Public Key Infrastructure: Transition from Classical to Quantum Paradigm

Goutam Paul

Indian Statistical Institute
Kolkata

September 4, 2025

Outline

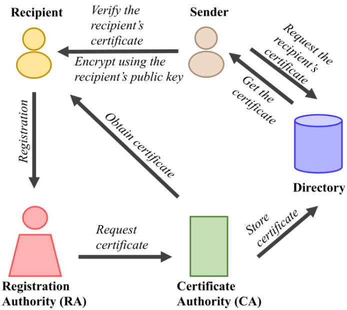
1. Classical PKI Architecture
2. Problems in Classical PKI
3. Shift to Quantum: PQC
4. PQC Standards
5. PKI using PQC
6. Introduction to Hybrid PKI
7. Other Issues in PQC Migration

Classical PKI Architecture

PKI Components and Operation

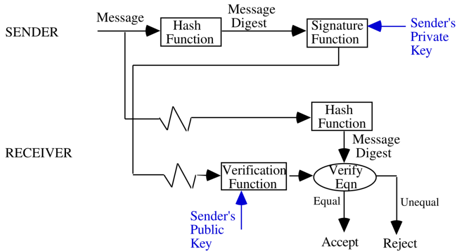
- PKI Offers a solution for managing encryption and secure authentication
- This is done by the creation and management of certificates and public keys.
- It aims to ensure that anyone using an open network can be clearly identified.

PKI Components and Operation



- **RA:** Responsible for confirming the identity of a certificate requester.
- **CA:** Issues the digital certificate once the identity is confirmed by the RA. The CA manages encryption and secure authentication.
- **Directory/Repository:** Stores certificates for retrieval by both the CA and the sender.

Digital Signatures in PKI



Sender:

- **Input:** Message M, Sender's private key SK
- Compute the digest: $h \leftarrow Hash(M)$
- Generate signature: $\sigma \leftarrow Sign(SK, h)$
- Send (M, σ) to the Receiver

Receiver:

- **Input:** Received (M, σ), Sender's public key PK
- Compute the digest: $h' \leftarrow Hash(M)$
- Verify the signature

Role of Hash Function in PKI

- **Efficiency & Fixed-Length Output**
  - Signing entire documents with public key crypto is slow.
  - Hashing reduces any message to a short digest (e.g., 160-bit SHA).
  - Signature is computed only on the digest $\Rightarrow$ efficient.
- **Message Integrity**
  - Digest = unique fingerprint of message.
  - Tiny change in input $\Rightarrow$ very different digest.
  - Receiver recomputes hash and verifies against signed digest.

(continued...)

Role of Hash Function in PKI

- **Authentication & Non-Repudiation**
  - Digest is signed with sender's private key.
  - Verification with public key confirms sender identity.
  - Prevents denial of sending (non-repudiation).
- **Role in PKI & Certificates**
  - CAs sign certificate hashes to bind identity $\leftrightarrow$ public key.
  - X.509 format includes signature fields.
  - Modern/PQC signature schemes (ML-DSA, SLH-DSA) rely heavily on hashing.

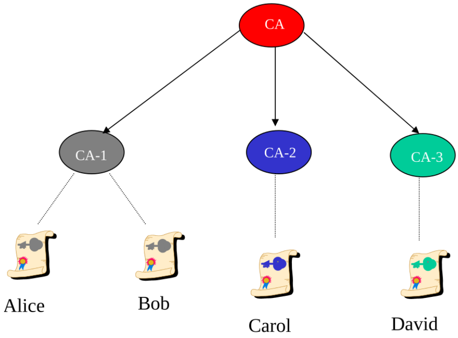
PKI Architectures

- It defines how Certificate Authorities (CAs) and users / Other CAs are structured to establish and manage trust.
- This also known as **trust models**.
- The choice depends on organizational needs, processes, and scalability.

PKI Architectures

- It defines how Certificate Authorities (CAs) and users / Other CAs are structured to establish and manage trust.
- This also known as **trust models**.
- The choice depends on organizational needs, processes, and scalability.
- **Examples:**
  - Hierarchical PKI
  - Mesh PKI
  - Bridge CAs

Hierarchical PKI Architecture (Tree Model)



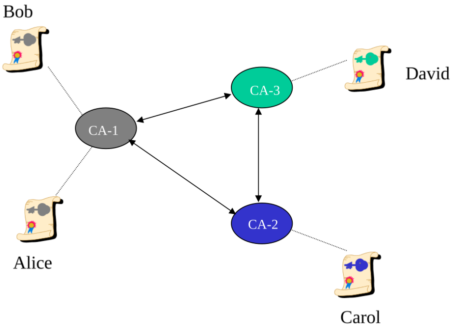
- A **Hierarchical PKI** arranges CAs in superior-subordinate relationships.

(continued...)

Hierarchical PKI Architecture (Tree Model)

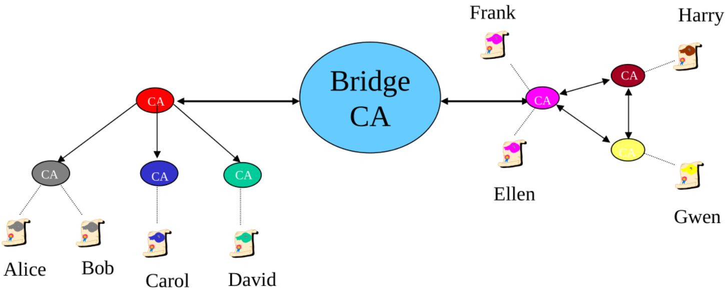
- **Root CA** (at top) is self-signed and acts as **trust anchor**.
- **Intermediate CAs (ICAs)** are issued certificates by the Root CA.
- Users (example, Alice) trust the root CA.
- This architecture offers a high level of control at all levels.
- It establishes trust in a public key's genuineness by a predetermined arrangement of certificates.

Mesh PKI Architecture (Cross-Certified CAs)



- CAs have peer-to-peer relationships.
- CAs cross-certify each other. Example, if Carol (certified by CA-2) needs to verify a certificate from David (certified by CA-3), she can do that, as CA-2 and CA-3 have cross-certified,
- Users trust the CA that issued their own certificate. Example, Carol can trust CA-2's certificate.

Bridge CAs for Interoperability



- A Bridge CA is designed to unify many PKIs into a single, interconnected PKI.
- They facilitate interoperability between different enterprise PKIs, regardless of their internal architecture (hierarchical or mesh).
- A Bridge CA establishes peer-to-peer relationships with various enterprise PKIs.

Problems in Classical PKI

Problems in Classical PKI

- **System Complexity:**
  - PKI involves policies, roles, procedures, hardware, and software.
  - This complexity makes deployment, interoperability, and management difficult.
- **Certificates as Weak Points:**
  - Certificates need to be issued, checked, and revoked properly.
  - If a certificate is wrong or hacked, the whole trust system breaks.
- **Reliance on Classical Algorithms:**
  - Security rests on hardness of factoring and discrete log problems.
  - These assumptions break under quantum computing.

(continued...)

Problems in Classical PKI

- **Classical Algorithms Broken in PKIs:**

Year	PKI	Algorithm	PKI Usage	Bits broken	What happened
1999	Early Web PKI	RSA-512	Certificate Encryption / Signatures	512 (modulus)	RSA-155 (512-bit) factored
2008	Web PKI (TLS/SSL)	MD5	Certificate Signatures	128 (hash)	Chosen-prefix collision found
2012	Microsoft Code-Signing PKI	MD5	Code-Signing Certificates	128 (hash)	"Flame" malware forged intermediate CA certificate
2017	Web PKI	SHA-1	Certificate Signatures	160 (hash)	First public practical collision found ("SHAttered")
2020	Web PKI	SHA-1	Certificate Signatures / Code-Signing	160 (hash)	Practical chosen-prefix collisions found

Problems in Classical PKI

- **Classical Algorithms Broken in PKIs:**

Year	PKI	Algorithm	PKI Usage	Bits broken	What happened
1999	Early Web PKI	RSA-512	Certificate Encryption / Signatures	512 (modulus)	RSA-155 (512-bit) factored
2008	Web PKI (TLS/SSL)	MD5	Certificate Signatures	128 (hash)	Chosen-prefix collision found
2012	Microsoft Code-Signing PKI	MD5	Code-Signing Certificates	128 (hash)	"Flame" malware forged intermediate CA certificate
2017	Web PKI	SHA-1	Certificate Signatures	160 (hash)	First public practical collision found ("SHAttered")
2020	Web PKI	SHA-1	Certificate Signatures / Code-Signing	160 (hash)	Practical chosen-prefix collisions found

- **Quantum threats:** Details next slide...

The Looming Quantum Threat to PKI

- Classical PKI cryptography (RSA, DH, ECC) will become obsolete with quantum computers.
- Shor's algorithm can efficiently break these public-key schemes.
- Quantum computers capable of breaking current algorithms are predicted to arrive within 5-15 years.
- This creates the "Harvest Now, Decrypt Later" (HNDL) threat, where encrypted data is stored now for future quantum decryption.
- An urgent transition to Post-Quantum Cryptography (PQC) is critical for future data security.

Centralization and Trust Model Flaws

- **Centralized Architecture:** Traditional PKI operates as a centralized system where the Certificate Authority (CA) is the single trusted party responsible for managing digital certificates.
- **Single Point of Failure:** This centralized design creates a single point of failure, making the CA a potential bottleneck.
- **Reliance on Absolute Trust:** PKI's security depends on unquestioned trust in CAs, which can be misplaced or exploited due to improper certificate issuance, leading to security breaches.
- **Scalability Challenges:** Single CA architectures often struggle with scalability, making large-scale client management and system interoperability difficult.

General Security Weaknesses and Operational Challenges

- **Implementation Errors:** PKI's complex code bases are prone to subtle implementation errors, compromising security and leading to incorrect SSL certificate issuance.
- **Lack of Cryptographic Agility:** Many PKI systems lack crypto-agility, hindering their ability to adapt to new threats and transition to Post-Quantum Cryptography (PQC).
- **Interoperability Issues:** Varying vendor implementations of PKI standards lead to interoperability problems across different deployments.

Shift to Quantum: PQC

Introduction to Post-Quantum Cryptography (PQC)

Definition

Post-Quantum Cryptography (PQC) refers to cryptographic algorithms and protocols designed to be secure against attacks using quantum computers.

Introduction to Post-Quantum Cryptography (PQC)

Definition

Post-Quantum Cryptography (PQC) refers to cryptographic algorithms and protocols designed to be secure against attacks using quantum computers.

Mathematical Foundations

Security is based on fundamentally different hard computational problems that are resistant to quantum attacks, such as:

- **Lattice-based:** Given a lattice basis and noisy linear equations, find the secret vector or the shortest nonzero vector in the lattice.
- **Hash-based:** Given a secure hash function, construct one-time or few-time signatures. The hard problem is finding collisions or preimages, which are infeasible if the hash is strong.

(continued...)

Introduction to Post-Quantum Cryptography (PQC)

Mathematical Foundations

- **Code-based:** Given a random linear code and a noisy codeword, recover the original message.
- **Multivariate:** Given a set of quadratic polynomial equations over a finite field, find the variable assignments (solutions).

Family	Problem	Examples	Hardness Status
Lattice-based	SVP, CVP, LWE	Kyber, Dilithium, FrodoKEM	SVP/CVP: NP-hard LWE: Believed to be NP-hard.
Hash-based	Collision / Preimage Resistance	SPHINCS+, XMSS, LMS	believed to be NP-hard
Code-based	Syndrome Decoding Problem (SDP)	Classic McEliece	NP-complete
Multivariate	MQ Problem	Rainbow, GeMSS	NP-complete

PQC Migration

Definition

The comprehensive process of transitioning cryptographic systems from traditional, classical algorithms to Post-Quantum Cryptography algorithms

- **Goal:** Ensure data and communications remain secure against quantum computers.

PQC Migration

Definition

The comprehensive process of transitioning cryptographic systems from traditional, classical algorithms to Post-Quantum Cryptography algorithms

- **Goal:** Ensure data and communications remain secure against quantum computers.

Key Aspects

- **Not a Simple Swap:** Migration requires redesign of protocols and infrastructure.
- **Crypto-Agility:** Systems must quickly switch algorithms without major changes.
- **PKI Modernization:**
  - Issue, manage, and revoke PQC or hybrid certificates.
  - X.509 remains the common format.

PQC Migration

Strategies for Transition

- **Hybrid Schemes:** Combine classical + PQC for redundancy.
  - Composite keys/certificates minimize PKI changes.
- **Phased Implementation:**
  - Threat assessment of crypto assets.
  - Pilot testing for performance and compatibility.
  - Strategic roadmap for gradual migration.

Challenges and Considerations

- Algorithm selection (Kyber, Dilithium, Falcon, SPHINCS+, NTRU, BIKE, McEliece).
- Trade-offs: key sizes, signature sizes, performance.
- Legacy system integration with limited crypto-agility.
- Larger keys/signatures → higher storage and computational demands.
- Extensive compatibility testing required.

PQC Standards

NIST PQC Standardization Process

- NIST launched PQC project to standardize quantum-safe algorithms.
- **Goal:** Replace classical crypto for authentication, communication, and data protection.
- **Round 1 (2017–2019):** 26 out of 69 submission selected.
- **Round 2 (2019):** 15 out of 26 candidates chosen.
- **Round 3 (2020–2022):** 4 algorithms are standardized: Kyber (FIPS 203), Dilithium (FIPS 204), FALCON (FIPS 206), SPHINCS+ (FIPS 205).
- **Round 4:**
  - 4 additional KEMs kept under study.
  - KEMs: BIKE, McEliece, HQC, SIKE.
  - SIKE was broken. Hence dropped in 2022.
  - HQC is Selected in Mar 2025.

NIST Standardization Efforts for PQC

Category	Algorithm	Standardization
Key Encapsulation Mechanism (KEM)	CRYSTALS-Kyber	FIPS 203 / ML-KEM
Digital Signatures	CRYSTALS-Dilithium	FIPS 204 / ML-DSA
Digital Signatures	FALCON	FIPS 206 / FN-DSA
Digital Signatures	SPHINCS+	FIPS 205 / SLH-DSA

Table: First Standardized PQC Algorithms (Announced 2022/2024)

NIST Standardization Efforts for PQC

Original Algorithm	NIST Standardized Name	FIPS No.	Input Type	Input Block Length (Bits)	Nature of Output	Ciphertext / Output Size (bits)	Public Key Size (bits)	Private Key Size (bits)	Structure	NIST Security Levels	Typical Applications
CRYSTALS-Kyber	ML-KEM	FIPS 203	Public key size, Random seed (for m)	256	Keys, Ciphertext	6144 to 12,544	6,400 to 12,800	13,000 to 26,000	Lattice, Module-LWE	Levels 1, 3, 5	TLS handshake, VPNs, messaging
CRYSTALS-Dilithium	ML-DSA	FIPS 204	Message (arbitrary length), Secret key, Public key	256	Signature	19,200 to 36,800	10,400 to 20,800	20,000 to 38,000	Lattice, Module-LWE / MSIS	Levels 2, 3, 5	Code signing, documents, certificates
SPHINCS+	SLH-DSA	FIPS 205	Message (arbitrary length), Secret key, Public key	256	Signature	64,000 to 2,40,000	256	512	Stateless hash-based (hypertree + FORS + WOTS+)	Levels 1, 3, 5	Conservative fallback signatures
FALCON	FN-DSA (draft pending)	(FIPS in progress)	Message (arbitrary length), Secret key, Public key	512	Signature	5328 to 10,240	7,200 to 10,400	14,400 to 20,000	Lattice, NTRU lattices (GPV + FFT sampling)	Levels 1, 5	Compact digital signatures for IoT/ embedded

NIST PQC Security Levels

- NIST defines security levels to measure resistance against known attacks.
- Levels are benchmarked against breaking AES and SHA.
- This helps organizations select suitable PQC algorithms.

Level	Equivalent Symmetric Security	Approx. RSA / DH Security	Resistance (Work Factor)
1	AES-128	3072-bit RSA / DH	$\geq 2^{128}$
2	AES-192	7680-bit RSA / DH	$\geq 2^{192}$
3	AES-192	7680-bit RSA / DH	$\geq 2^{192}$
4	AES-256	15360-bit RSA / DH	$\geq 2^{256}$
5	AES-256	15360-bit RSA / DH	$\geq 2^{256}$

Challenges and Strategies for PKI systems

Key Challenges in PQC Transition

- Updating old systems and libraries is difficult.
- Larger keys, ciphertexts, and signatures increase memory, computation, and network load.
- Many organizations underestimate quantum threats (e.g., "store now, decrypt later").

Challenges and Strategies for PKI systems

Key Challenges in PQC Transition

- Updating old systems and libraries is difficult.
- Larger keys, ciphertexts, and signatures increase memory, computation, and network load.
- Many organizations underestimate quantum threats (e.g., "store now, decrypt later").

Strategies for PKI systems → quantum-safe:

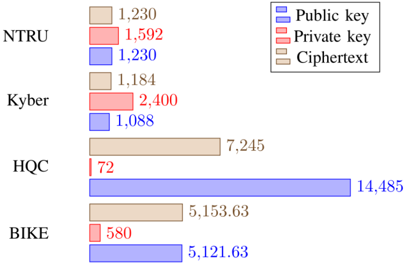
- **Complete Migration:** Replace entire PKI with quantum-safe system.
- **Transitional Migration:** Run classical and PQC PKI in parallel.
- **Hybrid Backwards Compatible:** Support old algorithms while adding hybrid certificates.
- **Hybrid Certificates:** Combine classical and PQC algorithms (e.g., RSA/ECDSA with Kyber/Dilithium).

Key Recommendations for PQC Readiness

- **Evaluate PQC Algorithms:** Choose suitable algorithms based on security and performance.
- **Pilot Testing:** Run proof-of-concept trials to check compatibility and performance.
- **Adopt TLS 1.3:** Leverage TLS 1.3 for hybrid PQC support and efficient exchanges.
- **Automation:** Automate certificate and key lifecycle management for resilience.

PQC Key and Signature Size Considerations

- PQC algorithms generally have much larger key and signature/ciphertext sizes than classical schemes. Example: Classic McEliece.
- **Implications:**
  - Greater memory/storage needs.
  - Higher computational cost.
  - Effects on network protocols: slower handshakes, more energy use, message fragmentation.



PKI using PQC

What is Hybrid PKI?

Definition

It refers to a modernized PKI approach that integrates both traditional (classical) cryptographic algorithms and Post-Quantum Cryptography (PQC) algorithms to ensure continued security against emerging quantum computing threats

Migration Bridge

It allows organizations to begin the transition now, rather than waiting for fully PQC-native systems, which might not be sufficiently studied or robust yet

Hybrid Solutions for Transition

- **Benefits:**
  - **Stronger security:** safe if one algorithm holds.
  - **Backward compatibility:** with legacy PKI.
  - **Gradual migration:** to quantum-safe systems.
- **Implementation:**
  - X.509 certificates can embed multiple keys.
  - Composite certs (e.g., MLDSA+RSA/ECDSA) support adoption.
  - TLS 1.3 enables PQC key exchange and signatures.

Hybrid Solutions for Transition

Hybrid Scheme	Classical Algorithm	Classical Function	Classical Key Size (bits)	PQC Algorithm	PQC Function	PQC Key Size (bits)
MLDSA-44 + RSA-2048 / SHA-256	RSA-2048	Digital Signatures	2048	MLDSA-44	Digital Signatures	10,496
MLDSA-65 + RSA-3072	RSA-3072	Digital Signatures	3072	MLDSA-65	Digital Signatures	15,616
MLDSA-44 + ECDSA-P256	ECDSA-P256	Digital Signatures	512	MLDSA-44	Digital Signatures	10,496
RSA-2048 + Kyber-512	RSA-2048	Key Exchange / Encryption	2048	Kyber-512	Key Encapsulation (KEM)	6,400
ECDSA-P256 + NTRU	ECDSA-P256	Digital Signatures	512	NTRU	Key Encapsulation (KEM)	5,600–8,000
RSA + MLDSA (composite)	RSA-2048	Digital Signatures	2048	MLDSA-44 / 65	Digital Signatures	10,496 / 15,616

PQC Implementation Examples

- **OpenSSL:** Actively integrating hybrid and PQC support.
- **Open Quantum Safe (OQS):** Includes ‘liboqs‘ (C library), protocol integrations like OpenSSL.
- **Python Tools:** Libraries like ‘Pycryptodome‘ (RSA) and ‘PQClean‘ (Kyber) enable PQC experiments in keygen, encryption, and decryption.
- **Hardware Security Modules (HSMs):** Offer PQC hardware acceleration and secure key handling, for stateful hash-based signatures.

Kyber + AES Example in Python

```
from pqc.kem import kyber512 as kemkyb
from Crypto.Cipher import AES
from Crypto.Util.Padding import pad, unpad
from Crypto.Random import get_random_bytes
import hashlib

# 1. Keypair generation
pk, sk = kemkyb.keypair()

# 2. Key encapsulation
ss, kem_ct = kemkyb.encap(pk)

# 3. Key de-encapsulation
ss_result = kemkyb.decap(kem_ct, sk)
assert ss_result == ss

# Convert the shared secret to a symmetric key
def derive_key(shared_secret):
# Hash the shared secret to get a 256-bit key for AES
return hashlib.sha256(shared_secret).digest()

# Derive AES key from shared secret
symmetric_key = derive_key(ss)

# Message to encrypt
message = b'hello world'

# Encrypt the message
def encrypt(message, key):
cipher = AES.new(key, AES.MODE_CBC)
ct_bytes = cipher.encrypt(pad(message, AES.block_size))
return cipher, ct_bytes
```

Issues after PQC Migration

PQC Migration Challenges

- Transitioning to post-quantum PKI is more than replacing algorithms. This often requires protocol and infrastructure redesign.
- The migration poses challenges in compatibility, interoperability, and system integration.
- Careful planning, proactive assessment, and significant engineering efforts are essential.
- Legacy systems may require hybrid approaches before full migration.
- Standardization is still evolving, creating uncertainty in long-term adoption.
- Security risks like key reuse or downgrade attacks must be mitigated.

Challenges in Hybrid PKI

Performance Overhead

- Larger key sizes and certificate sizes.
- Increased computational and bandwidth requirements.

Integration Complexity

- Integration with existing PKI infrastructure.
- Interoperability between different PQC algorithms and classical schemes.

Others

- Limited tooling and library support for hybrid certificates.
- Higher costs in deployment, maintenance, and training.

Cryptographic Agility

Definition

Cryptographic agility (crypto-agility) is the ability of a security system to rapidly switch between cryptographic algorithms, cryptographic primitives, and other encryption mechanisms without the rest of the system's infrastructure being significantly affected by these changes.

Necessity and Importance

- **PQC Migration:** Transition from RSA/ECC to PQC requires redesign of protocols and infrastructure.
- **"Harvest Now, Decrypt Later" Prevention:** Update algorithms to protect harvested data from future decryption.
- **Compliance and Standards:** Ensure alignment with emerging PQC standards and regulations.

Crypto-Agility

Key Characteristics

- **Minimal Disruption:** Replace algorithms without system-wide overhauls.
- **Standardized Interfaces:** Allow modular crypto services and reduce duplication/training costs.
- **Dynamic Algorithm Selection:** Enable configuration-based selection instead of hard-coded choices.
- **PQC-Ready PKI:** Modern PKI systems designed to transition easily to quantum-resistant methods.

Practical Implications

- **Resource Demands:** Larger PQC keys and signatures increase bandwidth, storage, and computation.
- **Constrained Environments:** Agility must support Industrial IoT and resource-limited devices.

Challenges in Achieving Crypto-Agility

- **Legacy Systems:** Require major updates to crypto libraries, protocols, and hardware.
- **Complexity of PQC:** Larger keys, stateful algorithms complicate smooth migration.
- **Cryptographic Inventory:** Difficult to maintain a complete view of algorithms used across systems.
- **Coordination:** Requires cooperation among governments, software vendors, and hardware manufacturers.

Hybrid Solutions for Transition

- **Strategy:** Combine classical + PQC algorithms in parallel.
- **Benefits:** Redundancy, backward compatibility, gradual adoption.
- **Implementation:** Hybrid certificates (e.g., X.509) and hybrid key exchange.

Side-Channel Attacks (SCA)

Definition

A **Side-Channel Attack (SCA)** is a type of attack that exploits physical or implementation-specific information leaked during the execution of a cryptographic algorithm, rather than directly breaking the mathematical security of the algorithm.

- **Examples:**
    - Timing information
    - Power consumption patterns
    - Electromagnetic emissions
    - Fault injection effects

Side-Channel Threats in PQC Migration

- **Implementation Vulnerabilities:** PQC algorithms can leak secret information through timing, power, or electromagnetic signals, even if mathematically secure.
- **Algorithm Complexity:** Larger keys and more complex operations increase the attack surface for side-channel attacks.
- **Hybrid Systems Risk:** During migration, classical and PQC systems coexist; a side-channel leak in PQC can compromise overall security.
- **Real-World Implications:** Without proper countermeasures, sensitive keys may be exposed despite quantum-resistant algorithms.

Automation and Key Management

Automation

Automation is the use of systems to handle cryptographic tasks, like key generation, rotation, and revocation without manual intervention.

- **Key Management Challenges:**
  - Coexistence of classical and post-quantum keys.
  - Secure storage and transport of larger PQC keys.
  - Policy enforcement across distributed systems.
- **Scalability:** IoT and cloud deployments will lead to millions of devices, making efficient automated key replacement and update strategies critical.
- **Monitoring:** Continuous auditing and anomaly detection help prevent misuse or compromise of PQC keys.

Conclusion

- PKI is essential for secure communication, enabling encryption, authentication, and digital signatures using RSA, ECC, DSA, and related algorithms.
- Quantum computing threatens classical PKI, making a timely transition to quantum-safe solutions critical.
- PQC standards like CRYSTALS-Kyber and CRYSTALS-Dilithium, FALCON, SPHINCS+ (DSAs) provide quantum-resistant alternatives.
- Challenges in migration include larger keys/signatures, system complexity, and lack of awareness.
- Phased adoption with hybrid schemes, pilot testing, and clear policies can be used.


ngiyabonga
рахмат
danke
謝謝
spas
teşekkür ederim
Баярлалаа
mersi
kia ora
barka
welalin
tack
спасибо
faafetai lava
vinaka
спасибі
blagodaram
dank je
misaotra
matondo
paldies
grazzi
mahalo
tapadh leat
kiitos
dankie
nanni
nandri
dhanyavad
thank you
gracias
хвала
asante
manana
obrigada
tenki
murakoze
bedankt
bayarlalaa
gracie
hvala
maururu
koszonom
dankon
aciu
akun
enkosi
dziękuję
djiere dieuf
tau
mochchakkeram
дякую
mamnun
chokrane
chnorakaloutioun
gratias ago
gracies
sulpay
taiku
go raibh maith agat
sobodi
dėkuji
sagolun
sukriya
kop khun krap
ありがとう
tanemirt
grazie
arigatô
takk
dakujem
trugarez
obrigado
mèsi
didi madloba
najis tuke
kam sah hamnida
terima kasih
rahmet
rahmat
diolch
dhanyavadagalu
shukriya
merce
мерси
তোমাকে ধন্যবাদ
감사합니다
xièxie
ευχαριστώ
merci