Topics in Post-Quantum Cryptography

Andreas Hülsing

PQCRYPTO
ICT-645622

State of affairs

- Standards track
  - Stateful hash-based signatures: XMSS, LMS (Internet drafts)
  - NTRUEncrypt (IEEE Std 1363.1, X9.98)
- Hundreds of proposed schemes

Initial recommendations

2015

- **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-, ...

„Official“ developments

- Feb `13: First PQC draft in **IRTF´s CFRG**
- Sep `13: **ETSI** holds first PQC WS (afterwards annually)
- April `15: **NIST** holds conference on PQC
- Aug `15: **NSA** announces transition to PQC
- Feb `16: **NIST** announces `PQC competition‘
- Dec `16: **NIST** opens call for proposals

Scheduled:

- Nov `17: **NIST** submission deadline
- 2024: „Draft standards ready“ (**NIST**, Feb `16)

NIST Competition


NIST National Institute of Standards and Technology
Information Technology Laboratory

SEARCH: Search

CONTACT SITE MAP

Computer Security Division
Computer Security Resource Center

CSRC Home | About | Projects / Research | Publications | News & Events

Post-Quantum Cryptography Project

- Documents
- Workshops / Timeline
- Federal Register Notices
- Email Listserve
- PQC Project Contact
- Archive Information

Post-Quantum Cryptography Standardization

CSRC HOME > GROUPS > CT > POST-QUANTUM CRYPTOGRAPHY PROJECT

POST-QUANTUM CRYPTO PROJECT

NEWS -- December 15, 2016: The National Institute of Standards and Technology (NIST) is now accepting submissions for quantum-resistant public-key cryptographic algorithms. The deadline for submission is ***November 30, 2017.*** Please see the Post-Quantum Cryptography Standardization menu at left for the complete submission requirements and evaluation criteria.

In recent years, there has been a substantial amount of research on quantum computers – machines that exploit quantum mechanical phenomena to solve mathematical problems that are difficult or intractable for conventional computers. If large-scale quantum computers are ever built, they will be able to break many of the public-key cryptosystems currently in use. This would seriously compromise

NIST Competition

SPHINCS

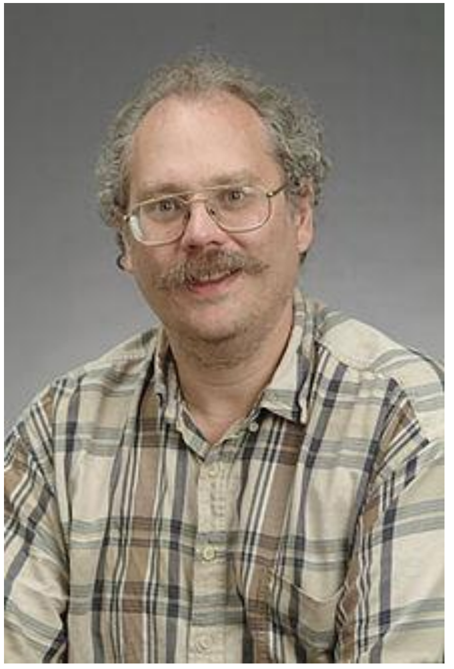
- Selection of
  - Digital signature and
  - Public key encryption / Key exchange
- Probably > 100 submissions
- No single winner
- Classically this will spark interest in cryptanalysis

Up next

(Quantum) security

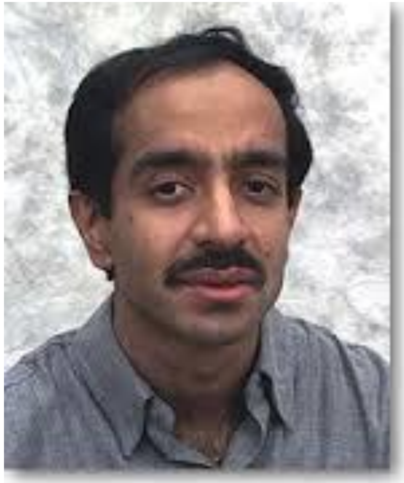
Shor‘s algorithm (1994)

- Quantum computers can do FFT very efficiently
- Can be used to find period of a function
- This can be exploited to factor efficiently (RSA)
- Shor also shows how to solve discrete log efficiently (DSA, DH, ECDSA, ECDH)



Grover‘s algorithm (1996)

- Quantum computers can search N entry DB in $\Theta(\sqrt{N})$
- Application to symmetric crypto
- Nice: Grover is provably optimal (For random function)
- Double security parameter.



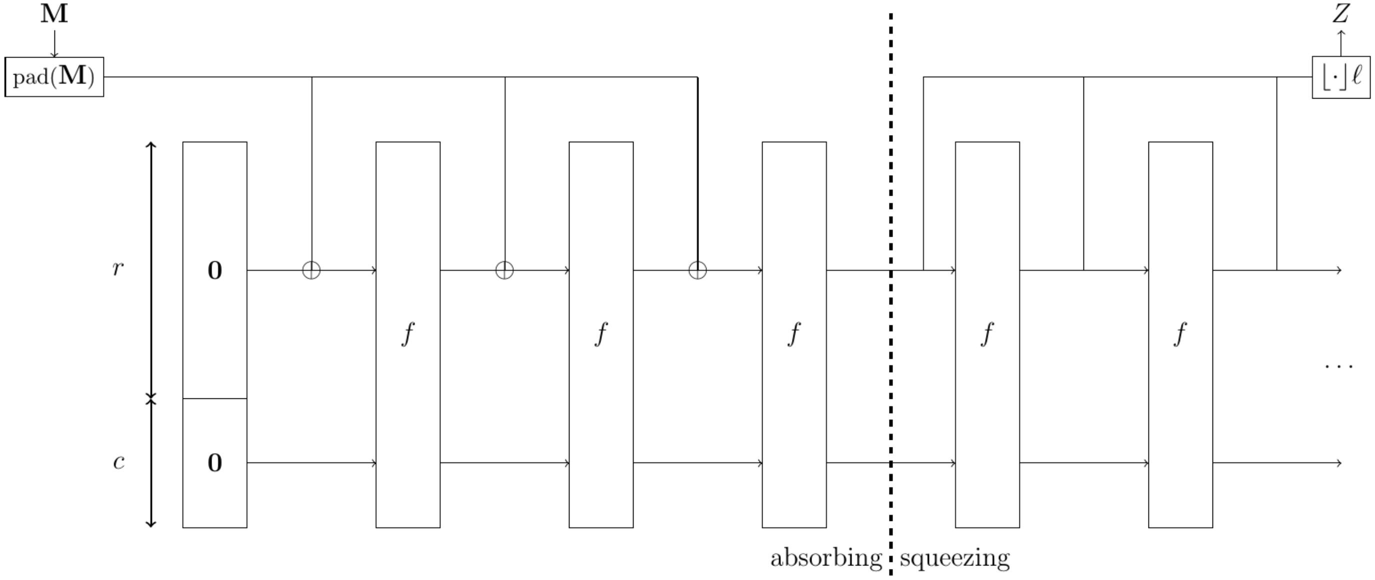
(Quantum) security

- Are attacks using Grover efficient?
- Is Grover speed-up the only thing we can get ?
  - Currently working to prove this for hash functions (under certain assumptions)
- Are the PQ problems classically secure?
- What is the exact security?

- We never had „provably secure crypto“
  - Can we classically break RSA? Who knows!

Results

- Sponges are quantum collision-resistant if block function is random function or random one-way permutation (does not cover SHA3!)



Quantum Cryptography

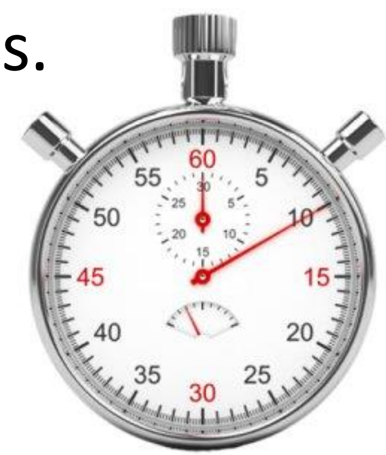
Why not beat ‘em with their own weapons?

- QKD: Quantum Key distribution.
  - Based on some nice quantum properties: entanglement & collapsing measurments
  - Information theoretic security (at least in theory) -> Great!
  - For sale today!
- So why don‘t we use this?
- Only short distance, point-to-point connections!
  - Internet? No way!
- Longer distances require „trusted-repeaters“ ☺
  - We all know where this leads...

Implementation security

Side-channels

- Implementations might leak secret information through
  - timing,
  - cache-access patterns,
  - electro-magnetic radiation,
  - power consumption...
- Not covered by standard security models.

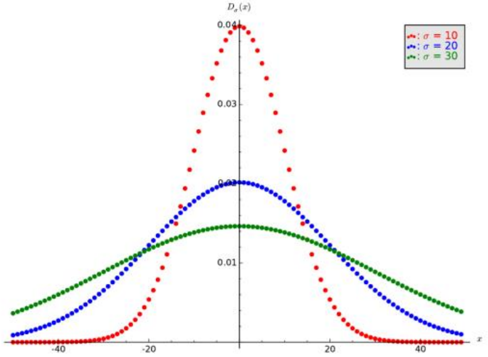


Implementation security

- Still hard for traditional schemes
- New PQ Problems come with new basic operations
- Not much research yet (for PQC)
- But a lot of experience

Discrete Gaussians

- Basic building block in lattice-based cryptography.
- Used to “hide” secret.
- Unknown how to implement efficiently in constant time.



Results

- Attack on BLISS [DDLL‘13], implemented in StrongSwan library.
- Practical cache attack on both implemented samplers.
- First algorithm to “un-hide” secret key given side-channel information for Gaussian noise.
- Can compute secret key after < 5000 signatures.

Ongoing:

- Solution: Allow for constant-time sampler by changing the distribution.

Integration

Signatures (Source https://ia.cr/2017/279)

Scheme	Gen [ms]	Sign [ms]	Verify [ms]	|sk| [bytes]	|pk| [bytes]	$\|\sigma\|$ [bytes]	Model
Fish-1-316	0.01	364.11	201.17	32	64	108013	ROM
Fish-10-38	0.01	29.73	17.46	32	64	118525	ROM
Fish-42-14	0.01	13.27	7.45	32	64	152689	ROM
Picnic-10-38	0.01	31.31	16.30	32	64	195458	QROM
MQ 5pass	0.96	7.21	5.17	32	74	40952	ROM
SPHINCS-256	0.82	13.44	0.58	1088	1056	41000	SM
BLISS-I	44.16	0.12	0.02	2048	7168	5732	ROM
Ring-TESLA*	16*k*	0.06	0.03	12288	8192	1568	ROM
TESLA-768	48*k*	0.65	0.36	3216*k*	4128*k*	2336	(Q)ROM
FS-Véron	n/a	n/a	n/a	32	160	129024	ROM
SIDHp751	16.41	7.3*k*	5.0*k*	48	768	141312	QROM

Integration

- Smaller but less conservative signature choices exist
- PKE / KEX: Sizes better
- Can your protocol fit a 40KB public key / signature?
- How to deal with immaturity of PQ Problems?
  - Combiners -> pay in size / speed

Conclusion

- A lot of important questions ahead
  - Strengthen confidence
  - Secure implementations
- All solvable but need time & money
- Might have to rethink existing protocols
  - Will not get MUCH smaller

Thank you!
Questions?