MINISTRY OF COMMUNICATIONS
AND MULTIMEDIA MALAYSIA

CyberSecurity
MALAYSIA

First edition
2020-07-16

Guideline on the Usage of AKSA MySEAL Recommended Cryptographic Algorithms

Reference number:
CD-5-GUI-0120-AKSA MySEAL Usage-v1

REGISTERED OFFICE:

CyberSecurity Malaysia,
Level 7 Tower 1,
Menara Cyber Axis,
Jalan Impact,
63000 Cyberjaya,
Selangor Darul Ehsan, Malaysia
Email: ctr@cybersecurity.my

COPYRIGHT © 2020 CYBERSECURITY MALAYSIA

NO ENDORSEMENT

Products and manufacturers discussed or referred to in this document, if any, are presented for informational purposes only and do not in any way constitute product approval or endorsement by CyberSecurity Malaysia.

TRADEMARKS

All terms mentioned in this document that are known to be trademarks or service marks have been appropriately capitalized. CyberSecurity Malaysia cannot attest to the accuracy of this information. Use of a term in this document should not be regarded as affecting the validity of any trademark or service mark.

DISCLAIMER

This document is for informational purposes only. It represents the current thinking of CyberSecurity Malaysia on the usage of *Algoritma Kriptografi Sedia Ada* (AKSA) *Senarai Algoritma Kriptografi Terpercaya Negara* (MySEAL) recommended cryptographic algorithms. It does not establish any rights for any person and is not binding on CyberSecurity Malaysia or the public. The information appearing on this guideline is not intended to provide technical advice to any individual or entity. We urge you to consult with your own organization before taking any action based on information appearing on this guideline or any other documents to which it may be linked.

Contents **Page**

1 Introduction

1.1 Overview

MySEAL is a project led by CyberSecurity Malaysia (CSM) to develop a portfolio of recommended national trusted cryptographic algorithms. The project commenced in 2016 and is expected to complete in the year 2020. The main goal of the project is to provide a list of recommended cryptographic algorithms for adoption by the Malaysian government. This goal is aligned with one of the objectives set out in the National Cryptography Policy (NCP) which is to protect information in government and critical national information infrastructure (CNII) agencies. The members of the project include representatives from several universities and research institutions in Malaysia. For further information on MySEAL project, please visit https:\\myseal.cybersecurity.my.

In 2018, a list of cryptographic algorithms called AKSA MySEAL was published. The AKSA algorithms were initially selected from various international standards and cryptography listing projects. These algorithms went through a thorough evaluation process during the course of the MySEAL project. The AKSA MySEAL list contains 29 cryptographic algorithms recommended by the MySEAL Focus Group members. The Focus Group consists of members from various universities and government agencies in Malaysia. The list of members is provided in Acknowledgements section.

The AKSA algorithms recommended by the MySEAL Focus Group members, henceforth referred to as AKSA MySEAL recommended cryptographic algorithms, include block ciphers, stream ciphers, hash functions, asymmetric encryption schemes, digital signature schemes, key agreement schemes, prime number generators and deterministic random bit generators. A second list, called AKBA, is expected to comprise newly proposed cryptographic algorithms. The AKBA MySEAL list is scheduled to be published in the year 2020.

As the backbone of secure systems, cryptography must be properly used and implemented in order to prevent attacks related to insecure usage and implementation. This document is developed to assist users who wish to incorporate cryptography in various applications.

1.2 Scope

This document takes the form of explanations of the recommended practices to securely use AKSA MySEAL recommended cryptographic algorithms. This includes aspects of key management and parameter selections for block ciphers, stream ciphers, hash functions, asymmetric encryption schemes, digital signature schemes, key agreement schemes, prime number generators, and deterministic random bit generators. Remarks on the implementation of these algorithms are also briefly discussed. Although this document is restricted to the discussion of the proper usage of AKSA MySEAL recommended cryptographic algorithms, other algorithms may be mentioned briefly if deemed necessary.

This document is not intended to replace existing guidelines and recommendations. Instead, any related guidelines and recommendations are stated to further guide the user. This document does not describe the specification of the AKSA MySEAL cryptographic algorithms. The specifications to the algorithms are provided in the references that are given in the appropriate sections of this document. This document applies for the protection of Official Secret and Official Information of the Malaysian government. This document also applies to Secret Information of non-government critical national information infrastructure (CNII) agencies or organisations. This is inline with the National Cryptography Policy (NCP) described in Section 4.7.

1.3 Objectives

The objective of this document is to serve as a reference document on the proper use of AKSA MySEAL recommended cryptographic algorithms in information security systems.

1.4 Intended audience

This document is intended to provide a general guide to individuals and organisations implementing AKSA MySEAL recommended cryptographic algorithms to protect digital information. The information may be in numerous forms such as text, file, image, audio and video. The audience of this document may include, but not limited to:

a) Developers implementing, possibly from scratch, AKSA MySEAL recommended algorithms as a cryptographic library.
b) Developers making use of an existing implementation of AKSA MySEAL recommended algorithms in an application.
c) Solution or enterprise architects overseeing cryptography-related projects.
d) Managers responsible for managing projects that make use of cryptography.
e) Auditors assessing an application, product or service provided by a third party that makes use of AKSA MySEAL recommended algorithms.
f) Researchers and learners who want to know how to properly use cryptography.

1.5 Preliminaries to Cryptography

Cryptography is an indispensable tool that provides various services to protect sensitive information. The services include confidentiality, integrity, data-origin authentication, entity authentication and non-repudiation. It is important for a user to understand the differences between these services to avoid misuse. The following is a brief review of each of the services provided by cryptography [1].

Confidentiality. To ensure that only authorised entities are able to access the protected information. Confidentiality is also referred to as secrecy.

Integrity. To ensure that any unauthorised changes to information can be detected. Integrity does not *prevent* data from being modified but offer a means to *detect* the modification.

Data-origin authentication. To ensure that the received data comes from the original entity that sends it, even though the data may be transmitted via one or more intermediate entities. The intermediate entity may be a web server or router.

Entity authentication. To ensure that entities in a communication session are validated and online at the given moment of the authentication process.

Non-repudiation. To ensure that an action performed by a user cannot be denied later. For instance, if an exchange of data occurs between two parties, or if a user digitally signs a document, then each party in these situations cannot later deny that an exchange has been made, or that the document has been signed.

With the exception of hash functions and DRBGs, all AKSA MySEAL recommended cryptographic primitives employ the use of keys. Symmetric-key primitives, which include block and stream ciphers, use a single secret key. Asymmetric-key primitives, comprise of asymmetric encryption and digital signature schemes, make use of a pair of keys called private and public keys. Both the secret and private keys must never be disclosed to unauthorised parties. The security of the cryptographic primitives rely on the secrecy of these keys, and not of the specific algorithms. Issues related to keys are discussed in Section 5. A cryptographic primitive is rarely employed on its own. It is often used with other primitives to provide functionality to users of a system.

1.6 AKSA MySEAL Recommended Cryptographic Algorithms

The 29 AKSA MySEAL recommended algorithms are listed in Table 1.

Table 1: AKSA MySEAL recommended cryptographic algorithms[1].

Block Ciphers (General Purpose)	Key Length
1. AES	128, 192, 256
2. Camellia	128, 192, 256
3. CLEFIA	128, 192, 256
Block Ciphers (Lightweight)	Key Length
1. PRESENT	80, 128
2. HIGHT	128
Stream Ciphers	Key Length
1. KCipher-2	128
2. Rabbit	128
3. ChaCHa20	256
Hash Functions (General-Purpose)	Digest Length
1. SHA2	384, 512, 512/224, 512/256
2. SHA3	224, 256, 384, 512
SHAKE (considered part of SHA3)	128, 256
Hash Functions (Lightweight)	
1. SPONGENT	80, 128, 160, 224, 256
2. PHOTON	80, 128, 160, 224, 256
Asymmetric Encryption Schemes	Key Length
1. PSEC-KEM	224*, 256, 384, 512
2. RSA-KEM	2048*, 3072
3. ACE-KEM	224*, 256, 384, 512
4. ECIES-KEM	224*, 256, 384, 512
5. RSA-OAEP	2048*, 3072
6. NTRU	4829, 5929, 8173
Digital Signature Schemes	Key Length
1. DSA	2048*, 3072
2. ECDSA	224*, 256, 384, 512
3. RSA-PSS	2048*, 3072
Key Agreement Schemes	Key Length
1. DH	2048*, 3072
2. ECDH	224*, 256, 384, 512
Prime Number Generators	
1. Miller-Rabin Primality Test	
2. Elliptic Curve Primality Proving	
3. Shawe-Taylor Algorithm	
Deterministic Random Bit Generators	Digest/Key Length
1. SHA-2-DRBG	384, 512, 512/224, 512/256
2. HMAC-SHA2-DRBG	384, 512
3. CTR-DRBG	AES: 128, 192; 3-Key TDEA: 168

[1] Key or message digest length marked with a '*' indicates that the length is recommended for use only for a certain period of time. Refer to Section 5.2 for more information

1.7 On the Non-Inclusion of Certain Cryptographic Primitives and Algorithms in AKSA MySEAL

There are several cryptographic primitives that are absent from AKSA MySEAL. These include, but are not limited to:

a) Message authentication code (MAC),
b) Authenticated encryption (AE), and
c) Key derivation function (KDF).

The above primitives may be included in a future version of AKSA MySEAL.

On the other hand, there are certain cryptographic algorithms that are not included in the AKSA MySEAL list. The non-inclusion of these algorithms does not imply that the algorithms are practically insecure, or that they should be immediately refrained from current use. The said algorithms were not recommended due to the algorithms not meeting certain evaluation criteria that are specific to MySEAL which may be of higher requirement than that of other standards, guidelines or recommendations. The continued deployment of the non-recommended cryptographic algorithms is to be done at one's own risk assessment by taking into account potential security issues that may arise in the future. Furthermore, the AKSA MySEAL list will be re-evaluated from time to time to ensure that the list is relevant with regards to the advancement in cryptanalysis and computing power.

1.7.1 SHA-256

One particular algorithm that is not recommended is SHA-256. At the time of writing, SHA-256 is currently being widely deployed in applications such as Bitcoin and protocols such as Transport Layer Security (TLS) 1.2. SHA-256 is an approved hash function algorithm of the US NIST and is included in the ISO/IEC 10118 standard. The CA/Browser Forum also includes SHA-256 as one of the endorsed hash function algorithms for digital certificate validity period beginning after 31 December 2010.

Although SHA-256 is not an AKSA MySEAL recommended algorithm, as stated in Section 4.7 of this document, the NCP can be interpreted by stating if a cryptographic algorithm is adopted by the industry, then the algorithm may still be used in practice.

1.7.2 RSA with 1024-bit Key

There are also applications that still use RSA-based cryptographic schemes that utilise 1024-bit keys. As key length of smaller than 2048 bits is not recommended for use under AKSA MySEAL, new applications should use key length of 2048 bits or larger. Applications that still use 1024-bit keys may still be continued to be used for legacy purposes.

1.7.3 Document Organisation

This document is organised as follows. Section 1 describes the purpose of this document and provides the list of AKSA MySEAL recommended cryptographic algorithms. The terms used throughout this document are defined in Section 2. This section also presents a list of symbols and abbreviated terms used throughout this document. International guidelines related to the use of cryptography are briefly described in Section 3. The relevant Malaysian statutes and policies related to cryptography are briefly described in Section 4. Issues related to key management are discussed in Section 5. Sections 6 to 13 respectively outline the algorithms and usage of AKSA MySEAL block ciphers, stream ciphers, hash functions, asymmetric encryption schemes, digital signature schemes, key agreement schemes, key agreement schemes, prime number generators and deterministic random bit generators. Remarks on implementation issues are given in Section 14.

2 Terms, definitions, symbols and abbreviated terms

2.1 Terms and definitions

For the purposes of this document, the following terms and definitions apply:

2.1.1
Algorithm
a particular specification of a cryptographic primitive. It consists of computational steps that take one or more inputs and produce one or more outputs. Examples are AES-128 and SHA-256.

2.1.2
bit
one of the two symbols ‘0’ or ‘1’.

2.1.3
bit string
a sequence of bits.

2.1.4
critical national information infrastructure
the nation’s vital infrastructure such that any disruption or destruction to the infrastructure has an impact to the nation’s defense, security, economic stability and image, the ability of the government to operate and the public’s health and safety.

[adopted from the National Cryptography Policy]

2.1.5
data encapsulation mechanism
cryptographic mechanism, based on symmetric cryptographic techniques, which protects both the confidentiality and integrity of data.

[ISO/IEC 18033-2:2006]

2.1.6
deterministic function
a function whose output is the same given the same set of input values.

2.1.7
forward secrecy with respect to entity A
property that knowledge of entity A’s long-term private key subsequent to a key agreement operation does not enable an opponent to recompute previously derived keys.

[ISO/IEC 11770-3:2015]

2.1.8
forward secrecy with respect to both entity A and entity B individually
property that knowledge of entity A’s long-term private key or knowledge of entity B’s long-term private key subsequent to a key agreement operation does not enable an opponent to recompute previously derived keys.

Note: This differs from mutual forward secrecy in which knowledge of both entity A’s and entity B’s long-term private keys do not enable recomputation of previously derived keys.

[ISO/IEC 11770-3:2015]

2.1.9
implicit key authentication from entity A to entity B
assurance for entity B that entity A is the only other entity that can possibly be in possession of the correct key.

[ISO/IEC 11770-3:2015]

2.1.10
key agreement
process of establishing a shared secret key between entities in such a way that neither of them can predetermine the value of that key.

Note: By predetermine it is meant that neither entity A nor entity B can, in a computationally efficient way, choose a smaller key space and force the computed key in the protocol to fall into that key space.

[ISO/IEC 11770-3:2015]

2.1.11
key confirmation from entity A to B
assurance for entity B that entity A is in possession of the correct key.

[ISO/IEC 11770-3:2015]

2.1.12
key encapsulation mechanism
similar to an asymmetric cipher, but the encryption algorithm takes as input a public key and generates a secret key and an encryption of this secret key.

[ISO/IEC 18033-2:2006]

2.1.13
key establishment
process of making available a shared secret key to one or more entities, where the process includes key agreement and key transport.

[ISO/IEC 11770-3:2015]

2.1.14
label
octet string that is input to both the encryption and decryption algorithms of an asymmetric cipher. A label is public information that is bound to the ciphertext in a non-malleable way.

[ISO/IEC 18033-2:2006]

2.1.15
may
this word, or the adjective “optional”, mean that an item is truly optional. One vendor may choose to include the item because a particular marketplace requires it or because the vendor feels that it enhances the product while another vendor may omit the same item. An implementation which does not include a particular option must be prepared to interoperate with another implementation which does include the option, though perhaps with reduced functionality. In the same vein an implementation which does

include a particular option must be prepared to interoperate with another implementation which does not include the option (except, of course, for the feature the option provides).

[IETF RFC 2119]

2.1.16
must
this word, or the terms “required” or “shall”, mean that the definition is an absolute requirement of this document.

[IETF RFC 2119]

2.1.17
must not
this phrase, or the phrase “shall not”, mean that the definition is an absolute prohibition of this document.

[IETF RFC 2119]

2.1.18
mutual forward secrecy
property that knowledge of both entity A’s and entity B’s long-term private keys subsequent to a key agreement operation does not enable an opponent to recompute previously derived keys.

[ISO/IEC 11770-3:2015]

2.1.19
octet
a bit string of length 8.

2.1.20
octet string
a sequence of octets.

2.1.21
official secret
any document specified in the Schedule (of the Official Secrets Act 1972) and any information and material relating thereto and includes any other official document, information and material as may be classified as ‘Top Secret’, ‘Secret’, ‘Confidential’ or ‘Restricted’, as the case may be, by a Minister, the Menteri Besar or Chief Minister of a State or such public officer appointed under section 2B (of the Act).

[Official Secrets Act 1972]

2.1.22
official information
any data, information and materials related to public service, apart from Official Secret.

[adopted from the National Cryptography Policy]

2.1.23
primitive
a cryptographic process or tool that provides one or more services. Examples are block ciphers and hash functions.

2.1.24
private key
the key of an entity’s asymmetric key pair which should only be used by that entity.

[ISO/IEC 11770-1:1996]

2.1.25
public key
the key of an entity’s asymmetric key pair which can be made public.

[ISO/IEC 11770-1:1996]

2.1.26
probabilistic function
a function whose output does not only depend on given input values, but also on a randomly chosen auxiliary value.

2.1.27
secret information
any data, information and related materials that needs protection and classified by any officer appointed under the rules, circular or laws adopted by the non-government CNII agencies/organisations.

[adopted from the National Cryptography Policy]

2.1.28
secret key
the key used with symmetric cryptographic techniques by a specified set of entities.

[ISO/IEC 11770-3:1999]

2.1.29
seed
a string of bits that is used as input to a deterministic random bit generator (DRBG) mechanism. The seed will determine a portion of the internal state of the DRBG, and its entropy must be sufficient to support the security strength of the DRBG.

[NIST SP800-90A Rev. 1]

2.1.30
seed period
the period of time between instantiating or reseeding a DRBG with one seed and reseeding that DRBG with another seed.

[NIST SP800-90A Rev. 1]

2.1.31
should
this word, or the adjective “recommended”, mean that there may exist valid reasons in particular circumstances to ignore a particular item, but the full implications must be understood and carefully weighed before choosing a different course.

[IETF RFC 2119]

2.1.32
should not
this phrase, or the phrase “not recommended” mean that there may exist valid reasons in particular circumstances when the particular behavior is acceptable or even useful, but the full implications should be understood and the case carefully weighed before implementing any behavior described with this label.

[IETF RFC 2119]

2.1.33
trusted third party
security authority or its agent, trusted by other entities with respect to security related activities.

[ISO/IEC 9798-1:2010]

2.2 Abbreviated terms and acronyms

For the purpose of this document, the following symbols and abbreviated terms apply:

[x]	the smallest integer greater than or equal to the real number x. For instance, [3] = 3, [3.1] = 4.
$x \oplus y$	if x and y are bit/octet strings of the same length, the bit-wise exclusive-OR (XOR) of the two strings.
0, 1, . . . , F	a number written in teletype font denotes a hexadecimal value which is from the range {0, 1, . . . , F}. For instance, the hexadecimal value A is used to represent the decimal 10.
AE	authenticated encryption
AKSA	Existing Cryptographic Algorithms (*Algoritma Kriptografi Sedia Ada*)
ANSI	American National Standards Institute
BNM	Bank Negara Malaysia (Central Bank of Malaysia)
CA	certificate authority
CGSO	Chief Government Security Office
CNII	critical national information infrastructure
CRYPTREC	Cryptography Research and Evaluation Committees
CSM	CyberSecurity Malaysia
DRBG	deterministic random bit generator
ENISA	European Union Agency for Network and Information Security
EPC	European Payments Council
ISO/IEC	International Organisation for Standardisation / International Electrotechnical Commission
IV	initialisation vector
KDF	key derivation function
MAC	message authentication code
MAMPU	Malaysian Administrative Modernisation and Management Planning Unit
MyMIS	Malaysian public sector management of information & communication technology security handbook
MySEAL	National Trusted Cryptographic Algorithms List (*Senarai Algoritma Kriptografi Terpercaya Negara*)

NCP	National Cryptography Policy
NIST	National Institute of Standards and Technology
OWASP	Open Web Application Security Project
RAKKSSA	Cyber Security Framework for the Public Sector (*Rangka Kerja Keselamatan Siber Sektor Awam*)

3 International Guidelines

This section describes international guidelines related to the use of cryptography. All the guidelines stated here are good reference points for a user in developing applications employing cryptography. A comparison is made to highlight the particular focus of the stated documents as compared to this document so that the user is aware of the differences.

3.1 Australian Government Information Security Manual (2019)

The manual outlines 22 guidelines related to cyber security that can be applied by organisations. One of the guidelines briefly discusses the use of cryptography to protect sensitive information. The guideline also contains a brief explanation on the use of cryptographic algorithms and protocols approved by the Australian Signals Directorate (ASD).

The focus of this AKSA MySEAL guideline document is on the usage of cryptography. Therefore, this guideline is narrower in scope and contains more information compared to the Australian government manual.

3.2 CRYPTREC Cryptographic Technology Guideline (Lightweight Cryptography) (2017)

The document outlines applications of lightweight cryptography. The majority of the document discusses analysis on the performance of selected lightweight block ciphers and authenticated encryption schemes.

This AKSA MySEAL document centres on the usage of AKSA MySEAL recommended cryptographic primitives, which includes lightweight primitives. Therefore, the scope of this document is larger than the CRYPTREC document.

3.3 ENISA Recommended Cryptographic Measures: Securing Personal Data (2013)

The document presents an overview of cryptographic methods that can be used to protect sensitive information. In order to demonstrate the applicability of the cryptographic methods, a case study that has five different types of data breaches is given. For each type of data breach, appropriate cryptographic protection measures are proposed. The document addresses cryptography from a high-level perspective without delving deeper into the usage aspects. Furthermore, the document does not reference any other standards or guidelines on the usage of cryptography.

This AKSA MySEAL document contains a deeper treatment of cryptographic usage compared to the ENISA document. Additionally, this document contains references to other relevant cryptographic standards or guidelines.

3.4 ENISA: Algorithms, Key Size and Parameters Report (2014)

The document provides a set of recommended cryptographic algorithms, their key sizes and parameters. The algorithms are classified as suitable for legacy or future use. If an algorithm is deemed not suitable for legacy use, then the algorithm should be replaced. The document considers schemes that are not part of AKSA MySEAL such as MAC and KDF schemes. The document also contains a brief section on side channels, random number generation and key management.

This AKSA MySEAL document only considers AKSA MySEAL recommended primitives and schemes which currently do not include MAC and KDF schemes. The key size and parameter recommendations given in this AKSA MySEAL follow that of NIST. The ENISA document contains its own set of recommendations.

3.5 EPC 342-08v8.0 (2018)

The document specifies a number of recommendations and best practices to the European payments community on the usage of cryptographic methods and key management. The document is written mainly for payment service providers.

This AKSA MySEAL document is not tailored for specific applications or industries. It is a cryptography guidelines document generally suitable for various applications and industries.

3.6 ISO/IEC 19790: 2012

The standard defines the security requirements for a cryptographic module utilised within a security system. There are four security levels defined in this standard for each of 11 requirement areas where a higher level means greater security than a lower level. The standard has been reviewed and confirmed by the ISO/IEC Technical Committee in 2018.

This AKSA MySEAL document does not define specific security requirements that a cryptographic module should meet. Therefore, the main purpose of this document and that of ISO/IEC 19790 is different.

3.7 NIST SP 800-175A (2016)

The document provides guidance on the basis for determining requirements for using cryptography to protect US Federal government's sensitive but unclassified information. The document does not provide specific guideline on the proper usage of cryptography. However, the document provides useful guideline whether or not the use of cryptography is required in the context of the US Federal government.

This document focuses on the context of the Malaysian government where relevant laws and policies pertaining to the use of cryptography are summarised.

3.8 NIST SP 800-175B (2016)

The document outlines a high-level view of the cryptographic methods and services available for the protection of sensitive information. The scope of NIST SP 800-175B is similar to this document. At the time of writing, a revised version of the document is currently in draft form and has completed a public comment period that ended on September 5, 2019.

This document differs from NIST SP 800-175B as this document includes relevant Malaysian laws and policies regarding the use of cryptography and provides a remark on issues related to cryptography implementation.

3.9 OWASP Guide to Cryptography (2018)

The guide is accessible online and provides a high-level overview of the cryptographic methods to protect sensitive information. The overview is less comprehensive than that of NIST SP 800-175B.

4 Malaysian Statutes, Policies and Guidelines

This section describes Malaysian statutes, policies and guidelines that are relevant to the use and misuse of cryptography. The information is provided here so that a user is aware of the applicable laws and regulations related to cryptography.

4.1 Official Secrets Act 1972

The Official Secrets Act (OSA) 1972 was enacted to protect both physical and digital information deemed as secret by the government of Malaysia. As cryptography can be applied to protect digital information classified as Official Secret, the provisions under the OSA must to be adhered to.

4.2 Copyright Act 1987

The Copyright Act 1987 concerns with matters related to copyright. Copyrighted materials may be secured via any technological protection measure which may involve the use of cryptography (e.g. [2]). Any unauthorised circumvention of such protection measures is an offence according to this Act.

4.3 Digital Signature Act 1997 and Digital Signature Regulations 1998

Both the Digital Signature Act 1997 and Digital Signature Regulations 1998 came into force on October 1, 1998. The main purpose of the Act and its accompanying Regulations is to regulate the use of digital signature in Malaysia. The Malaysian Communications and Multimedia Commission (MCMC) is responsible to administer, enforce, carry out and give effect to the provisions under the Digital Signature Act for the purpose of monitoring and overseeing the activities of certificate authorities.

4.4 Computer Crimes Act 1997

The Computer Crimes Act 1997 deals with offences related to computers. Among others, the Act states offences which include unauthorised access to applications and data residing in any computer, which may involve the use of cryptography. An example is ransomware, where an adversary uses an application to illegally access a user's computer and subsequently keeps a user's data hostage by encrypting the user's data. Cryptography must never be used for any unlawful activity.

4.5 MyMIS

The Malaysian Public Sector Management of Information and Communications Technology Security Handbook (MyMIS) was published by MAMPU in 2001. The handbook provides essential guidelines to Malaysian government employees on the ICT security process in the public sector. The scope of this handbook covers ICT security including cryptography. A number of general recommendations related to cryptography is stated in this handbook. However, as the handbook has never been revised since its publication in 2001, several recommendations made in the handbook may no longer be relevant. An as example, in the handbook, the minimum asymmetric cryptography key length is stated as 1024 bits. Currently, a minimum of 2048 bits is required. Section 5.2 discusses cryptographic key lengths.

4.6 Personal Data Protection Act 2010

The Personal Data Protection Act (PDPA) 2010 was enacted to regulate matters related to the processing of personal data in commercial transactions. Among others, Sections 9 and 11 of the PDPA respectively state the security and data integrity principles that apply to parties involved in the processing of personal data.

The mechanisms to provide security and data integrity as stipulated in these sections may involve the use of cryptography.

4.7 National Cryptography Policy

The National Cryptography Policy (NCP) was approved by the Cabinet in July 2013. One of the objectives of the NCP is to increase the use of trusted cryptography products among critical national information infrastructure (CNII) agencies/organisations. Clause 8 of the NCP states the following.

a) The use of trusted cryptography products is mandatory for matters involving Official Secret.
b) The use of trusted cryptography products or products adopted by the industry is encouraged for matters involving Official Information in public services.
c) The use of trusted cryptography products or products adopted by the industry is mandatory for matters involving Secret Information of non-government CNII agencies/organisations.

4.8 RAKKSSA

The Public Sector Cyber Security Framework, or RAKKSSA (*Rangka Kerja Keselamatan Siber Sektor Awam*) was developed as a result of strategic partnership between MAMPU, CGSO, CSM and MIMOS. The framework, published in 2016, provides a generic perspective of all cyber security components that ministries and Government agencies need to consider in protecting digital information. Section 2.2.4 of the framework briefly discusses about the use of cryptography. In line with the NCP, the said section on cryptography states that the use of trusted cryptography products in securing documents classified as Official Secret is mandatory. For Official documents, the use of trusted or other industry-accepted cryptography products is recommended.

4.9 BNM’s Policy on Risk Management in Technology

BNM’s policy document on Risk Management in Technology was issued on July 18, 2019. The policy applies to financial institutions as defined in paragraph 5.2 of the policy. Paragraphs 10.16 to 10.20 specifically address issues related to cryptography. In particular, the policy mandates the establishment of a robust and resilient cryptography policy and the proper management of cryptographic controls and keys.

5 Key Management

Management of keys is one of the most critical step in a system that employs cryptography. Keys (except for public keys) are the only data that need to be kept secret from unauthorised parties. Therefore, a poor key management may severely weaken the entire system. Key management can be a complex process and there is no single correct way of doing it. Determining a suitable key management system depends on specific requirements of the application that employs the system. For instance, key management for a secure cloud storage application may differ from an end-to-end encryption application. Therefore, this chapter is not intended to provide a one size fits-all solution for key management. Instead, several basic, but not necessarily sufficient, guidelines will be laid out to assist the user towards selecting and implementing a secure key management infrastructure.

5.1 Guidelines

The following guidelines are useful in developing a key management system and in selecting suitable key lengths:

a) CA/Browser Forum Baseline Requirements for the Issuance and Management of Publicly-Trusted Certificates v1.6.6 (2019) [3]. Subsection 6.1.5 defines allowed key lengths for generating digital certificates.
b) CPA Canada WebTrust for Certification Authorities v2.2 [4] (2019).
c) ENISA: Algorithms, Key Size and Parameters Report [5] (2014).

d) EPC 342-08 Version 8.0: Guidelines on Cryptographic Algorithms Usage and Key Management [6] (2018).
e) NIST SP 800-56B Revision 2: Recommendation for Pair-Wise Key Establishment Using Integer Factorization Cryptography [7].
f) NIST SP 800-57 Part 1 Revision 4: Recommendation for Key Management, Part 1: General [8] (2016).
g) NIST SP 800-57 Part 2 Revision 1: Recommendation for Key Management, Part 2: Best Practices for Key Management Organizations [9] (2019).
h) NIST SP 800-57 Part 3 Revision 1: Recommendation for Key Management, Part 3: Application-Specific Key Management Guidance [10] (2015).
i) NIST SP 800-130: A Framework for Designing Cryptographic Key Management Systems [11] (2013).
j) NIST SP 800-131A Revision 2: Transitioning the Use of Cryptographic Algorithms and Key Lengths [12] (2019).
k) OWASP Key Management Cheat Sheet[2].

5.2 Length of Key

In general, the longer is the key, the better is the security. A long key, however, requires more storage and computing requirements than a short key. Therefore, there is a trade-off between efficiency and security when selecting a particular key length. There are numerous recommendations for the appropriate key lengths and the duration of protection provide by these keys. In this document, the recommendations provided by the NIST are adopted [8]. The strength of a cryptographic primitive is typically attributed to the length of a symmetric key. Table 2 contains a guide for the selection of key lengths for symmetric key primitives. In the table, the information regarding key length of 112 bits and higher, is based on NIST recommendations.

Table 2: Symmetric key lengths.

Length	Remarks
80	Only for protecting data in resource-constrained (lightweight) devices such as IoT for a short period of time. Expect data to be secure only for a few minutes up to a day.
112	Can be used to protect data through the year 2030. Beyond the year 2030, data must be protected by a key longer than 112 bits, and data that was previously protected with 112-bit keys can still be processed for legacy use only.
128, 192	General use. Expect data to be protected beyond the year 2030.
256	Long-term security. Expected to provide adequate protection against quantum computers.

The lightweight block cipher PRESENT is the only cryptographic algorithm in the AKSA MySEAL recommended list that accepts a key smaller than 112 bits, which is 80 bits. The usage of this key should be restricted to protecting data for which its value is expected to last only for a short period of time. For instance, when sending an encrypted signal over the network to switch on a particular device (e.g. light, ceiling fan, surveillance camera) in an IoT environment. Once the device is activated, the previously

[2] https://cheatsheetseries.owasp.org/cheatsheets/Key_Management_Cheat_Sheet.html

protected signal is no more of value. In the general case, a minimum key length of 128 bits is recommended. A 256-bit key can be used as a master key that is expected to exist for a longer period of time.

Table 3 lists the recommended hash function digest lengths and asymmetric key primitives key lengths (all stated in bits) that are approximately equivalent to a given security strength. Details as follows:

a) The `Security' column refers to the strength of the primitive equivalent to the length of a symmetric key primitive in bits. For each security strength or level, each primitive has different key length due to its inherent properties and underlying components.
b) The `Hash' column lists the equivalent hash digest length. For instance, a 512-bit hash function algorithm, such as SHA3-512, provides 256-bit security.
c) The `DLP' column contains a pair of public and private key lengths[3] of cryptographic primitives based on discrete logarithm problem (DLP). Examples include DSA and DH.
d) The `IFP' column lists the key length (length of the modulus) of cryptographic primitives based on the integer factorization problem (IFP).
e) The `ECC' column contains the key lengths of cryptographic primitives based on elliptic curve cryptography (ECC). Examples include ECDSA and ECDH.

Table 3: Recommended hash digest and asymmetric key lengths for particular security strengths [8].

Security	Hash	DLP	IFP	ECC
112	224	(2048,224)	2048	224
128	256	(3072,256)	3072	256
182	384	(7680,384)	7680	384
256	512	(15360,512)	15360	512

5.2.1 Security Against Next Generation of Supercomputers

Computers that are widely in use today, including current supercomputers[4], are unable to efficiently break existing cryptographic primitives at the 128-bit security strength or higher. Assume that a hypothetical supercomputer is able to perform brute force at a rate of 2^{60} keys per second. The time it takes to exhaustively search a 128-bit key space is around 36.5 billion (≈36.5 x 109) years. Suppose that this speed doubles every year. Hence, in the next 40 years, the expected time it takes to brute force a 128-bit key is approximately 12 days.

To compare, at the time of writing, the fastest supercomputer as of June 2019, which is called Summit, is able to perform calculations at a theoretical peak rate of about 200,800 tera ($\approx 2^{57}$) FLOPS (floating point operations per second). Furthermore, the sum performance of all top 500 supercomputers is 1.6 exa ($\approx 2^{60}$) FLOPS[5]. On the other hand, the fastest Bitcoin mining hardware as of July 2019[6] is able to compute

[3] The public key length is stated first, followed by the private key

[4] A list of top 500 supercomputers can be obtained at https://www.top500.org

[5] See https://www.top500.org/statistics/perfdevel/

[6] See https://www.buybitcoinworldwide.com/mining/hardware/

2^{43} (14 tera) hashes per second. Therefore, it can be reasonably assumed that the 2^{60} brute force rate of the previously described hypothetical computer is still not within reach of today's single supercomputers.

The next generation of supercomputers, called quantum computers, are currently being researched and developed. Several small-scale quantum computers, such as D-Wave and IBM's Q, have also been constructed. If large-scale quantum computers are ever built, then the computers would able to efficiently break many cryptographic primitives based on mathematical-hardness problem, which include the entire AKSA MySEAL recommended asymmetric algorithms. The user then needs to immediately switch to post-quantum (PQ) cryptographic primitives. The AKSA MySEAL list contains one PQ primitive, i.e. NTRU.

The need to have a secure portfolio of PQ cryptographic primitives drew the US NIST to embark on the Post-Quantum Cryptography project in 2016. The project is expected to produce a draft standards document sometime in the year 2022. Consequently, this AKSA MySEAL guideline document is expected to be revised after the availability of the NIST PQ cryptography standard.

In the symmetric-key primitive space, a powerful quantum computer is expected to be able to perform brute force on a n-bit key with $2^{n/2}$ time with Grover's algorithm. On average, a conventional computer performs the same operation in 2^{n-1} time. For instance, a quantum computer needs to execute 2^{128} operations to brute force a 256-bit key whereas a conventional computer spends 2^{251}, on average. Therefore, a 256-bit key is expected to provide adequate protection against quantum computers as indicated in Table 2.

5.3 Key Period

Key period, or cryptoperiod, refers to the time span that a key is authorised to be used by participating entities. Once the period has elapsed, the key is considered expired and should no longer be used. Although in general, a key may be used to protect data for a long period of time, as depicted in Table 2, a user may wish to limit the key's period. Reasons for having finite key lifetimes include, but not limited to [1], Sect. 10.2.1]:

Protection against key compromise. Keys are set to have limited lifetime to restrict the damage caused by events such as loss of key-storage device, either intentionally or otherwise.

a) Mitigation against key management failures. There may exist loopholes in the key management system that allows unauthorised entities to have access to valid keys.
b) Mitigation against future attacks. Advances in attacking methods may lead to the compromise of keys. Hence, limiting a key's period is a prudent safeguarding measure.

Section 5.3 of NIST SP 800-57 Part 1 discusses the various issues pertaining to key period.

5.4 Key Revocation

A key may need to be explicitly revoked due to reasons including, but not limited to, expiry of the key period or the key has been compromised. The key revocation process can be performed by notifying all affected entities that the key has been revoked. An example is the Certificate Revocation List (CRL) issued by a CA that contains a list of digital certificates that has been revoked by the said CA.

5.5 Key Generation

Key generation involves the creation of keys for cryptographic usage. Depending on the application, a key can be generated in several ways:

a) Using any of the AKSA MySEAL recommended DRBG algorithms as discussed in Section 13. The output of these algorithms may be truncated to suit the desired key length. The initial input, or seed value, to these DRBG algorithms must be kept secret and may be derived from either a software or hardware based random source. In Unix-based systems, the seed value can be obtained from the device file */dev/urandom*. In Windows, this can be derived using the *BCryptGenRandom()* function included in the Cryptography API: Next Generation (CNG) API.

b) Deriving a key from another key, generally called the base key. The derived key is typically used only in a particular session or purpose. This base key may also be referred to as the master key, key deriving key, or key encrypting key, depending on context. The lifetime of the base key is generally longer than the keys it derives. An example of a base key is the one pre-loaded onto mobile phones or digital television set-top boxes.
c) From a user's secret input such as password or PIN, which is weak from a cryptographic standpoint. The resulting key is generated using a key derivation function such as the Password-Based Key Derivation Function 2 (PBKDF2) [13]. Note that PBKDF2 is currently not included in the AKSA MySEAL list.
d) Through any of the AKSA MySEAL recommended key agreement scheme algorithms as explained in Section 11.

A key must never be generated in the following manner:

a) Directly taking the user's weak secret input such as a password or PIN, and convert it to byte strings for use as a key.
b) Using a programming language's default or built-in random number generator that is not tailored for cryptographic use. Examples include, but not limited to, *rand()* in C*, java.util.Random* class in Java and *random()* in Python.

5.6 Key Transport

Key transport concerns with the distribution of keys to one or more entities where they will be used. The distribution can be done in various ways, depending on specific applications:

a) Pre-loaded onto a device such as mobile phones.
b) Through any of the AKSA MySEAL recommended key agreement scheme algorithms which inherently handles both key generation and transport.
c) By establishing a key distribution centre (KDC) that is trusted by all participating entities. Prior to the start of communication, each entity pre-shares a unique key with the KDC.
d) By encrypting the symmetric key with the public key of the recipient.
e) Manually transferred to the entities wishing to use the key.

5.7 Key Storage

Not all keys need to be stored. For instance, a key that is used to encrypt data for a particular session may be discarded immediately after the encryption is completed. Certain keys, such as long-term master and private keys, must be securely stored. Storing keys securely can be done by several means:

a) Using a hardware security module (HSM) to store the relevant keys. Most modern computers and smartphones come with the HSM already built-in. The module should be tamper-resistant in order to prevent an attacker from recovering the keys. The HSM may also be present in a cloud storage facility.
b) Storing the keys in encrypted form either on the hard disk, database, cloud or something similar. The key to encrypt these keys may come from the user-supplied secret.
c) Storing the key in component form where the key is broken up into several components and each component is stored at different locations. In order to use the key, all components or a minimum number of components must be combined.
d) Not storing any key at all but instead, rely on the user to supply the secret such as a password.

A key must never be stored in the following manner:

a) Hard-coded in the source code, even if the source code is proprietary and not disclosed to the public. The key may be recovered using reverse-engineering of the compiled code.
b) In the clear without any kind of masking in a text file or database, although access to the file or database is restricted.

5.8 Key Backup, Archival and Recovery

Keys may need to be backed up on an independent and secure storage media to prevent loss of access in normal operation. Similarly, keys that are not actively used may also need to be archived for a period of time (e.g. several years), due to legal requirements. For instance, a contract that has been digitally signed may need to be verified at later point of time.

There should be a mechanism to allow secure recovery of the keys in the event that the original keys are lost. Keys that are backed-up or archived should remain in storage for as long as the keys are needed in normal operation, or as required by the law. If the keys are no longer required, then they should be removed and subsequently destroyed.

5.9 Key Usage

In principle, the same key should never be used for two or more different purposes. For instance, a secret key that is used for encryption should never be re-used to produce a MAC tag. A different secret key should instead be generated to produce the MAC.

A long-term key, such as the master key, should be protected from direct use. Instead, a secure key derivation method such as the ones given above, can be used. A key hierarchical may be employed where the top level keys are the long-term master keys and the lower-level keys are derived from the upper level keys. For instance, the second level keys may consist of base keys that define the purpose of the third-level keys. Therefore, the third level keys are the actual keys used directly by the cryptographic algorithms.

5.9.1 Key Change

Periodically, keys may need to be changed, refreshed or removed from active use, due to the expiry of the keys (see Section 5.3). The following non-exhaustive list of scenarios may require the change of keys:

a) When performing cryptographic operations for a new session.
b) When a user opts to delete his account or an employee who has resigned from the organisation. In such situations, the keys can be removed from active use.
c) A key is compromised.
d) A security vulnerability of the developed system, cryptographic algorithm, hardware or operating system that may potentially lead to key compromise.

5.9.2 Key Destruction

A key must be securely destroyed if it is no longer required such as when the key has expired or has served the required archival period. A secure data erasure or sanitisation mechanism should be used to destroy the keys.

6 Block Ciphers

6.1 Algorithms

The AKSA MySEAL recommended block ciphers are categorised into general-purpose and lightweight. The former is suitable to be used for general applications while the latter is for environments where resources such as area, memory and power are limited such as IoT devices.

6.1.1 General-Purpose

There are three AKSA MySEAL recommended general-purpose block cipher algorithms, namely AES, Camellia and CLEFIA.

The AES is defined in the following standards:

a) ISO/IEC 18033-3:2010 Encryption Algorithms, Part 3: Block Ciphers, and
b) NIST FIPS PUB 197 - Advanced Encryption Standard.

In addition to the specification document [14], Camellia is defined in the following standard:

a) ISO/IEC 18033-3:2010: Encryption Algorithms, Part 3: Block Ciphers.

In addition to the specification document [15][16][17], CLEFIA is defined in the following standard:

a) ISO/IEC 29192-2:2012: Lightweight Cryptography, Part 2: Block Ciphers.

6.1.2 Lightweight

There are two AKSA MySEAL recommended lightweight block cipher algorithms, namely HIGHT and PRESENT.

HIGHT is defined in the following standard:

a) ISO/IEC 18033-3:2010: Encryption Algorithms, Part 3: Block Ciphers.

In addition to the specification document [18], PRESENT is defined in the following standard:

a) ISO/IEC 29192-2:2012: Lightweight Cryptography, Part 2: Block Ciphers.

6.2 Usage

This section outlines several guidelines on the usage of block ciphers.

6.2.1 Modes of Operation

A block cipher is typically used to encrypt bulk data that is longer than its own block length. Therefore, in such a case, the block cipher must be used in any of the following two types of modes of operation:

a) Confidentiality.
b) Authenticated encryption (AE).

Confidentiality Modes As the name suggests, confidentiality modes only provide secrecy of data. On its own, the mode does not provide data integrity and data-origin authentication protection. Therefore, usage of this mode is not recommended for applications involving secure communication between two or more entities, unless the confidentiality mode is combined with a secure MAC scheme. If properly implemented, the combined scheme provides two additional services, i.e. integrity and data-origin authentication. Should this approach be taken, the user needs to carefully implement them as they are prone to implementation errors.

A block cipher can be operated in any of the following confidentiality modes of operation [19]:

a) Electronic Code Book (ECB).
b) Cipher Block Chaining (CBC).
c) Cipher Feedback (CFB).
d) Output Feedback (OFB).
e) Counter (CTR).

Table 4: Block cipher confidentiality modes of operation.

Mode	IV	Remarks
ECB	n/a	For encrypting a message that is equal or less than the block length. Each message needs to be encrypted with a different key.
CBC	Random	Encryption is serialized (cannot be parallelized) but decryption can be parallelized.
CFB	Random	Only the encryption function of the block cipher is required to perform message encryption and decryption. The mode is similar to a stream cipher and requires the message to be present in order to generate the next output keystream.
OFB	Nonce	Only the encryption function of the block cipher is required to perform message encryption and decryption. The mode is similar to a stream cipher and does not require the message to be present in order to generate the next output keystream.
CTR	n/a	Make use of a counter to perform encryption. The mode is similar to a stream cipher.

Table 4 provide further information on block cipher confidentiality modes of operation. In general, a confidentiality mode of operation requires the following inputs:

a) Message. The message may compose of more than one block and its entire length may not be in multiple of the block length. The message may be a string, file, or an entry in a database.
b) Secret Key. The selected key length depends on the algorithm and application.
c) Initialisation Vector (IV). The IV is a public value (not secret) used to start the encryption process and is changed whenever a new message is to be encrypted while retaining the same key value. An IV may be a nonce or a random value.
    i) A nonce is a number used once. The same value cannot be reused to encrypt a different message with the same key. Therefore, the current nonce may be incremented by one to produce the next nonce value.
    ii) A random value has the requirement that the next value cannot be predicted from the current and past values. A sequence of random values can also be seen as nonce, with a stronger requirement that subsequent values cannot be predicted from current and past values.

The mode outputs the ciphertext which length is equal to the original message.

Authenticated Encryption Modes An authenticated encryption (AE) mode provides three essential properties: confidentiality, data-origin authentication and data integrity. In general, this mode is suitable to be used in applications involving communication between two or more parties. The following AE modes of operation may be used:

a) Galois Counter Mode (GCM) as defined in NIST SP800-38D.
b) Counter Mode with Cipher-block-chaining Message authentication code (CCM) as defined in NIST SP800-38C.

6.2.2 Counter and IV Generation

As stated earlier, CTR mode necessitates a counter to perform encryption and decryption. Likewise, CBC and CFB modes require an IV that needs to be random. OFB, on the other hand, requires that the IV is a nonce.

Counter Generation The counter in CTR mode can be implemented via an incrementing function. The function increments the current counter by one modulo an integer. Assume that a block cipher with a block length of b = 128 bits is employed. The incrementing function can be employed such that only the least $m \leq b$ bits are involved. The function then increments the value of the current counter modulo 2^m. The following is an example of five subsequent counter values for m = 48.

```
0000   0000   0000   0000   0000   FFFF   FFFF   FFFD

0000   0000   0000   0000   0000   FFFF   FFFF   FFFE

0000   0000   0000   0000   0000   FFFF   FFFF   FFFF

0000   0000   0000   0000   0000   0000   0000   0000

0000   0000   0000   0000   0000   0000   0000   0001
```

In the above example, the incremented bits of the counter starts with *FFFF FFFF FFFD*. After the value reached *FFFF FFFF FFFF*, the counter wraps around to *0000 0000 0000* since the incrementing function is performed modulo 2^{48}. Note that the incremented bits may start with any value, including all zeros.

So, the number of message blocks encrypted per the unique initial counter under a given key should be less than or equal to $2^m = 2^{48}$. If the number of blocks is greater than this, then the same counter values would be reused. This potentially allows an adversary to recover the original plaintext.

Initial Value of Counter According to NIST SP800-38 [19], there are two approaches in choosing the initial counter block values. The first approach is to use the incrementing function to the m bits of the counter for all plaintext messages. The entire plaintext messages is therefore treated as a single message.

Assume that in the current session, there are 3 different files (i.e. 3 different plaintext messages) to be encrypted and sent over the network. The lengths of the files are 160 bytes, 16 Kilobytes and 16 Megabytes. The underlying block cipher has a block length of b = 128 bits (16 bytes). Let the initial counter value be all zeros and let the least m = 24 bits of the counter denote the bits processed by the incrementing function.

The first file (i.e. first plaintext message) consists of 10 blocks, the second file 1,000 blocks while the third file 1,000,000 blocks. The initial and last counter block for the first file are therefore given below.

0000 0000 0000 0000 0000 0000 0000 0000

0000 0000 0000 0000 0000 0000 0000 0009

The initial and last counter block for the second file are:

0000 0000 0000 0000 0000 0000 0000 000A

0000 0000 0000 0000 0000 0000 0000 03F1

Lastly, the initial and last counter block for the third file are:

0000 0000 0000 0000 0000 0000 0000 03F2

0000 0000 0000 0000 0000 0000 000F 4631

If this approach is taken, then a mechanism should be in place to ensure that the total number of plaintext message blocks should not exceed 2^m.

The second approach for choosing the initial counter block values would be to treat the b-bit counter block as the concatenation of a $(b/2)$-bit nonce and a $(b/2)$-bit incrementing value. The first $(b/2)$ bits, i.e. the nonce, is changed whenever a new plaintext message is to be encrypted. The remaining bits are processed by the incrementing function. Let $T_{i,j}$ and N_i denote the counter block and nonce for the i-th plaintext message and j-th block. Then the counter block values for the previous example with the 3 different files can be viewed as follows.

$$T_{1,j} = N_1||(j \bmod 2^{\mathrm{m}}) \qquad \text{for } j = 1, 2, \ldots, 10$$

$$T_{2,j} = N_2||(j \bmod 2^{\mathrm{m}}) \qquad \text{for } j = 1, 2, \ldots, 1000$$

$$T_{3,j} = N_3||\ (j \bmod 2^{\mathrm{m}}) \qquad \text{for } j = 1, 2, \ldots, 1000000$$

As the counter block values are preceded by a nonce, then $N_i \neq N_j$ if $i \neq j$. The number of plaintext message block given an initial counter block value thus should be smaller than or equal to 2^m.

IV Generation As mentioned earlier, an IV is not secret and therefore, the IV, or information sufficient to construct the IV, can be transmitted in the clear. To generate random IVs, NIST SP800-38A recommends two methods. The first method is to encrypt a nonce with the same block cipher used for encryption for a given key. The output ciphertext block can be used as a random value for the IV. The IV for subsequent plaintext messages can be obtained by incrementing the nonce and use the corresponding output ciphertext blocks as the IVs.

If the mode requires the IV to be a nonce, then IV can be implemented using a counter or a message number.

7 Stream Ciphers

A stream cipher provides confidentiality of bulk data by applying the XOR operation between the plaintext message and a stream of secret bits known as the keystream. The keystream is produced by a keystream generator where its input is secret. Stream cipher is malleable, i.e. bits of its ciphertext can be flipped to produce a predictable change in the decrypted plaintext. When integrity and data-origin authenticity of the data is important, a stream cipher must be used in conjunction with a message authentication code (MAC).

7.1 Algorithms

There are three AKSA MySEAL recommended stream cipher algorithms: KCipher-2, Rabbit, and ChaCha20.

In addition to its specification document [20], KCipher-2 is defined in the following standard:

a) ISO/IEC 18033-4:2011: Encryption Algorithms, Part 4: Stream Ciphers.

In addition to its specification document [21][22], Rabbit is defined in the following standard:

a) ISO/IEC 18033-4:2011: Encryption Algorithms, Part 4: Stream Ciphers.

In addition to its specification document [23], ChaCha20 is defined in the following IETF memo:

a) IETF RFC 8439: ChaCha20 and Poly1305 for IETF Protocols.

7.2 Usage

In general, a stream cipher is composed of the following components:

a) Keystream generator. This component accepts a secret key, along with other inputs to produce a string of bits known as the keystream.
b) Encryption/decryption function. This component performs an XOR operation between the plaintext message bits and the keystream to produce the corresponding ciphertext.

A stream cipher algorithm typically describes how the keystream is produced (i.e. the keystream generator), as is the case for all AKSA MySEAL recommended stream ciphers. Encryption and decryption operations are simply the XOR operation.

The keystream generator requires the following inputs:

a) Secret Key. The selected key length depends on the algorithm and application.
b) IV. The IV is a public value (not secret) used to start the encryption process and is changed whenever a new message is to be encrypted while retaining the same key value. For all AKSA MySEAL recommended stream ciphers, the IV is a nonce.

The keystream generator outputs a sequence of bits to be used in the XOR operation in the encryption/decryption function.

At every cycle, KCipher-2 produces 64 bits of keystream. Under the same secret key and IV, the maximum number of cycles that can be performed is 2^{55} which resulted in 2^{58} x 64 = 2^{64} bits of keystream [24]. Rabbit produces 128 bits of keystream per cycle. Under the same key and IV, a maximum of 2^{64} cycles may be performed, which allows a maximum of 2^{64} x 128 = 2^{71} keystream bits [22]. ChaCha20 generates 512 bits of keystream per cycle. Under the same key and IV, ChaCha20 allows a maximum of 2^{64} cycles to be performed which leads to a maximum of 2^{64} x 512 = 2^{73} bits of keystream [25].

A user should be aware of the following:

a) There needs to be a mechanism to ensure that the maximum number of keystream bits produced by any of the AKSA MySEAL recommended stream ciphers do not exceed the stated bounds stated in Table 5. Failure to abide to this may allow an attacker to recover the original plaintext message.

b) A stream cipher, on its own, does not provide integrity and data-origin authentication. Therefore, a stream cipher is not recommended to be used in secure communication between two or more parties unless it is combined with a secure MAC scheme. In the specific case of ChaCha20, it is recommended to be used with the Poly1305 MAC scheme as stated in IETF RFC 8439.

c) A block cipher in CTR mode (see Section 6.2.1) provides similar functionality to that of a stream cipher. The decision whether to use a block cipher in CTR mode, or a stream cipher, depends on applications and possibly on the underlying hardware implementing the application. For instance, in an IETF note, it was reported that, on a specific Intel Xeon machine, ChaCha20 with Poly1305 performs about three times faster than an AES-GCM implementation with the AES New Instructions (AES-NI) enabled[7]. In certain applications, AES may be preferred to conserve area requirements since AES can be used in other primitive such as in AES-based DRBGs (see Section 13).

Table 5: Summary of stream cipher parameters.

Algorithm	Key Length (bits)	IV Length (bits)	Bits per cycle	Maximum Number of Keystream Bits
KCipher-2	128	128	64	2^{64}
Rabbit	128	64	128	2^{71}
ChaCha20	256	64	512	2^{73}

8 Hash Functions

A hash function provides integrity of a message by producing a fixed-length hash digest. The hashing process is non-reversible, i.e. given a hash digest, it is computationally difficult to recover the original message. Hash functions may be used in various applications. It can be used in applications where public verifiability is required, i.e. any other user can perform the same operation to verify the integrity of a message. It can also be used to produce a unique identification (ID) from a given set of data.

8.1 Algorithms

The AKSA MySEAL recommended hash functions can be categorised into general-purpose and lightweight.

8.2 General-Purpose

There are three AKSA MySEAL recommended hash function algorithms, namely SHA2, SHA3, and SHAKE. Note that not all digest lengths are recommended. Refer to Table 1 for details.

The SHA2 family of hash functions are defined in the following standards:

a) NIST FIPS PUB 180-4 – Secure Hash Standard (SHS), and
b) ISO/IEC 10118-3:2018 Hash Functions, Part 3: Dedicated Hash Functions.

The SHA3 family of hash functions and SHAKE are defined in the following standards:

[7] See https://datatracker.ietf.org/meeting/88/materials/slides-88-tls-1.

a) NIST FIPS PUB 202 – SHA-3 Standard: Permutation-Based Hash and Extendable-Output Functions.
b) ISO/IEC 10118-3:2018 Hash Functions, Part 3: Dedicated Hash Functions.

8.3 Lightweight

There are two AKSA MySEAL recommended lightweight hash function algorithms, namely SPONGENT and PHOTON. Apart from its specification document [26], SPONGENT is defined in the following standard:

a) ISO/IEC 29192-5:2016: Lightweight Cryptography, Part 5: Hash Functions.

Apart from its specification document [27], PHOTON is defined in the following standard:

a) ISO/IEC 29192-5:2016: Lightweight Cryptography, Part 5: Hash Functions.

A cryptographic hash function has the following properties:

a) Collision resistance. It is computationally infeasible to find two different messages that hashed to the same digest.
b) Pre-image resistance. Given a hash digest, it is computationally difficult to find the original message that produced the digest.
c) Second pre-image resistance. Given a message, it is computationally difficult to find another message that hashed to the same digest.

The above properties provide a useful guide to choose a suitable hash function for a particular purpose or application. For instance, in digital signature schemes, the hash function used must be strong in both collision and pre-image resistance. For public verifiability such as digest for a downloadable file, the hash function used must have a strong second pre-image resistance. In this setting, the hash function may have weak collision resistance. Intentionally, all AKSA MySEAL recommended hash functions meet the required expected strength in collision, pre-image and second pre-image resistance.

If n is the length of the hash digest in bits, then the collision resistance strength of all AKSA MySEAL recommended hash functions is $n/2$ bits. In other words, if the length of the hash digest is, say 256 bits, one may expect to find a collision by performing an average of 2^{128} operations. Additionally, the pre-image resistance is n bits. Except for SHA-512, the second pre-image resistance for all AKSA MySEAL recommended hash functions is n bits. The strength of all AKSA MySEAL recommended hash functions is summarised in Table 6 [28][29].

Table 6: Summary of the security strength provided by AKSA MySEAL recommended hash functions

Algorithm	Digest Length	Collision	Pre-image	2nd Pre-image
SHA-384	384	192	384	384
SHA-512*	512	256	512	512 - M
SHA-512/224	224	112	224	224
SHA-512/256	256	128	256	256
SHA3-224	224	112	224	224
SHA3-256	256	128	256	256
SHA3-384	384	192	384	384
SHA3-512	512	256	512	512
SHAKE128**	d	min(d/2,128)	≥ min(d, 128)	min(d, 128)
SHAKE256**	d	min(d/2,128)	≥ min(d, 128)	min(d, 128)
PHOTON-80	80	40	64	40
PHOTON-128	128	64	112	64
PHOTON-160	160	80	124	80
PHOTON-224	224	112	192	112
PHOTON-256	256	128	224	128
SPONGENT-88	88	40	80	40
SPONGENT-128	128	64	120	64
SPONGENT-160	160	80	144	80
SPONGENT-224	224	112	208	112
SPONGENT-256	256	128	240	128

*SHA-512 has an input block length of 2^{10} bits and let 2^M denote the number of message blocks. If the input message length is 2^{30} bits (1 gibibyte), then $M = \log_2(2^{30}/2^{10}) = 30 - 10 = 20$. Hence, the second pre-image resistance of SHA-512 for this particular input length is 512 − 20 = 492 bits.

**min(a, b) takes the smallest value of the two values a and b.

8.4 Usage

This section outlines several guidelines on the usage of hash functions.

8.4.1 Truncating a Hash Digest

If an application requires a hash digest that is shorter than the one produced by a particular hash function algorithm, then its hash digest length can be shortened. The truncated hash digest shall be obtained from the leftmost portion of the hash digest. For instance, if SHA3-256 is used, and the hash digest is to be truncated to 128 bits, then the leftmost 128 bits of the output of SHA3-256 is taken as the truncated hash digest.

As an example, the following is the 256-bit hash digest of SHA3-256 for the message “myseal”.

793cd5eb9037d698a5b7d2bfcfeee159d49ae83c7c7b3d49f3e03ac1d2e17ee8

If the user wants to truncate the output to 128 bits, then the following truncated hash digest is used.

793cd5eb9037d698a5b7d2bfcfeee159

Note that the security strength of truncated hash digest decreases accordingly based on the truncated length. An n-bit hash digest truncated to l bits (l < n) has a collision resistance strength of l /2 bits (instead of $n/2$). Both pre-image and second pre-image resistance is also down from n bits to l bits, except for SHA-512, where the length of message affects the second pre-image resistance strength.

If an application requires a collision-resistance strength of, say 112 bits, then the truncated hash digest length should be 224 bits, i.e. double the required strength.

8.4.2 Use in Digital Signature Schemes

A message is not directly signed by a digital signature scheme. The message is first hashed and the corresponding hash digest will be used in the actual signing process (e.g. elliptic curve or RSA private key operation). Let $H(m)$ be a hash function operation on a message m and let Sign(k_{priv}, h) denote the signing operation with the private key k_{priv} and digest h. The application of the hash function in the digital signing process is given below:

$$\text{Sign}(k_{\text{priv}}, H(m))$$

The output of this operation is the digital signature of the message. In many cryptographic libraries, this process is transparent to the user, i.e. the signing function already includes the hash function operation. In the case where such automated functionality is not available, the hash operation needs to be performed before the signing operation.

A developer should take note of the following when using a hash function in a digital signature scheme:

a) The output of the hash function may be used entirely or truncated to a desired length. For instance, the entire 384-bit output of SHA3-384 can be directly used or truncated to 320 bits. Note the reduction of security strength as discussed in Section 8.4.1.
b) It is not recommended to truncate the hash digest length if there is no specific need for it.
c) If a hash digest length is already supported by one of the AKSA MySEAL recommended hash function algorithms, then use that algorithm instead of another algorithm and later truncate its output. For instance, use SHA-384 or SHA3-384 for a 384-bit hash digest instead of using SHA3-512 and later truncate its output to 384 bits.

9 Asymmetric Encryption Schemes

Asymmetric encryption schemes provide confidentiality of data. Due to their lack of speed, these schemes are typically used to encrypt short data such as symmetric encryption keys. The length of a symmetric key ranges from 80 to 256 bits. The MySEAL AKSA recommended asymmetric encryption schemes include a special type of scheme called the *key encapsulation mechanism (KEM)*. Such schemes include methods to generate the symmetric encryption key and subsequently encapsulate the key in the asymmetric encryption ciphertext. KEMs and other asymmetric encryption schemes are normally used in a hybrid encryption setting where the bulk message is encrypted by symmetric encryption.

9.1 Algorithms

There are two AKSA MySEAL recommended schemes based on the RSA primitive, i.e.:

a) RSA-OAEP which is based on the PKCS#1 v2.2 RSA Cryptography Standard [30].
b) RSA-KEM.

There are three AKSA MySEAL recommended encryption schemes based on the elliptic curve primitive, and all of them are KEMs:

a) ECIES-KEM.
b) PSEC-KEM.
c) ACE-KEM.

Note that the specification of the above schemes is defined in ISO/IEC 18033-2.

There is only one AKSA MySEAL recommended lattice-based encryption scheme, i.e. NTRU. The adopted NTRU scheme variant is the Short Vector Encryption Scheme (SVES) defined in the following standards:

a) IEEE P1363.1-2008: IEEE Standard Specification for Public Key Cryptographic Techniques Based on Hard Problems over Lattices (2008).
b) ANSI X9.98 Lattice-Based Polynomial Public Key Establishment Algorithm for the Financial Services Industry (2011).
c) Efficient Embedded Security Standards (EESS) EESS #1: Implementation Aspects of NTRUEncrypt and NTRUSign (2003) [31].

9.2 Usage

The following subsections outline several guidelines on the usage of RSA, elliptic curve and lattice-based encryption schemes.

9.2.1 RSA-Based Encryption Schemes

The recommended lengths of the public modulus are given in Table 7. See also Section 5.2. A user should be aware of the following [7]:

a) Appendix B.3 of NIST FIPS 186 should be consulted to generate a pair of public and private keys.
b) The primes p and q may be generated by any AKSA MySEAL recommended prime number generators and their values shall be kept secret.
c) In general, the value of the public exponent e is an integer between 3 and $(p-1)(q-1)$ that satisfies $\text{GCD}(e, (p-1)(q-1)) = 1$. However, for both security and efficiency purposes, the value e shall be in the range $65537 \leq e < 2^{256}$. Note that the larger the value of e, the longer it takes to perform encryption or signature verification. To maximize efficiency, the value $e = 2^{16} + 1 = 65537$ is a prudent choice [32].
d) The key generation algorithm may be performed by a trusted third party that generates the key pair for a client. Any correspondence to this particular client will make use of the key pair for a specified period of validity. Therefore, depending on application, the key generating algorithm may not need to be performed every time prior to encryption.
e) The recipient of the public key should make the necessary checks to ensure it receives an authentic copy of the owner's public key. The checks can be done, among others, by verifying the digital certificate corresponding to the public key, that was signed by a trusted third party such as a CA.
f) A key pair for an encryption by one particular scheme should not be used for another scheme or purpose (e.g. signing).

Table 7: The lengths of the public modulus for RSA-based encryption schemes.

Security	Length of public modulus *n*
112	2048
128	3072
192	7680
256	15360

RSA-OAEP. RSA-OAEP makes use of the Optimal Asymmetric Encryption Padding (OAEP) scheme introduced by Bellare and Rogaway [33]. RSA-OAEP accepts messages with length up to ($nLen - 2 \times hLen - 2$) octets where *nLen* is the length of the RSA public modulus and *hLen* is the length in octets of the hash digest of the underlying hash function. Table 8 provides examples of the maximum plaintext lengths for a given length of the hash digest. RSA-OAEP is typically used to encrypt short messages such as symmetric encryption keys.

Table 8: Examples of maximum message lengths given a particular digest and public key length.

Digest Length		Public Key Length		Maximum Message Length	
bit	octet	bit	octet	bit	octet
256	32	3072	384	2544	318
384	48	7680	960	6896	862
512	64	15360	1920	14320	1790

RSA-KEM. RSA-KEM is a key encapsulation mechanism (KEM) that is capable of generating and encapsulating symmetric encryption keys. A KEM is often employed with a data encapsulation mechanism (DEM) to encrypt data. Both RSA-KEM and recommended DEMs are defined in ISO/IEC 18033-2.

9.2.2 Elliptic Curve-Based Encryption Schemes

In addition to the elliptic curves defined in FIPS 186-4, alternative recommended curves to be used by the schemes are:

a) Curve25519 due to Bernstein [34], and
b) Curve448 due to Hamburg [35]

Both of the above are as defined in IETF RFC 7748 [36].

9.2.3 Lattice-Based Encryption Scheme

The NTRU encryption scheme is owned by Security Innovation, Inc and has made available a sample open source implementation at GitHub[8]. The key length of the NTRU encryption scheme is $N \times \log_2 q$ bits as shown in Table 9.

[8] https://github.com/NTRUOpenSourceProject/ntru-crypto

Table 9: Parameters for the NTRU encryption scheme.

Security	N	q	Key Length
128	439	2048	4829
192	593	2048	5929
256	743	2048	8173

9.3 Selection of Scheme

At the time of this document is published, all AKSA MySEAL recommended asymmetric encryption schemes are currently secure against all practically known attacks. The selection of specific algorithm to be used in applications may depend on several factors.

a) **Compatibility with existing applications.** If an existing application already made use of any of the AKSA MySEAL recommended asymmetric encryption scheme, then a developer may select to use the already implemented algorithm.
b) **Small key length.** In devices where the application is running have limited storage or processing power, it is logical to choose an algorithm that supports smaller key lengths. For instance, for 256-bit security strength, an RSA-based primitive needs a key length of 15360 bits as opposed to an ECC-based primitive which require only 512 bits.
c) **Resistance against attacks using quantum computers.** If security is highly crucial, then NTRU is the only AKSA MySEAL recommended asymmetric encryption scheme that is known to be resistant against attacks using quantum computers. However, for the same security level, the key length for NTRU is significantly longer than ECC-based schemes, i.e. more than 15 times longer.

10 Digital Signature Schemes

A digital signature scheme performs its operation using a pair of public and private keys. A message is signed using the private key and the generated signature can be verified by anyone in possession of the corresponding public key. A digital signature scheme provides non-repudiation, i.e. once an entity has signed a data, the entity cannot later deny that it has signed the data. Additionally, a digital signature also offers protection on the integrity and data-origin authenticity of the data. A digital signature scheme does not provide confidentiality of the data to be signed.

10.1 Algorithms

There are three AKSA MySEAL recommended digital signature schemes:

a) DSA,
b) ECDSA, and
c) RSA-PSS.

DSA and ECDSA are both based on discrete logarithm, while RSA-PSS is based on the RSA primitive. All of the above schemes fall under the category of digital signature scheme with appendix, as opposed to digital signature scheme with message recovery. In the former, the message is not concealed while in the latter, the message is partially or fully embedded in the signature.

DSA is defined and included in the following standards:

a) NIST FIPS PUB 186-4 – Digital Signature Standard (DSS),

b) ISO/IEC 14888-3:2016 Digital Signatures with Appendix, Part 3: Discrete Logarithm Based Mechanisms, and
c) IEEE P1363: IEEE Standard Specifications for Public-Key Cryptography.

ECDSA is defined and included in the following standards:

a) ISO/IEC 14888-3:2016 Digital Signatures with Appendix, Part 3: Discrete Logarithm Based Mechanisms,
b) NIST FIPS PUB 186-4 – Digital Signature Standard (DSS),
c) ANSI X9.62 – Public Key Cryptography for the Financial Services Industry: The Elliptic Curve Digital Signature Algorithm (ECDSA),
d) IEEE P1363: IEEE Standard Specifications for Public-Key Cryptography (2000), and
e) Standards for Efficient Cryptography 1 – SEC 1: Elliptic Curve Cryptography Version 2.0 (2009).

RSA-PSS is defined and included in the following standards and IETF memo:

a) ISO/IEC 14888-2:2008 Digital Signatures with Appendix, Part 2: Integer Factorization-Based Mechanisms, and
b) IETF RFC 8017: Public-Key Cryptography Standards (PKCS) #1: RSA Cryptography Specifications Version 2.2.

10.2 Usage

A user should be aware of the following:

a) The verifier of a signature should make the necessary checks to ensure it receives an authentic copy of the signer's public key. The checks can be done, among others, by verifying the digital certificate corresponding to the public key, that was signed by a trusted third party such as a CA.
b) A key pair for digital signature generation and verification shall not be used for any other purpose such as key establishment.
c) The message to be signed is not necessarily be made secret.

10.2.1 DSA

In FIPS 186-4, the DSA standard specifies, among others, the parameters p and q, where p is a prime modulus and q is a prime divisor of $(p - 1)$. The lengths of p and q in bits are denoted as L and N, respectively. In ISO/IEC 14888-3, these parameters are labelled as α and β, respectively. Table 10 provides the recommended values for L and N for given security strengths. Note that in FIPS 186-4, a prime modulus of 1024 bits, which provides 80 bits of security, is defined. However, according to NIST SP800-57 Part 1 (published after FIPS 186-4), a security strength of less than 80 bits is no longer allowed.

Table 10: Recommended parameters for DSA.

Security	L	N
112	204	22
112	204	25
128	307	25

A user should be aware of the following:

a) The secret message number (denoted r in FIPS 186-4 and called randomiser K in ISO/IEC 14888-3) must be unique and random for any new message to be signed.

b) FIPS 186-4 specifies two methods to generate this secret message number that makes the signature generation process probabilistic (i.e. the signature generation process produces a unique signature for the same message at different times).
c) IETF RFC 6979 specifies a method to generate this secret message number that depends on the values of the private key and the message to be signed. This makes the signature generation process deterministic.
d) The same secret message number must never be used to sign different messages.
e) The next value of the secret message number must never be trivially predicted given the current and/or any previous message number.

10.2.2 ECDSA

The ISO/IEC 18333-4 standard recommends that users check the proper generation of the public parameters according to FIPS 186-4 and ANSI X9.62-2005. In addition to the elliptic curves defined in these standards, alternative recommended curves to be used by the schemes are:

a) Curve25519 due to Bernstein [34], and
b) Curve448 due to Hamburg [35]

Both of the above are as defined in IETF RFC 7748 [36].

10.2.3 RSA-PSS

The signature process generation of RSA-PSS is probabilistic, i.e. a unique signature is generated for the same message at different times. The randomness is provided by the salt value that is used as input to the EMSA-PSS encoding method defined in IETF RFC 8017. The user should be aware of the following:

a) The length of the salt value may be an octet string of length 0. If this is the case, then the salt is the empty string.
b) It is recommended that the hash functions used in the EMSA-PSS encoding, including in the underlying mask generation function, are the same. For typical applications, all AKSA MySEAL recommended hash function algorithms that produce a minimum digest length of 224 bits may be used. This includes selected SPONGENT and PHOTON variants.
c) The user needs to ensure that the value of the seed in the EMSA-PSS encoding is generated pseudorandomly.
d) The same private key can be used to sign multiple messages, as long as the salt value is properly generated each time the signature generation process is executed.

11 Key Agreement Schemes

A key agreement scheme provides a way for two parties to share a common secret key without having to pre-share a secret. Both parties exchange public parameters that allow each party to compute the same secret value. No single party has the ultimate influence in determining the final secret value.

11.1 Algorithms

There are two AKSA MySEAL recommended key agreement schemes:

a) DH, and
b) ECDH

11.2 Usage

There are various ways to securely implement DH and its elliptic curve variant, ECDH. The specific mechanisms of the AKSA MySEAL recommended key agreement schemes are defined in ISO/IEC 11770-3: 2015, which contains 12 key agreement mechanisms.

The mechanisms differ in the following properties.

a) No. of passes, which denote the number of exchanges between two or more entities.
b) Implicit key authentication, i.e. where an entity B is assured that an entity A is the only other entity that can possibly be in possession of the correct key, and vice versa.
c) Key confirmation, i.e. whether or not the mechanism provides key confirmation from the participating entities.
d) Entity authentication, i.e. whether or not the mechanism provides one-way or mutual authentication.
e) Forward secrecy, i.e. whether or not the mechanism provides one-way or mutual forward secrecy.
f) Key freshness, i.e. whether or not the participating entities are protected from replay attacks.

A user should be aware of the following.

a) There needs to be a mechanism to ensure that each entity receives an authentic copy of the public parameters of the other entity. The checks can be done, among others, by verifying the digital certificate corresponding to the public key, that was signed by a trusted third party such as a CA.
b) A key agreement scheme can be used to generate a temporary key that is valid only for a particular session. A different shared key is computed for every new subsequent session.

12 Prime Number Generators

A prime number generator produces prime numbers suitable for use in other cryptographic primitives that require prime numbers as input. Among others, a prime number generator involves running specific tests to validate whether a given integer is probably prime, composite, or inconclusive. The number is accepted as prime if the result of the validation process indicates that the number is probably prime, else the number is rejected. Prime numbers are primarily used in asymmetric cryptographic schemes.

12.1 Algorithms

There are three AKSA MySEAL recommended prime number generator algorithms:

a) Miller-Rabin primality test,
b) Elliptic curve primality proving (ECPP), and
c) Shawe-Taylor.

Both Miller-Rabin primality test and Shawe-Taylor algorithm are defined in FIPS 186-4. The definition for ECPP can be found in Atkin and Morain [37]. The Miller-Rabin primality test may be run in parallel with ECPP to produce an output for prime validation[9].

12.2 Usage

A user should refer to the following recommended documents on generating prime numbers.

a) NIST FIPS 186-4: Digital Signature Standard (DSS).

[9] https://maths-people.anu.edu.au/~brent/pd/comp4600_primality.pdf.

b) ANSI X9.80: Prime Number Generation, Primality Testing and Primality Certificates.
c) ISO/IEC 18032:2005 Prime Number Generation. Note that at the time of writing, a revised version of this standard is currently in the works.

13 Deterministic Random Bit Generators

A deterministic random bit generator (DRBG) produces a bit string that appears random[10]. The output bit string can be used directly or converted to a different representation such as integer, if required by the cryptographic primitive. A DRBG is deterministic because it will produce the same output bits given the same initial input, i.e. the seed.

13.1 Algorithms

There are four AKSA MySEAL recommended DRBGs:

a) SHA2-DRBG,
b) HMAC-SHA2-DRBG,
c) 3-Key TDEA-CTR-DRBG, and
d) AES-CTR-DRBG.

The specific key or digest lengths of the above algorithms are given in Table 1 The definition for each of the above algorithms can be obtained from NIST SP-800-90A Revision 1. In the categorisation used by the NIST document, SHA2-DRBG and HMAC-SHA2-DRBG fall under the Hash_DRBG and HMAC_DRBG categories, respectively. On the other hand, 3-Key TDEA-CTR-DRBG and AES-CTR-DRBG are in the CTR_DRBG category. Although the NIST SP-800-90A allows a broader range of algorithm selection, the AKSA MySEAL recommended DRBGs are the instances specified in Table 1. For example, if the user wishes to use HMAC-SHA2-DRBG, then the only variants recommended are the ones utilising SHA2-384 and SHA2-512.

13.2 Usage

A user should be aware of the following:

a) The input seed to the DRBG shall be generated from a randomness source and its value be kept secret.
b) Apart from the input seed, the DRBG may receive other additional inputs such as time and nonce. These other inputs may not necessarily be kept secret.
c) A DRBG is instantiated with a seed and may be further reseeded with different seed values.
d) Each seed should have a seed period so that it can be reseeded after the period has elapsed.
e) The user is recommended to refer to NIST SP-800-90A Revision 1 for an extensive resource for secure implementation of DRBGs.

The maximum security strength supported by a DRBG based on a hash function (Hash_DRBG and HMAC_DRBG) is the security strength of the underlying hash function for pre-image resistance. For a DRBG based on a block cipher (CTR_DRBG), the maximum security strength supported is the security strength of the underlying block cipher, based on its block and key lengths. This is summarised in Table 111 where the digest lengths, key lengths and security strengths are in bits.

[10] DRBGs are also referred to as pseudorandom number (or bit) generators

Table 11: Security strengths of DRBGs.

Algorithm	Digest/Key Length	Security Strength
SHA2-DRBG	384	384
	512	512
	512/224	224
	512/256	256
HMAC-SHA2-DRBG	384	384
	512	512
3-Key TDEA-CTR-DRBG	168	112
AES-CTR-DRBG	128	128
	192	192

14 Remarks on Implementation

A user may opt to implement the AKSA MySEAL recommended algorithms from scratch or choose from the various established cryptographic libraries that already contain the implementation of one or more AKSA MySEAL recommended algorithms. This section briefly discusses these two issues.

14.1 Implementing Algorithms from Scratch

A cryptographic algorithm that has been shown to be theoretically secure does not necessarily be secure in practice. There are numerous real-world attacks involving cryptography that are not due to the algorithm itself, but due to how the algorithm is implemented. There are also attacks caused by insecure algorithms such as RC4, DES, MD5 and SHA-1. However, these algorithms have already been excluded in the AKSA MySEAL list and no longer recommended by international bodies such as the US NIST.

Securely implementing cryptographic algorithms from scratch is non-trivial and such task is prone to errors [38][39][40]. A developer may not be adept at secure programming which potentially allow for a flawed implementation to be exploited by attackers. Even well-known cryptographic libraries such as OpenSSL has been subjected to various attacks (e.g. [41][42]). A prominent example is the Heartbleed bug, when exploited, allows memory contents of the server to be leaked to the client, and vice versa. The memory content contains cryptographic key materials that should be protected from being exposed.

Developers who wish to implement AKSA MySEAL cryptographic algorithms from scratch must be competent in secure programming and must be aware of the various cryptographic usage issues raised in this document.

14.2 Making Use of Cryptographic Libraries

Since most AKSA MySEAL recommended algorithms are included in various international standards, there are existing implementation of the algorithms, typically packaged as a software or hardware library in various programming languages. A user may wish to use these existing implementations. Examples of well-known cryptographic libraries include, but not limited to:

a) Bouncy Castle[11]

[11] https://bouncycastle.org/

b) Google Tink[12]
c) LibreSSL[13]
d) Mbed TLS (formerly known as PolarSSL)[14]
e) OpenSSL[15]
f) WolfSSL[16]

Note that the above list should not be interpreted as recommendations by AKSA MySEAL. They are given only as examples. The user should perform the necessary due diligence when selecting an open-source cryptographic library and be aware of the latest attacks and updates to the library. An alternative way is to appoint a competent third party to implement the algorithms for the user.

There are several cryptographic libraries that are certified according to the Federal Information Processing Standard (FIPS) 140-2 [43][17]. FIPS 140-2 certified cryptographic libraries are validated by the US National Voluntary Laboratory Accreditation Program (NVLAP) accredited laboratories based on four pre-defined security levels. Cryptographic libraries such as Bouncy Castle, OpenSSL and WolfSSL offer FIPS-certified versions, in addition to the regular non-certified versions. Note that due to the time and cost it takes to perform the certification, a FIPS 140-2 certified cryptographic library may not be regularly released or updated. Using a FIPS 140-2 certified cryptographic library is mandatory if the application or product employing the library is to be used by the US or Canadian government. For the Malaysian government, it is currently not a requirement for developers to use a FIPS 140-2 certified cryptographic library.

14.3 References Related to Implementation

Note that secure implementation must not be focused solely on cryptography. In fact, all aspects of the application that uses cryptography must be secure as well. Useful references to a developer implementing or incorporating cryptography include, but not limited to, the following:

a) Android Developers' App Security Best Practices[18].
b) Apple Developer Documentation: Security[19].
c) Carnegie Mellon University's Software Engineering Institute (SEI) CERT Coding Standards[20]. The site includes secure programming standards for C, C++, Java and Perl programming languages.
d) OWASP Cheat Sheet Series Project[21].
e) Red Hat Secure Coding[22].

[12] https://github.com/google/tink

[13] https://www.libressl.org/

[14] https://tls.mbed.org

[15] https://www.openssl.org/

[16] https://www.wolfssl.com

[17] At the time of writing, FIPS 140-3 has just been published on March 22, 2019 and will take effect on September 22, 2019

[18] https://developer.android.com/topic/security/best-practices

[19] https://developer.apple.com/documentation/security

[20] https://wiki.sei.cmu.edu/confluence/display/seccode/SEI+CERT+Coding+Standards

[21] https://cheatsheetseries.owasp.org

[22] https://developers.redhat.com/topics/secure-coding/

Bibliography

[1] K. M. Martin, “Everyday Cryptography,” *Oxford Univ. Press*, 2012.

[2] N. Fazio, “On Cryptographic Techniques for Digital Rights Management,” PhD thesis. New York University, 2006.

[3] CA/Browser Forum, “Baseline Requirements for the Issuance and Management of Publicly-Trusted Certificates Version 1.6.6,” *CA/Browser Forum Website*, 2019.

[4] WebTrust/PKI Assurance Task Force, “WebTrust for Certification Authorities: WebTrust Principles and Criteria for Certification Authorities Version 2.2,” *Chart. Prof. Accountants Canada*, 2019.

[5] N. P. Smart *et al.*, “No Title,” in *Algorithms, Key Size and Parameters Report - 2014*, 2014.

[6] European Payments Council, “Guidelines on Cryptographic Algorithms Usage and Key Management Version 8.0,” *Eur. Payments Counc. Website*, 2018.

[7] E. Barker, L. Chen, A. Roginsky, A. Vassilev, R. Davis, and S. Simon, “SP 800-56B Revision 2: Recommendation for Pair-Wise Key Establishment Using Integer Factorization Cryptography,” *Natl. Inst. Stand. Technol.*, 2019.

[8] E. Barker, “SP 800-57 Part 1 Revision 4: Recommendation for Key Management, Part 1: General,” *Natl. Inst. Stand. Technol.*, 2016.

[9] E. Barker and W. Barker, “SP 800-57 Part 2 Revision 1: Best Practices for Key Management,” *Natl. Inst. Stand. Technol.*, 2019.

[10] E. Barker and Q. Dang, “SP 800-57 Part 3 Revision 1: Application-Specific Key Management Guidance,” *Natl. Inst. Stand. Technol.*, 2015.

[11] E. Barker, M. Smid, D. Branstad, and S. Chokhani, “SP 800-130: A Framework for Designing Cryptographic Key Management Systems,” *Natl. Inst. Stand. Technol.*, 2013.

[12] E. Barker and A. Roginsky, “SP 800-131A Revision 2: Transitioning the Use of Cryptographic Algorithms and Key Lengths,” *Natl. Inst. Stand. Technol.*, 2019.

[13] and A. R. K. Moriarty, B. Kaliski, “PKCS #5: Password-Based Cryptography Specification Version 2.1,” *IETF*, 2017.

[14] K. Aoki *et al.*, “Camellia: A 128-Bit Block Cipher Suitable for Multiple Platforms – Design and Analysis,” in *Selected Areas in Cryptography, SAC 2000*, 2001, pp. 39–56.

[15] T. Shirai, K. Shibutani, T. Akishita, S. Moriai, and T. Iwata, “The 128-bit Blockcipher CLEFIA,” in *Fast Software Encryption: 14th International Workshop, FSE 2007*, 2007, pp. 181–195.

[16] Sony Corporation, “The 128-bit Blockcipher CLEFIA Algorithm Specification,” vol. 1, 2007.

[17] Sony Corporation, “The 128-bit Blockcipher CLEFIA Security and Performance Evaluations,” 2007.

[18] A. Bogdanov *et al.*, “PRESENT: An Ultra-Lightweight Block Cipher,” in *Cryptographic Hardware and Embedded Systems - CHES 2007*, 2007, pp. 450–466.

[19] M. Dworkin, “SP 800-38A, Recommendation for Block Cipher Modes of Operation: Methods and Techniques,” *Natl. Inst. Stand. Technol.*, 2001.

[20] S. Kiyomoto, T. Tanaka, and K. Sakurai, “K2 stream cipher,” in *E-business and Telecommunications*, 2007, pp. 214–226.

[21] M. Boesgaard, M. Vesterager, T. Pedersen, J. Christiansen, and O. Scavenius, “Rabbit: A New High-Performance Stream Cipher,” in *Fast Software Encryption, FSE 2003*, 2003, pp. 307–329.

[22] M. Boesgaard, M. Vesterager, and E. Zenner, “The Rabbit Stream Cipher,” in *New Stream Cipher Designs*, 2008, pp. 69–83.

[23] D. J. Bernstein, “ChaCha, A Variant of Salsa20,” in *Workshop Record of SASC*, 2008.

[24] KDDI Corporation, “Stream Cipher KCipher-2.” [Online]. Available: https://www.cryptrec.go.jp/english/%0Acryptrec_13_spec_cypherlist_files/PDF/21_09spec_e_1.2.pdf.

[25] D. J. Bernstein, “The Salsa20 Family of Stream Ciphers,” in *New Stream Cipher Designs*, 2008, pp. 84–97.

[26] A. Bogdanov, M. Knežević, G. Leander, D. Toz, K. Varici, and I. Verbauwhede, “Spongent: A Lightweight Hash Function,” in *Cryptographic Hardware and Embedded Systems - CHES 2011*, 2011, pp. 312–325.

[27] J. Guo, T. Peyrin, and A. Poschmann, “The PHOTON Family of Lightweight Hash Functions,” in *Advances in Cryptology - CRYPTO 2011*, 2011, pp. 222–239.

[28] Q. Dang, “SP 800-107 Rev. 1: Recommendation for Applications Using Approved Hash Algorithms,” *Natl. Inst. Stand. Technol.*, 2012.

[29] National Institute of Standards and Technology, “SHA-3 standard: Permutation-Based Hash and Extendable-Output Functions,” 2015.

[30] RSA Laboratories, “PKCS#1 v2.2: RSA Cryptography Standard,” *EMC Corp. website*, 2012.

[31] “EESS #1: Implementation Aspects of NTRUEncrypt and NTRUSign,” *Effic. Embed. Secur. Stand.*, 2003.

[32] D. Boneh, R. Rivest, A. Shamir, and L. Adleman, “Twenty Years of Attacks on the RSA Cryptosystem,” *Not. AMS*, 1999.

[33] M. Bellare and P. Rogaway, “Optimal Asymmetric Encryption,” in *Advances in Cryptology - EUROCRYPT ’94*, 1995, pp. 92–111.

[34] D. J. Bernstein, “Curve25519: New Diffie-Hellman Speed Records,” in *Public Key Cryptography - PKC 2006*, 2006, pp. 207–228.

[35] M. Hamburg, “Ed448-Goldilocks, A New Elliptic Curve,” *IACR Cryptol. ePrint Arch.*, 2015.

[36] A. Langley, M. Hamburg, and S. Turner, “Elliptic Curves for Security,” 2016.

[37] A. O. L. Atkin and F. Morain, “Elliptic Curves and Primality Proving,” in *Mathematics of Computation*, 1993, pp. 29–68.

[38] B. Schneier, “So, You Want to be a Cryptographer,” *Crypto-Gram Newsl.*, 1999.

[39] D. Lazar, H. Chen, X. Wang, and N. Zeldovich, “Why Does Cryptographic Software Fail? A Case Study and Open Problems,” in *Asia-Pacific Workshop on Systems, APSys ’14*, 2014.

[40] J.-P. Aumasson, *Serious Cryptography: A Practical Introduction to Modern Encryption*. No Starch Press, 2017.

[41] S. Yilek, E. Rescorla, H. Shacham, B. Enright, and S. Savage, “When Private Keys are Public: Results from the 2008 Debian OpenSSL Vulnerability,” in *Internet Measurement Conference, IMC 2009*, 2009, pp. 15–27.

[42] Z. Durumeric *et al.*, “The Matter of Heartbleed,” in *Internet Measurement Conference, IMC 2014*, 2014, pp. 475–488.

[43] National Institute of Standards and Technology, “FIPS 140-2: Security Requirements for Cryptographic Modules,” *Fed. Inf. Process. Stand.*, 2001.

Acknowledgements

CyberSecurity Malaysia would like to express our appreciation and gratitude to all members in developing the Guideline on the Usage of Recommended AKSA MySEAL Cryptographic Algorithms. Members are as follows:

External Contributors/Reviewers:

Mr. Chris Liaw Man Cheon	Augmented Technology Sdn. Bhd.
Mr. Mohammad Azizi Yacob/	Bank Negara Malaysia
Mr. Sza'zuan Izaham Saat	
Dr. Mohd Anuar Mat Isa	iExploTech
Mr. Lee Kay Win/	MIMOS Berhad
Ms. Moesfa Soeheila Mohamad	
Mr. Hazhar Ismail	MSC Trustgate.com Sdn. Bhd.
Dr. Noorul Halimin Mansol/	Pos Digicert Sdn. Bhd.
Mr. Nazril Mohd Ghani	
Mr. Mohd Fakhri Aris Muhammad Ghazi/	Suruhanjaya Komunikasi dan Multimedia Malaysia
Ms. Nuraffiza Ahmad/	
Ms. Rafina Mimi Muhamad	
Mr. Fazli Azran	TeleAwan Sdn. Bhd.
Prof. Ts. Dr. Miss Laiha Mat Kiah/	Universiti Malaya
Dr. Muhammad Reza Z'aba	
Prof. Ts. Dr. Heng Swee Huay/	Universiti Multimedia
Dr. Chin Ji Jian	
Prof. Dr. Ramlan Mahmod/	Universiti Putra Malaysia
Assoc. Prof. Dr. Muhammad Rezal Kamel Ariffin/	
Dr. Amir Hamzah Abd Ghafar	
Prof. Ts. Dr. Rabiah Ahmad/	Universiti Teknikal Malaysia Melaka
Assoc. Prof. Dr. Nor Azman Abu/	
Dr. Shekh Faisal bin Abdul Latip	
Mr. Mohd Saufy Rohmad	Universiti Teknologi MARA
Assoc. Prof. Ts. Dr. Norziana Jamil	Universiti Tenaga Nasional
Assoc. Prof. Ts. Dr. Yap Wun She	Universiti Tunku Abdul Rahman
Mr. Gan Chin Sam	WannaStation Sdn. Bhd.

Internal Contributors/Reviewers:
All CyberSecurity Malaysia staff that have been involved in the development of this document.

The MySEAL Focus Group members involved in recommending the AKSA list in 2018 are given below.

Dr. Noor Hayati Hashim	Agensi Keselamatan Siber Negara
Dr. Poh Geong Sen/ Ms. Moesfa Soeheila Mohamad	MIMOS Berhad
Prof. Ts. Dr. Miss Laiha Mat Kiah/ Dr. Muhammad Reza Z'aba	Universiti Malaya
Prof. Ts. Dr. Heng Swee Huay/ Dr. Chin Ji Jian/ Dr. Tan Syh Yuan	Universiti Multimedia
Prof. Dr. Ramlan Mahmod/ Assoc. Prof. Dr. Muhammad Rezal Kamel Ariffin	Universiti Putra Malaysia
Prof. Dr. Kamaruzzaman Seman	Universiti Sains Islam Malaysia
Assoc. Prof. Dr. Nor Azman Abu/ Dr. Shekh Faisal bin Abdul Latip	Universiti Teknikal Malaysia Melaka
Assoc. Prof. Ts. Dr. Norziana Jamil	Universiti Tenaga Nasional
Assoc. Prof. Ts. Dr. Yap Wun She	Universiti Tunku Abdul Rahman

As of the publication date of this guideline document, Dr. Poh Geong Sen and Dr. Tan Syh Yuan are no longer members of the MySEAL Focus Group and are no longer affiliated with MIMOS Berhad and Universiti Multimedia.