Hash-based Signature, the Round-3 Candidate: SPHINCS+

Tanja Lange

Academia Sinica

Eindhoven University of Technology

Post-quantum cryptography forum
14 January 2022

Public-key signatures

- Prerequisite: Alice has a private key  and public key .
- Prerequisite: Everyone knows  as belonging to Alice.
- Alice signs messages using . Other people verify using .

Public-key signatures

- Prerequisite: Alice has a private key  and public key .
- Prerequisite: Everyone knows  as belonging to Alice.
- Alice signs messages using . Other people verify using .
- Security goals: Integrity and authenticity.
- Nobody can produce signatures valid under  without .
- Modifications to signed message get caught.

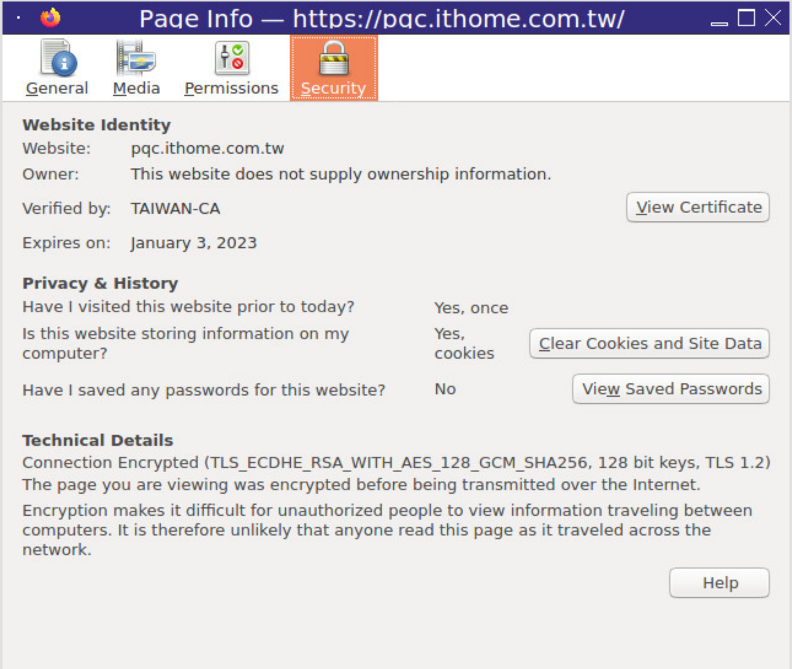

https://pqc.ithome.com.tw
Connection Security for pqc.ithome.com.tw
You are securely connected to this site.
Verified by: TAIWAN-CA
More Information
活動資訊
專業議程
國際講者
相關報導
第一屆後量子密碼論
Post-quantum Cryptography Foru

Connection Security for pqc.ithome.com.tw

You are securely connected to this site.

Verified by: TAIWAN-CA

More Information



Page Info — https://pqc.ithome.com.tw/
General
Media
Permissions
Security
Website Identity
Website: pqc.ithome.com.tw
Owner: This website does not supply ownership information.
Verified by: TAIWAN-CA
View Certificate
Expires on: January 3, 2023
Privacy & History
Have I visited this website prior to today? Yes, once
Is this website storing information on my computer? Yes, cookies
Clear Cookies and Site Data
Have I saved any passwords for this website? No
View Saved Passwords
Technical Details
Connection Encrypted (TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256, 128 bit keys, TLS 1.2)
The page you are viewing was encrypted before being transmitted over the Internet.
Encryption makes it difficult for unauthorized people to view information traveling between computers. It is therefore unlikely that anyone read this page as it traveled across the network.
Help

Firefox about:certificate?cert=MIIIODCCByCgAwIBAgIQR%2BUAAAAE6hVKWIXDsS0aUTANBgkqhkiG9w0BAQsFADBvMQswCQYDVQQGEwJUVzESMBAGA1UEChMJVEFJV0FOLUNBMRowGAYDVQQLExF

Certificate

*.ithome.com.tw | TWCA Secure SSL Certification Authority | TWCA Global Root CA

Subject Name

Country	TW
State/Province	Taiwan
Locality	Taipei
Organization	ITHOME PUBLICATIONS INC.
Organizational Unit	SYSTEM
Common Name	*.ithome.com.tw

Issuer Name

Country	TW
Organization	TAIWAN-CA
Organizational Unit	Secure SSL Sub-CA
Common Name	TWCA Secure SSL Certification Authority

Validity

Not Before	Thu, 16 Dec 2021 08:43:55 GMT
Not After	Tue, 03 Jan 2023 15:59:59 GMT

Subject Alt Names

DNS Name	*.ithome.com.tw
DNS Name	ithome.com.tw

Public Key Info

Algorithm	RSA
Key Size	2048
Exponent	65537
Modulus	C5:16:B9:74:75:83:F5:F4:37:6A:5F:27:A2:1B:6D:F9:AB:C5:8B:DC:D...

Miscellaneous

Serial Number	47:E5:00:00:00:04:EA:15:4A:58:85:C3:B1:2D:1A:51
Signature Algorithm	SHA-256 with RSA Encryption

Public Key Info

Algorithm	RSA
Key Size	2048
Exponent	65537
Modulus	C5:16:B9:74:75:83:F5:F4:37:6A:5F:27:A2:1B:6D:F9:AB:C5:8B:DC:D...

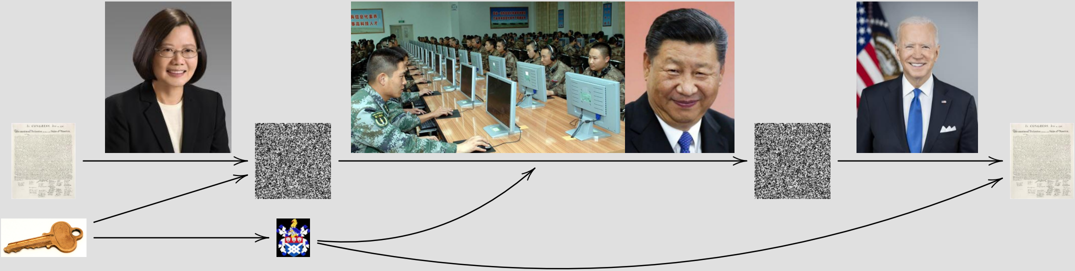
Miscellaneous

Serial Number	47:E5:00:00:00:04:EA:15:4A:58:85:C3:B1:2D:1A:51
Signature Algorithm	SHA-256 with RSA Encryption
Version	3
Download	PEM (cert) PEM (chain)

Fingerprints

SHA-256	BB:D8:99:8A:7B:9A:06:FE:81:A1:F2:18:92:1D:93:CB:62:1F:42:BE:36...
SHA-1	8E:E3:DA:17:00:DD:4F:7C:89:1A:33:E3:C2:9C:C1:ED:C4:3F:87:6B

Post-quantum public-key signatures: hash-based

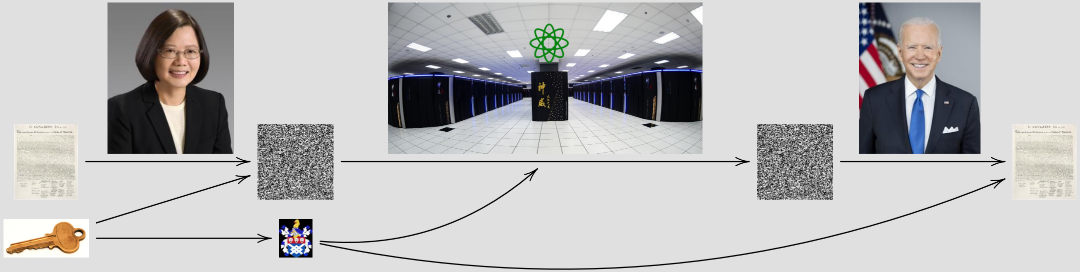


- Only one prerequisite: a good hash function, e.g. SHA3-512, ...
  Hash functions map long strings to fixed-length strings.
  $H : \{0,1\}^* \to \{0,1\}^n$.

  Signature schemes use hash functions in handling  .

- Old idea: 1979 Lamport one-time signatures;
  1979 Merkle extends to more signatures.

Post-quantum public-key signatures: hash-based



- Only one prerequisite: a good hash function, e.g. SHA3-512, . . .
  Hash functions map long strings to fixed-length strings.
  $H : \{0,1\}^* \to \{0,1\}^n$.

  Signature schemes use hash functions in handling .
- Quantum computers affect the hardness only marginally (Grover, not Shor).
- Old idea: 1979 Lamport one-time signatures;
  1979 Merkle extends to more signatures.

One-time signatures (Lamport and Winternitz)

Idea: Use one-wayness of cryptographic hash function to authenticate.
Private key: bit string s, public key: $H(s)$.

One-time signatures (Lamport and Winternitz)

Idea: Use one-wayness of cryptographic hash function to authenticate.
Private key: bit string s, public key: $H(s)$.
Can only use once as s known after first use.

One-time signatures (Lamport and Winternitz)

Idea: Use one-wayness of cryptographic hash function to authenticate.
Private key: bit string s, public key: $H(s)$.
Can only use once as s known after first use.

Extend to signing bit by having two values:
Private key: 2 bit strings (s_0, s_1), public key: $(H(s_0), H(s_1))$.

One-time signatures (Lamport and Winternitz)

Idea: Use one-wayness of cryptographic hash function to authenticate.
Private key: bit string s, public key: $H(s)$.
Can only use once as s known after first use.

Extend to signing bit by having two values:
Private key: 2 bit strings (s_0, s_1), public key: $(H(s_0), H(s_1))$.
To sign 0 reveal s_0, to sign 1 reveal s_1.

One-time signatures (Lamport and Winternitz)

Idea: Use one-wayness of cryptographic hash function to authenticate.
Private key: bit string s, public key: $H(s)$.
Can only use once as s known after first use.

Extend to signing bit by having two values:
Private key: 2 bit strings (s_0, s_1), public key: $(H(s_0), H(s_1))$.
To sign 0 reveal s_0, to sign 1 reveal s_1.

Lamport signs m via $H(m) = (h_0, h_1, \ldots, h_{255})$.
Private key: 256×2 bit strings $\mathbf{s} = (s_{0,0}, s_{0,1}, s_{1,0}, s_{1,1}, \ldots, s_{255,0}, s_{255,1})$,
public key: $\mathbf{p} = (H(s_{0,0}), H(s_{0,1}), H(s_{1,0}), H(s_{1,1}), \ldots, H(s_{255,0}), H(s_{255,1}))$.
To sign m reveal $s_{0,h_0}, s_{1,h_1}, \ldots, s_{255,h_{255}}$.

One-time signatures (Lamport and Winternitz)

Idea: Use one-wayness of cryptographic hash function to authenticate.
Private key: bit string s, public key: $H(s)$.
Can only use once as s known after first use.

Extend to signing bit by having two values:
Private key: 2 bit strings (s_0, s_1), public key: $(H(s_0), H(s_1))$.
To sign 0 reveal s_0, to sign 1 reveal s_1.

Lamport signs m via $H(m) = (h_0, h_1, \ldots, h_{255})$.
Private key: 256×2 bit strings $\mathbf{s} = (s_{0,0}, s_{0,1}, s_{1,0}, s_{1,1}, \ldots, s_{255,0}, s_{255,1})$,
public key: $\mathbf{p} = (H(s_{0,0}), H(s_{0,1}), H(s_{1,0}), H(s_{1,1}), \ldots, H(s_{255,0}), H(s_{255,1}))$.
To sign m reveal $s_{0,h_0}, s_{1,h_1}, \ldots, s_{255,h_{255}}$.
Tradeoff: define public key as $H(\mathbf{p})$, also reveal rest of $\mathbf{p}$ to sign,
for short public key at expense of longer signature.

One-time signatures (Lamport and Winternitz)

Idea: Use one-wayness of cryptographic hash function to authenticate.
Private key: bit string s, public key: $H(s)$.
Can only use once as s known after first use.

Extend to signing bit by having two values:
Private key: 2 bit strings (s_0, s_1), public key: $(H(s_0), H(s_1))$.
To sign 0 reveal s_0, to sign 1 reveal s_1.

Lamport signs m via $H(m) = (h_0, h_1, \ldots, h_{255})$.
Private key: 256×2 bit strings $\mathbf{s} = (s_{0,0}, s_{0,1}, s_{1,0}, s_{1,1}, \ldots, s_{255,0}, s_{255,1})$,
public key: $\mathbf{p} = (H(s_{0,0}), H(s_{0,1}), H(s_{1,0}), H(s_{1,1}), \ldots, H(s_{255,0}), H(s_{255,1}))$.
To sign m reveal $s_{0,h_0}, s_{1,h_1}, \ldots, s_{255,h_{255}}$.
Tradeoff: define public key as $H(\mathbf{p})$, also reveal rest of $\mathbf{p}$ to sign,
for short public key at expense of longer signature.

Winternitz achieves short public keys and signatures costing more calls to H.

On the fast track: stateful hash-based signatures

- CFRG has published 2 RFCs: RFC 8391 and RFC 8554

IETF Datatracker Groups Documents Meetings Other User

```
Internet Research Task Force (IRTF)                          A. Huelsing
Request for Comments: 8391                                 TU Eindhoven
Category: Informational                                        D. Butin
ISSN: 2070-1721                                            TU Darmstadt
                                                              S. Gazdag
                                                             genua GmbH
                                                            J. Rijneveld
                                                      Radboud University
                                                             A. Mohaisen
                                         University of Central Florida
                                                                May 2018

               XMSS: eXtended Merkle Signature Scheme
```

IETF Datatracker Groups Documents Meetings Other User

```
Internet Research Task Force (IRTF)                           D. McGrew
Request for Comments: 8554                                     M. Curcio
Category: Informational                                       S. Fluhrer
ISSN: 2070-1721                                            Cisco Systems
                                                              April 2019

               Leighton-Micali Hash-Based Signatures
```

On the fast track: stateful hash-based signatures

- CFRG has published 2 RFCs: RFC 8391 and RFC 8554
- NIST has standardized the same two schemes.

On the fast track: stateful hash-based signatures

- CFRG has published 2 RFCs: RFC 8391 and RFC 8554
- NIST has standardized the same two schemes.

- ISO SC27 JTC1 WG2 is working on standard for stateful hash-based signatures.

Merkle's (e.g.) 8-time signature system

Hash 8 one-time public keys into a single Merkle public key P_{15}.

$$P_{15} = H(P_{13}, P_{14})$$

$$P_{13} = H(P_9, P_{10}) \qquad P_{14} = H(P_{11}, P_{12})$$

$$P_9 = H(P_1, P_2) \quad P_{10} = H(P_3, P_4) \quad P_{11} = H(P_5, P_6) \quad P_{12} = H(P_7, P_8)$$

$$S_1 \to P_1 \quad S_2 \to P_2 \quad S_3 \to P_3 \quad S_4 \to P_4 \quad S_5 \to P_5 \quad S_6 \to P_6 \quad S_7 \to P_7 \quad S_8 \to P_8$$

$S_i \to P_i$ can be Lamport or Winternitz one-time signature system.
Each such pair (S_i, P_i) may be used only once.

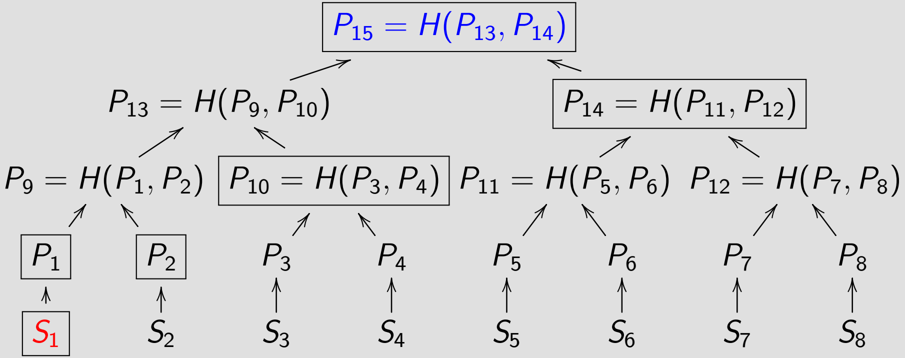
Signature in 8-time Merkle hash tree

Signature of first message: $(\texttt{sign}(m, S_1), P_1, P_2, P_{10}, P_{14})$.

$$\begin{tikzcd}
\boxed{P_{15} = H(P_{13}, P_{14})} & & & & & & & \\
P_{13} = H(P_9, P_{10}) \arrow[u] & & & & \boxed{P_{14} = H(P_{11}, P_{12})} \arrow[ullll] & & & \\
P_9 = H(P_1, P_2) \arrow[u] & & \boxed{P_{10} = H(P_3, P_4)} \arrow[ull] & & P_{11} = H(P_5, P_6) \arrow[u] & & P_{12} = H(P_7, P_8) \arrow[ull] & \\
\boxed{P_1} \arrow[u] & \boxed{P_2} \arrow[ul] & P_3 \arrow[u] & P_4 \arrow[ul] & P_5 \arrow[u] & P_6 \arrow[ul] & P_7 \arrow[u] & P_8 \arrow[ul] \\
\boxed{S_1} \arrow[u] & S_2 \arrow[u] & S_3 \arrow[u] & S_4 \arrow[u] & S_5 \arrow[u] & S_6 \arrow[u] & S_7 \arrow[u] & S_8 \arrow[u]
\end{tikzcd}$$

Signature in 8-time Merkle hash tree

Signature of first message: $(\texttt{sign}(m, S_1), P_1, P_2, P_{10}, P_{14})$.



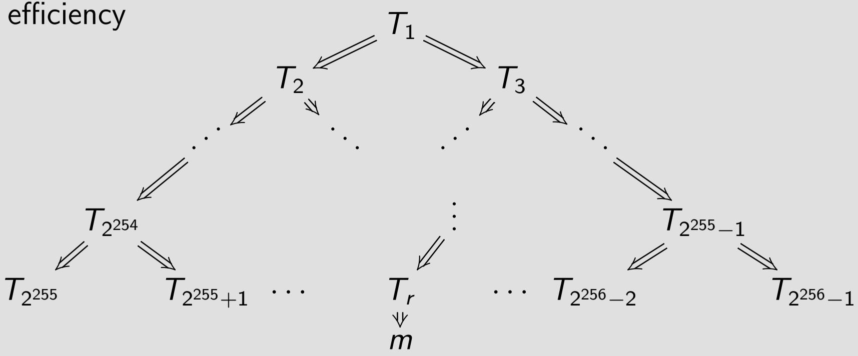
Verify signature $\texttt{sign}(m, S_1)$ with public key P_1 (provided in signature).
Link P_1 against public key P_{15} by computing $P'_9 = H(P_1, P_2)$, $P'_{13} = H(P'_9, P_{10})$, and comparing $H(P'_{13}, P_{14})$ with P_{15}. Reject if $H(P'_{13}, P_{14}) \neq P_{15}$.

Huge trees (1987 Goldreich), keys on demand (Levin)

Signer chooses random $r \in \{2^{255}, 2^{255}+1, \ldots, 2^{256}-1\}$, uses one-time public key T_r to sign m; uses one-time public key T_i to sign (T_{2i}, T_{2i+1}) on path to T_1.

Generates ith secret key deterministically as $H_k(i)$ where k is master secret.

Important for efficiency



Huge trees (1987 Goldreich), keys on demand (Levin)

Signer chooses random $r \in \{2^{255}, 2^{255}+1, \ldots, 2^{256}-1\}$, uses one-time public key T_r to sign m; uses one-time public key T_i to sign (T_{2i}, T_{2i+1}) on path to T_1.

Generates ith secret key deterministically as $H_k(i)$ where k is master secret.

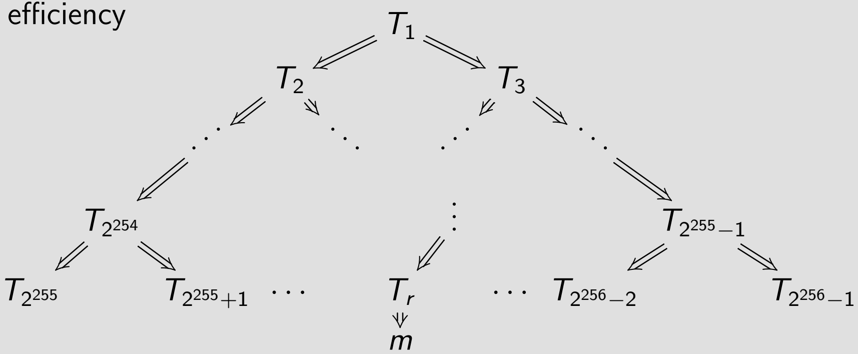
Important for efficiency

$$
\begin{tikzcd}
 & & & T_1 \arrow[dl, Rightarrow] \arrow[dr, Rightarrow] & & & \\
 & & T_2 \arrow[dl, Rightarrow] \arrow[d, Rightarrow] & & T_3 \arrow[d, Rightarrow] \arrow[dr, Rightarrow] & & \\
 & \iddots \arrow[d, Rightarrow] & \ddots & & \iddots & \ddots \arrow[d, Rightarrow] & \\
 & T_{2^{254}} \arrow[dl, Rightarrow] \arrow[d, Rightarrow] & & \vdots \arrow[d, Rightarrow] & & T_{2^{255}-1} \arrow[d, Rightarrow] \arrow[dr, Rightarrow] & \\
T_{2^{255}} & T_{2^{255}+1} & \cdots & T_r \arrow[d, Rightarrow] & \cdots & T_{2^{256}-2} & T_{2^{256}-1} \\
 & & & m & & &
\end{tikzcd}
$$

T_i for small i gets used repeatedly (each time an m falls in that sub-tree)

Huge trees (1987 Goldreich), keys on demand (Levin)

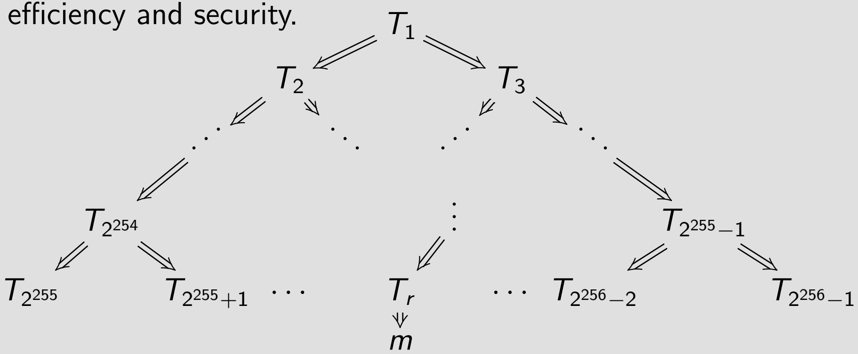
Signer chooses random $r \in \{2^{255}, 2^{255}+1, \ldots, 2^{256}-1\}$, uses one-time public key T_r to sign m; uses one-time public key T_i to sign (T_{2i}, T_{2i+1}) on path to T_1.
Generates ith secret key deterministically as $H_k(i)$ where k is master secret.
Important for efficiency



T_i for small i gets used repeatedly (each time an m falls in that sub-tree) but $H_k(i)$ being deterministic means it signs the same value, so no break.

Huge trees (1987 Goldreich), keys on demand (Levin)

Signer chooses random $r \in \{2^{255}, 2^{255}+1, \ldots, 2^{256}-1\}$, uses one-time public key T_r to sign m; uses one-time public key T_i to sign (T_{2i}, T_{2i+1}) on path to T_1.
Generates ith secret key deterministically as $H_k(i)$ where k is master secret.
Important for efficiency and security.



T_i for small i gets used repeatedly (each time an m falls in that sub-tree) but $H_k(i)$ being deterministic means it signs the same value, so no break.

NIST submission SPHINCS+

- Post-quantum signature based on hash functions.
- Requires only a secure hash function, no further assumptions.
- Based on ideas of Lamport (1979) and Merkle (1979).
- Developed starting from SPHINCS with
  - improve multi-signature,
  - smaller keys,
  - Option for shorter signatures (30kB instead of 41kB) if "only" 2^{50} messages signed.
- Three versions (using different hash functions)
  - SPHINCS+-SHA3 (with SHAKE256),
  - SPHINCS+-SHA2 (with SHA-256),
  - SPHINCS+-Haraka (with Haraka, a hash function for short inputs).

More info at `https://sphincs.org/`.

See also my course page for more detailed videos and slides.