Post-Quantum Cryptography

Simon Swenson

Problem: Quantum Computers

- Quantum Turing machines can solve hard problems used in public/private cryptosystems in polynomial time (Shor, 1995).
  - Integer factorization
  - Discrete logarithms in abelian groups
- TLS (under HTTPS) uses such algorithms for the client and server to agree on a shared key.

Possible Solutions

- Lattice-based methods
- Code-based methods
- Multivariate methods

What is a lattice?

- Let $v_1, v_2, \ldots, v_n$ form a basis for $\mathbb{R}^n$. Then $L = \{ (z_1*v_1, z_2*v_2, \ldots, z_n*v_n) \mid z_1, z_2, \ldots, z_n \in \mathbb{Z} \}$ is a lattice.
- Just like a linear combination (matrix multiplication), except we constrain the coefficients to integers instead of real numbers.

Hard Lattice Problems

- Shortest vector problem
- Gap shortest vector problem
- Closest vector problem
- Gap closest vector problem

Ring Learning with Errors

- Equivalent to finding the approximate shortest vector for a lattice (hard).
- Let p be prime. Then recall that Z/pZ is a field over (+, ·).
- Recall that the polynomial ring $Z/pZ[x] = a_0 + a_1x + a_2x^2 + \ldots, a_nx^n$, with $n \in Z$ and $a_0, a_1, \ldots, a_n \in Z/pZ$
- However, this is an infinite ring, and computers can't work without bounds. Thus, we create a finite ring from that polynomial ring by picking an cyclotomic polynomial, $\phi(x)$, and letting $\phi(x) = 1$ in the ring. This forms the finite ring $(Z/pZ[X])/\phi(x)$.

Ring Learning with Errors

- Let b be a bound < p.
- Let $A(x) = \{ a_1(x), a_2(x), \ldots, a_m(x) \mid a_i(x) \in (Z/pZ[X])/\phi(x) \}$ be public knowledge.
- Let $E(x) = \{ e_1(x), e_2(x), \ldots, e_m(x) \mid e_i(x) \in (Z/pZ[X])/\phi(x) \wedge e_i(x)$ is bounded by b $\}$ be private knowledge.
- Let s(x) be a polynomial bounded by b and private knowledge.
- Let $B(x) = \{ b_1(x), b_2(x), \ldots, b_m(x) \mid b_i(x) = (a_i(x) \cdot s(x)) + e_i(x) \}$ be public knowledge.
- Then, it is hard to find s(x) given the public knowledge.

Lattice-based Cryptography for the Internet (Peikert)

- Peikert’s suite is one step towards implementing the ring learning with errors approach for the internet.
- Peikert’s suite includes methods for key transport, encryption, and authenticated key exchange.

Error-correcting Codes

- Originally, ECC methods like Hamming codes were used to preserve data integrity.
- However, ECC methods can be applied to cryptography as well.
- What if, instead of using codes for data integrity, we sent data with errors in it, and only we know the code to correct the error (and thus decrypt)?

McEliece Cryptosystem – Key Generation

- Probabilistic key generation and encryption
- Let n, k, t be public integers.
- Def: A binary linear code C(n, k) is a linear subspace with dimension k of the vector space F_2^n, with F_2 a field with two elements.
- A picks a binary linear code C(n, k) capable of correcting t errors.
- A generates G, a k x n generator matrix for C.
- A picks a k x k binary, non-singular matrix S.
- A picks an n x n permutation matrix P.
- A computes G' = SGP
- Pub_A = (G', t), $Priv_A$ = (S, G, P)

McEliece Cryptosystem – Encryption

- B pads the message, m , to k bits.
- B computes c’ = mG’.
- B picks z $\in$ $(Z/2Z)^n$ with exactly t one bits.
- B computes c = c’ + z.

McEliece Cryptosystem – Decryption

- A computes P^{-1}.
- A computes $\hat{c} = cP^{-1}$.
- A uses C(n, k) to “correct” the errors in $\hat{c}$ to $\hat{m}$.
- A computes $m = \hat{m}S^{-1}$.

Multivariate Cryptography

- Based on multivariate polynomials. So, for example, $p = 1 + x + y + xy + x^2 + x^2y + xy^2 + x^2y^2$ is a multivariate polynomial of degree 2.
- Suitable for low-power scenarios, like smart card processors.
- Example: PFLASH, a multivariate signature system

PFLASH

- Based on the “big field” cryptosystem, C*.
- Consider a field F_q, a field with q elements.
- We can form a degree n polynomial field extension of F_q. Call it k. Note that we can treat this as an n-dimensional vector of F_q.
- Consider the C* monomial map $f(x) = x^{q^{\theta}+1}$, picking θ carefully so that $\gcd(q^n - 1, q^{\theta} + 1) = 1$. Note that $f: F_q{}^n \rightarrow F_q{}^n$.
- On it’s own f(x) is easily invertible, thus, we introduce two affine transformations: T, U.

PFLASH

- The type of multivariate cryptosystem depends on the properties of T and U.
- If T and U are both invertible, the system is “C*,” easily broken.
- If T and U are both singular (not invertible), the system is known as “pC*-,” what PFLASH uses.
- The scheme is defined by four parameters: q, n, r, and d, where q and n are as defined above, r is the corank of T, and d is the corank of U.

PFLASH Signature

- A publishes $P = T \circ f \circ U$, the public key.
- To reverse the public key operation, since neither T nor U are invertible, A must find a preimage under T, call it T', a preimage under U, call it U', and f^{-1}.
- To sign a message m, A first hashes the message into F_q^n, call the hashed version w, then computes $U'(f^{-1}(T'(w)))$. To verify, B applies the public key: $T(f(U(U'(f^{-1}(T'(w)))))) = T(f(f^{-1}(T'(w)))) = T(T'(w)) = w$.