



# INTERNATIONAL CRYPTOGRAPHIC MODULE CONFERENCE 2017

May 16-19 | Westin Arlington Gateway | Washington, DC

## Preparing Today for Key Management in a Post-Quantum Computing World

Tony Cox

[tony.cox@cryptsoft.com](mailto:tony.cox@cryptsoft.com)

Prof Dr Tanja Lange

[tanja@hyperelliptic.org](mailto:tanja@hyperelliptic.org)

Q13a 17-May-2017 14:40 (Ballroom A)

# Introductions

## Tony Cox

- ▶ VP Partners, Alliances & Sales - Cryptsoft
- ▶ Chair - OASIS KMIP Technical Committee
- ▶ Co-Editor - KMIP Specification v1.3, v1.4 & v2.0
- ▶ Chair - OASIS PKCS11 Technical Committee

## Prof Dr Tanja Lange

- ▶ Professor at Technische Universiteit Eindhoven (The Netherlands)
- ▶ Expert on curve-based cryptography
- ▶ Early proponent of post-quantum cryptography.
- ▶ 3 steering committees, including PQCrypto workshop.
- ▶ Coordinator of EU-H2020 project PQCrypto – Post-quantum cryptography for long-term security  
<https://pqcrypto.eu.org>



# INTERNATIONAL CRYPTOGRAPHIC MODULE CONFERENCE 2017

May 16-19 | Westin Arlington Gateway | Washington, DC

## Introduction

Why is this topic important?

# Not just crypto...

- ▶ Plenty of discussion on QC and PQC at this event and in the wider industry
- ▶ In a key management context we need to consider:
  - ▶ Nature of PQC threats to managed encryption keys
  - ▶ Responses to coming threats
  - ▶ Ongoing ability to change and adapt

# Keys in use for decades

- ▶ How do we put together a framework to deal with both quantified and as-yet unquantified threats and their impact on:
  - ▶ Encrypted data
  - ▶ Signatures and legal contexts
  - ▶ Authentication systems



# INTERNATIONAL CRYPTOGRAPHIC MODULE CONFERENCE 2017

May 16-19 | Westin Arlington Gateway | Washington, DC

## Threats and recommendations

Post Quantum Crypto overview







## Algorithms for Quantum Computation: Discrete Logarithms and Factoring

Peter W. Shor  
AT&T Bell Labs  
Room 2D-149  
600 Mountain Ave.  
Murray Hill, NJ 07974, USA

### Abstract

*A computer is generally considered to be a universal computational device; i.e., it is believed able to simulate any physical computational device with a cost in computation time of at most a polynomial factor. It is not clear whether this is still true when quantum mechanics is taken into consideration. Several researchers, starting with David Deutsch, have developed models for quantum mechanical computers and have investigated their computational properties. This paper gives Las Vegas algorithms for finding discrete logarithms and factoring integers on a quantum computer that take a number of steps which is polynomial in the input size, e.g., the number of digits of the integer to be factored. These two problems are generally considered hard on a classical computer and have been used as the basis of several proposed cryptosystems. (We thus give the first examples of quantum cryptanalysis.)*

[1, 2]. Although he did not ask whether quantum mechanics conferred extra power to computation, he did show that a Turing machine could be simulated by the reversible unitary evolution of a quantum process, which is a necessary prerequisite for quantum computation. Deutsch [9, 10] was the first to give an explicit model of quantum computation. He defined both quantum Turing machines and quantum circuits and investigated some of their properties.

The next part of this paper discusses how quantum computation relates to classical complexity classes. We will thus first give a brief intuitive discussion of complexity classes for those readers who do not have this background. There are generally two resources which limit the ability of computers to solve large problems: time and space (i.e., memory). The field of analysis of algorithms considers the asymptotic demands that algorithms make for these resources as a function of the problem size. Theoretical computer scientists generally classify algorithms as efficient when the number of steps of the algorithms grows as a polynomial in the size of the input. The class of prob-



# Threats and recommendations

- ▶ Shor's algorithm solves in polynomial time:
  - ▶ Integer factorization. RSA is dead.
  - ▶ The discrete-logarithm problem in finite fields. DSA is dead.
  - ▶ The discrete-logarithm problem on elliptic curves. ECDSA is dead.
- ▶ This breaks all current public-key cryptography on the Internet!
- ▶ Massive research effort. Tons of progress summarized in, e.g., [https://en.wikipedia.org/wiki/Timeline\\_of\\_quantum\\_computing](https://en.wikipedia.org/wiki/Timeline_of_quantum_computing).
- ▶ Mark Ketchen, IBM Research, 2012, on quantum computing: "Were actually doing things that are making us think like, 'hey this isn't 50 years off, this is maybe just 10 years off, or 15 years off.' It's within reach."
- ▶ Also, Grover's algorithm speeds up brute-force searches.
  - ▶ Example: Only  $2^{64}$  quantum operations to break AES-128;
  - ▶ Example: Only  $2^{128}$  quantum operations to break AES-256.

# Threats and recommendations

Name	function	pre-quantum security level	post-quantum security level
<b>Symmetric cryptography</b>			
AES-128 [8]	symmetric encryption	128	64 (Grover)
AES-256 [8]	symmetric encryption	256	128 (Grover)
Salsa20 [9]	symmetric encryption	256	128 (Grover)
GMAC [10]	MAC	128	128 (no impact)
Poly1305 [11]	MAC	128	128 (no impact)
SHA-256 [12]	hash function	256	128 (Grover)
SHA3-256 [13]	hash function	256	128 (Grover)
<b>Public-key cryptography</b>			
RSA-3072 [1]	encryption	128	broken (Shor)
RSA-3072 [1]	signature	128	broken (Shor)
DH-3072 [14]	key exchange	128	broken (Shor)
DSA-3072 [15, 16]	signature	128	broken (Shor)
256-bit ECDH [4, 5, 17]	key exchange	128	broken (Shor)
256-bit ECDSA [18, 19]	signature	128	broken (Shor)

# Physical Security

## A return to the dark ages?

- ▶ Locked briefcases, quantum key distribution, etc.
- ▶ Horrendously expensive.
- ▶ Not suitable for today's networks and end points.
- ▶ “Provably secure” under highly questionable assumptions.
- ▶ Broken again and again. Much worse track record than normal crypto.
- ▶ Easy to screw up. Easy to backdoor. Hard to audit.
- ▶ Very limited functionality: e.g., no public-key signatures.



# Security advantages of algorithmic cryptography

- ▶ Keep secrets heavily shielded inside authorized computers.
  - ▶ Reduce trust in third parties:
  - ▶ Reduce reliance on closed-source software and hardware.
  - ▶ Increase comprehensiveness of audits and certifications.
  - ▶ Increase comprehensiveness of formal verification.
  - ▶ Design systems to be secure even if keys are public.
  - ▶ Critical example: signed software updates.
- ▶ Understand security as thoroughly as possible:
  - ▶ Publish comprehensive specifications.
  - ▶ Build large research community with clear security goals.
  - ▶ Publicly document attack efforts.
  - ▶ Require systems to convincingly survive many years of analysis.



# Post-Quantum Cryptography

- ▶ Post-quantum crypto is crypto that resists attacks by quantum computers.
- ▶ 2003 Daniel J. Bernstein introduces term Post-quantum cryptography.
- ▶ PQCrypto 2006: International Workshop on Post-Quantum Cryptography.
- ▶ PQCrypto 2016: 22-26 Feb in Fukuoka, Japan, > 200 people



# What to do now?

- ▶ Upgrade now!
  - ▶ Rolling out crypto takes long time.
  - ▶ Every message encrypted with pre-quantum crypto is lost.
  - ▶ Need to be up & running when quantum computers come.
- ▶ Upgrade later!
  - ▶ NIST will receive >100 great submissions, sure better than old crap
  - ▶ Once rolled out, it's hard to change systems.
  - ▶ (That said, easier to choose now than after November).

# What to do now?

- ▶ Recommend very conservative systems now.
  - ▶ Users who care will accept performance issues and gladly update to faster/smaller options later.
  - ▶ Recommend now, standardize later. General roll out later.
- ▶ Make sure to secure update mechanisms for long-lived products
  - ▶ car manufacturers
  - ▶ energy companies
  - ▶ banking industry (ISO/TC68/WG2)
- ▶ Find out now where you rely on crypto; make an inventory.





# INTERNATIONAL CRYPTOGRAPHIC MODULE CONFERENCE 2017

May 16-19 | Westin Arlington Gateway | Washington, DC

## KMIP & PQC


# Industry recommendations





In most cryptographic functions, the key length is an important security parameter. Both academic and private organizations provide recommendations and mathematical formulas to approximate the minimum key size requirement for security. Despite the availability of these publications, choosing an appropriate key size to protect your system from attacks remains a headache as you need to read and understand all these papers.

This web site implements mathematical formulas and summarizes reports from well-known organizations allowing you to quickly evaluate the minimum security requirements for your system. You can also easily compare all these techniques and find the appropriate key length for your desired level of protection. The lengths provided here are designed to resist mathematic attacks; they do not take algorithmic attacks, hardware flaws, etc. into account.



Choose a Method 

Lenstra and Verheul Equations (2000)  
 Lenstra Updated Equations (2004)  
 ECRYPT II Recommendations (2012)  
 NIST Recommendations (2016)  
 ANSSI Recommendations (2014)  
 IAD-NSA CNSA Suite (2016)  
 Network Working Group RFC3766 (2004)  
 BSI Recommendations (2017)


 Compare all Methods
 

© 2017 BlueKrypt - v 30.4 - February 23, 2017  
 Author: Damien Giry  
 Approved by Prof. Jean-Jacques Quisquater  
 Contact: [keylength@bluekrypt.com](mailto:keylength@bluekrypt.com)

I would like to thank Prof. Arjen K. Lenstra for his kind authorization and comments.  
 Surveys of laws and regulations on cryptology: [Crypto Law Survey](#) / [Digital Signature Law Survey](#).

[Privacy Policy \(P3P\)](#) | [Disclaimer / Copyright](#) | [Release Notes](#)

# Industry (NIST) recommendations

NIST is a non-regulatory federal agency within the U.S. Commerce Department's Technology Administration. Recommendations in this report [4] are aimed to be use by Federal agencies and provide key sizes together with algorithms. The first table provides cryptoperiod for 19 types of key uses. A cryptoperiod is the time span during which a specific key is authorized for use by legitimate entities, or the keys for a given system will remain in effect. The second table presents the key length recommendations.

Key Type <i>Move the cursor over a type for description</i>	Originator Usage Period (OUP)	Cryptoperiod Recipient Usage Period
Private Signature Key	1-3 years	-
Public Signature Key	Several years (depends on key size)	
Symmetric Authentication Key	$\leq 2$ years	$\leq \text{OUP} + 3$ years
Private Authentication Key		1-2 years
Public Authentication Key		1-2 years
Symmetric Data Encryption Key	$\leq 2$ years	$\leq \text{OUP} + 3$ years
Symmetric Key Wrapping Key	$\leq 2$ years	$\leq \text{OUP} + 3$ years
Symmetric RBG keys	Determined by design	-
Symmetric Master Key	About 1 year	-
Private Key Transport Key		$\leq 2$ years <sup>(1)</sup>
Public Key Transport Key		1-2 years
Symmetric Key Agreement Key		1-2 years <sup>(2)</sup>
Private Static Key Agreement Key		1-2 years <sup>(3)</sup>
Public Static Key Agreement Key		1-2 years
Private Ephemeral Key Agreement Key		One key agreement transaction
Public Ephemeral Key Agreement Key		One key agreement transaction
Symmetric Authorization Key		$\leq 2$ years
Private Authorization Key		$\leq 2$ years
Public Authorization Key		$\leq 2$ years

In some cases risk factors affect the cryptoperiod selection (see section 5.3.1 in report [4]).

(1) In certain email applications where received messages are stored and decrypted at a later time, the cryptoperiod of the Private Key Transport Key may exceed the cryptoperiod of the Public Key Transport Key.

(2) In certain email applications where received messages are stored and decrypted at a later time, the key's recipient-usage period key may exceed the originator-usage period.

(3) In certain email applications whereby received messages are stored and decrypted at a later time, the cryptoperiod of the Private Static Key Agreement Key may exceed the cryptoperiod of the Public Static Key Agreement Key.

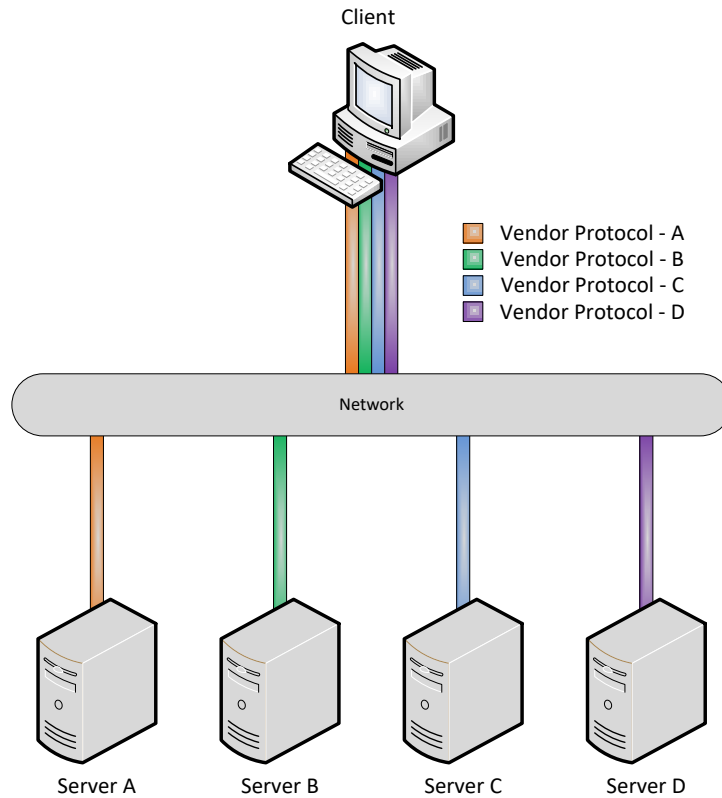
# Approach

- ▶ Be mindful of, but not tied to the specific recommendations. Instead:
  - ▶ Focus on the framework that can enable recommendations to be implemented quickly and easily
  - ▶ Ensure the framework enables sufficient agility to respond to new & different threats
  - ▶ Ensure that the framework will work in a commercial implementation
- ▶ Externalize key management from applications
  - ▶ Use a standardized protocol to deliver interoperability across the enterprise/vendors
  - ▶ Ensure the standardized protocol has wide industry & vendor support

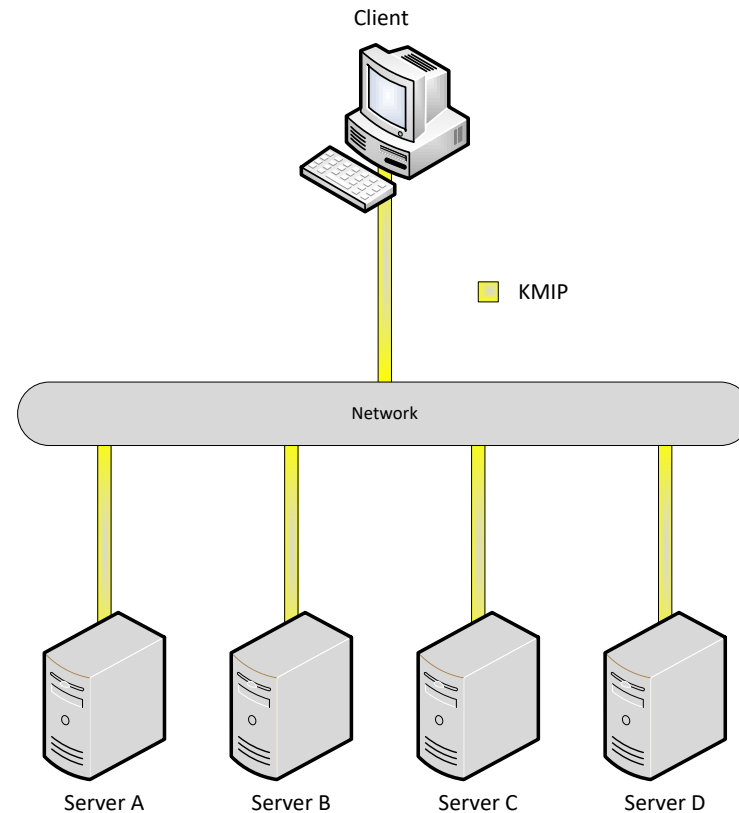
# KMIP Vendors



# KMIP 101

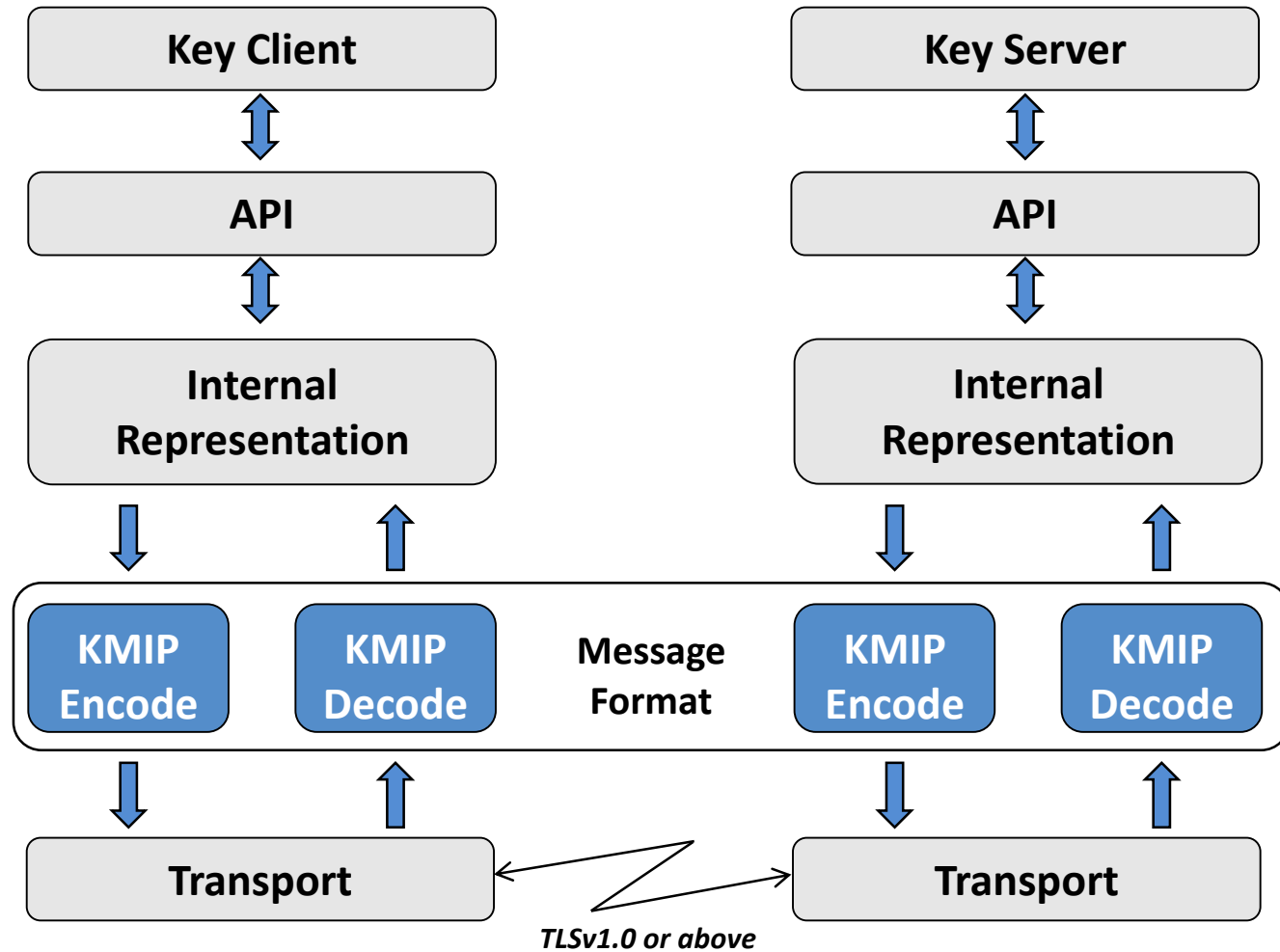


Prior to KMIP each application had to support each vendor protocol



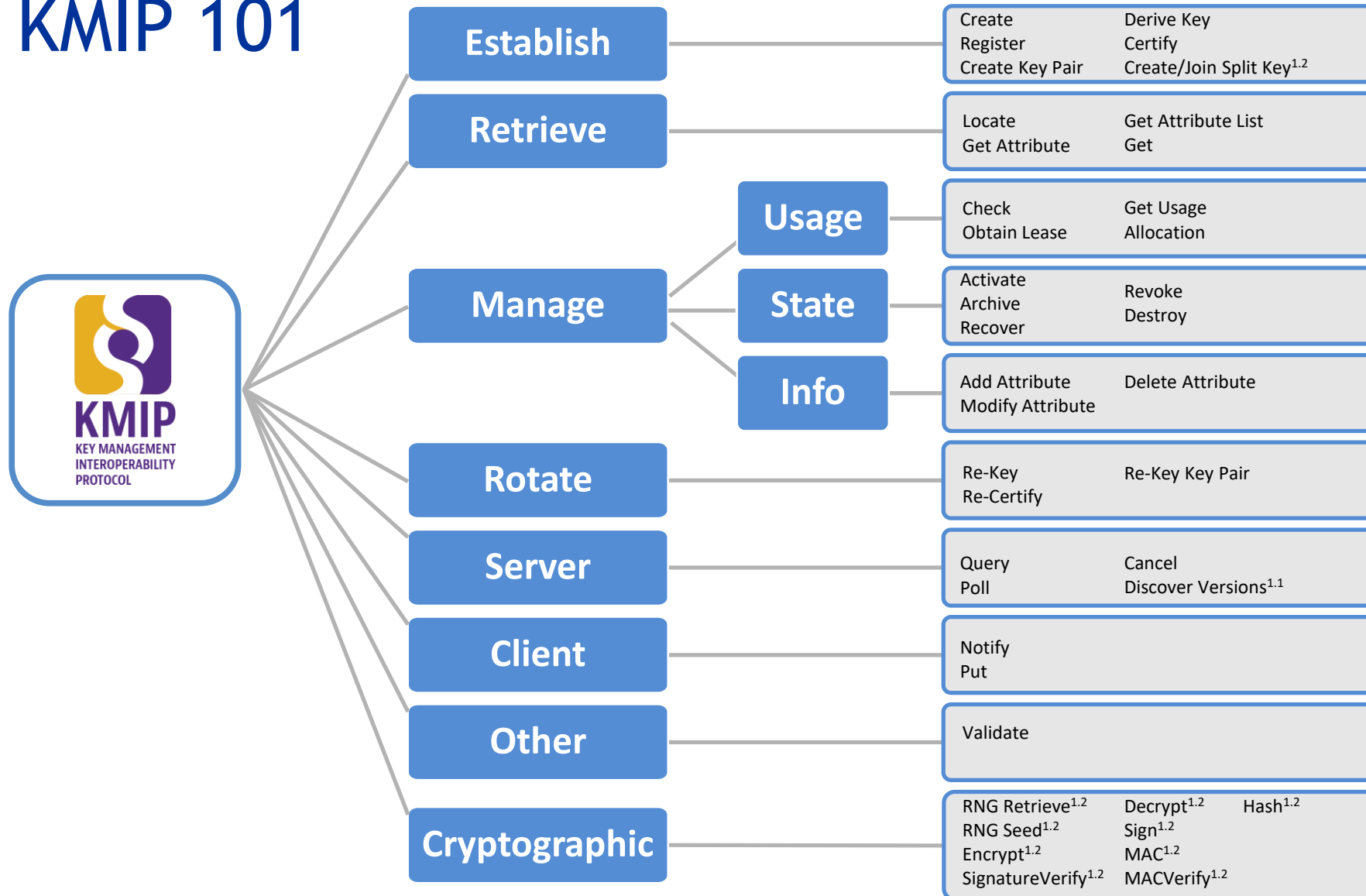
With KMIP each application only requires support for one protocol

# KMIP 101





# KMIP 101





# KMIP 101

## Operations

- |                 |                                    |                                 |                             |                                  |                                   |
|-----------------|------------------------------------|---------------------------------|-----------------------------|----------------------------------|-----------------------------------|
| ▪ Activate      | ▪ Create Key Pair                  | ▪ Encrypt <sup>1,2</sup>        | ▪ Locate                    | ▪ Put                            | ▪ Revoke                          |
| ▪ Add Attribute | ▪ Create Split Key <sup>1,2</sup>  | ▪ Get                           | ▪ MAC <sup>1,2</sup>        | ▪ Register                       | ▪ RNG Retrieve <sup>1,2</sup>     |
| ▪ Archive       | ▪ Decrypt <sup>1,2</sup>           | ▪ Get Attribute List            | ▪ MAC Verify <sup>1,2</sup> | ▪ Register Query                 | ▪ RNG Seed <sup>1,2</sup>         |
| ▪ Cancel        | ▪ Delete Attribute                 | ▪ Get Attributes                | ▪ Modify Attribute          | ▪ Re-certify                     | ▪ Sign <sup>1,2</sup>             |
| ▪ Certify       | ▪ Derive Key                       | ▪ Get Usage Allocation          | ▪ Notify                    | ▪ Recover                        | ▪ Signature Verify <sup>1,2</sup> |
| ▪ Check         | ▪ Destroy                          | ▪ Hash <sup>1,2</sup>           | ▪ Obtain Lease              | ▪ Re-Key                         | ▪ Validate                        |
| ▪ Create        | ▪ Discover Versions <sup>1,1</sup> | ▪ Join Split Key <sup>1,2</sup> | ▪ Poll                      | ▪ Re-key Key Pair <sup>1,1</sup> |                                   |

## Object Types

- |                          |               |                 |
|--------------------------|---------------|-----------------|
| ▪ Certificate            | ▪ Private Key | ▪ Split Key     |
| ▪ Opaque Object          | ▪ Public Key  | ▪ Symmetric Key |
| ▪ PGPKKey <sup>1,2</sup> | ▪ Secret Key  | ▪ Template      |

## States

- |               |                         |
|---------------|-------------------------|
| ▪ Pre Active  | ▪ Compromised           |
| ▪ Active      | ▪ Destroyed             |
| ▪ Deactivated | ▪ Destroyed Compromised |

## Encodings

- TTLV
- HTTPS/TTLV<sup>1,2</sup>
- HTTPS/JSON<sup>1,2</sup>
- HTTPS/XML<sup>1,2</sup>

## Profiles

- |   |  |   |
|---|--|---|
| ▪ Advanced Cryptographic Client & Server <sup>1,2</sup> | ▪ Basic Symmetric Key Foundry Client & Server        | ▪ Storage Array With SED Client & Server  |
| ▪ Advanced Symmetric Key Foundry Client & Server        | ▪ HTTPS, JSON, XML Client & Server                   | ▪ Suite-B MinLOS_128 Client & Server      |
| ▪ Asymmetric Key Lifecycle Client & Server              | ▪ Intermediate Symmetric Key Foundry Client & Server | ▪ Suite-B MinLOS_192 Client & Server      |
| ▪ Baseline Client & Server Basic                        | ▪ Opaque Managed Object Store Client & Server        | ▪ Symmetric Key Lifecycle Client & Server |
| ▪ Baseline Client & Server TLSv1_2                      | ▪ RNG Cryptographic Client & Server <sup>1,2</sup>   | ▪ Tape Library Client & Server            |
| ▪ Basic Cryptographic Client & Server <sup>1,2</sup>    |  | ▪ Complete Server                         |

# KMIP Deployment Overview

## Disk Arrays, Flash Storage Arrays, NAS Appliances, Storage Operating Systems

- ▶ Vaulting master authentication key
- ▶ Cluster-wide sharing of configuration settings
- ▶ Specific Usage Limits checking (policy)
- ▶ FIPS140-2 external key generation (create, retrieve)
- ▶ Multi-version key support during Rekey
- ▶ Backup and recovery of device specific key sets

## Tape Libraries, Virtual Tape Libraries

- ▶ External key generation (create, retrieve)
- ▶ FIPS140-2 external key generation (create, retrieve)
- ▶ Multi-version key support during Rekey

# KMIP Deployment Overview

## Encrypting Switches, Storage Controllers

- ▶ Vaulting device or port specific encryption keys
- ▶ Cluster-wide sharing of configuration settings
- ▶ Specific Usage Limits checking (policy)

## Encryption Gateways, Virtualisation Managers

- ▶ Vaulting device, port or user specific encryption keys
- ▶ External key generation (create, retrieve)
- ▶ Cluster-wide sharing of configuration settings
- ▶ Specific Usage Limits checking (policy)

# KMIP Deployment Overview

## Compliance Platforms, Information Managers, Enterprise Security

- ▶ Policy Enforcement for Access
- ▶ Policy Enforcement for Operation Usage
- ▶ Audit and Compliance Management
- ▶ Cross-device and cross-application coordination
- ▶ User and device authentication enforcement
- ▶ Multi-tenancy and multi-jurisdictional enforcement

## Endpoint Security

- ▶ Vaulting device, port or user specific encryption keys
- ▶ External key generation (create, retrieve)
- ▶ Cluster-wide sharing of configuration settings
- ▶ Specific Usage Limits checking (policy)

# KMIP Deployment Overview

## Key Managers

- ▶ Key and other Object Vault (store)
- ▶ Key and other Object Creator (generate)
- ▶ Secure Cryptographic Operations (use)
- ▶ Policy Enforcement for Access
- ▶ Policy Enforcement for Operation Usage
- ▶ Audit and Compliance Management
- ▶ Cross-device and cross-application coordination
- ▶ User and device authentication enforcement
- ▶ Multi-tenancy and multi-jurisdictional enforcement

# KMIP Deployment Overview

## Hardware Security Modules (HSM)

- ▶ Key and other Object Vault (store)
- ▶ Policy Enforcement for Access
- ▶ Policy Enforcement for Operation Usage
- ▶ Audit and Compliance Management
- ▶ Multi-tenancy and multi-jurisdictional enforcement
- ▶ Key management / HSM gateways

## Authentication and Identity Management

- ▶ Vaulting user specific information
- ▶ External authentication storage and generation
- ▶ Validation of authentication for multi-protocol support over KMIP



# Deployment considerations

- ▶ The impact of realized PQC threats, must be considered across many deployment scenarios
  - ▶ External key generation
  - ▶ Greater key volumes
  - ▶ Greater key lengths
  - ▶ Additional key metadata (Attributes)
  - ▶ Additional/stronger authentication requirements
  - ▶ Differing jurisdictional requirements

# Approach

- ▶ Client side vs Server side
- ▶ Clients
  - ▶ Far greater numbers
  - ▶ Longer technology turnover rate
  - ▶ In place longer
- ▶ Servers
  - ▶ Fewer in number
  - ▶ Faster refresh rate
  - ▶ Greater focus on security status
- ▶ **Plan = Focus on the clients first!!!**

# Objective

- ▶ In designing the new framework items, three objectives guided the proposal:
  - ▶ Ensure KMIP specification operations contains sufficient context to enable more agile response to quantum computing threats as they arise
  - ▶ Provide the framework to enable clients to request keys and operations such that the server “decide” what is safe
  - ▶ Provide the ability to notify clients of a need to rekey based on factors other than compromise



# KMIP Specification Changes

- ▶ Specification changes - new attributes
  - ▶ **Protection Period** - The period a given key, encryption, signing or certification is able to remain “safe” for a period. Specified as an interval in seconds (max~135 years).
  - ▶ **Protection Level** - The level of protection required for a given object. Specified as “High” or Low”
  - ▶ **Post-Quantum Crypto** - flag to be set if a given object is required to be safe for the given protection Period and Level in the face of a Quantum Computer attack.

# Profiles in use

- ▶ Profiles outline a mandatory (with some allowed variation) set of conformance requirements.
- ▶ Requirements are usually a subset of specific operations, attributes and other items combined with one or more request/response traces.

# New Profiles Defined

- ▶ **Query** - obtain PQC-relevant information
  - ▶ Illustrates a Client determining the capability of a given Server, in this case specifically requesting the PQC capability the Server supports
- ▶ **Create** - Client requests key creation with no algorithm set - provides Protection Period, Protection Level & Post-Quantum Crypto requirements instead.
  - ▶ Illustrates a Client delegating authority for selecting an appropriate key size and algorithm to the Server.

# PQC Profile

## 2 Post-Quantum Cryptography Profile

The Post-Quantum Cryptography Profile describes a KMIP client interacting with a KMIP server in a manner that should also remain secure long-term against attacks by quantum computers, whilst providing a more flexible set of options for handling known or suspected PQC vulnerabilities.

### 2.1 Authentication Suite

Implementations conformant to this profile SHALL use TLS to negotiate a mutually-authenticated connection.

#### 2.1.1 Protocols

Conformant KMIP clients and servers SHOULD support:

- TLS v1.3 [RFC-PENDING]

Conformant KMIP clients and servers MAY support:

- TLS v1.2 [RFC5246]

Conformant KMIP clients and servers SHALL NOT support:

- Any other TLS or SSL protocol version

#### 2.1.2 Cipher Suites

Conformant KMIP servers SHALL support the following cipher suites for TLSv1.3 if TLSv1.3 is supported:

- TLS13-CHACHA20-POLY1305-SHA256
- TLS13-AES-256-GCM-SHA384

Conformant KMIP servers SHALL support the following cipher suites for TLSv1.2 if TLSv1.2 is supported:

- TLS\_ECDHE\_ECDSA\_WITH\_CHACHA20\_POLY1305\_SHA256
- TLS\_ECDHE\_ECDSA\_WITH\_AES\_256\_GCM\_SHA384

#### 2.1.3 Client Authenticity

Conformant KMIP servers and clients SHALL handle client authenticity in accordance with section 3.1.3 of the Basic Authentication Suite [KMIP-PROF].





# PQC Profile

- 85 **2.3 Post-Quantum Cryptography - Server**
- 86 KMIP servers conformant to this profile under [KMIP-SPEC]:
- 87 1. SHALL conform to the *Baseline Server* of [KMIP-PROF]
- 88 2. SHALL support the following *Objects* [KMIP-SPEC]
- 89 a. *Certificate* [KMIP-SPEC]
- 90 b. *Symmetric Key* [KMIP-SPEC]
- 91 c. *Public Key* [KMIP-SPEC]
- 92 d. *Private Key* [KMIP-SPEC]
- 93 3. SHALL support the following *Attributes* [KMIP-SPEC]
- 94 a. *Cryptographic Algorithm* [KMIP-SPEC]
- 95 b. *Cryptographic Length* [KMIP-SPEC] value:
- 96 c. *Protection Period* [KMIP-SPEC] [INTERVAL]
- 97 d. *Protection Type* [KMIP-SPEC] (LEVEL - LOW, MEDIUM, HIGH)
- 98 e. *Post-Quantum Crypto* [KMIP-SPEC] (Boolean)
- 99 4. SHALL support the following *Client-to-Server Operations* [KMIP-SPEC]:
- 100 a. *Create* [KMIP-SPEC]
- 101 b. *Create Key Pair* [KMIP-SPEC]
- 102 c. *Register* [KMIP-SPEC]
- 103 d. *Re-key* [KMIP-SPEC]
- 104 e. *Re-key Key Pair* [KMIP-SPEC]
- 105 f. *Certify* [KMIP-SPEC]
- 106 g. *Re-Certify* [KMIP-SPEC]
- 107 h. *Encrypt* [KMIP-SPEC]
- 108 i. *Decrypt* [KMIP-SPEC]
- 109 j. *Sign* [KMIP-SPEC]
- 110 k. *SignatureVerify* [KMIP-SPEC]
- 111 5. SHALL support the following *Server-to-Client Operations* [KMIP-SPEC]:

# PQC Profile

Encryption	SHOULD ChaCha20 (with 256-bit key) MAY AES-256
Digital Signature	SHOULD SPHINCS-256 (stateless) SHOULD XMSS (statefull)
Key Exchange	SHALL McEliece (with binary Goppa codes using length $n = 6960$ , dimension $k = 5413$ and adding $t = 119$ errors).
Encryption with Authentication	SHOULD ChaCha20Poly1305 (with 256-bit key) MAY AES-256 (with 96 bit nonce in GCM)
Hashes	SHOULD SHA3-384 or SHA3-512 MAY SHA-384 or SHA-512



# INTERNATIONAL CRYPTOGRAPHIC MODULE CONFERENCE 2017

May 16-19 | Westin Arlington Gateway | Washington, DC

## Securing KMS for QC attacks

# Code-based Crypto

- ▶ McEliece with binary Goppa codes:
- ▶ length  $n = 6960$ , dimension  $k = 5413$ ,  $t = 119$  errors.
- ▶ Key size: 1MB
- ▶ What does this mean?

# Error correction

- ▶ Digital media is exposed to memory corruption.
- ▶ Many systems check whether data was corrupted in transit:
  - ▶ ISBN numbers have check digit to detect corruption.
  - ▶ ECC RAM detects up to two errors and can correct one error. 64 bits are stored as 72 bits: extra 8 bits for checks and recovery.
- ▶ In general,  $k$  bits of data get stored in  $n$  bits, adding some redundancy.
- ▶ If no error occurred, these  $n$  bits satisfy  $n - k$  parity check equations; else can correct errors from the error pattern.
- ▶ Good codes can correct many errors without blowing up storage too much; offer guarantee to correct  $t$  errors (often can correct or at least detect more).
- ▶ To represent these check equations we need a matrix.



RESEARCH REPORT

NO. 100

RESEARCH REPORT

NO. 100

RESEARCH REPORT

RESEARCH REPORT





# Hamming code

Parity check matrix ( $n = 7, k = 4$ ):

$$H = \begin{pmatrix} 1 & 0 & 0 & 1 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 & 1 \end{pmatrix}$$

An error-free string of 7 bits  $\mathbf{b} = (b_0, b_1, b_2, b_3, b_4, b_5, b_6)$  satisfies these three equations:

$$\begin{array}{rcccccl} b_0 & & +b_3 & +b_4 & & +b_6 & = & 0 \\ & b_1 & +b_3 & & +b_5 & +b_6 & = & 0 \\ & & b_2 & +b_4 & +b_5 & +b_6 & = & 0 \end{array}$$

If one error occurred at least one of these equations will not hold.  
Failure pattern uniquely identifies the error location,  
e.g., 1, 0, 1





# Hamming code

Parity check matrix ( $n = 7, k = 4$ ):

$$H = \begin{pmatrix} 1 & 0 & 0 & 1 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 & 1 \end{pmatrix}$$

An error-free string of 7 bits  $\mathbf{b} = (b_0, b_1, b_2, b_3, b_4, b_5, b_6)$  satisfies these three equations:

$$\begin{array}{rcccccl} b_0 & & +b_3 & +b_4 & & +b_6 & = & 0 \\ & b_1 & +b_3 & & +b_5 & +b_6 & = & 0 \\ & & b_2 & +b_4 & +b_5 & +b_6 & = & 0 \end{array}$$

If one error occurred at least one of these equations will not hold.  
Failure pattern uniquely identifies the error location,  
e.g., 1, 0, 1 means  $b_4$  flipped.  
In math notation, the failure pattern is  $H \cdot \mathbf{b}$ .

# Code-based crypto

- ▶ Proposed 1978 by McEliece, this version from 1985 Niederreiter.
- ▶ Many special constructions discovered in 65 years of coding theory:
  - ▶ Large matrix  $H$ .
  - ▶ Fast decoding algorithm to find  $e$  given  $s = H \cdot (c + e)$ , whenever  $e$  doesn't have too many bits set.
- ▶ Given large  $H$ , usually very hard to find fast decoding algorithm.
  - ▶ Use this difference in complexities for encryption.
  - ▶ Public key: random looking matrix; secret key: efficient decoder.
- ▶ Length  $n = 6960$ , dimension  $k = 5413$ ,  $t = 119$  errors means:
  - ▶  $H$  has 6960 columns and 1547 rows;
  - ▶ Attacker is given  $s$ , knows it comes from a length-6960 string with 119 one-bits.

# Further Resources

- ▶ Summer school on post-quantum crypto
  - ▶ Eindhoven, 19-23 June 2017 - <https://2017.pqcrypto.org/school/index.html>
- ▶ Executive school on post-quantum crypto
  - ▶ Eindhoven, 22-23 June 2017 - <https://2017.pqcrypto.org/exec/index.html>
- ▶ PQCrypto 2017
  - ▶ Utrecht, 26-28 June 2017 - <https://2017.pqcrypto.org/conference/index.html>
- ▶ Post-quantum survey site - <https://pqcrypto.org>
- ▶ PQCrypto EU project - <https://pqcrypto.eu.org>

