Montgomery Ladder and ECDSA

September 11, 2013

ECDSA

The ElGamal Signature Scheme is the basis of the US 1994 NIST standard, Digital Signature Algorithm (DSA). The Elliptic Curve Digital Signature Algorithm (ECDSA) is the adaptation of this algorithm from the multiplicative group of a finite field to the group of points on an elliptic cure. The main benefit of using this group over field elements is ....

Parameters: An elliptic curve E defined over a finite field $\mathbb{F}_q$; a point $G \in E$ of large prime order n (generator of the group of points of order n). Paramaters chosen as such are generally believed to offer a security level of $\frac{n}{2}$ given current knowledge and technologies. Parameters are recommended to be generated following [2]. The field size q is usually taken to be a large, odd prime or a power of 2. The implementation of OpenSSL...

Public-Private Key pairs: the private key is an integer d, $1 < d < n-1$ and the public key is the point $Q = dG$. Calculating the private key from the public key requires solving the elliptic curve discrete logarithm problem (ECDLP), which is known to be hard in practice for the correctly chosen parameters. The most efficient algorithms currently know which solve the ECDLP have a square root run time in the size of the group and thus the security level is given as $\frac{n}{2}$.

Suppose Bob, with public-private Key pair $\{d_B, Q_B\}$, wishes to send a signed message m to Alice, he follows the following steps:

1. Using an approved hash algorithm, compute $e = Hash(m)$, take $\bar{e}$ to be the leftmost n bits of e.
2. Randomly select $k \leftarrow_R \mathbb{Z}_n$ with $1 < k < p-1$ and $(k, p-1) = 1$.
3. Compute the point $(x, y) = kG \in E$.

4. Take $r = x \mod n$; if $r = 0$ then return to step 2.

5. Compute $s = k^{-1}(z + rd_B) \mod n$; if $s = 0$ then return to step 2.

6. Bob sends (m, r, s) to Alice.

The message m is not necessarily encrypted, the contents may not be secret, but a valid signature gives Alice strong evidence that the message was indeed sent by Bob. She verifies that the message came from Bob by

1. cheching that all received parameters are correct, that $r, s \in \mathbb{Z}_n$ and that Bob's public key is valid, that is $Q_b \neq \mathcal{O}$ and $Q_B \in E$ is of order n.

2. Using the same hash function and method as above, compute $\bar{e}$.

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

4. Find the point $(x, y) = \bar{e}\bar{s}G + r\bar{s}Q_B$.

5. Verify that $r = x \mod n$ otherwise reject the signature.

Step 2 of the signing algorithm is of vital importance, inapproproate reuse of the random integer is what lead to the breaking of PS3 signature scheme implementation. Knowledge of the random value k leads to knowledge of the secret key as all values (m, r, s) can be observed by an eavesdropper, $\bar{e}$ can be found from m, $r^{-1} \mod n$ can be easily found from n, and if k is discovered then an adversary can find Bob's secret key through the simple calculation

$$d_B = (sk - \bar{e})r^{-1}.$$

Montgomery Ladder

Scalar multiplication is a common operation in cryptography and in a number of incidences (such as the multiplication by the secret, randomly generated element required in ECDSA), the scalar is intended to remain secret. This scalar multiplication is most efficiently performed using a square-and-multiply method (or the related Right-to-left method) as outlined in Algorithm 1

Double-and-add methods, though efficient, are vulnerable to simple power analysis. The addition law for points on Weirstrass curves is not complete, that is, the computation of $P+Q$ differs between the cases $P = Q$ and $P \neq Q$. Consequently, by examining the power consumption of the computation it is

possible to distingush when the if loop is executed and hence when a bit of n is 0.

As described by Montgomery in [1]

References

[1] P. L. Montgomery. Speeding the pollard and elliptic curve methods of factorization. *Mathematics of Computation*, 48:243–264, 1987.

[2] National Institute of Standards and Technology. Fips pub 186-4 digital signature standard (dss). pages 26–30, 2013.

Input: Point P, scalar n, k bits
Output: Point nP
$Q \leftarrow \mathcal{O}$
for *i from k to* 0 **do**
    $Q \leftarrow 2Q$ **if** $n_i = 0$ **then**
        $Q \leftarrow Q + P$
    end
end

Algorithm 1: Double-and-Add Point Multiplication

Input: Point P, scalar n, k bits
Output: Point nP
$R_0 \leftarrow \mathcal{O}$
$R_1 \leftarrow P$
for *i from k to* 0 **do**
    if $n_i = 0$ **then**
        $R_1 \leftarrow R_0 + R_1$
        $R_0 \leftarrow 2R_0$
    else
        $R_0 \leftarrow R_0 + R_1$
        $R_1 \leftarrow 2R_1$
    end
end

Algorithm 2: Montgomery Ladder Point Multiplication