Applications of elliptic curves to cryptology

Jang SCHILTZ

Centre Universitaire de Luxembourg
Séminaire de Mathématiques
162A, avenue de la Faïencerie
L-1511 Luxembourg
Luxembourg
E-mail:schiltzj@cu.lu

1 Properties of elliptic curves

The central operation of cryptographic schemes based on elliptic curve cryptography (ECC) is the elliptic scalar multiplication (operation analogue of the exponentiation in multiplicative groups)

Definition 1.1 *Given an integer k and a point P in a finite field $\mathbb{F}$, the elliptic scalar multiplication kP is the result of adding P to itself k times.*

Definition 1.2 *The order of a point P on an elliptic curve is the smallest positive integer r such that $rP = \mathcal{O}$. If k and l are integers, then $kP = lP$ if and only if $k \equiv l (mod\ r)$.*

Definition 1.3 *The number of points of $E(\mathbb{F})$, denoted by $\#E(\mathbb{F})$ is called the curve order of the curve $E(\mathbb{F})$.*

Definition 1.4 *The trace $Tr(\cdot)$ is the linear map from $\mathbb{F}_{2^m}$ to $\mathbb{F}_2$ defined by*

$$Tr(a) = \sum_{i=0}^{m-1} a^{2^i}.$$

Proposition 1.5 *Let E be an elliptic curve ofer a finite field $\mathbb{F}_q$. Then:*

- *Hasse's theorem states that $\#E(\mathbb{F}_q) = q + 1 - t$, where $|t| \leq 2\sqrt{q}$. That is, the number of points in $E(\mathbb{F}_q)$ is approximately q.*

- *If q is a power of 2, then $\#E(\mathbb{F}_q)$ is even. More specifically, $\#E(\mathbb{F}_q) \equiv 0(mod\ 4)$ if $Tr(a) = 0$, and $\#E(\mathbb{F}_q) \equiv 2(mod\ 4)$ if $Tr(a) = 1$.*
- *$E(\mathbb{F}_q)$ is an abelian group of rank 1 or 2. That is, $E(\mathbb{F}_q)$ is isomorphic to $\mathbb{Z}_{n_1} \times \mathbb{Z}_{n_2}$, where n_2 divides n_1 and $q - 1$.*
- *If q is a power of two and $P = (x, y) \in E(\mathbb{F}_q)$ is a point of odd order, then the trace of the x-coordinate of all multiples of P is equal to the trace of the parameter a. That is, $Tr\big(x(kP)\big) = Tr(a)$ for each integer k.*

2 Elliptic curve cryptography

Unlike the ordinary discrete logarithm problem and the integer factorization problem, no subexponential-time algorithm is known for the elliptic curve discrete logarithm problem. For this reason, the strength-per-key-bit is substantially greater in an algorithm that uses elliptic curves. Thus, smaller parameters, but with equivalent levels of security, can be used with elliptic curve cryptosystems than with discrete logarithm systems.

2.1 Digital signature schemes

Digital signature schemes are designed to provide the digital counterpart to handwritten signatures (and more). A digital signature is a number dependent on some secret known only to the signer (the signer's private key), and, additionnaly, on the contents of the message being signed. Signatures must be verifiable - if a dispute arises as to whether an entity signed a document, an unbiased third party should be able to resolve the matter equitably, without requiring access to the signer's private key. disputes may arise when a signer tries to repudiate a signature it did create, or when a forger makes a fraudulent claim.

We speak here about asymmetric digital signature schemes with an appendix. "Asymmetric" means that each entity selects a key pair consisting of a private key and a related public key. The entity maintains the secrecy of the private key that it uses for signing messages, and makes authentic copies of its public key available to other entities which use it to verify signatures. "Appendix" means that a cryptographic hash function is used to create a message digest of the message, and the signing transformation is applied to the message digest rather than to the message itelf.

Security. Ideally, a digital signature scheme should be existentially unforgeable under chosen-message attack. This notion of security asserts that an adversery who is able to obtain entity A's signatures for any message of its choice is unable to successfully forge A's signature on a single other message.

Applications. Digital signature schemes can be used to provide the following basic cryptographic services: data integrity (the assurance that data has not been altered by uauthorized or unknown means), data origin authentication (the assurance that the source of data is as claimed), and non-repudiation (the assurance that an entity cannot deny previous actions or commitments). Digital signature schemes are commonly used as primitives in cryptographic protocols that provide other services including entity authentication, authenticated key-transport and authenticated key agreement.

Classification. The digital signature schemes in use today can be classified according to the hard underlying mathematical problem which provides the basis for their security:

1. Integer factorization (IF) schemes, which base their security on the intracibility of the integer factorization problem. Examples of these include the RSA and Rabin signature schemes.

2. Discrete logarithm (DL) schemes, which base their security on the intractability of the (ordinary)discrete logarithm problem in a finite field. Examples of these include the ELGamal, Schnorr, DSA, and Nyberg-Ryppel signature schemes.

3. Elliptic curve (EC)schemes, which base their security on the intractability of the elliptic curve discrete logarithm problem.

2.2 The Elliptic Curve Digital Signature Algorithm (ECDSA)

Elliptic curve requirements. In order to avoid Pollard's rho and the Pohlig-Hellman attacks on the elliptic curve discrete logarithm problem, it is necessary that the number of $\mathbb{F}_q$-rational points on E be divisible by a sufficiently large prime $n > 2^{160}$. Having fixed an underlying field $\mathbb{F}_q$, n should be selected to be as large as possible, i.e. one should have $n \approx q$, so $\#E(\mathbb{F}_q)$ is almost prime. Usually one selects $n > 4\sqrt{q}$. The co-factor is defined to be $h = \#E(\mathbb{F}_q)/n$.

Some further precautions should be exercised when selecting the elliptic curve. To avoid the reduction algorithm of Menezes, respectively Frey and Rück, the curve should be non-supersingular (i.e. p should not divide $\big(q + 1 - \#E(\mathbb{F}_q)\big)$). More generally, one should verify that n does not divide $q^k - 1$ for all $1 \leq k \leq C$, where C is a large enough so that is computationally infeasible to, find discrete logarithms in $\mathbb{F}_{q^C}$ ($C = 20$ suffices in practice). Finally, to avoid the attack of Semaev, Smart and Satoh and Araki on $\mathbb{F}_q$-anomalous curves, the curve should not be $\mathbb{F}_q$-anomalous i.e. $\#E(\mathbb{F}_q) \neq q$.

A prudent way to guard against these attacks and similar attacks against special classes of curves that may be discovered in the future, is to select the elliptic curve at random subject to the condition that$\#E(\mathbb{F}_q)$ is divisible by a large prime. Indeed, the probability that a random curve succombs to these special-purpose attacks is negligible. A curve can be selected verifiably at random by choosing the coefficients of the defining elliptic curve equation as the outpouts of a one-way function such as SHA-1 according to some pre-specified procedure.

Domain parameters. ECDSA domain parameters $D = (q, \mathrm{FR}, \mathrm{seedE}, a, b, G, n, h)$are comprised of

1. A field size q, where either $q = p$, an odd prime, or $q = 2^m$.
2. An indication FR (*field representation*) of the representation used ofr the elements in $\mathbb{F}_q$.
3. A bit string seedE of lenght at least 160 bits, if the elliptic curve was generated randomly (ptional).
4. Two field elements a and b in $\mathbb{F}_q$ which define the equation of the elliptic curve E over $\mathbb{F}_q$ (i.e $y^2 + xy = x^3 + ax^2 + b$ in a field of caracteristics 2 and $y^2 = x^3 + ax + b$ otherwhise).
5. Two field elements x_G and y_G in $\mathbb{F}_q$ which define a finite point $G = (x_G, y_G)$ of prime order in $E(\mathbb{F}_q)$.
6. The order n of the point G, with $n > 2^{160}$ and $n > 4\sqrt{q}$.
7. The cofactor $h = \#E(\mathbb{F}_q)/n$.

Key pair generation. An entity A's key pair is associated with a particular set of ECDSA domain parameters $D = (q, \mathrm{FR}, \mathrm{seedE}, a, b, G, n, h)$. This association can be assured cryptographically (e.g. with certificates) or by context (e.g. all entities use the same domain parameters). The entity A must have the assurance that the domain parameters are valid prior to key generation.

To generate a key pair, each entity A does the following:

1. Select a random or pseudorandom 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, an entity A with domain parameters $D = (q, \mathrm{FR}, \mathrm{seedE}, a, b, G, n, h)$ and associated key pair (d, Q) does the following:

1. Select a random or pseudorandom integer k, $1 \leq k \leq n-1$.
2. Compute $kG = (x_1, y_1)$ and convert x_1 to an integer $\overline{x}_1$.
3. Compute $r = x_1 \mod n$. If $r = 0$ then go to step 1.

4. Compute $k^{-1} \mod n$.

5. Compute SHA-1(m) and convert this bit string to an integer e.

6. Compute $s = k^{-1}(e + dr) \mod n$. If $s = 0$ then go to step 1.

7. A's signature for the message m is (r, s).

ECDSA signature verification. To verify A's signature (r, s) on m, B obtains an authentic copy of A's domain parameters $D = (q, \mathrm{FR}, \mathrm{seedE}, a, b, G, n, h)$ and associated public key Q. B then does the following:

1. Verify that r and s are integers in the interval $[1, n - 1]$.

2. Compute SHA-1(m) and convert this bit string to an integer e.

3. Compute $w = s^{-1} \mod n$.

4. Compute $u_1 = ew \mod n$ and $u_2 = rw \mod n$.

5. Compute $X = u_1 G + u_2 Q$.

6. If $X = \mathcal{O}$, then reject the signature. Otherwise, convert the x coordinate x_1 of X to an integer $\overline{x}_1$, and compute $v = \overline{x}_1 \mod n$.

7. Accept the signature if and only if $v = r$.

Proof: If a signature (r, s) on a message m was indeed generated by A, then $s = k^{-1}(e+dr) \mod n$. Rearranging gives

$$\begin{aligned} k &\equiv s^{-1}(e + dr) \\ &\equiv s^{-1}e + s^{-1}rd \\ &\equiv we + wrd \\ &\equiv u_1 + u_2 d(\mod n). \end{aligned}$$