Post-quantum cryptography

Tanja Lange
slides jointly with Daniel J. Bernstein

Technische Universiteit Eindhoven

PQCRYPTO
ICT-645622

17 September 2017

ASCrypto Summer School:

Cryptographic applications in daily life

Most applications happen behind the scenes, no activation needed by user.
Cryptography is essential in obtaining security against attacks and impersonation.

- Mobile phones connecting to cell towers.
- Credit cards, EC-cards, access codes for banks.
- Electronic passports; soon ID cards.
- Internet commerce, online tax declarations, webmail.
- Facebook, Gmail, WhatsApp, iMessage on iPhone.
- Any webpage with `https`.
- Encrypted file system on iPhone: see Apple vs. FBI.

Cryptographic applications in daily life

Most applications happen behind the scenes, no activation needed by user.
Cryptography is essential in obtaining security against attacks and impersonation.

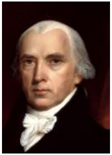
- Mobile phones connecting to cell towers.
- Credit cards, EC-cards, access codes for banks.
- Electronic passports; soon ID cards.
- Internet commerce, online tax declarations, webmail.
- Facebook, Gmail, WhatsApp, iMessage on iPhone.
- Any webpage with `https`.
- Encrypted file system on iPhone: see Apple vs. FBI.
- PGP encrypted email, Signal, Tor, Qubes OS, Subgraph OS, Tails.

Cryptography

- Motivation #1: Communication channels are spying on our data.
- Motivation #2: Communication channels are modifying our data.

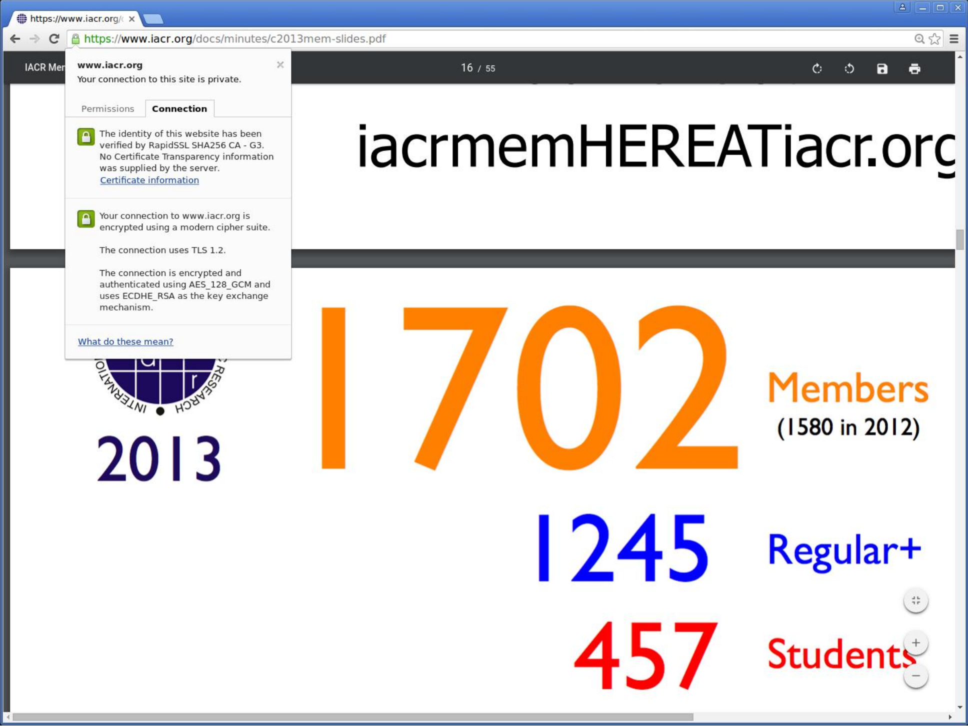
Sender
"Alice"

Untrustworthy network
"Eve"



Receiver
"Bob"

- Literal meaning of cryptography: "secret writing".
- Achieves various security goals by secretly transforming messages.


https://www.iacr.org/docs/minutes/c2013mem-slides.pdf
IACR Mem
16 / 55
www.iacr.org
Your connection to this site is private.
Permissions
Connection
The identity of this website has been verified by RapidSSL SHA256 CA - G3. No Certificate Transparency information was supplied by the server.
Certificate information
Your connection to www.iacr.org is encrypted using a modern cipher suite.
The connection uses TLS 1.2.
The connection is encrypted and authenticated using AES_128_GCM and uses ECDHE_RSA as the key exchange mechanism.
What do these mean?
iacrmemHEREATiacr.org
2013
1702
Members
(1580 in 2012)
1245
Regular+
457
Students


https://www.iacr.org/docs/minutes/c2013mem-slides.pdf
IACR Mer
16 /
www.iacr.org
Your connection to this site is private.
Permissions
Connection
The identity of this website has been verified by RapidSSL SHA256 CA - G3. No Certificate Transparency information was supplied by the server.
Certificate information
Your connection to www.iacr.org is encrypted using a modern cipher suite.
The connection uses TLS 1.2.
The connection is encrypted and authenticated using AES_128_GCM and uses ECDHE_RSA as the key exchange mechanism.
What do these mean?
iacrm

Secret-key encryption

- Prerequisite: Alice and Bob share a secret key .
- Prerequisite: Eve doesn't know .
- Alice and Bob exchange any number of messages.
- Security goal #1: **Confidentiality** despite Eve's espionage.

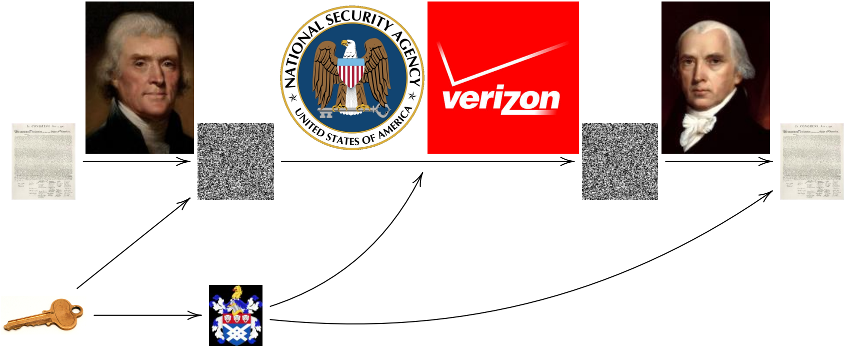
Secret-key authenticated encryption

- Prerequisite: Alice and Bob share a secret key .
- Prerequisite: Eve doesn't know .
- Alice and Bob exchange any number of messages.
- Security goal #1: **Confidentiality** despite Eve's espionage.
- Security goal #2: **Integrity**, i.e., recognizing Eve's sabotage.

Secret-key authenticated encryption

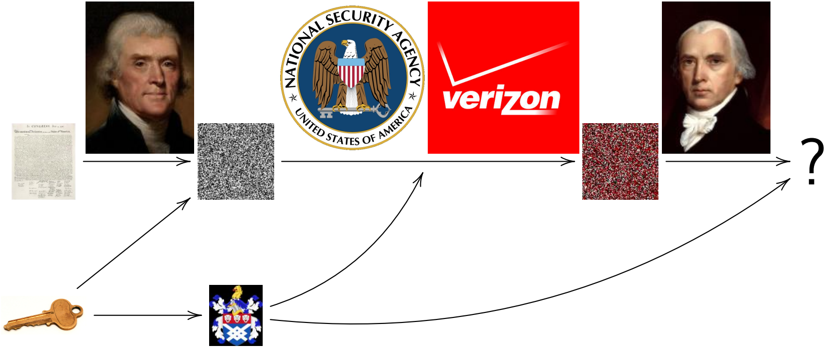
- Prerequisite: Alice and Bob share a secret key .
- Prerequisite: Eve doesn't know .
- Alice and Bob exchange any number of messages.
- Security goal #1: **Confidentiality** despite Eve's espionage.
- Security goal #2: **Integrity**, i.e., recognizing Eve's sabotage.

Public-key signatures



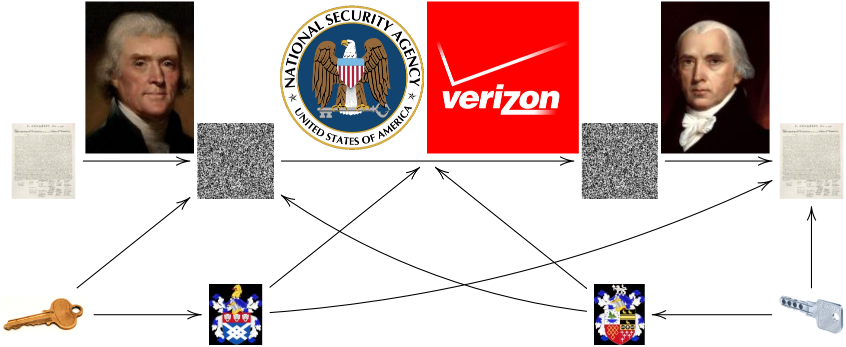
- Prerequisite: Alice has a secret key  and public key .
- Prerequisite: Eve doesn't know . Everyone knows .
- Alice publishes any number of messages.
- Security goal: Integrity.

Public-key signatures



- Prerequisite: Alice has a secret key  and public key .
- Prerequisite: Eve doesn't know . Everyone knows .
- Alice publishes any number of messages.
- Security goal: Integrity.

Public-key authenticated encryption ("DH" data flow)



- Prerequisite: Alice has a secret key  and public key .
- Prerequisite: Bob has a secret key  and public key .
- Alice and Bob exchange any number of messages.
- Security goal #1: Confidentiality.
- Security goal #2: Integrity.

Many more security goals studied in cryptography

- Protecting against denial of service.
- Stopping traffic analysis.
- Securely tallying votes.
- Searching encrypted data.
- Much more.

Cryptographic tools

Many factors influence the security and privacy of data:

- Secure storage, physical security; access control.
- Protection against alteration of data
  $\Rightarrow$ public-key signatures, message-authentication codes.
- Protection of sensitive content against reading
  $\Rightarrow$ encryption.

Currently used crypto (check the lock icon in your browser) starts with RSA, Diffie-Hellman (DH) in finite fields, or elliptic curve DH, followed by AES or ChaCha20.

Internet currently moving over to Curve25519 (Bernstein) and Ed25519 (Bernstein, Duif, Lange, Schwabe, and Yang).

Security is getting better. Some obstacles: bugs; untrustworthy hardware;

Cryptographic tools

Many factors influence the security and privacy of data:

- Secure storage, physical security; access control.
- Protection against alteration of data
  $\Rightarrow$ public-key signatures, message-authentication codes.
- Protection of sensitive content against reading
  $\Rightarrow$ encryption.

Currently used crypto (check the lock icon in your browser) starts with RSA, Diffie-Hellman (DH) in finite fields, or elliptic curve DH, followed by AES or ChaCha20.

Internet currently moving over to Curve25519 (Bernstein) and Ed25519 (Bernstein, Duif, Lange, Schwabe, and Yang).

Security is getting better. Some obstacles: bugs; untrustworthy hardware; let alone anti-security measures such as backdoors or restrictions on use.

Algorithms for Quantum Computation: Discrete Logarithms and Factoring

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

Abstract

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

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

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


D:Wave

D-Wave quantum computer isn't universal . . .

- Can't store stable qubits.
- Can't perform basic qubit operations.
- Can't run Shor's algorithm.
- Can't run other quantum algorithms we care about.

D-Wave quantum computer isn't universal . . .

- Can't store stable qubits.
- Can't perform basic qubit operations.
- Can't run Shor's algorithm.
- Can't run other quantum algorithms we care about.
- Hasn't managed to find any computation justifying its price.
- Hasn't managed to find any computation justifying 1% of its price.

. . . but universal quantum computers are coming . . .

- Massive research effort. Tons of progress summarized in, e.g., https://en.wikipedia.org/wiki/Timeline_of_quantum_computing.

. . . but universal quantum computers are coming . . .

- Massive research effort. Tons of progress summarized in, e.g., https://en.wikipedia.org/wiki/Timeline_of_quantum_computing.
- Mark Ketchen, IBM Research, 2012, on quantum computing: "We're actually doing things that are making us think like, 'hey this isn't 50 years off, this is maybe just 10 years off, or 15 years off.' It's within reach."
- Fast-forward to 2022, or 2027. Universal quantum computers exist.

... but universal quantum computers are coming ...

- Massive research effort. Tons of progress summarized in, e.g., https://en.wikipedia.org/wiki/Timeline_of_quantum_computing.
- Mark Ketchen, IBM Research, 2012, on quantum computing: "We're actually doing things that are making us think like, 'hey this isn't 50 years off, this is maybe just 10 years off, or 15 years off.' It's within reach."
- Fast-forward to 2022, or 2027. Universal quantum computers exist.
- Shor's algorithm solves in polynomial time:
  - Integer factorization. RSA is dead.
  - The discrete-logarithm problem in finite fields. DSA is dead.
  - The discrete-logarithm problem on elliptic curves. ECDSA is dead.
- This breaks all current public-key cryptography on the Internet!

. . . but universal quantum computers are coming . . .

- Massive research effort. Tons of progress summarized in, e.g., https://en.wikipedia.org/wiki/Timeline_of_quantum_computing.
- Mark Ketchen, IBM Research, 2012, on quantum computing: "We're actually doing things that are making us think like, 'hey this isn't 50 years off, this is maybe just 10 years off, or 15 years off.' It's within reach."
- Fast-forward to 2022, or 2027. Universal quantum computers exist.
- Shor's algorithm solves in polynomial time:
  - Integer factorization. RSA is dead.
  - The discrete-logarithm problem in finite fields. DSA is dead.
  - The discrete-logarithm problem on elliptic curves. ECDSA is dead.
- This breaks all current public-key cryptography on the Internet!
- Also, Grover's algorithm speeds up brute-force searches.
- Example: Only 2^{64} quantum operations to break AES-128;
  2^{128} quantum operations to break AES-256.

Physical cryptography: a return to the dark ages

- Example: Locked briefcases.
- One-time pad is information-theoretically secure, i.e. no computational assumptions.
- Horrendously expensive.
- Can call it "locked-briefcase cryptography" but it's much more expensive than normal crypto.

Physical cryptography: a return to the dark ages

- Example: Locked briefcases.
- One-time pad is information-theoretically secure, i.e. no computational assumptions.
- Horrendously expensive.
- Can call it "locked-briefcase cryptography" but it's much more expensive than normal crypto.
- Broken again and again. Much worse track record than normal crypto.
- Easy to screw up. Easy to backdoor. Hard to audit.

Physical cryptography: a return to the dark ages

- Example: Locked briefcases.
- One-time pad is information-theoretically secure,
  i.e. no computational assumptions.
- Horrendously expensive.
- Can call it "locked-briefcase cryptography"
  but it's much more expensive than normal crypto.
- Broken again and again.
  Much worse track record than normal crypto.
- Easy to screw up. Easy to backdoor. Hard to audit.
- Very limited functionality: e.g., no public-key signatures.

Security advantages of algorithmic cryptography

- Keep secrets heavily shielded inside authorized computers.
- Reduce trust in third parties:
  - Reduce reliance on closed-source software and hardware.
  - Increase comprehensiveness of audits.
  - Increase comprehensiveness of formal verification.
  - Design systems to be secure even if **algorithm and public keys are public**.

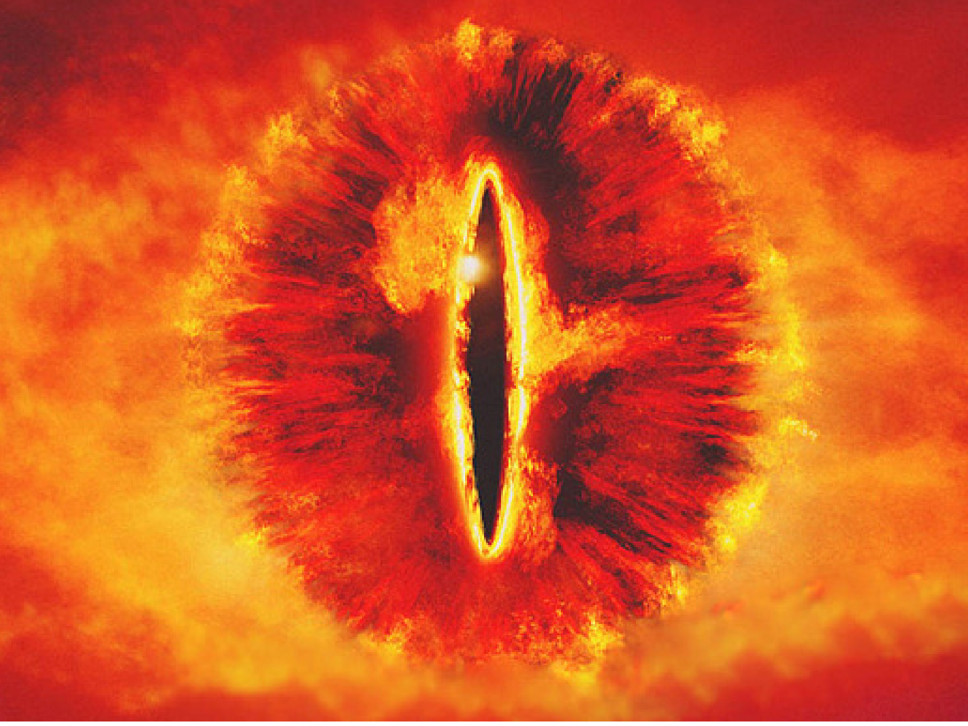
    Critical example: **signed** software updates.
- Understand security as thoroughly as possible:
  - Publish comprehensive specifications.
  - Build large research community with clear security goals.
  - Publicly document attack efforts.
  - Require systems to convincingly survive many years of analysis.

History of post-quantum cryptography

- 2003 Daniel J. Bernstein introduces term Post- quantum cryptography.
- PQCrypto 2006: International Workshop on Post-Quantum Cryptography.

History of post-quantum cryptography

- 2003 Daniel J. Bernstein introduces term Post- quantum cryptography.
- PQCrypto 2006: International Workshop on Post-Quantum Cryptography.
- PQCrypto 2008, PQCrypto 2010, PQCrypto 2011, PQCrypto 2013.
- 2014 EU publishes H2020 call including post-quantum crypto as topic.
- ETSI working group on "Quantum-safe" crypto.
- PQCrypto 2014.
- April 2015 NIST hosts first workshop on post-quantum cryptography
- August 2015 NSA (US National Security Agency) wakes up



NSA announcements

August 11, 2015

> *IAD recognizes that there will be a move, in the not distant future, to a quantum resistant algorithm suite.*

NSA announcements

August 11, 2015

> *IAD recognizes that there will be a move, in the not distant future, to a quantum resistant algorithm suite.*

August 19, 2015

> *IAD will initiate a transition to quantum resistant algorithms in the not too distant future.*

NSA comes late to the party and botches its grand entrance.

NSA announcements

August 11, 2015

> *IAD recognizes that there will be a move, in the not distant future, to a quantum resistant algorithm suite.*

August 19, 2015

> *IAD will initiate a transition to quantum resistant algorithms in the not too distant future.*

NSA comes late to the party and botches its grand entrance.

Worse, now we get people saying “Don’t use post-quantum crypto, the NSA wants you to use it!”.

NSA announcements

August 11, 2015

> *IAD recognizes that there will be a move, in the not distant future, to a quantum resistant algorithm suite.*

August 19, 2015

> *IAD will initiate a transition to quantum resistant algorithms in the not too distant future.*

NSA comes late to the party and botches its grand entrance.

Worse, now we get people saying "Don't use post-quantum crypto, the NSA wants you to use it!". Or "NSA says NIST P-384 is post-quantum secure".

NSA announcements

August 11, 2015

> *IAD recognizes that there will be a move, in the not distant future, to a quantum resistant algorithm suite.*

August 19, 2015

> *IAD will initiate a transition to quantum resistant algorithms in the not too distant future.*

NSA comes late to the party and botches its grand entrance.

Worse, now we get people saying "Don't use post-quantum crypto, the NSA wants you to use it!". Or "NSA says NIST P-384 is post-quantum secure". Or "NSA has abandoned ECC."

Post-quantum becoming mainstream

- PQCrypto 2016: 22–26 Feb in Fukuoka, Japan, with more than 200 participants

- PQCrypto 2017 took place last June in Utrecht, Netherlands, again more than 200 participants.
- NIST is calling for post-quantum proposals: 5–7 year competition.

Confidence-inspiring crypto takes time to build

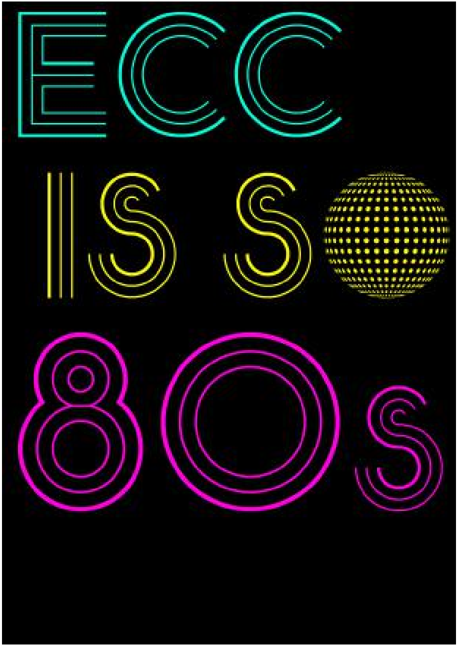
- Many stages of research from cryptographic design to deployment:
  - Explore space of cryptosystems.
  - Study algorithms for the attackers.
  - Focus on secure cryptosystems.

Confidence-inspiring crypto takes time to build

- Many stages of research from cryptographic design to deployment:
  - Explore space of cryptosystems.
  - Study algorithms for the attackers.
  - Focus on secure cryptosystems.
  - Study algorithms for the users.
  - Study implementations on real hardware.
  - Study side-channel attacks, fault attacks, etc.
  - Focus on secure, reliable implementations.
  - Focus on implementations meeting performance requirements.
  - Integrate securely into real-world applications.

Confidence-inspiring crypto takes time to build

- Many stages of research from cryptographic design to deployment:
  - Explore space of cryptosystems.
  - Study algorithms for the attackers.
  - Focus on secure cryptosystems.
  - Study algorithms for the users.
  - Study implementations on real hardware.
  - Study side-channel attacks, fault attacks, etc.
  - Focus on secure, reliable implementations.
  - Focus on implementations meeting performance requirements.
  - Integrate securely into real-world applications.
- Example: ECC introduced **1985**; big advantages over RSA. Robust ECC started to take over the Internet in **2015**.
- Can't wait for quantum computers before finding a solution!


ECC
IS SO
80s

Even higher urgency for long-term confidentiality

- Attacker can break currentlyused encryption (ECC, RSA) with a quantum cmputer.
- Even worse, today's encrypted communication is being stored by attackers and will be decrypted years later with quantum computers. All data can be recovered in clear from recording traffic and breaking the public key scheme.
- How many years do you want to keep your data secret? From whom?

- Signature schemes can be replaced once a quantum computer is built – but there will not be a public announcement

Even higher urgency for long-term confidentiality

- Attacker can break currentlyused encryption (ECC, RSA) with a quantum cmputer.
- Even worse, today's encrypted communication is being stored by attackers and will be decrypted years later with quantum computers. All data can be recovered in clear from recording traffic and breaking the public key scheme.
- How many years do you want to keep your data secret? From whom?

- Signature schemes can be replaced once a quantum computer is built – but there will not be a public announcement . . . and an important function of signatures is to protect operating system upgrades.
- Protect your upgrades *now* with post-quantum signatures.

Initial recommendations of long-term secure post-quantum systems

Daniel Augot, Lejla Batina, Daniel J. Bernstein, Joppe Bos,
Johannes Buchmann, Wouter Castryck, Orr Dunkelman,
Tim Güneysu, Shay Gueron, Andreas Hülsing,
Tanja Lange, Mohamed Saied Emam Mohamed,
Christian Rechberger, Peter Schwabe, Nicolas Sendrier,
Frederik Vercauteren, Bo-Yin Yang

Initial recommendations

- **Symmetric encryption** Thoroughly analyzed, 256-bit keys:
  - AES-256
  - Salsa20 with a 256-bit key

  Evaluating: Serpent-256, ...
- **Symmetric authentication** Information-theoretic MACs:
  - GCM using a 96-bit nonce and a 128-bit authenticator
  - Poly1305
- **Public-key encryption** McEliece with binary Goppa codes:
  - length $n = 6960$, dimension $k = 5413$, $t = 119$ errors

  Evaluating: QC-MDPC, Stehlé-Steinfeld NTRU, ...
- **Public-key signatures** Hash-based (minimal assumptions):
  - XMSS with any of the parameters specified in CFRG draft
  - SPHINCS-256

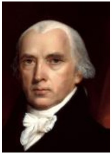
  Evaluating: HFEv-, ...

Systems expected to survive

- Code-based crypto, see next talk
- Hash-based signatures, see next talk
- Isogeny-based crypto: new kid on the block, promising short keys and key exchange without communication (static-static) as possibility; needs more reserach on security.
- Lattice-based crypto
- Multivariate crypto
- Symmetric crypto.

Maybe some more, maybe some less.

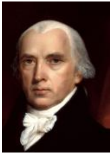
Post-quantum secret-key authenticated encryption



$$m \xrightarrow[k]{} c \longrightarrow c \xrightarrow[k]{} m$$

- Very easy solutions if secret key k is long uniform random string:
  - "One-time pad" for encryption.
  - "Wegman–Carter MAC" for authentication.
- AES-256: Standardized method to expand 256-bit k into string indistinguishable from long k.
- AES introduced in 1998 by Daemen and Rijmen. Security analyzed in papers by dozens of cryptanalysts.
- No credible threat from quantum algorithms. Grover costs 2^{128}.
- Some recent results assume attacker has quantum access to compuation, then some systems are weaker

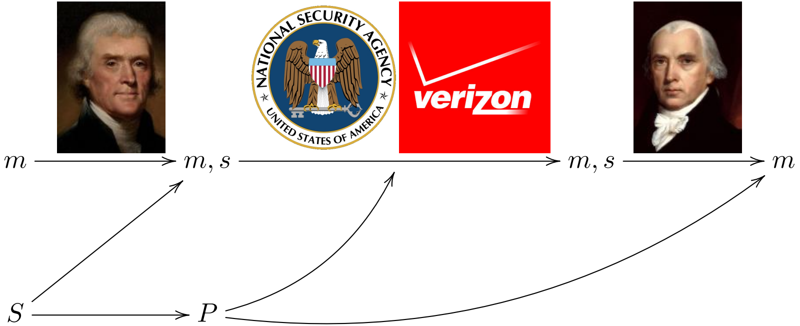
Post-quantum secret-key authenticated encryption



$$m \xrightarrow[k]{} c \longrightarrow c \xrightarrow[k]{} m$$

- Very easy solutions if secret key k is long uniform random string:
  - "One-time pad" for encryption.
  - "Wegman–Carter MAC" for authentication.
- AES-256: Standardized method to expand 256-bit k into string indistinguishable from long k.
- AES introduced in 1998 by Daemen and Rijmen. Security analyzed in papers by dozens of cryptanalysts.
- No credible threat from quantum algorithms. Grover costs 2^{128}.
- Some recent results assume attacker has quantum access to compuation, then some systems are weaker . . . but I'd know if my laptop had turned into a quantum computer.

Post-quantum public-key signatures: hash-based



- Secret key S, public key P.
- Only one prerequisite: a good hash function, e.g. SHA3-512, ...
  Hash functions map long strings to fixed-length strings.
  Signature schemes use hash functions in handling m.
- Old idea: 1979 Lamport one-time signatures.
- 1979 Merkle extends to more signatures.
- Many further improvements.
- Security thoroughly analyzed.

A signature scheme for 1-bit messages: key generation, signing

KeyGen:

- Pick random $s_0, s_1 \in \{0,1\}^{256}$
  Secret key is $S = (s_0, s_1)$.
- Compute public key $P = (p_0, p_1) = (H(s_0), H(s_1))$,
  where H is a cryptographic hash function.

A signature scheme for 1-bit messages: key generation, signing

KeyGen:

- Pick random $s_0, s_1 \in \{0,1\}^{256}$
  Secret key is $S = (s_0, s_1)$.
- Compute public key $P = (p_0, p_1) = (H(s_0), H(s_1))$,
  where H is a cryptographic hash function.

Sign:

- To sign bit b, i.e., to sign 0 or 1,
  send s_b.

A signature scheme for 1-bit messages: key generation, signing

KeyGen:

- Pick random $s_0, s_1 \in \{0,1\}^{256}$
  Secret key is $S = (s_0, s_1)$.
- Compute public key $P = (p_0, p_1) = (H(s_0), H(s_1))$,
  where H is a cryptographic hash function.

Sign:

- To sign bit b, i.e., to sign 0 or 1,
  send s_b.

Verify:

- To verify signature s on message b check that
  $H(s) = p_b$ using the public key.

A signature scheme for $\underline{4\text{-bit}}$ messages: key generation

KeyGen:

- Pick 4 pairs of random $s_{i0}, s_{i1} \in \{0,1\}^{256}$
  Secret key is $S = (s_{00}, s_{01}, s_{10}, s_{11}, s_{20}, s_{21}, s_{30}, s_{31})$.
- Compute public key $P =$
  $(H(s_{00}), H(s_{01}), H(s_{10}), H(s_{11}), H(s_{20}), H(s_{21}), H(s_{30}), H(s_{31}))$.

A signature scheme for 4-bit messages: key generation

KeyGen:

- Pick 4 pairs of random $s_{i0}, s_{i1} \in \{0,1\}^{256}$
  Secret key is $S = (s_{00}, s_{01}, s_{10}, s_{11}, s_{20}, s_{21}, s_{30}, s_{31})$.
- Compute public key $P =$
  $(H(s_{00}), H(s_{01}), H(s_{10}), H(s_{11}), H(s_{20}), H(s_{21}), H(s_{30}), H(s_{31}))$.

Sign:

- To sign message b_0, b_1, b_2, b_3 send $s = (s_{0b_0}, s_{1b_1}, s_{2b_2}, s_{3b_3})$.

A signature scheme for 4-bit messages: key generation

KeyGen:

- Pick 4 pairs of random $s_{i0}, s_{i1} \in \{0,1\}^{256}$
  Secret key is $S = (s_{00}, s_{01}, s_{10}, s_{11}, s_{20}, s_{21}, s_{30}, s_{31})$.
- Compute public key $P =$
  $(H(s_{00}), H(s_{01}), H(s_{10}), H(s_{11}), H(s_{20}), H(s_{21}), H(s_{30}), H(s_{31}))$.

Sign:

- To sign message b_0, b_1, b_2, b_3 send $s = (s_{0b_0}, s_{1b_1}, s_{2b_2}, s_{3b_3})$.

Verify:

- To verify signature $s = (s_0, s_1, s_2, s_3)$ on message b_0, b_1, b_2, b_3 check that
  $H(s_i) = p_{ib_i}$ using the public key.

Lamport's 1-time signature system

- Scale up to 256-bit messages.
  Secret and public key now consist of 2×256 strings of 256 bits each.
- Sign arbitrary-length message by signing its 256-bit hash

$$H(m) = (b_0, b_1, \ldots, b_{255}).$$

- Attention: This is called a one-time signature for a reason!
  Given
  signature $(s_{00}, s_{11}, s_{20}, s_{30})$ on $(0, 1, 0, 0)$ and
  signature $(s_{00}, s_{10}, s_{21}, s_{31})$ on $(0, 0, 1, 1)$
  we can combine them to sign,

Lamport's 1-time signature system

- Scale up to 256-bit messages.
  Secret and public key now consist of 2×256 strings of 256 bits each.
- Sign arbitrary-length message by signing its 256-bit hash

$$H(m) = (b_0, b_1, \ldots, b_{255}).$$

- Attention: This is called a one-time signature for a reason!
  Given
  signature $(s_{00}, s_{11}, s_{20}, s_{30})$ on $(0, 1, 0, 0)$ and
  signature $(s_{00}, s_{10}, s_{21}, s_{31})$ on $(0, 0, 1, 1)$
  we can combine them to sign, e.g., $(0, 0, 0, 0)$, ...
- Space improvement: "Winternitz signatures".

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

$$
\begin{tikzcd}[column sep=tiny]
S_1 \arrow[d] & & S_2 \arrow[d] & & S_3 \arrow[d] & & S_4 \arrow[d] & & S_5 \arrow[d] & & S_6 \arrow[d] & & S_7 \arrow[d] & & S_8 \arrow[d] \\
P_1 \arrow[dr] & & P_2 \arrow[dl] & & P_3 \arrow[dr] & & P_4 \arrow[dl] & & P_5 \arrow[dr] & & P_6 \arrow[dl] & & P_7 \arrow[dr] & & P_8 \arrow[dl] \\
& P_9 = H(P_1, P_2) \arrow[drr] & & & & P_{10} = H(P_3, P_4) \arrow[dll] & & & & P_{11} = H(P_5, P_6) \arrow[drr] & & & & P_{12} = H(P_7, P_8) \arrow[dll] & \\
& & & P_{13} = H(P_9, P_{10}) \arrow[drrrr] & & & & & & & & P_{14} = H(P_{11}, P_{12}) \arrow[dllll] & & & \\
& & & & & & & {\color{blue} P_{15} = H(P_{13}, P_{14})} & & & & & & &
\end{tikzcd}
$$

Eight Lamport one-time keys $P_1, P_2, \ldots, P_8$ with corresponding $S_1, S_2, \ldots, S_8$ at leaves of tree.
Merkle public key is P_{15} at root of tree.

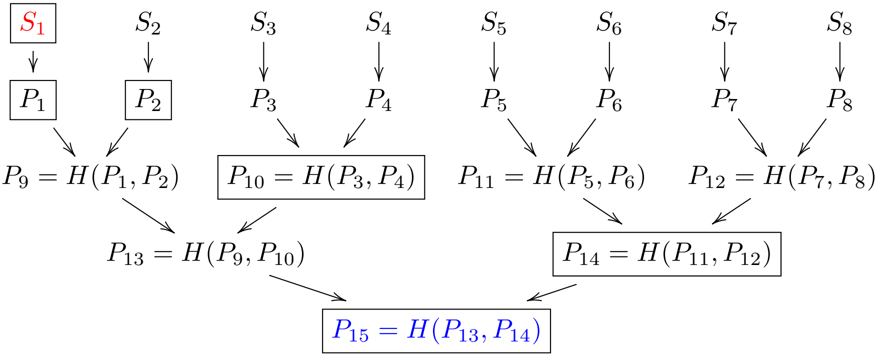
Signature in 8-time Merkle hash tree

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

$$\begin{tikzcd}
\boxed{{\color{red}S_1}} \arrow[d] & & S_2 \arrow[d] & & S_3 \arrow[d] & & S_4 \arrow[d] & & S_5 \arrow[d] & & S_6 \arrow[d] & & S_7 \arrow[d] & & S_8 \arrow[d] \\
\boxed{P_1} \arrow[dr] & & \boxed{P_2} \arrow[dl] & & P_3 \arrow[dr] & & P_4 \arrow[dl] & & P_5 \arrow[dr] & & P_6 \arrow[dl] & & P_7 \arrow[dr] & & P_8 \arrow[dl] \\
& P_9 = H(P_1, P_2) \arrow[drr] & & & & \boxed{P_{10} = H(P_3, P_4)} \arrow[dll] & & & & P_{11} = H(P_5, P_6) \arrow[drr] & & & & P_{12} = H(P_7, P_8) \arrow[dll] & \\
& & & P_{13} = H(P_9, P_{10}) \arrow[drrrr] & & & & & & & & \boxed{P_{14} = H(P_{11}, P_{12})} \arrow[dllll] & & & \\
& & & & & & & \boxed{{\color{blue}P_{15} = H(P_{13}, P_{14})}} & & & & & & &
\end{tikzcd}$$

Signature in 8-time Merkle hash tree

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



Verify by checking one-time signature $\mathtt{sign}(m, S_1)$ on m against P_1.
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}.

Pros and cons

Pros:

- Post quantum
- Only need secure hash function
- Small public key
- Security well understood
- Fast
- Proposed for standards: https://tools.ietf.org/html/draft-irtf-cfrg-xmss-hash-based-signatures-XX

[Docs] [txt|pdf|xml|html] [Tracker] [WG] [Email] [Diff1] [Diff2] [Nits]

Versions: (draft-huelsing-cfrg-hash-sig-xmss)
00 01

```
Crypto Forum Research Group                                  A. Huelsing
Internet-Draft                                            TU Eindhoven
Intended status: Informational                                D. Butin
Expires: January 4, 2016                                  TU Darmstadt
                                                             S. Gazdag
                                                            genua GmbH
                                                           A. Mohaisen
                                                         Verisign Labs
                                                          July 3, 2015
```

XMSS: Extended Hash-Based Signatures
draft-irtf-cfrg-xmss-hash-based-signatures-01

Abstract

This note describes the eXtended Merkle Signature Scheme (XMSS), a hash-based digital signature system. It follows existing descriptions in scientific literature. The note specifies the WOTS+ one-time signature scheme, a single-tree (XMSS) and a multi-tree variant (XMSS^MT) of XMSS. Both variants use WOTS+ as a main building block. XMSS provides cryptographic digital signatures without relying on the conjectured hardness of mathematical problems.

Pros and cons

```
[Docs] [txt|pdf|xml|html] [Tracker] [WG] [Email] [Diff1] [Diff2] [Nits]

Versions: (draft-huelsing-cfrg-hash-sig-xmss)
          00 01

Crypto Forum Research Group                                  A. Huelsing
Internet-Draft                                             TU Eindhoven
Intended status: Informational                                 D. Butin
Expires: January 4, 2016                                   TU Darmstadt
                                                             S. Gazdag
                                                            genua GmbH
                                                           A. Mohaisen
                                                          Verisign Labs
                                                           July 3, 2015


                 XMSS: Extended Hash-Based Signatures
              draft-irtf-cfrg-xmss-hash-based-signatures-01

Abstract

   This note describes the eXtended Merkle Signature Scheme (XMSS), a
   hash-based digital signature system.  It follows existing
   descriptions in scientific literature.  The note specifies the WOTS+
   one-time signature scheme, a single-tree (XMSS) and a multi-tree
   variant (XMSS^MT) of XMSS.  Both variants use WOTS+ as a main
   building block.  XMSS provides cryptographic digital signatures
   without relying on the conjectured hardness of mathematical problems.
```

Pros:

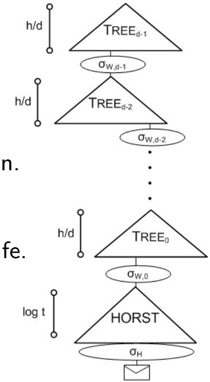
- Post quantum
- Only need secure hash function
- Small public key
- Security well understood
- Fast
- Proposed for standards: https://tools.ietf.org/html/draft-irtf-cfrg-xmss-hash-based-signatures-XX

Cons:

- Biggish signature
- Stateful
  Adam Langley "for most environments it's a huge foot-cannon."

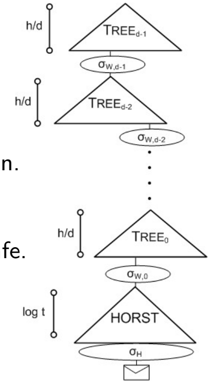
Stateless hash-based signatures

- Idea from 1987 Goldreich:
  - Signer builds huge tree of certificate authorities.
  - Signature includes certificate chain.
  - Each CA is a hash of master secret and tree position. This is deterministic, so don't need to store results.
  - **Random** bottom-level CA signs message. Many bottom-level CAs, so one-time signature is safe.

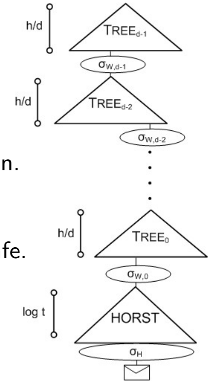


Stateless hash-based signatures

- Idea from 1987 Goldreich:
  - Signer builds huge tree of certificate authorities.
  - Signature includes certificate chain.
  - Each CA is a hash of master secret and tree position. This is deterministic, so don't need to store results.
  - **Random** bottom-level CA signs message. Many bottom-level CAs, so one-time signature is safe.
- 0.6 MB: Goldreich's signature with good 1-time signature scheme.
- 1.2 MB: average Debian package size.
- 1.8 MB: average web page in Alexa Top 1000000.



Stateless hash-based signatures



- Idea from 1987 Goldreich:
  - Signer builds huge tree of certificate authorities.
  - Signature includes certificate chain.
  - Each CA is a hash of master secret and tree position. This is deterministic, so don't need to store results.
  - **Random** bottom-level CA signs message. Many bottom-level CAs, so one-time signature is safe.
- 0.6 MB: Goldreich's signature with good 1-time signature scheme.
- 1.2 MB: average Debian package size.
- 1.8 MB: average web page in Alexa Top 1000000.
- 0.041 MB: SPHINCS signature, new optimization of Goldreich. Modular, guaranteed as strong as its components (hash, PRNG). Well-known components chosen for 2^{128} post-quantum security. sphincs.cr.yp.to

Further resources

- https://pqcrypto.org: Our survey site.
  - Many pointers: e.g., PQCrypto conference series.
  - Bibliography for 4 major PQC systemss.
- PQCrypto 2016 with slides and videos from lectures (incl. winter school)
- PQCrypto 2017 and two schools (incl. complete course on PQC on video + slides and exercises)
- https://pqcrypto.eu.org: PQCRYPTO EU project.
  - Expert recommendations.
  - Free software libraries. (Coming soon)
  - More benchmarking to compare cryptosystems. (Coming soon)
  - 2017: workshop and spring/summer school.
- https://twitter.com/pqc_eu: PQCRYPTO Twitter feed.
  - Get used to post-quantum cryptosystems.
  - Improve; implement; integrate into real-world systems.