

Post-Quantum Cryptography QIC 891

Geovandro C. C. F. Pereira

Institute for Quantum Computing
University of Waterloo

CryptoWorks21

Agenda

- Preliminaries: security and post-quantum security
- A little history and awareness for post-quantum cryptography
- Hash-based cryptography
- Multivariate public-key cryptography

What is security after all?

Computer Security

Computer security is the **protection afforded** to an information system in order to attain the applicable goals of preserving the **integrity/authenticity, availability, and confidentiality** of the resources (includes hardware, software, information/data, and telecommunications).

– NIST Computer Security Handbook

What is security after all?

Computer Security

Computer security is the **protection afforded** to an information system in order to attain the applicable goals of preserving the **integrity/authenticity, availability, and confidentiality** of the resources (includes hardware, software, information/data, and telecommunications).

– NIST Computer Security Handbook

- Security requires more than confidentiality. Solutions are not fully supported by **information-theoretic security**. Hardness assumptions and bug-susceptible implementations should be taken into account.

What is security after all?

Computer Security

Computer security is the **protection afforded** to an information system in order to attain the applicable goals of preserving the **integrity/authenticity, availability, and confidentiality** of the resources (includes hardware, software, information/data, and telecommunications).

– NIST Computer Security Handbook

- Security requires more than confidentiality. Solutions are not fully supported by **information-theoretic security**. Hardness assumptions and bug-susceptible implementations should be taken into account.
 - ▶ \uparrow security \implies \uparrow cost

What is security after all?

Computer Security

Computer security is the **protection afforded** to an information system in order to attain the applicable goals of preserving the **integrity/authenticity, availability, and confidentiality** of the resources (includes hardware, software, information/data, and telecommunications).

– NIST Computer Security Handbook

- Security requires more than confidentiality. Solutions are not fully supported by **information-theoretic security**. Hardness assumptions and bug-susceptible implementations should be taken into account.
 - ▶ \uparrow security \implies \uparrow cost
 - ▶ Investments: Parameter sizes, code analysis, cryptanalysis, etc.

Security Goals (CIA, not the agency)

- **Confidentiality** (symmetric): prevent eavesdropping



Security Goals (CIA, not the agency)

- **Confidentiality** (symmetric): prevent eavesdropping
 - ▶ instant messaging, local storage (Capital One?)



Security Goals (CIA, not the agency)

- **Confidentiality** (symmetric): prevent eavesdropping
 - ▶ instant messaging, local storage (Capital One?)
 - ▶ **symmetric ciphers**: OTP, AES (block cipher), ChaCha20 (stream cipher) ...



Security Goals (CIA, not the agency)

- Integrity (symmetric): prevent data tampering



Security Goals (CIA, not the agency)

- Integrity (symmetric): prevent data tampering
 - ▶ control messages: pacemakers, drones



Security Goals (CIA, not the agency)

- Integrity (symmetric): prevent data tampering

- ▶ control messages: pacemakers, drones



- ▶ hash functions: SHA-2, SHA-3, Blake2
- ▶ message authentication codes (MACs): HMAC (keyed hash), Poly1305, AES-GCM.



Security Goals (CIA, not the agency)

- **Authenticity** / non-repudiation: check the origin



Security Goals (CIA, not the agency)

- **Authenticity / non-repudiation:** check the origin
 - ▶ web browsing, software updates



Security Goals (CIA, not the agency)

- **Authenticity / non-repudiation:** check the origin

- ▶ web browsing, software updates
- ▶ **public key crypto (asymmetric):** RSA,
(EC)DSA



Quantum cryptanalysis effect I

Public-key crypto broken

- ▶ Shor 1994: *algorithms for quantum computation: discrete logarithms and factoring*
 - ★ ✗ No more RSA and DSA (discrete log-based)
- ▶ ✗ Proos and Zalka 2003: extension to ECDSA



Quantum cryptanalysis effect I

Public-key crypto broken

- ▶ Shor 1994: *algorithms for quantum computation: discrete logarithms and factoring*
 - ★ ✗ No more RSA and DSA (discrete log-based)
- ▶ ✗ Proos and Zalka 2003: extension to ECDSA



of qubits required

- ▶ **Factoring:** $2n + 2$ logical qubits^a
 - ★ RSA-3072: 6146 qubits

^aHäner, Roetteler, and Svore 2016.

^bRoetteler et al. 2017.

Quantum cryptanalysis effect I

Public-key crypto broken

- ▶ Shor 1994: *algorithms for quantum computation: discrete logarithms and factoring*
 - ★ ✗ No more RSA and DSA (discrete log-based)
- ▶ ✗ Proos and Zalka 2003: extension to ECDSA



of qubits required

- ▶ **Factoring:** $2n + 2$ logical qubits^a
 - ★ RSA-3072: 6146 qubits
- ▶ **ECDLP:** $9n + 2\lceil \log n \rceil + 10$ logical qubits^b
 - ★ ECC-256: 2330 qubits

^aHäner, Roetteler, and Svore 2016.

^bRoetteler et al. 2017.

Quantum cryptanalysis effect I

Public-key crypto broken

- ▶ Shor 1994: *algorithms for quantum computation: discrete logarithms and factoring*
 - ★ **X** No more RSA and DSA (discrete log-based)
- ▶ **X** Proos and Zalka 2003: extension to ECDSA



of qubits required

- ▶ **Factoring:** $2n + 2$ logical qubits^a
 - ★ RSA-3072: 6146 qubits
- ▶ **ECDLP:** $9n + 2\lceil \log n \rceil + 10$ logical qubits^b
 - ★ ECC-256: 2330 qubits
- ▶ For 128-bit security "ECC easier target than RSA"^b.

^aHäner, Roetteler, and Svore 2016.

^bRoetteler et al. 2017.

Quantum cryptanalysis effect II

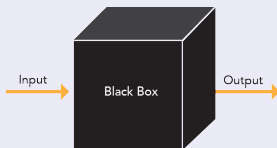
Quantum application II

- Grover 1996: *a fast quantum mechanical algorithm for database search*

Quantum cryptanalysis effect II

Quantum application II

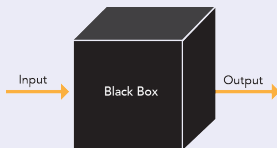
- Grover 1996: *a fast quantum mechanical algorithm for database search*
- **unstructured search**: find the unique input to black box function for a given output



Quantum cryptanalysis effect II

Quantum application II

- Grover 1996: *a fast quantum mechanical algorithm for database search*
- **unstructured search**: find the unique input to black box function for a given output

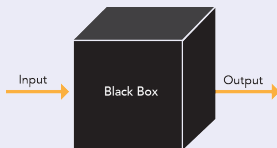


- Complexity is **optimal** Bennett et al. 1997:
No hope of solving NP-complete problems. $\Omega(\sqrt{N})$ lower bound.

Quantum cryptanalysis effect II

Quantum application II

- Grover 1996: *a fast quantum mechanical algorithm for database search*
- **unstructured search**: find the unique input to black box function for a given output



- Complexity is **optimal** Bennett et al. 1997:
No hope of solving NP-complete problems. $\omega(\sqrt{N})$ lower bound.
- Partially affects block-ciphers: breaks 128-bit keys in $O(2^{64})$ steps
Langenberg, Pham, and S. 2019: 865 qubits required (but $\approx 2^{77}$ circuit depth)

What are the affected families of cryptosystems?

- Confidentiality, Integrity (symmetric) ✓

Model of block ciphers

If the encryption key X is chosen at random, then an attacker who does not know the key cannot distinguish between the block cipher and a truly random permutation.

$$f(X) = E_X(I)$$

What are the affected families of cryptosystems?

- Confidentiality, Integrity (symmetric) ✓

Model of block ciphers

If the encryption key X is chosen at random, then an attacker who does not know the key cannot distinguish between the block cipher and a truly random permutation.

$$f(X) = E_X(I)$$

- Moreover, hash functions are also modeled as one-way functions.

What are the affected families of cryptosystems?

- Confidentiality, Integrity (symmetric) ✓

Model of block ciphers

If the encryption key X is chosen at random, then an attacker who does not know the key cannot distinguish between the block cipher and a truly random permutation.

$$f(X) = E_X(I)$$

- Moreover, hash functions are also modeled as one-way functions.
- Authentication (asymmetric) ✗

Assumptions do not hold in a quantum setting

One-way trapdoor functions where **trapdoor in {integer factoring, discrete logarithm}**

But there is hope

- Surprisingly, some old not very interesting algorithms naturally resist to quantum attacks.

But there is hope

- Surprisingly, some old not very interesting algorithms naturally resist to quantum attacks.

Definition

Post-quantum cryptography consists of classical cryptographic algorithms whose security assumption does not suffer an exponential speedup by quantum attacks.

But there is hope

- Surprisingly, some old not very interesting algorithms naturally resist to quantum attacks.

Definition

Post-quantum cryptography consists of classical cryptographic algorithms whose security assumption does not suffer an exponential speedup by quantum attacks.

Definition

Quantum-safe cryptography includes a broader set of cryptographic algorithms including non-classical assumptions such as laws of quantum physics, e.g. Quantum Key Distribution.

Provable security of post-quantum cryptography

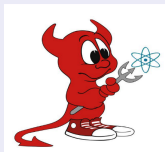
Security notions:

- 1 The underlying computational problem must be hard on a quantum computer

Provable security of post-quantum cryptography

Security notions:

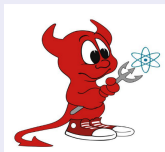
- 1 The underlying computational problem must be hard on a quantum computer
- 2 The security reduction from the hard problem must hold against quantum adversaries.



Provable security of post-quantum cryptography

Security notions:

- 1 The underlying computational problem must be hard on a quantum computer
- 2 The security reduction from the hard problem must hold against quantum adversaries.

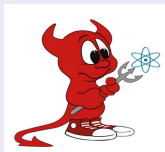


- Security proofs for classical full-domain hash (provable secure signature) or the Fiat-Shamir do not hold for post-quantum security.

Provable security of post-quantum cryptography

Security notions:

- 1 The underlying computational problem must be hard on a quantum computer
- 2 The security reduction from the hard problem must hold against quantum adversaries.



- Security proofs for classical full-domain hash (provable secure signature) or the Fiat-Shamir do not hold for post-quantum security.
 - ▶ Do not assume adversary can query an oracle $H(m)$ in superposition.

Hash-based crypto:

One-way functions exist, Lamport-Diffie: 40+ years



Hash-based crypto:

One-way functions exist, Lamport-Diffie: 40+ years



Multivariate crypto:

Random-looking multivariate system of non-linear equations is hard, UOV: 20 years

$$y_1 = x_1^2 + x_1x_2 + x_1x_4 + x_3$$

$$y_2 = x_3^2 + x_2x_3 + x_2x_4 + x_1 + 1$$

$$y_3 = \dots$$

Post-quantum algorithms and security assumptions (beliefs)

Hash-based crypto:

One-way functions exist, Lamport-Diffie: 40+ years



Multivariate crypto:

Random-looking multivariate system of non-linear equations is hard, UOV: 20 years

$$\begin{aligned}y_1 &= x_1^2 + x_1x_2 + x_1x_4 + x_3 \\y_2 &= x_3^2 + x_2x_3 + x_2x_4 + x_1 + 1 \\y_3 &= \dots\end{aligned}$$

Code-based crypto:

Syndrome decoding for error-correcting codes is hard,
McEliece/Goppa: 40 years



Post-quantum algorithms and security assumptions (beliefs)

Hash-based crypto:

One-way functions exist, Lamport-Diffie: 40+ years



Multivariate crypto:

Random-looking multivariate system of non-linear equations is hard, UOV: 20 years

$$\begin{aligned}y_1 &= x_1^2 + x_1x_2 + x_1x_4 + x_3 \\y_2 &= x_3^2 + x_2x_3 + x_2x_4 + x_1 + 1 \\y_3 &= \dots\end{aligned}$$

Code-based crypto:

Syndrome decoding for error-correcting codes is hard,
McEliece/Goppa: 40 years



Lattice-based crypto:

Finding short/close vectors on a lattice is hard,
NTRU: 20+ years, LWE: 14 years



Post-quantum algorithms and security assumptions (beliefs)

Hash-based crypto:

One-way functions exist, Lamport-Diffie: 40+ years



Multivariate crypto:

Random-looking multivariate system of non-linear equations is hard, UOV: 20 years

$$\begin{aligned}y_1 &= x_1^2 + x_1x_2 + x_1x_4 + x_3 \\y_2 &= x_3^2 + x_2x_3 + x_2x_4 + x_1 + 1 \\y_3 &= \dots\end{aligned}$$

Code-based crypto:

Syndrome decoding for error-correcting codes is hard, McEliece/Goppa: 40 years



Lattice-based crypto:

Finding short/close vectors on a lattice is hard, NTRU: 20+ years, LWE: 14 years



Supersingular Isogeny-based crypto:

Computing isogenies between supersingular elliptic curves is hard, CGL hash: 14 years, SIDH kex: 8 years



Post-quantum algorithms aspects

Hash-based crypto:

- ★ Pros: simplicity (hash), efficient, high confidence
- ★ Cons: large signatures (1–40 KiB), only signatures



Post-quantum algorithms aspects

Hash-based crypto:

- ★ Pros: simplicity (hash), efficient, high confidence
- ★ Cons: large signatures (1–40 KiB), only signatures



Multivariate-quadratic crypto:

- ★ Pros: small finite fields, matrix-vector ops, fast
- ★ Cons: large keys (>10 KiB), **only signatures** safe

$$y_1 = x_1^2 + x_1x_2 + x_1x_4 + x_3$$

$$y_2 = x_3^2 + x_2x_3 + x_2x_4 + x_1 + 1$$

$$y_3 = \dots$$

Post-quantum algorithms aspects

Hash-based crypto:

- ★ Pros: simplicity (hash), efficient, high confidence
- ★ Cons: large signatures (1–40 KiB), only signatures



Multivariate-quadratic crypto:

- ★ Pros: small finite fields, matrix-vector ops, fast
- ★ Cons: large keys (>10 KiB), **only signatures** safe

$$\begin{aligned}y_1 &= x_1^2 + x_1x_2 + x_1x_4 + x_3 \\y_2 &= x_3^2 + x_2x_3 + x_2x_4 + x_1 + 1 \\y_3 &= \dots\end{aligned}$$

Code-based crypto:

- ★ Pros: small finite fields, matrix-vector ops, fast
- ★ Cons: large keys (MiB), **only encryption** shown safe



Post-quantum algorithms and security assumptions

Lattice-based crypto:

- ★ Pros: versatile, moderate finite fields sizes, matrix/vector operations, very fast
- ★ Cons: moderate key sizes (few KiB)



Post-quantum algorithms and security assumptions

Lattice-based crypto:

- ★ Pros: versatile, moderate finite fields sizes, matrix/vector operations, very fast
- ★ Cons: moderate key sizes (few KiB)



Supersingular Isogeny-based crypto:

- ★ Pros: Small key sizes (few hundred bytes), many primitives
- ★ Cons: relatively slower



What is the risk of sticking to classical cryptography?



- In August 2015, NSA announces it plans to replace Suite B with a new cipher suite due to concerns about quantum computing attacks on ECC (even 256-bit curves).

What is the risk of sticking to classical cryptography?



- In August 2015, NSA announces it plans to replace Suite B with a new cipher suite due to concerns about quantum computing attacks on ECC (even 256-bit curves).
- 2015, Kobitz N., Menezes A. *"A riddle wrapped in an enigma"*

What is the risk of sticking to classical cryptography?



- In August 2015, NSA announces it plans to replace Suite B with a new cipher suite due to concerns about quantum computing attacks on ECC (even 256-bit curves).
- 2015, Koblitz N., Menezes A. *"A riddle wrapped in an enigma"*
- 2016, NIST announces post-quantum standardization
 - ▶ Deadline for submission Nov 30th, 2017 (Round 1).
 - ▶ Focus on digital sigs, encryption and key establishment.

What is the risk of sticking to classical cryptography?



- In August 2015, NSA announces it plans to replace Suite B with a new cipher suite due to concerns about quantum computing attacks on ECC (even 256-bit curves).
- 2015, Kobitz N., Menezes A. *"A riddle wrapped in an enigma"*
- 2016, NIST announces post-quantum standardization
 - ▶ Deadline for submission Nov 30th, 2017 (Round 1).
 - ▶ Focus on digital sigs, encryption and key establishment.
- 2018, NIST PQC Standardization Conference (submitter's talks)

What is the risk of sticking to classical cryptography?

Mosca's risk analysis formula

Let X be the time to have certain information protected.

Let Y be the time to deploy post-quantum.

Let Z be the Y2Q (countdown of years to quantum).

If $Z < X + Y$: trouble!

What can one expect from Z?

- Huge investments in Quantum Computing research

NATURE | NEWS



Europe plans giant billion-euro quantum technologies project

Third European Union flagship will be similar in size and ambition to graphene and human brain initiatives.

Elizabeth Gibney

21 April 2016 | Updated: 26 April 2016

What can one expect from Z?

- Huge investments in Quantum Computing research

NATURE | NEWS



Europe plans giant billion-euro quantum technologies project

Third European Union flagship will be similar in size and ambition to graphene and human brain initiatives.

Elizabeth Gibney

21 April 2016 | Updated: 26 April 2016

- Industry competing for quantum supremacy (no crypto purpose).
 - ▶ 2017, Google and IBM building general-purpose small prototypes of QCs. Google has no fault-tolerance design plans.

What can one expect from Z?

- Huge investments in Quantum Computing research

NATURE | NEWS



Europe plans giant billion-euro quantum technologies project

Third European Union flagship will be similar in size and ambition to graphene and human brain initiatives.

Elizabeth Gibney

21 April 2016 | Updated: 26 April 2016

- Industry competing for quantum supremacy (no crypto purpose).
 - ▶ 2017, Google and IBM building general-purpose small prototypes of QCs. Google has no fault-tolerance design plans.
- Experts estimate that large QCs (thousand+ qubits) will be around by 2031 with 50% chance.

What can one expect from Y ?

Y : Time to deploy new cryptography with widespread adoption

What can one expect from Y ?

Y : Time to deploy new cryptography with widespread adoption

- Let's look at the history of ECC:

What can one expect from Y ?

Y : Time to deploy new cryptography with widespread adoption

- Let's look at the history of ECC:
 - ▶ 1985: Elliptic curves in cryptography by Koblitz and Miller

What can one expect from Y ?

Y : Time to deploy new cryptography with widespread adoption

- Let's look at the history of ECC:
 - ▶ 1985: Elliptic curves in cryptography by Koblitz and Miller
 - ▶ **1992**: ECDSA signature suggested by Scott Vanstone to DSS

What can one expect from \mathcal{Y} ?

\mathcal{Y} : Time to deploy new cryptography with widespread adoption

- Let's look at the history of ECC:
 - ▶ 1985: Elliptic curves in cryptography by Koblitz and Miller
 - ▶ **1992**: ECDSA signature suggested by Scott Vanstone to DSS
 - ▶ 1996: Paul Kotcher introduces side-channel attacks.

What can one expect from Y ?

Y : Time to deploy new cryptography with widespread adoption

- Let's look at the history of ECC:
 - ▶ 1985: Elliptic curves in cryptography by Koblitz and Miller
 - ▶ **1992**: ECDSA signature suggested by Scott Vanstone to DSS
 - ▶ 1996: Paul Kotcher introduces side-channel attacks.
 - ★ 1997: Joye-Quisquater introduce fault attacks against ECC

What can one expect from Y ?

Y : Time to deploy new cryptography with widespread adoption

- Let's look at the history of ECC:
 - ▶ 1985: Elliptic curves in cryptography by Koblitz and Miller
 - ▶ **1992**: ECDSA signature suggested by Scott Vanstone to DSS
 - ▶ 1996: Paul Kotcher introduces side-channel attacks.
 - ★ 1997: Joye-Quisquater introduce fault attacks against ECC
 - ★ 1998: The ECDLP is actually easy in some (anomalous) curves

What can one expect from \mathcal{Y} ?

\mathcal{Y} : Time to deploy new cryptography with widespread adoption

- Let's look at the history of ECC:
 - ▶ 1985: Elliptic curves in cryptography by Koblitz and Miller
 - ▶ **1992**: ECDSA signature suggested by Scott Vanstone to DSS
 - ▶ 1996: Paul Kotcher introduces side-channel attacks.
 - ★ 1997: Joye-Quisquater introduce fault attacks against ECC
 - ★ 1998: The ECDLP is actually easy in some (anomalous) curves
 - ★ 2000: Biehl-Meyer-Muller devise invalid curve attacks

What can one expect from \mathcal{Y} ?

\mathcal{Y} : Time to deploy new cryptography with widespread adoption

- Let's look at the history of ECC:
 - ▶ 1985: Elliptic curves in cryptography by Koblitz and Miller
 - ▶ **1992**: ECDSA signature suggested by Scott Vanstone to DSS
 - ▶ 1996: Paul Kotcher introduces side-channel attacks.
 - ★ 1997: Joye-Quisquater introduce fault attacks against ECC
 - ★ 1998: The ECDLP is actually easy in some (anomalous) curves
 - ★ 2000: Biehl-Meyer-Muller devise invalid curve attacks
 - ▶ **2000**: NIST FIPS 186-2 includes ECDSA and the 15 NIST curves

What can one expect from Y ?

- Let's look at the history of ECC (cont.):

2014 Cloudflare's post dedicated to Scott Vanstone

W.r.t. **https** certificates, despite ECDSA being much faster than RSA for TLS handshake/signing, **> 90% of the certificates used on the web in 2014 were RSA-based.**

What can one expect from Y ?

- Let's look at the history of ECC (cont.):

2014 Cloudflare's post dedicated to Scott Vanstone

W.r.t. **https** certificates, despite ECDSA being much faster than RSA for TLS handshake/signing, **> 90% of the certificates used on the web in 2014 were RSA-based.**

Main reason

Web sites owners slow to adopt new certificates due to maintainance of **compatibility with legacy browsers** that do not support the new algorithms. – Sullivan, N. 2014

What can one expect from \mathcal{Y} ?

The good news to ECC

In **Apr'17**, ECDSA finally **surpassed RSA** with 60% of all TLS connections. Recall that ECDSA was proposed in 1992 for DSS. It took 25 years for ECDSA to become widely deployed from its conception and **17** years from its standardization. – Cloudfare report.

What can one expect from \mathcal{Y} ?

The good news to ECC

In **Apr'17**, ECDSA finally **surpassed RSA** with 60% of all TLS connections. Recall that ECDSA was proposed in 1992 for DSS. It took 25 years for ECDSA to become widely deployed from its conception and **17** years from its standardization. – Cloudflare report.

The bad news to ECC

"For those partners and vendors that have not yet made the transition to Suite B algorithms, we recommend **not** making a significant expenditure to do so at this point but instead to prepare for the upcoming quantum resistant algorithm transition." – NSA 2015 announcement

What can one expect for Y ?

Jul'18, Pereira's suggested bound:

Conjectured bound $Y \geq 20$

If $Z < X + 20 \Rightarrow$ trouble!

What can one expect for Y ?

Jul'18, Pereira's suggested bound:

Conjectured bound $Y \geq 20$

If $Z < X + 20 \Rightarrow$ trouble!

- It is true that the more robust TLS infrastructure and experienced community will be faster at deploying implementations.

What can one expect for Y ?

Jul'18, Pereira's suggested bound:

Conjectured bound $Y \geq 20$

If $Z < X + 20 \Rightarrow$ trouble!

- It is true that the more robust TLS infrastructure and experienced community will be faster at deploying implementations.
- On the other hand:
 - ▶ NIST standardization analysis phase will take 5 years and 2 more for the drafts – D. Moody
 - ▶ The field of quantum cryptanalysis has only just begun (recent attacks against NTRU and binary MQ)
 - ▶ Two lines of attacks imply higher chances to break post-quantum assumptions.

Agenda

- Preliminaries: security and post-quantum security
- A little history and awareness for post-quantum
- Hash-based cryptography
- Multivariate public-key cryptography

Hash-based Signatures (HBS)

1976, Diffie-Hellman in "New directions in cryptography" introduce HBS:

One-way message authentication has a partial solution suggested to the authors by Leslie Lamport of Massachusetts Computer Associates. This technique employs a one-way function f mapping k -dimensional binary space into itself for k on the order of 100. If the transmitter wishes to send an N bit message he generates $2N$, randomly chosen, k -dimensional binary vectors $x_1, X_1, x_2, X_2, \dots, x_N, X_N$ which he keeps secret. The receiver is given the corresponding images under f , namely $y_1, Y_1, y_2, Y_2, \dots, y_N, Y_N$. Later, when the message $\mathbf{m} = (m_1, m_2, \dots, m_N)$ is to be sent, the transmitter sends x_1 or X_1 depending on whether $m_1 = 0$ or 1. He sends x_2 or X_2 depending on whether $m_2 = 0$ or 1, etc. The receiver operates with f on the first received block and sees whether it yields y_1 or Y_1 as its image and thus learns whether it was x_1 or X_1 , and whether $m_1 = 0$ or 1. In a similar manner the receiver is able to determine m_2, m_3, \dots, m_N . But the receiver is incapable of forging a change in even one bit of \mathbf{m} .

1976, Lamport-Diffie signature: sign a n -bit message

Let $f : \{0, 1\}^k \rightarrow \{0, 1\}^k$ be a public one-way function

1976, Lamport-Diffie signature: sign a n -bit message

Let $f : \{0, 1\}^k \rightarrow \{0, 1\}^k$ be a public one-way function

Signer generates at random n pairs of k -bit strings $(x_i, X_i)_{i=1}^n$:

$$sk = \{(x_1, X_1), (x_2, X_2), \dots, (x_n, X_n)\}$$

1976, Lamport-Diffie signature: sign a n -bit message

Let $f : \{0, 1\}^k \rightarrow \{0, 1\}^k$ be a public one-way function

Signer generates at random n pairs of k -bit strings $(x_i, X_i)_{i=1}^n$:

$$sk = \{(x_1, X_1), (x_2, X_2), \dots, (x_n, X_n)\}$$

Evaluate f on them: $y_i, Y_i \leftarrow f(x_i), f(X_i)$ and publishes:

$$pk = \{(y_1, Y_1), (y_2, Y_2), \dots, (y_n, Y_n)\}$$

1976, Lamport-Diffie signature: sign a n -bit message

Let $f : \{0, 1\}^k \rightarrow \{0, 1\}^k$ be a public one-way function

Signer generates at random n pairs of k -bit strings $(x_i, X_i)_{i=1}^n$:

$$sk = \{(x_1, X_1), (x_2, X_2), \dots, (x_n, X_n)\}$$

Evaluate f on them: $y_i, Y_i \leftarrow f(x_i), f(X_i)$ and publishes:

$$pk = \{(y_1, Y_1), (y_2, Y_2), \dots, (y_n, Y_n)\}$$

To sign $m \in \{0, 1\}^n$ reveal

$$\sigma_i \leftarrow \overline{m_i} x_i + m_i X_i \quad // m_i \text{ selects the } i\text{-th element}$$

1976, Lamport-Diffie signature: sign a n -bit message

Let $f : \{0, 1\}^k \rightarrow \{0, 1\}^k$ be a public one-way function

Signer generates at random n pairs of k -bit strings $(x_i, X_i)_{i=1}^n$:

$$sk = \{(x_1, X_1), (x_2, X_2), \dots, (x_n, X_n)\}$$

Evaluate f on them: $y_i, Y_i \leftarrow f(x_i), f(X_i)$ and publishes:

$$pk = \{(y_1, Y_1), (y_2, Y_2), \dots, (y_n, Y_n)\}$$

To sign $m \in \{0, 1\}^n$ reveal

$$\sigma_i \leftarrow \overline{m_i} x_i + m_i X_i \quad // m_i \text{ selects the } i\text{-th element}$$

To verify $\sigma = (\sigma_1, \dots, \sigma_n)$ check if

$$f(\sigma_i) \stackrel{?}{=} \begin{cases} y_i \Rightarrow m_i = 0 \\ Y_i \Rightarrow m_i = 1 \end{cases}$$

Hash-based Signatures (HBS)

1976, Lamport's digital signature idea

- A problem pointed by Diffie and Hellman:

Hash-based Signatures (HBS)

1976, Lamport's digital signature idea

- A problem pointed by Diffie and Hellman:
Imagine the message to be signed is $N = 1Mbit$.

Hash-based Signatures (HBS)

1976, Lamport's digital signature idea

- A problem pointed by Diffie and Hellman:

Imagine the message to be signed is $N = 1Mbit$.

The size of sk, pk would be $2kN \approx 2 \cdot 100 \cdot 10^6 = 200M$ bits

Hash-based Signatures (HBS)

1976, Lamport's digital signature idea

- A problem pointed by Diffie and Hellman:

Imagine the message to be signed is $N = 1Mbit$.

The size of sk, pk would be $2kN \approx 2 \cdot 100 \cdot 10^6 = 200M$ bits

- An improvement would be:

Hash-based Signatures (HBS)

1976, Lamport's digital signature idea

- A problem pointed by Diffie and Hellman:

Imagine the message to be signed is $N = 1Mbit$.

The size of sk , pk would be $2kN \approx 2 \cdot 100 \cdot 10^6 = 200M$ bits

- An improvement would be:

Define a new one-way function $g : \{0, 1\}^* \rightarrow \{0, 1\}^n$.

Hash-based Signatures (HBS)

1976, Lamport's digital signature idea

- A problem pointed by Diffie and Hellman:

Imagine the message to be signed is $N = 1\text{Mbit}$.

The size of sk , pk would be $2kN \approx 2 \cdot 100 \cdot 10^6 = 200M$ bits

- An improvement would be:

Define a new one-way function $g : \{0, 1\}^* \rightarrow \{0, 1\}^n$.

Under message $m \in \{0, 1\}^*$, compute $m' = g(m) \in \{0, 1\}^n$

Hash-based Signatures (HBS)

1976, Lamport's digital signature idea

- A problem pointed by Diffie and Hellman:

Imagine the message to be signed is $N = 1\text{Mbit}$.

The size of sk , pk would be $2kN \approx 2 \cdot 100 \cdot 10^6 = 200M$ bits

- An improvement would be:

Define a new one-way function $g : \{0, 1\}^* \rightarrow \{0, 1\}^n$.

Under message $m \in \{0, 1\}^*$, compute $m' = g(m) \in \{0, 1\}^n$

Generate sk and pk using $f : \{0, 1\}^k \rightarrow \{0, 1\}^k$ for each bit $1, \dots, n$.

Hash-based Signatures (HBS)

1976, Lamport's digital signature idea

- A problem pointed by Diffie and Hellman:

Imagine the message to be signed is $N = 1\text{Mbit}$.

The size of sk , pk would be $2kN \approx 2 \cdot 100 \cdot 10^6 = 200M$ bits

- An improvement would be:

Define a new one-way function $g : \{0, 1\}^* \rightarrow \{0, 1\}^n$.

Under message $m \in \{0, 1\}^*$, compute $m' = g(m) \in \{0, 1\}^n$

Generate sk and pk using $f : \{0, 1\}^k \rightarrow \{0, 1\}^k$ for each bit $1, \dots, n$.

Note that g can have many inputs mapped to a same output.

Hash-based Signatures (HBS)

1976, Lamport's digital signature idea

- A problem pointed by Diffie and Hellman:

Imagine the message to be signed is $N = 1\text{Mbit}$.

The size of sk , pk would be $2kN \approx 2 \cdot 100 \cdot 10^6 = 200M$ bits

- An improvement would be:

Define a new one-way function $g : \{0, 1\}^* \rightarrow \{0, 1\}^n$.

Under message $m \in \{0, 1\}^*$, compute $m' = g(m) \in \{0, 1\}^n$

Generate sk and pk using $f : \{0, 1\}^k \rightarrow \{0, 1\}^k$ for each bit $1, \dots, n$.

Note that g can have many inputs mapped to a same output.

Therefore, g should have stronger properties than f (collision resistance).

Hash-based Signatures (HBS)

Remark: Lamport-Diffie is a one-time signature (OTS)!

- ① Assume a message $m_1 = (\mathbf{0}, \mathbf{1}, \mathbf{1}) \in \{0, 1\}^3$ is signed. Thus

$$\begin{aligned}\sigma &= \{\bar{\mathbf{0}}x_1 + \mathbf{0}X_1, \bar{\mathbf{1}}x_2 + \mathbf{1}X_2, \bar{\mathbf{1}}x_3 + \mathbf{1}X_3\} \\ &= \{x_1, X_2, X_3\}\end{aligned}$$

Hash-based Signatures (HBS)

Remark: Lamport-Diffie is a one-time signature (OTS)!

- ① Assume a message $m_1 = (\mathbf{0}, \mathbf{1}, \mathbf{1}) \in \{0, 1\}^3$ is signed. Thus

$$\begin{aligned}\sigma &= \{\bar{\mathbf{0}}x_1 + \mathbf{0}X_1, \bar{\mathbf{1}}x_2 + \mathbf{1}X_2, \bar{\mathbf{1}}x_3 + \mathbf{1}X_3\} \\ &= \{x_1, X_2, X_3\}\end{aligned}$$

- ② Let the signer produce a new signature of $m_2 = (1, 1, 0)$ under sk :

$$\sigma' = \{X_1, X_2, x_3\}$$

Hash-based Signatures (HBS)

Remark: Lamport-Diffie is a one-time signature (OTS)!

- ① Assume a message $m_1 = (\mathbf{0}, \mathbf{1}, \mathbf{1}) \in \{0, 1\}^3$ is signed. Thus

$$\begin{aligned}\sigma &= \{\bar{\mathbf{0}}x_1 + \mathbf{0}X_1, \bar{\mathbf{1}}x_2 + \mathbf{1}X_2, \bar{\mathbf{1}}x_3 + \mathbf{1}X_3\} \\ &= \{x_1, X_2, X_3\}\end{aligned}$$

- ② Let the signer produce a new signature of $m_2 = (1, 1, 0)$ under sk :

$$\sigma' = \{X_1, X_2, x_3\}$$

- ③ Notice that $\{x_1, X_1, X_2, x_3, X_3\}$ are now public.

Hash-based Signatures (HBS)

Remark: Lamport-Diffie is a one-time signature (OTS)!

- ① Assume a message $m_1 = (\mathbf{0}, \mathbf{1}, \mathbf{1}) \in \{0, 1\}^3$ is signed. Thus

$$\begin{aligned}\sigma &= \{\bar{\mathbf{0}}x_1 + \mathbf{0}X_1, \bar{\mathbf{1}}x_2 + \mathbf{1}X_2, \bar{\mathbf{1}}x_3 + \mathbf{1}X_3\} \\ &= \{x_1, X_2, X_3\}\end{aligned}$$

- ② Let the signer produce a new signature of $m_2 = (1, 1, 0)$ under sk :

$$\sigma' = \{X_1, X_2, x_3\}$$

- ③ Notice that $\{x_1, X_1, X_2, x_3, X_3\}$ are now public.
- ④ Then it's easy to forge a signature of $m_3 = (1, 1, 1)$ for example. Thus, Lamport-Diffie signature is OTS and each key pair can be only used once.

Hash-based Signatures (HBS)

Security assumption of Lamport-Diffie (LD)

- ① one-way function f is hard to invert and
- ② it is hard to find different input values that map to a same output

Put the above together, HBS rely on the existence of modern **cryptographically secure hash functions**.

Hash-based Signatures (HBS)

- Can LD be improved in terms of space?

Hash-based Signatures (HBS)

- Can LD be improved in terms of space?

Merkle 1979, an optimization of LD due to Winternitz

Idea: instead of processing $m \in \{0,1\}^n$ bit-by-bit, use w -bit chunks.

$$m = (m_1 || \cdots || m_{\lceil n/w \rceil})$$

where $m_i \in \{0,1\}^w$ can be viewed as w -bit integers.

Hash-based Signatures (HBS)

- Can LD be improved in terms of space?

Merkle 1979, an optimization of LD due to Winternitz

Idea: instead of processing $m \in \{0, 1\}^n$ bit-by-bit, use w -bit chunks.

$$m = (m_1 || \dots || m_{\lceil n/w \rceil})$$

where $m_i \in \{0, 1\}^w$ can be viewed as w -bit integers.

- KeyGen: Precompute $x_i, y_i = f^{2^w-1}(x_i)$ for $i = 1, \dots, \lceil n/w \rceil$
where $f^t(x) = f(\dots f(x) \dots)$ means t applications of f

Hash-based Signatures (HBS)

- Can LD be improved in terms of space?

Merkle 1979, an optimization of LD due to Winternitz

Idea: instead of processing $m \in \{0, 1\}^n$ bit-by-bit, use w -bit chunks.

$$m = (m_1 || \cdots || m_{\lceil n/w \rceil})$$

where $m_i \in \{0, 1\}^w$ can be viewed as w -bit integers.

- ▶ KeyGen: Precompute $x_i, y_i = f^{2^w-1}(x_i)$ for $i = 1, \dots, \lceil n/w \rceil$ where $f^t(x) = f(\cdots f(x) \cdots)$ means t applications of f
- ▶ Actually, it is possible to do better:

$$pk = y = g(y_1 || \cdots || y_{\lceil n/w \rceil})$$

- ▶ Public key pk boils down to one hash value (instead of $2n$).

Hash-based Signatures (HBS)

(cont. ...)

Winternitz OTS

- Sign: compute

$$\sigma = (f^{m_1}(x_1), \dots, f^{m_{\lceil n/w \rceil}}(x_{\lceil n/w \rceil}))$$

Hash-based Signatures (HBS)

(cont. ...)

Winternitz OTS

- Sign: compute

$$\sigma = (f^{m_1}(x_1), \dots, f^{m_{\lceil n/w \rceil}}(x_{\lceil n/w \rceil}))$$

- Verify: compute

$$y'_i = f^{2^w - 1 - m_i}(\sigma_i), \text{ for all } i$$

$$y' = g(y'_1 \parallel \dots \parallel y'_{\lceil n/w \rceil})$$

Check

$$y' \stackrel{?}{=} y$$

Hash-based Signatures (HBS)

- **Problem:** Winternitz defined exactly as previously is **insecure**!

Hash-based Signatures (HBS)

- **Problem:** Winternitz defined exactly as previously is **insecure**!

Assume a message $m = (m_1, \dots, \mathbf{m}_i, \dots, m_{\lceil n/w \rceil})$ with signature

$$\sigma = (f^{m_1}(x_1), \dots, \mathbf{f}^{\mathbf{m}_i}(\mathbf{x}_i), \dots)$$

and suppose we have $m_i < 2^w - 1$

Hash-based Signatures (HBS)

- **Problem:** Winternitz defined exactly as previously is **insecure!**

Assume a message $m = (m_1, \dots, m_i, \dots, m_{\lceil n/w \rceil})$ with signature

$$\sigma = (f^{m_1}(x_1), \dots, f^{m_i}(x_i), \dots)$$

and suppose we have $m_i < 2^w - 1$

- Then, an adversary can produce a valid signature for the message

$$m' = (m_1, \dots, m_i + 1, \dots, m_{\lceil n/w \rceil})$$

Hash-based Signatures (HBS)

- **Problem:** Winternitz defined exactly as previously is **insecure!**

Assume a message $m = (m_1, \dots, m_i, \dots, m_{\lceil n/w \rceil})$ with signature

$$\sigma = (f^{m_1}(x_1), \dots, f^{m_i}(x_i), \dots)$$

and suppose we have $m_i < 2^w - 1$

- Then, an adversary can produce a valid signature for the message

$$m' = (m_1, \dots, m_i + 1, \dots, m_{\lceil n/w \rceil})$$

given by

$$\begin{aligned}\sigma' &= (f^{m_1}(x_1), \dots, f(f^{m_i}(x_i)), \dots) \\ &= (f^{m_1}(x_1), \dots, f^{m_i+1}(x_i), \dots) \\ &= \sigma(m')\end{aligned}$$

Hash-based Signatures (HBS)

- **Problem:** Winternitz defined exactly as previously is **insecure!**

Assume a message $m = (m_1, \dots, \mathbf{m}_i, \dots, m_{\lceil n/w \rceil})$ with signature

$$\sigma = (f^{m_1}(x_1), \dots, \mathbf{f}^{\mathbf{m}_i}(\mathbf{x}_i), \dots)$$

and suppose we have $m_i < 2^w - 1$

- Then, an adversary can produce a valid signature for the message

$$m' = (m_1, \dots, m_i + 1, \dots, m_{\lceil n/w \rceil})$$

given by

$$\begin{aligned}\sigma' &= (f^{m_1}(x_1), \dots, f(\mathbf{f}^{\mathbf{m}_i}(\mathbf{x}_i)), \dots) \\ &= (f^{m_1}(x_1), \dots, \mathbf{f}^{\mathbf{m}_i+1}(\mathbf{x}_i), \dots) \\ &= \sigma(m')\end{aligned}$$

- Violates the notion of existential unforgeability!

Hash-based Signatures (HBS)

- **Solution:** 1979, Merkle introduces a checksum.

Hash-based Signatures (HBS)

- **Solution:** 1979, Merkle introduces a checksum.
- From $m = (m_1, \dots, m_{\lceil n/w \rceil})$, compute

$$CS = \sum_{i=1}^{\lceil n/w \rceil} 2^w - 1 - m_i$$

which can be stored in $t = \log_2(\lceil n/w \rceil \cdot (2^w - 1))$ bits.

Hash-based Signatures (HBS)

- **Solution:** 1979, Merkle introduces a checksum.
- From $m = (m_1, \dots, m_{\lceil n/w \rceil})$, compute

$$CS = \sum_{i=1}^{\lceil n/w \rceil} 2^w - 1 - m_i$$

which can be stored in $t = \log_2(\lceil n/w \rceil \cdot (2^w - 1))$ bits.

- **Idea:** represent CS as w -bit chunks as well:

$$CS = (m_{\lceil n/w \rceil + 1}, \dots, m_{\lceil n/w \rceil + \lceil t/w \rceil})$$

Hash-based Signatures (HBS)

- **Solution:** 1979, Merkle introduces a checksum.
- From $m = (m_1, \dots, m_{\lceil n/w \rceil})$, compute

$$CS = \sum_{i=1}^{\lceil n/w \rceil} 2^w - 1 - m_i$$

which can be stored in $t = \log_2(\lceil n/w \rceil \cdot (2^w - 1))$ bits.

- **Idea:** represent CS as w -bit chunks as well:

$$CS = (m_{\lceil n/w \rceil + 1}, \dots, m_{\lceil n/w \rceil + \lceil t/w \rceil})$$

- Extend the message to be $m || CS = (m_1, \dots, m_{\lceil n/w \rceil + \lceil t/w \rceil})$

Hash-based Signatures (HBS)

(checksum: cont ...)

- Given a signature

$$\sigma = (f^{m_1}(x_1), \dots, \mathbf{f}^{m_i}(\mathbf{x}_i), \dots)$$

Hash-based Signatures (HBS)

(checksum: cont ...)

- Given a signature

$$\sigma = (f^{m_1}(x_1), \dots, \mathbf{f}^{m_i}(\mathbf{x}_i), \dots)$$

- Let $i \leq \lceil n/w \rceil$. If the adversary tries to go forward on $\mathbf{f}^{m_i}(\mathbf{x}_i)$:

$$\begin{aligned}\sigma' &= (f^{m_1}(x_1), \dots, f(\mathbf{f}^{m_i}(\mathbf{x}_i)), \dots) \\ &= (f^{m_1}(x_1), \dots, \mathbf{f}^{m_i+1}(\mathbf{x}_i), \dots)\end{aligned}$$

Hash-based Signatures (HBS)

(checksum: cont ...)

- Given a signature

$$\sigma = (f^{m_1}(x_1), \dots, \mathbf{f}^{m_i}(\mathbf{x}_i), \dots)$$

- Let $i \leq \lceil n/w \rceil$. If the adversary tries to go forward on $\mathbf{f}^{m_i}(\mathbf{x}_i)$:

$$\begin{aligned}\sigma' &= (f^{m_1}(x_1), \dots, f(\mathbf{f}^{m_i}(\mathbf{x}_i)), \dots) \\ &= (f^{m_1}(x_1), \dots, \mathbf{f}^{m_i+1}(\mathbf{x}_i), \dots)\end{aligned}$$

- Then the new checksum is $CS' = CS - 1$ which implies an inversion $f^{-1}(f^{m_j}(x_j))$ for some $m_j \in CS$.

Hash-based Signatures (HBS)

(checksum: cont ...)

- Given a signature

$$\sigma = (f^{m_1}(x_1), \dots, f^{m_i}(x_i), \dots)$$

- Let $i \leq \lceil n/w \rceil$. If the adversary tries to go forward on $f^{m_i}(x_i)$:

$$\begin{aligned}\sigma' &= (f^{m_1}(x_1), \dots, f(f^{m_i}(x_i)), \dots) \\ &= (f^{m_1}(x_1), \dots, f^{m_i+1}(x_i), \dots)\end{aligned}$$

- Then the new checksum is $CS' = CS - 1$ which implies an inversion $f^{-1}(f^{m_j}(x_j))$ for some $m_j \in CS$.
- If a forger targets $m_j \in CS$, an inversion is also implied on m_j . Thus, the checksum protects against such attacks.

Winternitz OTS example

- Let $w = 2$ and one wants to sign the $(n = 4)$ -bit message

$$m = (1011)$$

with $\lceil n/w \rceil = 2$ and $m = (m_1 = 2, m_2 = 3)$.

Winternitz OTS example

- Let $w = 2$ and one wants to sign the $(n = 4)$ -bit message

$$m = (1011)$$

with $\lceil n/w \rceil = 2$ and $m = (m_1 = 2, m_2 = 3)$.

- Compute the checksum:

$$\begin{aligned} CS &= \sum_{i=1}^2 2^2 - 1 - m_i \\ &= 3 - 2 + 3 - 3 = 1 \end{aligned}$$

Winternitz OTS example

- Let $w = 2$ and one wants to sign the $(n = 4)$ -bit message

$$m = (1011)$$

with $\lceil n/w \rceil = 2$ and $m = (m_1 = 2, m_2 = 3)$.

- Compute the checksum:

$$\begin{aligned} CS &= \sum_{i=1}^2 2^2 - 1 - m_i \\ &= 3 - 2 + 3 - 3 = 1 \end{aligned}$$

Thus, $CS = (0001) = (m_3 = 0, m_4 = 1)$ and the actual message is

$$m || CS = (2, 3, 1, 0)$$

(cont ...) Winternitz OTS: example

- Key generation including CS:

$$sk = (x_1, \dots, x_4)$$

$$pk = y = g(f^3(x_1) \parallel \dots \parallel f^3(x_4))$$

(cont ...) Winternitz OTS: example

- Key generation including CS:

$$sk = (x_1, \dots, x_4)$$

$$pk = y = g(f^3(x_1) \parallel \dots \parallel f^3(x_4))$$

- The signature of $m \parallel CS = (2, 3, 1, 0)$ will be

$$\sigma = (f^2(x_1), f^3(x_2), f(x_3), x_4) = (\sigma_1, \sigma_2, \sigma_3, \sigma_4)$$

(cont ...) Winternitz OTS: example

- Key generation including CS:

$$sk = (x_1, \dots, x_4)$$

$$pk = y = g(f^3(x_1) \parallel \dots \parallel f^3(x_4))$$

- The signature of $m \parallel CS = (2, 3, 1, 0)$ will be

$$\sigma = (f^2(x_1), f^3(x_2), f(x_3), x_4) = (\sigma_1, \sigma_2, \sigma_3, \sigma_4)$$

- Verification of σ will be

(cont ...) Winternitz OTS: example

- Key generation including CS:

$$sk = (x_1, \dots, x_4)$$

$$pk = y = g(f^3(x_1) \parallel \dots \parallel f^3(x_4))$$

- The signature of $m \parallel CS = (2, 3, 1, 0)$ will be

$$\sigma = (f^2(x_1), f^3(x_2), f(x_3), x_4) = (\sigma_1, \sigma_2, \sigma_3, \sigma_4)$$

- Verification of σ will be
 - ▶ Recompute CS from m getting $m \parallel CS = (2, 3, 1, 0)$

(cont ...) Winternitz OTS: example

- Key generation including CS:

$$sk = (x_1, \dots, x_4)$$

$$pk = y = g(f^3(x_1) \parallel \dots \parallel f^3(x_4))$$

- The signature of $m \parallel CS = (2, 3, 1, 0)$ will be

$$\sigma = (f^2(x_1), f^3(x_2), f(x_3), x_4) = (\sigma_1, \sigma_2, \sigma_3, \sigma_4)$$

- Verification of σ will be

- ▶ Recompute CS from m getting $m \parallel CS = (2, 3, 1, 0)$
- ▶ Compute and check

$$(y'_1, y'_2, y'_3, y'_4) = (f^{3-2}(\sigma_1), f^{3-3}(\sigma_2), f^{3-1}(\sigma_3), f^{3-0}(\sigma_4))$$

$$g(y'_1 \parallel y'_2 \parallel y'_3 \parallel y'_4) \stackrel{?}{=} y$$

Hash-based Signatures (HBS)

Note on the efficiency of Winternitz OTS

- $|sk| = |\sigma| \approx (\mathbf{n}/\mathbf{w})\mathbf{k}$ (ignoring the checksum)

Hash-based Signatures (HBS)

Note on the efficiency of Winternitz OTS

- $|sk| = |\sigma| \approx (\mathbf{n/w})\mathbf{k}$ (ignoring the checksum)
 - ▶ Compares well with the $\mathbf{2nk}$ bits in $|sk|, |\sigma|$ for LD
 - ▶ A $\mathbf{2w}$ reduction factor for sk
 - ▶ A \mathbf{w} reduction factor for σ

Hash-based Signatures (HBS)

Note on the efficiency of Winternitz OTS

- $|sk| = |\sigma| \approx (\mathbf{n/w})\mathbf{k}$ (ignoring the checksum)
 - ▶ Compares well with the $2\mathbf{n}\mathbf{k}$ bits in $|sk|, |\sigma|$ for LD
 - ▶ A $2\mathbf{w}$ reduction factor for sk
 - ▶ A \mathbf{w} reduction factor for σ
- $|pk| = \mathbf{n}$

Hash-based Signatures (HBS)

Note on the efficiency of Winternitz OTS

- $|sk| = |\sigma| \approx (\mathbf{n}/\mathbf{w})\mathbf{k}$ (ignoring the checksum)
 - ▶ Compares well with the $2\mathbf{n}\mathbf{k}$ bits in $|sk|, |\sigma|$ for LD
 - ▶ A $2\mathbf{w}$ reduction factor for sk
 - ▶ A \mathbf{w} reduction factor for σ
- $|pk| = \mathbf{n}$
- But, WOTS requires $(2^{\mathbf{w}} - 1)/\mathbf{w}$ hash evaluations per bit
 - ▶ While LD requires 2 evaluations per bit
 - ▶ Notice that for $w = 1$ we get exactly Lamport-Diffie

Hash-based Signatures (HBS)

Note on the efficiency of Winternitz OTS

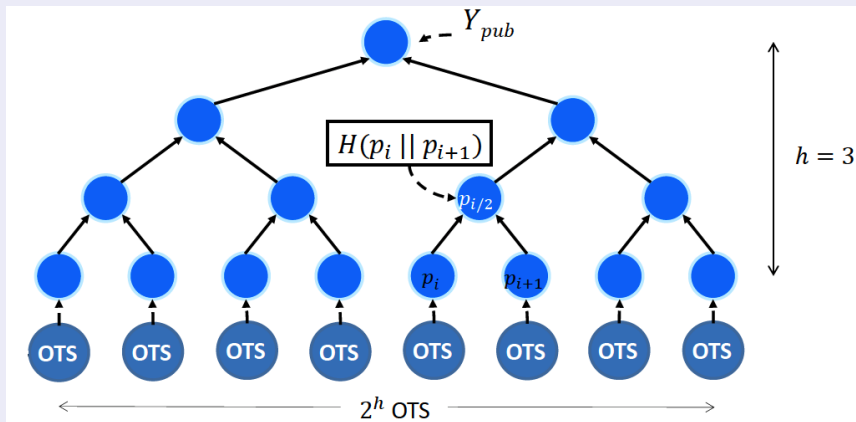
- $|sk| = |\sigma| \approx (\mathbf{n}/\mathbf{w})\mathbf{k}$ (ignoring the checksum)
 - ▶ Compares well with the $2\mathbf{n}\mathbf{k}$ bits in $|sk|, |\sigma|$ for LD
 - ▶ A $2\mathbf{w}$ reduction factor for sk
 - ▶ A \mathbf{w} reduction factor for σ
- $|pk| = \mathbf{n}$
- But, WOTS requires $(2^{\mathbf{w}} - 1)/\mathbf{w}$ hash evaluations per bit
 - ▶ While LD requires 2 evaluations per bit
 - ▶ Notice that for $w = 1$ we get exactly Lamport-Diffie
- Since hash evaluations can be very fast, it is a reasonable tradeoff

Hash-based Signatures (HBS)

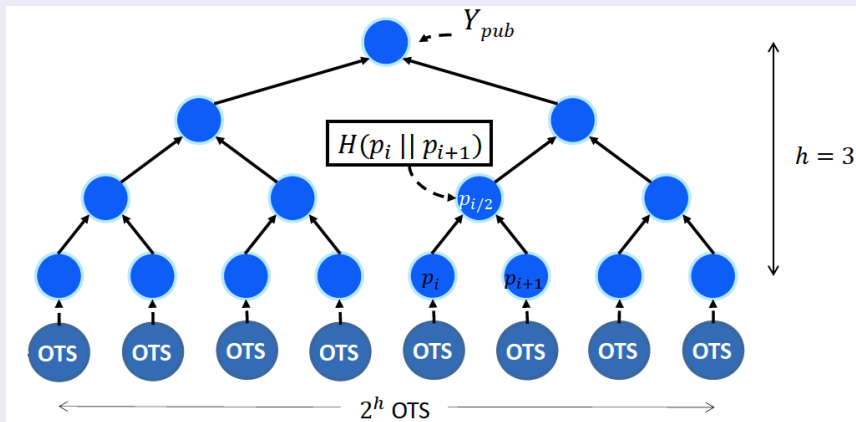
Scheme	$n = k$	$ PrivKey $	$ PubKey $	$ Sig $
LD OTS	256	16	16	16
WOTS ($w=2$)	256	4.2	32 bytes	4.2
WOTS ($w=8$)	256	1.1	32 bytes	1.1
WOTS ($w=16$)	256	0.6	32 bytes	0.6

Table: Parameter sizes for one-time signatures in KiB

1979, Merkle turns OTS into multi-time signatures



1979, Merkle turns OTS into multi-time signatures

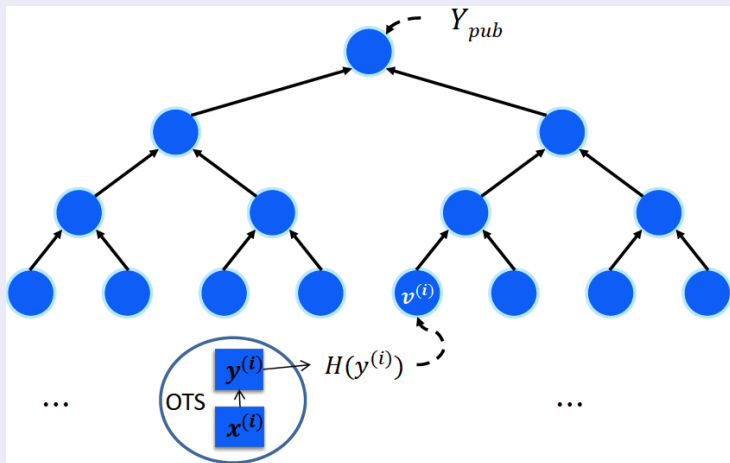


- ▶ OTS can be **any** one-time signature scheme.
- ▶ Y_{pub} authenticates 2^h OTS key pairs.



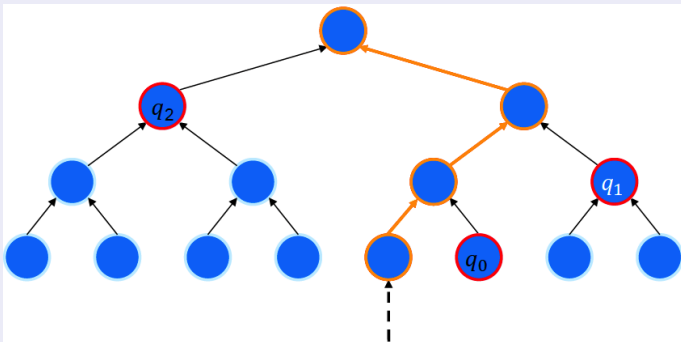
Hash-based Signatures (HBS)

1979, Merkle signature: **Key Generation**



Hash-based Signatures (HBS)

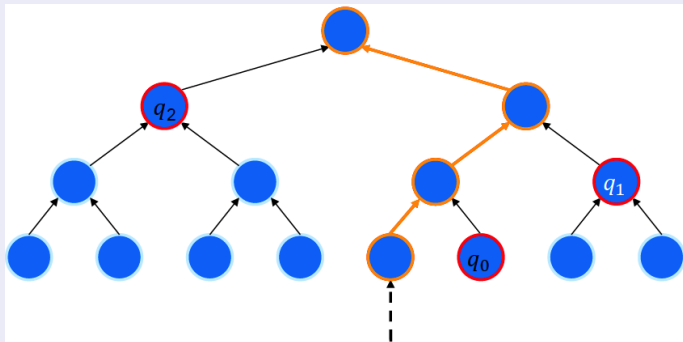
1979, Merkle signature: **Sign**



$$\sigma(data) = (i, \text{Sign}_{OTS}(x^{(i)}, data), v^{(i)}, (q_0, \dots, q_{h-1}))$$

Hash-based Signatures (HBS)

1979, Merkle signature: **Sign**

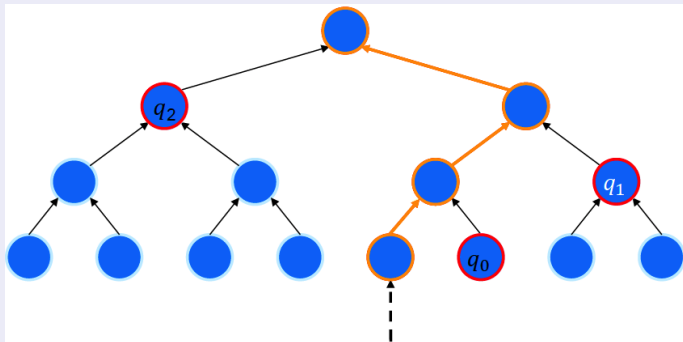


$$\sigma(data) = (i, \text{Sign}_{OTS}(x^{(i)}, data), v^{(i)}, (q_0, \dots, q_{h-1}))$$

- Nodes q_i are called the **authentication path** of i -th signature

Hash-based Signatures (HBS)

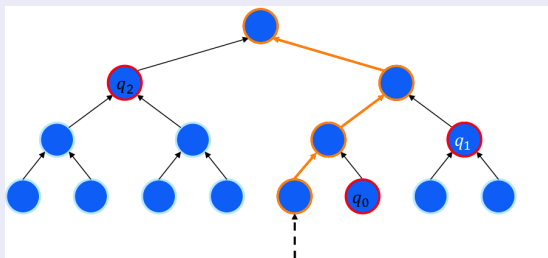
1979, Merkle signature: **Sign**



$$\sigma(data) = (i, \text{Sign}_{OTS}(x^{(i)}, data), v^{(i)}, (q_0, \dots, q_{h-1}))$$

- Nodes q_i are called the **authentication path** of i -th signature
- Stateful: susceptible to some attacks, e.g. 'restart attacks'

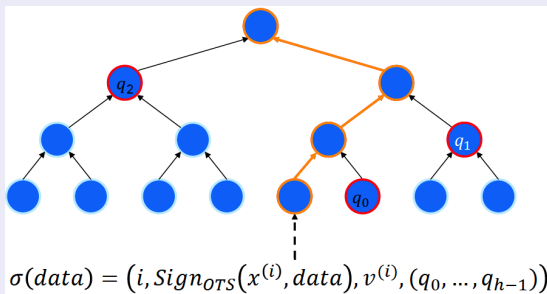
Time efficiency of the Merkle signature



$$\sigma(data) = (i, \text{Sign}_{OTS}(x^{(i)}, data), v^{(i)}, (q_0, \dots, q_{h-1}))$$

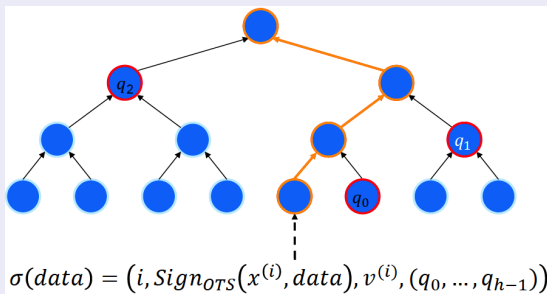
- Auth. path computation required $O(2^h)$ hash evaluations per signature

Time efficiency of the Merkle signature



- ▶ Auth. path computation required $O(2^h)$ hash evaluations per signature
- ▶ Improvement by Buchmann, Dahmen, and Schneider 2008: $O(h)$
 - ★ Store strategic nodes on a state and some temporarily on a stack.

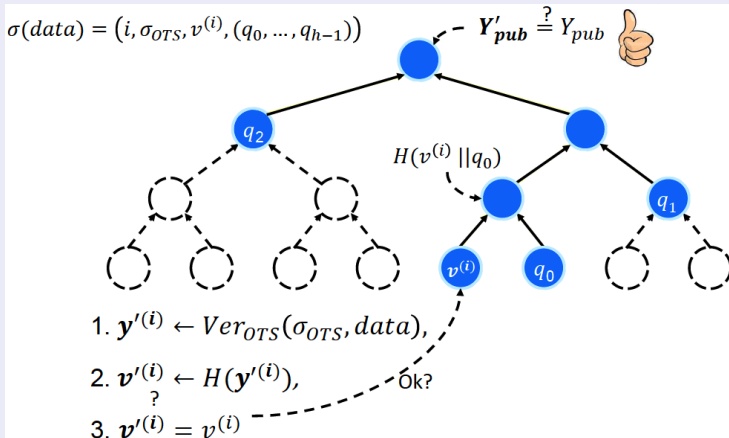
Time efficiency of the Merkle signature



- ▶ Auth. path computation required $O(2^h)$ hash evaluations per signature
- ▶ Improvement by Buchmann, Dahmen, and Schneider 2008: $O(h)$
 - ★ Store strategic nodes on a state and some temporarily on a stack.
 - ★ Possible tradeoff: size of the state vs # leaf computations per signature.

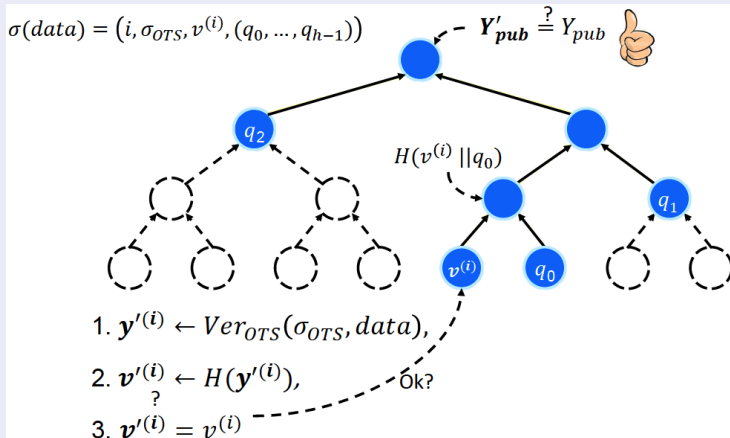
Hash-based Signatures (HBS)

1979, Merkle signature: **Verify**



Hash-based Signatures (HBS)

1979, Merkle signature: **Verify**



- An obvious optimization is not sending $v^{(i)}$. Verifier only checks the root.

Space efficiency of the Merkle signature

- ▶ Private key size: $2^h \cdot |sk_{OTS}|$
- ▶ Public key size: size of hash H , e.g. 256 bits.
- ▶ Signature size: $|\sigma| = |i| + |\sigma_{OTS}| + |v^{(i)}| + |(q_0, \dots, q_{h-1})|$

Space efficiency of the Merkle signature

- ▶ Private key size: $2^h \cdot |sk_{OTS}|$
- ▶ Public key size: size of hash H , e.g. 256 bits.
- ▶ Signature size: $|\sigma| = |i| + |\sigma_{OTS}| + |v^{(i)}| + |(q_0, \dots, q_{h-1})|$

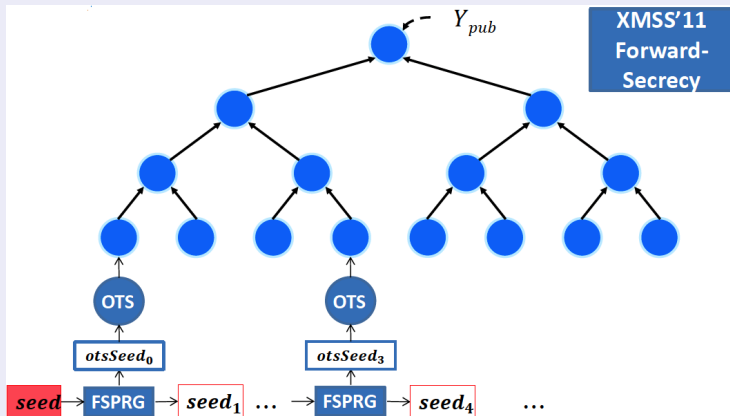
Merkle parameter sizes example

- ▶ $|f| = |H| = n = 256, h = 10$

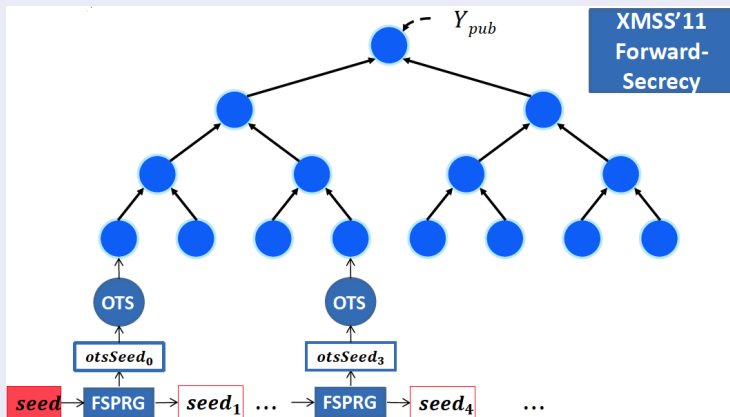
Scheme	$ PrivKey $	$ PubKey $	$ Sig $
Merkle+LD	16 MiB	32 bytes	16.4 KiB
Merkle+WOTS ($w=2$)	4.2 MiB	32 bytes	4.5 KiB
Merkle+WOTS ($w=16$)	0.6 MiB	32 bytes	0.9 KiB

Table: Parameter sizes for Merkle multi-time signature (**1024 signatures**)

Merkle signature: XMSS'11 introduces additional properties

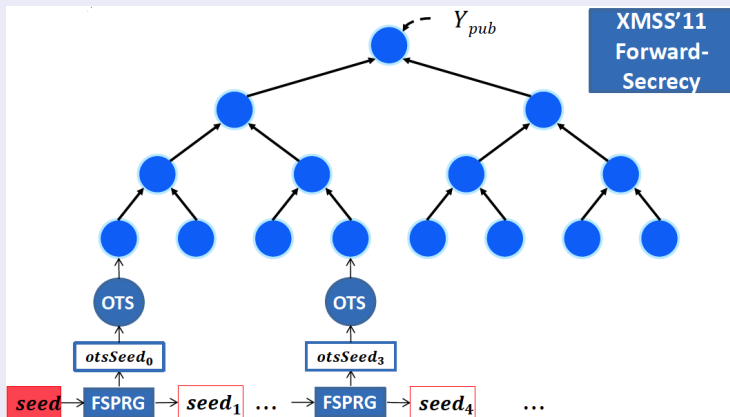


Merkle signature: XMSS'11 introduces additional properties



- XMSS uses the variant $WOTS^+$. Collision-resistance unnecessary.

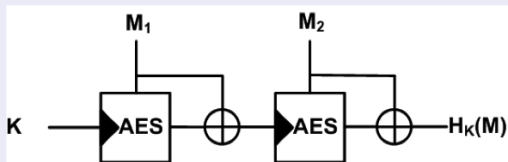
Merkle signature: XMSS'11 introduces additional properties



- XMSS uses the variant $WOTS^+$. Collision-resistance unnecessary.
- Implication: half-size hashes can be used safely.

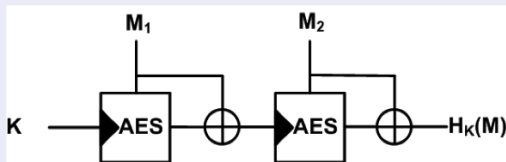
Merkle signature: implementation of PRNG and hash function

- Matyas-Meyer-Oseas: block-cipher-based hash function



Merkle signature: implementation of PRNG and hash function

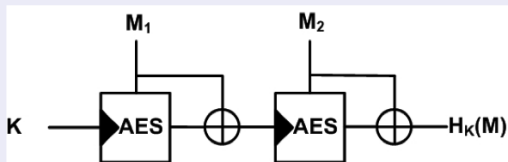
- Matyas-Meyer-Oseas: block-cipher-based hash function



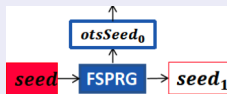
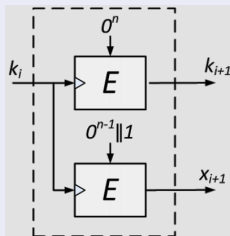
- Fast optimized (hw/sw) block-ciphers available in many platforms

Merkle signature: implementation of PRNG and hash function

- Matyas-Meyer-Oseas: block-cipher-based hash function

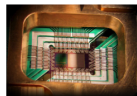


- Fast optimized (hw/sw) block-ciphers available in many platforms
- FSPRG by Standaert et al. 2010:

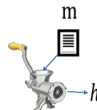


Hash-based Signatures (HBS) – A holistic view

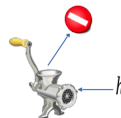
Post-quantum security



Only require hash functions
(efficient/minimal security assumption)



No reliance on trapdoors



Robust security (1976)
(cryptanalysis with little progress)



www.ieee.org/about/awards/bios/hamming_recipients

Larger signatures



References I



Shor, Peter W (1994).

“Algorithms for quantum computation: Discrete logarithms and factoring”.



Proos, John and Christof Zalka (2003).

“Shor’s discrete logarithm quantum algorithm for elliptic curves”.



Häner, Thomas, Martin Roetteler, and Krysta M Svore (2016).

“Factoring using $2n+2$ qubits with Toffoli based modular multiplication”.



Roetteler, Martin et al. (2017).

“Quantum resource estimates for computing elliptic curve discrete logarithms”.



Grover, Lov K (1996).

“A fast quantum mechanical algorithm for database search”.



Bennett, Charles H et al. (1997).

“Strengths and weaknesses of quantum computing”.



Langenberg, B., H. Pham, and Rainer S. (2019).

“Reducing the Cost of Implementing AES as a Quantum Circuit”.

References II



Merkle, Ralph Charles (1979).

“Secrecy, authentication, and public key systems”.



Buchmann, Johannes, Erik Dahmen, and Michael Schneider (2008).

“Merkle tree traversal revisited”.



Standaert, FranCcois-Xavier et al. (2010).

“Leakage resilient cryptography in practice”.