

Authenticated Key Exchange from Ring Learning with Errors

Jiang Zhang Zhenfeng Zhang Jintai Ding
Michael Snook Özgür Dagdelen

April 29, 2015

The context of our work - PQC

- Shor's quantum algorithm
- Post-quantum cryptography
 - Develop public key cryptosystems that could resist resist future quantum computer attacks

The Preparation for the Future

- The first Quantum-Safe-Crypto Workshop
26 - 27 September, 2013

ETSI – the European Telecommunications Standards Institute at SOPHIA ANTIPOLIS, FRANCE

- The second Quantum-Safe-Crypto Workshop
6 - 2 October , 2014, Ottawa, Canada
White paper

- The Quantum-Safe-Crypto Workshop at **NIST: National Institute of Standard of Technology**,
April 7-8, 2015, Washington DC

What do we really need ?– a slides of L. Chen from NIST

Practical Challenge

- ▶ Quantum computing will break many public-key cryptographic algorithms/schemes
 - Key agreement (e.g. DH and MQV)
 - Digital signatures (e.g. RSA and DSA)
 - Encryption (e.g. RSA)
- ▶ These algorithms have been used to protect Internet protocols (e.g. IPsec) and applications (e.g. TLS)
- ▶ NIST is studying “quantum-safe” replacements
- ▶ This talk will focus on practical aspects
 - For security, see Yi-Kai Liu’s talk later today

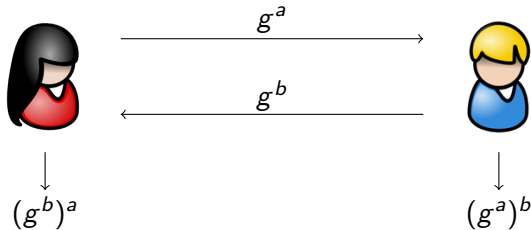
Post Quantum Needs – Functionality

- Key Exchange
- Signatures
- Authentication

Key Exchange Applications — SSL/TLS

- RSA
- Diffie–Hellman
- Our goal – replacements for post quantum world

Diffie-Hellman Key Exchange



Generalizing DH

- DH works because maps $f(x) = x^a$ and $h(x) = x^b$ commute
- When do we have commuting maps?
 - Powers of x (normal DH)
 - Iterates of a polynomial
 - J. Ritt (1923) – Power polynomials, Chebychev polynomials. Elliptic curve
- (Ring) LWE approximately commutes—use to build DH generalization
$$(s_1 \times a) \times s_2 = s_1 \times (a \times s_2)$$
$$(as_1 + e_1)s_2 + e_2 \approx s_1as_2 \approx (as_2 + e_2)s_1 + e_1.$$

Learning with Errors [2006, Regev]

$$\underbrace{\begin{pmatrix} b_1 \\ b_2 \\ \vdots \\ b_m \end{pmatrix}}_{\vec{b}} = \underbrace{\begin{pmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & \dots & a_{mn} \end{pmatrix}}_A \underbrace{\begin{pmatrix} s_1 \\ s_2 \\ \vdots \\ s_n \end{pmatrix}}_{\vec{s}} + \underbrace{\begin{pmatrix} e_1 \\ e_2 \\ \vdots \\ e_m \end{pmatrix}}_{\vec{e}}$$

- Approximate system over \mathbb{Z}_q
- Hard to find \vec{s} from A, \vec{b} .
- Hard to tell if \vec{s} even exists
- Reduction to lattice approximation problems

Ring LWE

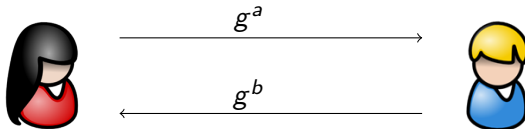
Definition

Let n be a power of 2, $q \equiv 1 \pmod{2n}$ prime. Define the ring

$$R_q = \frac{\mathbb{Z}_q[x]}{(x^n + 1)}.$$

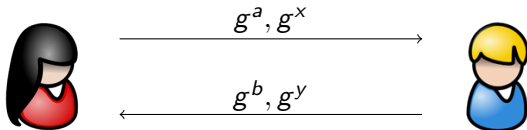
- Again, $b = as + e$ hard to find s
- Hard to distinguish from uniform b
- Approximation problems on *ideal* lattices
- More efficient than standard LWE

Authentication: HMQV – To Resist Man-in-the-middle Attack and Achieve Forward Security



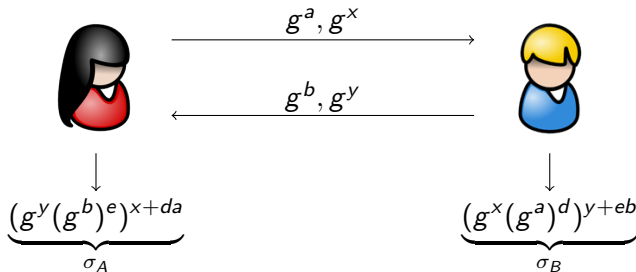
- Static keys a , b ; tied to each party's identity.

Authentication: HMQV – To Resist Man-in-the-middle Attack and Achieve Forward Security



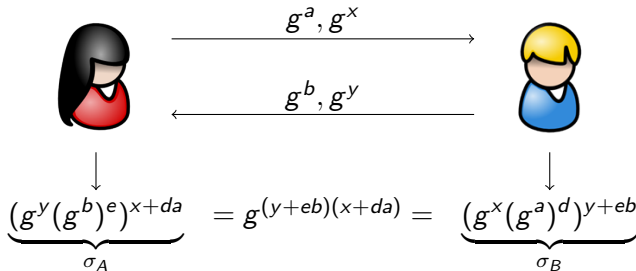
- Static keys a, b ; tied to each party's identity.
- Ephemeral keys x, y : **forward security**.

Authentication: HMQR – To Resist Man-in-the-middle Attack and Achieve Forward Security



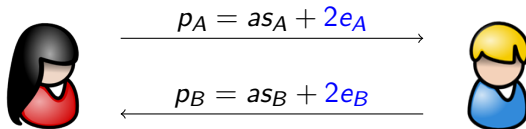
- Static keys a, b ; tied to each party's identity.
- Ephemeral keys x, y : **forward security**.
- Publicly derivable computations d, e .

Authentication: HMQR – To Resist Man-in-the-middle Attack and Achieve Forward Security



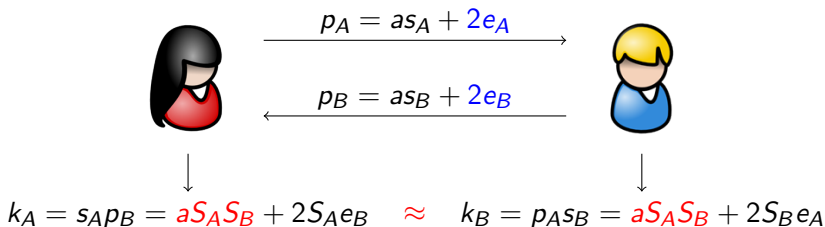
- Static keys a, b ; tied to each party's identity.
- Ephemeral keys x, y : **forward security**.
- Publicly derivable computations d, e .
- Shared key is $K = H(\sigma_A) = H(\sigma_B)$

Diffie-Hellman from Ideal Lattices



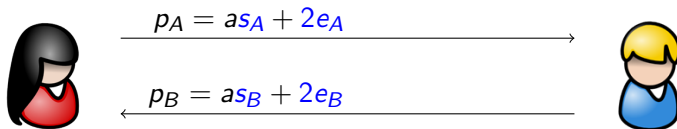
- Public $a \in R_q$. Acts like generator g in DH.

Diffie-Hellman from Ideal Lattices



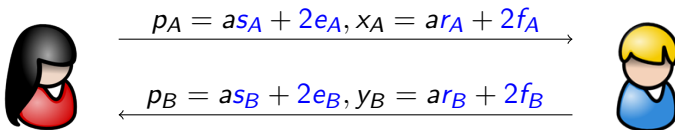
- Public $a \in R_q$. Acts like generator g in DH.
- Each side's key is only *approximately* equal to the other.
- Difference is even—same low bits.
- No authentication—MitM

HMQV from Ideal Lattices



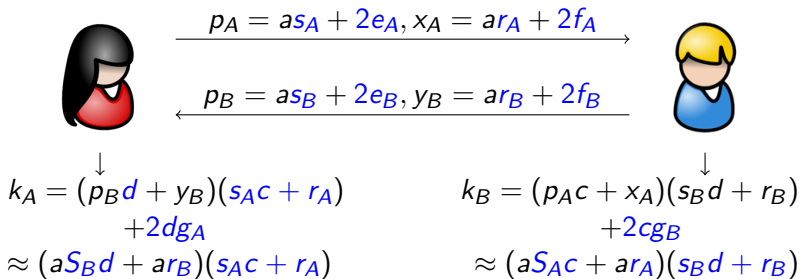
- p_A, p_B as above. Public, static keys for authentication

HMQV from Ideal Lattices



- p_A, p_B as above. Public, static keys for authentication
- x_A, y_B same form. Forward secrecy.

HMQV from Ideal Lattices



- p_A, p_B as above. Public, static keys for authentication
- x_A, y_B same form. Forward secrecy.
- c, d publicly derivable; g_A, g_B random, small.

Key Derivation

Obtaining shared secret from approximate shared secret:

$$k_A = (k_A^{(0)}, k_A^{(1)}, \dots, k_A^{(n-1)})$$

$$k_B = (k_B^{(0)}, k_B^{(1)}, \dots, k_B^{(n-1)})$$

$$\tilde{g} = (g^{(0)}, g^{(1)}, \dots, g^{(n-1)})$$

$$k_A - k_B = 2\tilde{g}$$

$$k_A \equiv k_B \pmod{2}$$

Key Derivation

Obtaining shared secret from approximate shared secret:

$$k_A = (k_A^{(0)}, k_A^{(1)}, \dots, k_A^{(n-1)})$$

$$k_B = (k_B^{(0)}, k_B^{(1)}, \dots, k_B^{(n-1)})$$

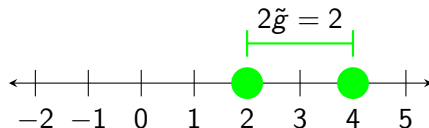
$$\tilde{g} = (g^{(0)}, g^{(1)}, \dots, g^{(n-1)})$$

$$k_A - k_B = 2\tilde{g}$$

$$k_A \equiv k_B \pmod{2}$$

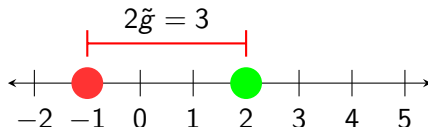
- Each $k_A^{(j)} = k_B^{(j)} + 2g^{(j)}$.
- Each $g^{(j)}$ is small ($|g^{(j)}| < \frac{q}{8}$).
- Matching coefficients differ by small multiple of 2
- Take each coefficient mod 2, get n bit secret

Wrap-around Illustrated



- Difference 2, both even.

Wrap-around Illustrated



- Difference 2, both even.
- But wait! If $q = 5$, $\mathbb{Z}_q = \{-2, -1, 0, 1, 2\}$.
- 4 becomes -1 , now parities disagree!

Compensating for Wrap-Around

- Recall: $|g^{(j)}| < \frac{q}{8}$
- Define $E = \{-\lfloor \frac{q}{4} \rfloor, \dots, \lfloor \frac{q}{4} \rfloor\}$. Middle half of \mathbb{Z}_q .
- If $k_B^{(j)} \in E$, no wrap-around occurs; $k_A^{(j)} \equiv k_B^{(j)}$.
- If $k_B^{(j)} \notin E$, then $k_B^{(j)} + \frac{q-1}{2} \in E$
- If $k_B^{(j)} \notin E$, $k_A^{(j)} + \frac{q-1}{2} \equiv k_B^{(j)} + \frac{q-1}{2}$.

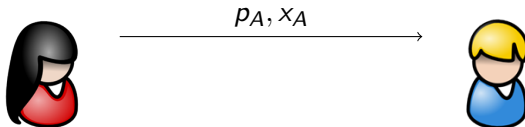
Wrap-around Defeated

Define $w_B^{(j)} = \begin{cases} 0 & k_B^{(j)} \in E, \\ 1 & k_B^{(j)} \notin E. \end{cases}$ Then $k_B^{(j)} + w_B^{(j)} \frac{q-1}{2} \in E$.

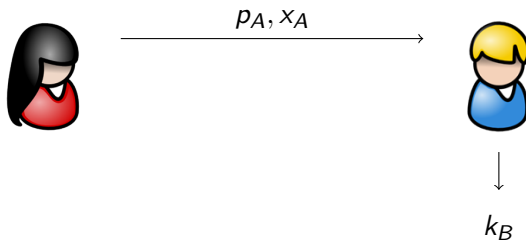
Also, $k_B^{(j)} + w_B^{(j)} \frac{q-1}{2} \equiv k_A^{(j)} + w_B^{(j)} \frac{q-1}{2} \pmod{2}$.

- $k_B^{(j)} + w_B^{(j)} \frac{q-1}{2} \pmod{q} \pmod{2} = k_A^{(j)} + w_B^{(j)} \frac{q-1}{2} \pmod{q} \pmod{2}$.
- Wrap-around correction $w_B = (w_B^{(0)}, w_B^{(1)}, \dots, w_B^{(n-1)})$
- $\sigma_B = k_B + w_B \frac{q-1}{2} \pmod{2}$.
- $\sigma_A = k_A + w_B \frac{q-1}{2} \pmod{2}$.

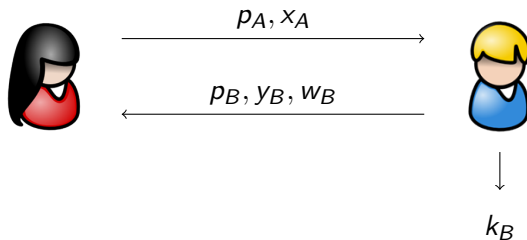
HMQV from Ideal Lattices—Corrected



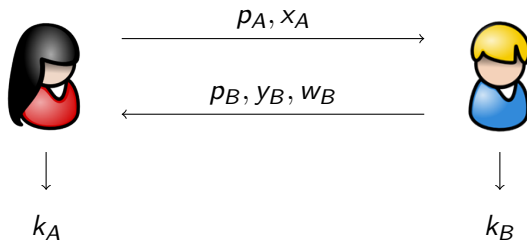
HMQV from Ideal Lattices—Corrected



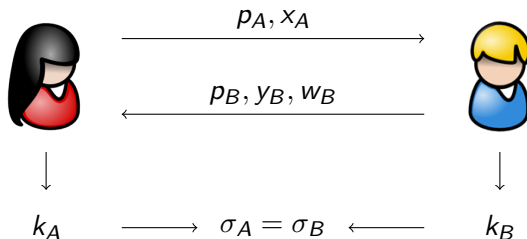
HMQV from Ideal Lattices—Corrected



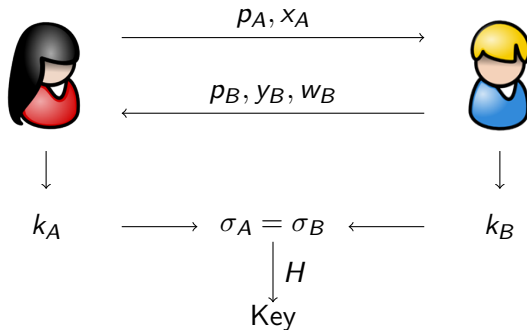
HMQV from Ideal Lattices—Corrected



HMQV from Ideal Lattices—Corrected



HMQV from Ideal Lattices—Corrected



Proof Games

Proof proceeds by series of games:

- Begin with simulated protocol
- Replace one hash output with true random value, back-program random oracle
- Adversary cannot distinguish from previous game
- Eventually, if original protocol can be distinguished from random, rLWE can be broken
- The modification using rejecting sampling

Forward Security

- If static keys compromised, previous session keys remain secure
- Notion captured in proof by giving adversaries ability to corrupt static key
- Use Bellare–Rogaway model restricted to two-pass

Quantum Hardness

- Proof uses Random Oracle Model—quantum implications not fully understood
- Important step to post quantum key exchange

Implementations Parameters

Parameters	n	Security (expt.)	α	γ	$\log \frac{\beta}{\alpha}$	$\log q$ (bits)
I*	1024	80 bits	3.397	101.919	8.5	40
II	2048	80 bits	3.397	161.371	27	78
III	2048	128 bits	3.397	161.371	19	63
IV	4096	128 bits	3.397	256.495	50	125
V	4096	192 bits	3.397	256.495	36	97
VI	4096	256 bits	3.397	256.495	28	81

Communication Overheads

Choice of Parameters	Size (KB)			
	pk	sk (expt.)	init. msg	resp. msg
I*	5 KB	0.75 KB	5 KB	5.125 KB
II	19.5 KB	1.5 KB	19.5 KB	19.75 KB
III	15.75 KB	1.5 KB	15.75 KB	16 KB
IV	62.5 KB	3 KB	62.5 KB	63 KB
V	48.5 KB	3 KB	48.5 KB	49 KB
VI	40.5 KB	3 KB	40.5 KB	41 KB

The bound 6α with $\operatorname{erfc}(6) \approx 2^{-55}$ is used to estimate the size of secret keys.

Timings

Parameters	Initiation	Response	Finish
I	3.22 ms (0.02 ms)	8.50 ms (4.69 ms)	5.23 ms (4.73 ms)
II	12.00 ms (0.04 ms)	29.33 ms (14.64 ms)	17.28 ms (14.61 ms)
III	10.33 ms (0.04 ms)	25.83 ms (13.46 ms)	15.58 ms (13.40 ms)
IV	83.61 ms (0.08 ms)	156.58 ms (39.86 ms)	73.11 ms (39.73 ms)
V	61.74 ms (0.08 ms)	117.81 ms (32.58 ms)	55.64 ms (32.20 ms)
VI	25.42 ms (0.08 ms)	62.31 ms (31.32 ms)	36.80 ms (31.29 ms)

Table: Timings of Proof-of-Concept Implementations in ms (The figures in the parentheses indicate the timings with pre-computing. For comparison, by simply using the “speed” command in openssl on the same machine, the timing for dsa1024 signing algorithm is about 0.7 ms, and for dsa2048 is about 2.3 ms).

Summary

- We build a simple AKE based on RLWE.
 - They are provably secure.
 - We can prove the Forward Security of the AKE.
 - Our preliminary implementations are very efficient.
- Our AKE are strong candidates for the post-quantum world.

Thank You

