher security level.

**Forward secrecy**:

> **Key Establishment**: CNSA introduced more secure key establishment techniques to provide forward secrecy, a feature that protects past sessions against future compromises of secret keys.

**Modernized cryptographic primitives**:

> CNSA introduced more contemporary cryptographic primitives to offer enhanced security, including recommendations on more secure elliptic curves for public key cryptography.

### Current state of algorithm suites and their usage in systems today

It is important to note the widespread use of the CNSA Suite in cryptographic implementations. The suite is designed to protect classified and unclassified National Security Systems (NSS) and information.

The CNSA Suite not only establishes a framework for secure communications but also emphasizes interoperability and consistency in cryptographic mechanisms. It is foundational in present secure communications within various operating systems.

Moreover, operating systems like Windows have taken steps to implement CNSA Suite, proving its pivotal role in securing modern infrastructures.

## Implementation in Windows operating system

the Windows Operating System stands as one of the most widely used platforms, making its compatibility with various cryptographic algorithms a topic of significant relevance.

Windows has exhibited a progressive approach in adopting robust cryptographic suites to enhance security in its operating systems. It incorporates support for many cryptographic algorithms, including those specified in the Commercial National Security Algorithm Suite (CNSA), such as modern elliptic curve algorithms, amongst others. This support ensures secure communications in a multitude of applications and environments.

## Challenges and considerations for using ECC

Despite the many advantages that ECC (Elliptic Curve Cryptography) presents, including faster computations and lower power consumption for a similar level of security compared to RSA, there are several challenges and considerations to note when looking to implement it.

Firstly, ECC is relatively newer and thus may not be supported across all platforms and systems, potentially causing compatibility issues. Moreover, the migration from RSA to ECC is non-trivial and involves a considerable amount of time and expertise to execute seamlessly.

Furthermore, selecting the correct curve is key. While ECC allows for smaller key sizes, which means faster performance and less storage, it introduces complexity in choosing the right elliptic curve, a choice that impacts both security and performance.

Given the evolving cryptographic landscape, the NSA has recommended organizations that have not transitioned to ECC to hold off on making the switch. Instead, it advises waiting for the advent of quantum-resistant algorithms, which are designed to offer security against the threats posed by quantum computers, marking the next frontier in cryptographic security.

Therefore, when contemplating the shift to ECC, organizations must undertake a meticulous assessment of their existing infrastructure to discern the feasibility and implications of such a move. Moreover, a comprehensive understanding of ECC, right from curve selection to its integration with existing systems, is vital to leveraging its benefits while navigating its intricacies proficiently.

## Setting up certificates in an enterprise

### Root CA: Key lengths and lifetimes

In a Public Key Infrastructure (PKI), the root certificate authority, or Root CA, plays a central role as the most trusted part of the chain. That's why it's important to carefully choose the key length and how long the Root CA should be valid, to keep things secure and manageable in the long run.

When considering key length, it is often advised to opt for a lengthier key to ensure enhanced security. A 4096-bit key length strikes a suitable balance, offering robust security while still being feasible for most computational environments. It's a choice

aligned with a vision for long-term security, positioning your infrastructure to be resilient against cryptographic attacks for many years to come.

Equally important is determining a suitable lifetime for the Root CA. A prolonged lifetime, such as 20 years, offers the convenience of a less frequent renewal cycle. However, it necessitates a robust key with a longer length to maintain a high level of security over time. Conversely, a shorter lifetime demands more frequent renewals but allows for a more adaptable approach to the evolving security landscape.

## Microsoft's recommendation

In a session once dedicated to key usage titled *"Demystifying Native Mode Security to Deliver Internet-based Client Management"*, Microsoft recommended setting the Root CA validity period to **no longer than 16 years**. At the time, this represented a careful balance between long-term stability and cryptographic security. However, the cryptographic landscape has evolved significantly since then, and so have the guidelines for secure key management.

Today, more up-to-date recommendations can be found in NIST Special Publication 800-57 Part 1 Revision 5 (2020), which outlines best practices for cryptoperiods, the maximum time a key should be used, based on key type, size, and use case. NIST explicitly advises *shorter lifespans for weaker keys* and allows for longer periods only when stronger keys are combined with strong operational controls.

For example, here's how NIST maps key sizes to maximum recommended cryptoperiods for digital signature:

## NIST Key Lifespan Recommendations (SP 800-57 Part 1 Rev. 5)

> **Note:** These are not certificate validity periods, but the maximum recommended duration a private key should be used to sign other certificates or data — also known as its **cryptoperiod**.

## Special note on Root CA keys

NIST clarifies that:

> *"Root CA keys used for self-signing or signing Subordinate CA certificates may be used for longer periods than keys used for end-entity certificates, but caution should be exercised to ensure that algorithm strength and key management remain adequate."*

In other words, it is possible to use a Root CA key for a longer time — but only if the following conditions are met:

1. The key strength is sufficient for the entire intended period (e.g., RSA 4096 or ECC P-384).

2. The Root CA is operated under strict security controls (e.g., kept offline, limited access, HSM-protected).
3. A clear *rollover plan* is in place to transition to a new Root CA before cryptographic or operational risks emerge.

With this in mind, the 16-year recommendation from Microsoft should be seen as a product of its time. In modern deployments, particularly those involving 4096-bit RSA keys, organizations may confidently define Root CA lifetimes of up to 15 years, in line with NIST's more rigorous guidance. This offers a robust level of security and interoperability, while still ensuring flexibility to rotate keys before algorithmic or operational risks emerge.

For any organization planning to deploy a new Root CA today, NIST's recommendations provide the most reliable framework to follow. Not only do they reflect current cryptographic standards, but they also promote regular reassessment of risks over time, ensuring the PKI remains aligned with both security demands and technological progress.

## Enterprise CA and leaf certificates: Best practices

Building upon the foundation established by the Root CA, the Enterprise Certificate Authority (Enterprise CA) undertakes the task of issuing certificates to end entities or other subordinate CA's. When configuring the Enterprise CA, it is typical to opt for a key length of 3072 bits and a lifetime of 10 years, striking a balance between security and operational efficiency.

Leaf certificates, issued by the Enterprise CA, inherit the security configurations of their issuer. It is pertinent to note that the leaf certificates cannot possess a lifetime exceeding that of the issuing CA, emphasizing the need for careful planning when determining the lifetime of your Enterprise CA.

**Note!** *A Certificate Authority (CA) is bound by the limitations of its own certificate's validity period, meaning it cannot issue certificates with a validity period exceeding its own. Moreover, as time progresses, the CA's capacity to issue long-term certificates diminishes; it is unable to grant certificates with a lifespan surpassing its own remaining validity period. This not only is a rule to adhere to but also serves as a constantly reducing timeframe within which the CA can operate.*

## Revocation check on Root CA in Windows

In Windows environments, it is essential to address the dynamics of certificate revocation checks on the Root CA. Windows does conduct revocation checks on certificates; however, it bypasses the check for Root CAs since they are self-signed, and there's no higher authority to validate their legitimacy. Instead, the trust is established through a different mechanism, often involving the manual installation of the root certificate in the trust store of the systems in your infrastructure, effectively marking it as trusted.

Thus, while revocation checks are a critical aspect of PKI, Root CAs operate on a basis of inherent trust, bypassing the usual need for such checks. Keeping abreast of the best practices in managing Root CA revocation settings is essential to maintaining a secure and functioning PKI environment.

## Looking ahead

It is imperative to keep an eye on the horizon to anticipate future advancements and shifts in technology. From enhanced security protocols to a new generation of cryptographic standards, in an ever-evolving cybersecurity landscape.

### Introduction to CRYSTALS-Kyber

Navigating the ongoing advancements in cryptography, it is vital to spotlight emerging cryptographic algorithms that are poised to play a significant role in the future of secure communications. One such algorithm is CRYSTALS-Kyber, a lattice-based cryptographic algorithm. This mechanism was developed as a part of the Cryptographic Suite for Algebraic Lattices (CRYSTALS) project, positioning it at the forefront of the post-quantum cryptography era.

Though the name may stir up images of the famous Kyber crystals from the Star Wars series, the reality is far more grounded yet equally fascinating. This algorithm, designed to be resistant to attacks from quantum computers, represents a quantum leap towards more secure cryptographic practices in a world anticipating the advent of quantum computing.

### Potential future developments in cryptography

As we stand on the cusp of a new era in cryptography, propelled by advancements in quantum computing, it is imperative to look towards future developments that could redefine the security landscape.

While ECC and RSA have served as reliable staples in the cryptographic community, the entrance of quantum computing paints a complex picture where these stalwarts might find themselves susceptible to unprecedented levels of attacks. It heralds a shift towards post-quantum cryptographic algorithms, like CRYSTALS-Kyber, which are designed to be quantum-resistant, providing a fortress of security in a quantum computing age.

Apart from the advent of quantum-resistant algorithms, we may also witness enhancements in existing cryptographic suites, with institutions continuously working towards updating and fortifying existing security protocols to meet the demands of the evolving threat landscape.

Looking forward, it is vital to keep abreast of developments in the cryptographic space, as it undergoes a transformation influenced by rapid technological advancements, offering a glimpse into a future where security and innovation walk hand in hand.

## BSI Guidelines: A European perspective on key length and validity

In addition to the NIST recommendations, the German Federal Office for Information Security (BSI)provides valuable guidance on key management within Public Key Infrastructures. Their technical guideline, *BSI TR-02102-1* (as of January 2025), sets clear expectations for key lengths and operational lifetime of cryptographic systems, including certification authorities.

### Recommended key lengths

To ensure cryptographic security at least until 2031, BSI recommends the following **minimum key lengths** for commonly used asymmetric algorithms:

These values aim to meet a cryptographic strength of at least 120 bits, considered robust for the coming years. For high-trust, long-term roles such as a Root CA, BSI further recommends opting for RSA 4096 bits or ECC 384 bits or higher, providing an additional security buffer.

### Limited validity

BSI emphasizes that predicting the long-term strength of asymmetric cryptography becomes highly uncertain beyond 6–7 years. For that reason, systems should be designed with crypto-agility in mind — the ability to transition to stronger algorithms or longer keys without disrupting the PKI or its dependent services.

### Root CA: Long validity, but well-protected

While BSI does not define a strict maximum lifetime for Root CA keys, it makes clear that long-term use is only acceptable under the following conditions:

1. The key strength is more than sufficient for the entire period (e.g., RSA 4096 bits or ECC 384+).
2. The Root CA is well-protected — preferably stored offline and secured with HSMs.
3. A clear rollover or replacement plan is in place, allowing timely transition before security assumptions become outdated.

These recommendations align closely with NIST's position: long-lived Root CAs are possible, but only when designed and managed with a strong focus on cryptographic hygiene and lifecycle planning.

## Choosing the key Length and validity period: The conclusion

In this deep dive into key lengths and cryptographic algorithms, I explored some of the key considerations around RSA and ECC. Based on everything covered so far, it's clear that designing a solid cryptographic strategy means striking the right balance between

security requirements and real-world compatibility. To wrap up this part of the series, I've put together a concise set of recommendations based on both research and practical experience:

### RSA key length vs. recommended usage

### Recommended key lengths and validity by certificate role

### Cryptographic strategy advisory

**Note!** *Although a Microsoft CA can generate RSA keys up to 16384 bits, it's computational not feasible to use these keys. If there's a necessity to use stronger keys, consider moving to ECC.*

Building your cryptographic strategy on a solid RSA foundation gives you a secure and proven starting point. It offers both strong protection today and the flexibility to adapt when new technologies, like quantum-resistant algorithms, become necessary. The key is to make informed decisions now that keep your environment resilient against both current and future threats. I that I hope this updated blog helps!

## References


Barker, E. (2020). *Recommendation for key management:* https://doi.org/10.6028/nist.sp.800-57pt1r5

Waterman, M. (2022, December 27). . https://michaelwaterman.nl/2022/12/27/the-basics-on-diffie-hellman-key-exchange/

Giry, D. (n.d.). *Keylength – Cryptographic key length recommendation*. http://creativecommons.org/licenses/by-sa/3.0/. https://www.keylength.com/

*BSI TR-02102-1 "Cryptographic Mechanisms: Recommendations and Key Lengths" Version: 2025-1*. (n.d.). Federal Office for Information Security. https://www.bsi.bund.de/SharedDocs/Downloads/EN/BSI/Publications/TechGuidelines/TG02102/BSI-TR-02102-1.html