THE PAST FIFTEEN YEARS OF PUBLIC-KEY CRYPTOGRAPHY

Scott Vanstone

University of Waterloo
&
Certicom Corp

ECC 2000, Essen
October 5, 2000

Outline

1. The Early Years
2. Early Deployment of Public-Key Cryptography
3. Elliptic Curve Cryptography
4. Other Public-Key Schemes
5. Provable Security
6. Deployment of Public-Key Cryptography in the 90's
7. Conclusions

The Early Years

- **1976**: The fundamental ideas behind public-key cryptography were discovered by Diffie and Hellman.
  - The recognition that the discrete log problem was hard and could conceivably be used to build public-key schemes.
- **1977**: Merkle and Hellman propose a public-key encryption scheme whose security is believed to be based on the hardness of the subset sum problem.

The Early Years (2)

- **1977**: The fundamental ideas behind the RSA scheme were discovered by Rivest, Shamir and Adleman.
  - The recognition that the integer factorization problem was hard and could be used to build public-key schemes.
  - First concrete scheme providing digital signatures.
- **1978**: McEliece recognized that the decoding problem for general linear codes being NP-hard could be used as the basis for a public-key encryption scheme.
  - Public key size is about 2^{19} bits.
    (Goppa code, $n = 1024$, $k \geq 644$, $t = 38$.)
  - No digital signature scheme.

The Early Years (3)

- **1983**: Shamir describes a "complete break" of the original Merkle-Hellman knapsack scheme.
- **1984**: Brickell describes a "complete break" of multiple-iterated Merkle-Hellman knapsack scheme.
- **1984**: Blake, Fuji-Hara, Mullin, Vanstone show that the DLP in $\mathbb{F}_{2^{127}}$ can be solved efficiently.
  - Coppersmith extends these results to obtain an algorithm with an asymptotically better running time.
- **1984**: ElGamal introduces public-key encryption and digital signature schemes based on the DLP.

The Early Years: Summary

- This was a time of academic exploration.
- From this era emerged hard problems which could be used to build public-key schemes:
  - The discrete logarithm problem (DLP).
  - The integer factorization problem (IFP).
- The basic principles underlying public-key cryptography were studied but the subtleties of practical use and wide-scale deployment were yet to come.

Early Deployment of Public-Key Cryptography

- From its discovery in the mid 70's until well into the 80's, there was little wide-scale deployment of public-key cryptography.
- Some proprietary public-key hardware implementations for point-to-point communications:
  - Certicom (TradeSecrets, Fax encryptor)
  - Cylink (Diffie-Hellman)
  - Racal (RSA)

Early Deployment (2)

Possible reasons for lack of deployment

- Lack of market.
- Lack of general awareness of benefits of public-key cryptography in the user community.
- Existing symmetric-key infrastructure to overcome.
- Suspicions about the security of public-key schemes.
- Lack of standards.
- Lack of public-key infrastructure and legal infrastructure.
- Patent coverage by Public Key Partners and RSA.

Elliptic Curve Cryptography

- Discovered independently by Koblitz and Miller.
- Presented at Crypto '85 by Miller.
- Security is based on hardness of the ECDLP.
- Any protocol based on DLP can be converted to one based on ECDLP.

Elliptic Curve Digital Signature Algorithm (ECDSA)

- The Elliptic Curve Digital Signature Algorithm (ECDSA) is the elliptic curve analogue of the DSA.
- Can be used to provide data origin authentication, data integrity, and non-repudiation.
- Widely standardized:
  - January 1999: ANSI X9.62.
  - January 2000: NIST FIPS 186-2.
  - August 2000: IEEE 1363-2000.

EC Key Pair Generation

- Domain parameters: E, $\mathbb{F}_q$, $G \in E(\mathbb{F}_q)$, $n = \text{ord}(G)$, $h = \#E(\mathbb{F}_q)/n$.
- Each entity A does the following:
  1. Select a random integer d in the interval $[1, n-1]$.
  2. Compute $Q = dG$.
  3. A's public key is Q; A's private key is d.

ECDSA Signature Generation

To sign a message m, A does the following:

1. Select a random integer k, $1 \leq k \leq n-1$.
2. Compute $kG = (x_1, y_1)$ and $r = x_1 \bmod n$.
   If $r = 0$ then go to step 1.
3. Compute $k^{-1} \bmod n$.
4. Compute $e = \text{SHA-1}(m)$.
5. Compute $s = k^{-1}\{e + dr\} \bmod n$.
   If $s = 0$ then go to step 1.
6. A's signature for the message m is (r, s).

ECDSA Signature Verification

To verify A's signature (r, s) on m, B should do the following:

1. Verify that r and s are integers in the interval $[1, n-1]$.
2. Compute $e = \text{SHA-1}(m)$.
3. Compute $w = s^{-1} \bmod n$.
4. Compute $u_1 = ew \bmod n$ and $u_2 = rw \bmod n$.
5. Compute $u_1 G + u_2 Q = (x_1, y_1)$ and $v = x_1 \bmod n$.
6. Accept the signature if and only if $v = r$.

Elliptic Curve Integrated Encryption Scheme (ECIES)

- ElGamal-like public-key encryption scheme.
- Designed by Bellare and Rogaway.
- Semantically secure in the presence of an adversary capable of launching chosen-ciphertext attacks.
- ECIES is being standardized by ANSI, IEEE and SECG.

ECIES Encryption Procedure

To encrypt a message m for B, A does the following:

1. Select a random integer r_A, $1 \le r_A \le n-1$.
2. Compute $R_A = r_A G$.
3. Compute $K = hr_A Q_B$. Check that $K \ne \mathcal{O}$.
4. Compute $k_1 \parallel k_2 = \mathsf{KDF}(K)$.
5. Compute $c = \mathrm{ENC}_{k_1}(m)$.
6. Compute $t = \mathrm{MAC}_{k_2}(c)$.
7. Send (R_A, c, t) to B.

ECIES Decryption Procedure

To decryption ciphertext (R_A, c, t), B does the following:

1. Perform a partial key validation on R_A.
2. Compute $K = hd_B R_A$. Check that $K \ne \mathcal{O}$.
3. Compute $k_1 \parallel k_2 = \mathsf{KDF}(K)$.
4. Verify that $t = \mathrm{MAC}_{k_2}(c)$.
5. Compute $m = \mathrm{ENC}_{k_1}^{-1}(m)$.

Certicom's Role in Development of ECC

- Recognized immediately that ECC had tremendous advantages over existing systems, especially in constrained environments.
- Began to evangelize the merits of ECC in the early 90's.
- Became editor of the ECC portion of IEEE P1363 in 1994.
- Fielded our first ECC product in 1997 (12 years after the discovery).

Some Products Using ECC

- Smart cards.
- wTLS (standardized by WAP).
- Palm Pilot VII.
- Hand-held VPN client (Nortel, Cisco, Alcatel, Intel, etc.).
- Mobiletrust: Complete certificate services especially well suited for constrained devices.
- Motorola's MPC180e security processor.
  - Offload the processing or IPSEC, IKE, SSL, WAP, and wTLS procotols used in broadband access, routers, and WAP gateways.

Other Public-Key Schemes

Over the past 15 years a number of public-key schemes have been proposed including:

- Chor-Rivest knapsack scheme (1984).
- Jacobians of hyperelliptic curves (1989).
- NTRU (1996).
- XTR (2000).

Chor-Rivest Knapsack Scheme

- Proposed in 1984.
- Does not use some form of modular multiplication to disguise an easy subset problem.
- Defined over $\mathbb{F}_q$, $q = p^h$.
- Example: $p = 197$, $h = 24$.

Key Generation

1. Select the following:
   1.1. $q = p^h$ so that DLP in $\mathbb{F}_q$ is easy.
   1.2. Random irreducible poly $f(x)$ of degree h over $\mathbb{Z}_p$.
   1.3. Random generator $g(x)$ of $\mathbb{F}_q$.
   1.4. Random permutation π of $\{0, 1, \ldots, p-1\}$.
   1.5. Random integer $d \in [0, p^h - 2]$.
2. For each $i \in \mathbb{Z}_p$, compute $a_i = \log_g(x+i)$.
3. Compute $c_i = (a_{\pi(i)} + d) \bmod (p^h - 1)$ for each $i \in \mathbb{Z}_p$.
4. Public key: $((c_0, c_1, \ldots, c_{p-1}), p, h)$
5. Private key: $(f(x), g(x), \pi, d)$.

Chor-Rivest Encryption

1. Represent the message m as a binary vector $M = (M_0, M_1, \ldots, M_{p-1})$ having exactly h 1's.
2. Compute $c = \Sigma_{i=0}^{p-1} M_i c_i \bmod (p^h - 1)$.
3. Ciphertext is c.

Chor-Rivest Decryption

1. Compute $r = (c - hd) \bmod (p^h - 1)$.
2. Compute $u(x) = g(x)^r \bmod f(x)$.
3. Compute $s(x) = u(x) + f(x)$, a monic polynomial of degree h over $\mathbb{Z}_p$.
4. Factor $s(x)$ into linear factors over $\mathbb{Z}_p$: $s(x) = \Pi_{j=1}^{h}(x + t_j)$, where $t_j \in \mathbb{Z}_p$.
5. The components of the vector M that are 1 have indices $\pi^{-1}(t_j)$, $1 \le j \le h$. The remaining components are 0.

Properties

- Large public key, e.g., 36,000 bits for $p = 197$, $h = 24$.
- No digital signature scheme.
- Knapsack has density greater than 1.
- Appears resistant to attacks based on lattice-basis reduction.
  - Schnorr, Hörner, 1995.
  - Broke Chor-Rivest with $p = 103$, $h = 12$.
  - Some success with $p = 151$, $h = 16$.
- No proof that security is based on the subset-sum problem.

Vaudenay's Attack (1998)

- For each public key there are at least hp^2 private keys:
  - Replacing x and g by x^{p^i} and g^{p^i} for $0 \le i \le h-1$ gives same public key.
  - Replace x and $\pi(i)$ by $x+u$ and $\pi(i)-u$ for any $u \in \mathbb{Z}_p$.
  - Replace $(x, d, \pi(i))$ by $(ux, d - \log_g u, u\pi(i))$ for any $u \in \mathbb{Z}_p$.
- **Fact:** For any $r|h$ there exists a generator g' of $\mathbb{F}_{p^r} \subseteq \mathbb{F}_{p^h}$ and a polynomial Q of degree h/r with coefficients in $\mathbb{F}_{p^r}$ having $-x$ as a root. Also, $Q(\pi(i)) = (g')^{c_i}$.

Vaudenay's Attack (2)

1. Known g' and π Attack:
   - Interpolate Q with the $h/r+1$ pairs $(\pi(i), (g')^{c_i})$.
   - Solve Q to obtain x.
   - Perform a known x attack.
2. Known g' and Unknown π Attack:
   - Devise an algorithm to deduce π from g'.
   - Algorithm is based on the minimal poly of x over $\mathbb{F}_{p^r}$.
3. Finding g':
   - Devise an algorithm to find g'.
   - Best performance when $r|h$ and $r > \sqrt{h + \frac{1}{4}} + \frac{1}{2}$.
   - Complexity is $O(h^3 p^r / r^2)$.

Practical Results

- Suggested parameters are $p = 197$, $h = 24$.
- Take $r = 6$.
- Private key can be found much faster than the time required to construct the public key.

NTRU (1996)

Parameters:

	NTRU1	NTRU2	NTRU3
p	3	3	3
q	128	128	256
N	167	263	503
d_ϕ	18	16	55
d_f	60	50	215
d_g	20	24	72

Claim: security of NTRU1, NTRU2, NTRU3 is equivalent to RSA-512, RSA-768, RSA-1024, respectively.

NTRU Key Generation

- Select two polynomial $f, g \in \{0, 1, -1\}[x]$ at random with:
  - The number of 1's in f is $d_f + 1$, the number of -1's is d_f, all other coefficients are 0.
  - The number of 1's and -1's in g are each d_g.
  - Compute polynomial F_p and F_q such that $F_q \star f = 1 \bmod q$ and $F_p \star f = 1 \bmod p$.
  - Public key is $h = F_q \star g$.
  - Private key is (f, F_p).

NTRU Encryption

- Consider the plaintext m as a 0-1 polynomial in $\mathbb{Z}_q[x]/(x^N - 1)$.
- Choose a random polynomial ϕ with d_ϕ 1's and d_ϕ -1's in its coefficients; other coefficients are 0.
- Ciphertext is $c = p \cdot \phi \cdot h + m$ (mod q).

NTRU Decryption

- Let $a = f \star c \pmod{q}$.
- The plaintext is $F_p \star a \pmod{p}$.

NTRU Hard Problem

- Given a polynomial $h \in \mathbb{Z}_q[x]/(x^N - 1)$, find two "small" polynomials f and g such that $f \star h = g \bmod q$, where $\star$ is multiplication in the truncated ring.
- Here "small" means the Euclidean norm of the vector is small.
- The best way to attack NTRU appears to be using lattice basis reduction.
- However, no **proof** that this is the best approach.

XTR

- $q | p^2 - p + 1 | p^3 + 1 | p^6 - 1$, where $p \equiv 2 \pmod 3$.
- The XTR subgroup S is the subgroup of order $p^2 - p + 1$ of $\mathbb{F}^*_{p^6}$.
- Claim: XTR has no relation to elliptic curves.

ECSTR: Elliptic Curve Singular Trace Representation

- Let d be a quadratic non-residue in $\mathbb{F}_{p^3}$.
- Consider the singular elliptic curve $E : y^2 = x^3 + dx^2$ over $\mathbb{F}_{p^3}$.
- There is a group isomorphim between the group of non-singular points on $E(\mathbb{F}_{p^3})$ and the subgroup of order $p^3 + 1$ of $\mathbb{F}^*_{p^6}$.
- This isomorphism can be efficiently computed in both directions.

ECSTR: Elliptic Curve Supersingular Trace Representation?

- There is a supersingular curve E of order $p^2 - p + 1$ over $\mathbb{F}_{p^2}$ when $p \equiv 2 \pmod 3$.
- The Weil or Tate pairing can be used to efficiently reduce the DLP in $E(\mathbb{F}_{p^2})$ to the log problem in the subgroup of order $p^2 - p + 1$ of $\mathbb{F}^*_{p^6}$ — the XTR subgroup.
- Q: Can the inverse isomorphism be efficiently computed?
- Can the Weil descent be used to do this?
- Comment: No one uses this supersingular curve because of the Weil/Tate pairing reduction to the XTR subgroup.
- Q: Is it more efficient to compute on this supersingular curve than in the XTR subgroup using the Trace representation?

Provable Security

- Assuming that the underlying math problem is hard, do we have any assurance that the **protocol** itself is secure?
- Until quite recently, protocol design has been ad hoc.
- Methodology:
  - Design protocol.
  - Protocol broken.
  - Re-design protocol.
  - Protocol broken again.
  - Re-design protocol again.
  - etc.

Example: Textbook RSA Encryption

- Encryption: $c = m^e \bmod n$.
- Decryption: $m = c^d \bmod n$.
- Chosen-ciphertext attack on Textbook RSA Encryption.
  - Adversary's goal: decrypt c.
  - Adversary can launch a chosen-ciphertext attack.
  - Adversary selects random x, presents $\overline{c} = cx^e \bmod n$ to the oracle.
  - The oracle returns $\overline{m} = (\overline{c})^d = c^d x^{ed} = mx \bmod n$.
  - The adversary computes $m = \overline{m} x^{-1} \bmod n$.

Example: RSA Encryption PKCS #1 (1993)

- Idea: Induce structure into the plaintext to defeat chosen-ciphertext attacks.
- Format the message m prior to encryption:

00	**02**	**padding**	**00**	**m**

- Decryptor checks that the decrypted message has the required structure before accepting the message.
- PKCS #1 is widely deployed, including in SSL.

Chosen-ciphertext attack on PKCS #1

- Bleichenbacher (1998).
- Target ciphertext is c.
- Main idea:
  - Carefully select x and compute $\widehat{c} = cx^e \bmod n$.
  - Present $\widehat{c}$ to decryption oracle.
  - If oracle returns an error message "incorrectly formatted plaintext", then this yields some information about m.
  - Repeat until enough information is collected to recover m.
- For 1024-bit RSA moduli, about 2^{20} chosen-ciphertexts suffice.
- The attack is real: implementations of SSL had to be changed to circumvent this attack.

Provable Security

Designing a provable-secure protocol entails:

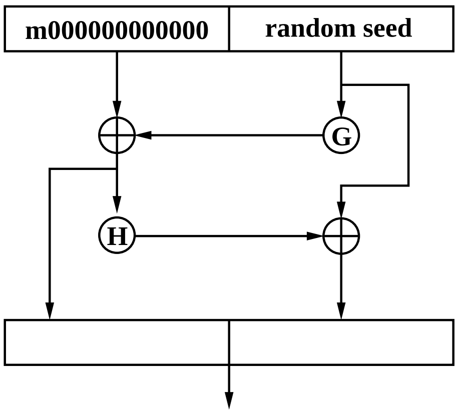
1. Specification of model (legitimate entities and adversary).
2. Definition of protocol goals within this model.
3. Statement of assumptions.
4. Description of protocol.
5. Proof that the protocol meets its goals within the model.

Why Provable Security is Important

- Clarifies and minimizes assumptions.
- Assumptions are "well-studied", and not ad-hoc.
- Allows precise definitions of security.
- Prevents against nasty surprises in the future.

How to Encrypt with RSA

- Optimal Asymmetric Encryption Padding (OAEP).
- Bellare and Rogaway (1994).



- G and H are hash functions.
- This is efficient!

Provable Aspects of OAEP

- **Semantic security**: A computationally bounded adversary cannot deduce any information whatsoever about the plaintext from the ciphertext, except possibly its length.

- **Theorem**: If the RSA problem is hard (finding eth roots modulo n), and if G and H are random functions, then OAEP RSA encryption is semantically secure against adaptive chosen-ciphertext attacks.

Provable Security in Practice

- Trend in cryptographic research:
  - Precise definitions of security.
  - Designing efficient protocols that can be proven to satisfy the security definitions.
- Standards are being rewritten to incorporate the protocols that are provably secure.
- Example: The new version of PKCS #1 specifies OAEP encryption.

Deployment of Public-Key Cryptography in the 90's

The 90's saw the first wide-scale deployment. Possible reasons:

- The Internet.
- Increased confidence and understanding.
- Public-Key Infrastructures (PKIs).
- Legal enforcement infrastructure.
- Application demand.
- Standards.

Conclusions

- The work done over the past 15 years is finally reaping its rewards.
- Public-key cryptography will enable e-commerce and m-commerce.
- Public-key cryptography is here to stay.