A COMPARISON OF ECDSA, SM2 AND SM3

Fitzroy Nembhard

Florida Institute of Technology

December 2, 2014

There are several algorithms that are used in ensuring data integrity and authenticity in commercial applications and communication between parties. Some algorithms are based on hashing, some on public-key cryptography and others on elliptic curve cryptography or some variant of these algorithms. Examples of public-key cryptography include El-Gamal, Diffie-Hellman, RSA, and SM2. One common example of elliptic curve cryptography is the Elliptic Curve Digital Signature Algorithm (ECDSA). One hashing algorithm is SM3. In this report, I will compare ECDSA with SM2 and the SM3 hash function.

SM3

SM3 was published by the Chinese Commercial Cryptography Administration Office in 1997. In SM3, a message *m* is divided into blocks of 512 bits (16 32-bit words). Message padding is done to meet the required length. An iterative operation that utilizes a compression function is then applied to the padded message blocks. The hashing process basically consists of two parts, namely message expansion and a state update function (compression function) (Shen and Lee, SM3 Hash function). The message expansion function takes as input the 512-bit long message block and expands it to 68 32-bit words. The compression function starts at the first word and performs an iteration that updates each word in 64 rounds (Youssef). Figure 1 shows the 8 registers (A,B,C,D,E,F,G,H) used in the compression function. The design of SM3 resembles the design of SHA-2 but includes additional fortifying features such as feeding two message-derived words into each round, as opposed to only one in the case of SHA-2 (Youssef).

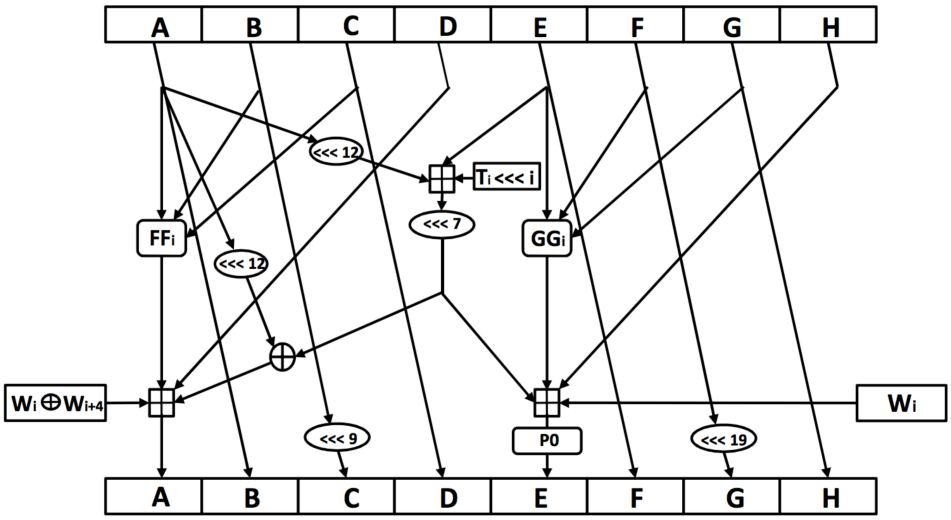

Figure 1: One round of the SM3 hash function (Youssef)

ECDSA

ECDSA is based on the standards specified in the Digital Signature Algorithm that was adopted as FIPS 186 by the National Institute of Standards and Technology (NIST) in 1993. Elliptic curve cryptography uses curves whose variables and coefficients are finite. ECDSA offers good security with smaller bit sizes as compared to the majority of public-key cryptosystems such as RSA and Diffie-Hellman. Table 1 shows a comparison of various key sizes.

Symmetric	ECC	DH/DSA/RSA
80	163	1024
112	233	2048
128	283	3072
192	409	7680
256	571	15360

Table 1: ECC key-size comparison (Blake-Wilson)

How ECDSA works

If two parties (Alice & Bob) decide to communicate by signing, sending and verifying a signature applied to a message, they must agree on the following parameters (C, G, n), where C is a curve, G is a base point on the curve and n is an integer of multiplicative order G.

ECDSA Signature Generation

A preliminary step to signature generation is the creation of a public and private key. Alice creates a key pair consisting of a random private key d_A in the interval [*1, n-1*] and a public key curve point $Q_A = d_A \times G$, where $\times$ denotes EC point multiplication by a scalar.

For Alice to sign a message *m,* do the following:

1. Calculate *e=H(m),* where *H* is a cryptographic hash function
2. Copy *n* leftmost bits of *e* as *z*
3. Select a random integer *k* from [*1,n-1*] for each message to be signed
4. Calculate a curve point $(x_1, y_1) = k \times G$
5. Create a signature pair (*r, s*) as follows:
   $r = x_1 \bmod n$. If $r=0$, go to step 3
   $s = k^{-1}(z + rd_A) \bmod n$. If $s=0$, go to step 3

ECDSA Signature Verification

To verify a signature signed by Alice, Bob should do the following:

1. Verify that *r* and *s* are integers in [*1,n-1*]. If not signature is invalid
2. Copy *n* leftmost bits of *e* as *z,* where *e=H(m)*
3. Let $w = s^{-1} \bmod n$
4. Calculate $u_1 = zw \bmod n$ and $u_2 = rw \bmod n$
5. Calculate a curve point $(x_1, y1) = u_1 \times G + u_2 \times Q_A$
6. Check if $r \cong x_1 \bmod n$. Otherwise signature is invalid

SM2

SM2 is an Elliptic Curve Cryptography (ECC) algorithm published by the Chinese Commercial Cryptography Administration Office. In SM2, a signer generates a digital signature over given data and a verifier verifies the validation of the signature.

SM2 SIGNATURE GENERATION

Before generation of the digital signature, the message m and a certain parameter, ZA, need to be compressed via a hash function. (Shen and Lee, SM2 Digital Signature Algorithm). ZA is a padded value, which consists of elliptic curve parameters and the public key of the signer.

For Alice to sign a message *m,* do the following:

1. Set $m\tilde{}=ZA \mid\mid m$
2. Calculate $e=Hv(m\tilde{})$, where *Hv* is an approved hash function such as SM3
3. Pick a random number *k* in [*1, n-1*] via a random number generator
4. Calculate the elliptic curve point $(x1, y1)=[k]G$
5. Create a signature pair (*r, s*) as follows:
   $r=(e+x1)$ mod *n*. return to step 3 if $r=0$ or $r + k=n$
   $s=((1+d_A)^{-1} \times (k-r \times d_A))$ mod *n*. return to step 3 if $s=0$
   $*d_A$ *is Alice's private key* (Shen and Lee, SM2 Digital Signature Algorithm)

SM2 SIGNATURE VERIFICATION

To verify a signature, the verifier does the following:

1. Verify that *r'* in [*1,n-1*]. Otherwise verification failed
2. Verify that *s'* in [*1,n-1*]. Otherwise verification failed
3. Set $m'\tilde{}=ZA \mid\mid m'$
4. Calculate $e'=Hv(M'\tilde{})$
5. Calculate $t = (r' + s')$ mod *n*, verification failed if $t=0$
6. Calculate the point $(x_1', y_1')=[s']G + [t]PA$
7. Calculate $R=(e'+x_1') \bmod n$. Verification pass if yes, otherwise failed
   *where PA=[dA]G=(xA, yA) and dA is the private key (Shen and Lee, SM2 Digital Signature Algorithm)

COMPARISON OF ECDSA, SM2 AND SM3

As can be seen in the explanations above, SM3 is a hash function designed by the Chinese Commercial Cryptography Administration Office to be used in an electronic authentication service system. SM2 is an Elliptic Curve Cryptography (ECC) algorithm also published by the Chinese Commercial Cryptography Administration Office. A few differences and some similarities can be noted between ECDSA and SM2. First, whereas the recommended hash function used in ECDSA can be SHA-2, SM3 is the recommended hash function for SM2. SM3 introduces additional strengthening features that surpass the security of SHA-2 (Youssef).

Additionally, SM2 requires extra padding and hashing that are not a part of the ECDSA standard. It can be seen in the ECDSA signature generation that the leftmost bits of the hash of the message (*e*), given the identifier *z,* is used in the calculation of the s value in the signature key pair, whereas in the SM2 signature generation, the message is padded with a parameter ZA, where ZA is derived as follows (Shen and Lee, SM2 Digital Signature Algorithm):

```
ZA=H256(ENTLA || IDA || a || b || xG || yG || xA || yA)
```

For signature verification, it can be seen in the steps above that ECDSA requires the calculation of a curve point $(x_1, y1) = u_1 \times G + u_2 \times Q_A$ and the verification that $r \cong x_1 \bmod n$, whereas SM2 requires the calculation of the point $(x_1', y_1')=[s']G + [t]P_A$ and verification that $R=(e'+x_1') \bmod n$. There is also a step in SM2 that requires that the verifier calculate $t = (r' + s') \bmod n$. This calculation checks to ensure that *t* is not equal to zero before calculating the point (x_1', y_1'). This step is not a part of ECDSA. In conclusion, both ECDSA and SM2 rely on the Elliptic Curve Discrete Logarithm Problem (ECDLP). However, the signature of SM2 requires a different signing procedure and different parameters (M. Liu).

REFERENCES

1. Blake-Wilson, et al. "Elliptic Curve Cryptography (ECC) Cipher Suites for Transport Layer Security (TLS)." 2006. Network Working Group. 2 12 2014 <https://tools.ietf.org/html/rfc4492>.

2. M. Liu, et al. "Partionally Known Nonces and Fault Injection Acttacks on SM2 Signature Algorithm." Lin, Dongdai and Moti Yung Shouhuai Xu. Information Security and Cryptohraphy. Guangzou: Springer International Publishing, 2013. 343-358.

3. Shen, S. and X. Lee. "SM2 Digital Signature Algorithm." 2014. Internet Engineering Task Force . 2 12 2014 <http://tools.ietf.org/html/draft-shen-sm2-ecdsa-02>.

4. —. "SM3 Hash function." 2014. Internet Engineering Task Force. 2 12 2014 <http://tools.ietf.org/html/draft-shen-sm3-hash-01>.

5. Youssef, Aleksandar Kircanski and Amr M. "Boomerang and Slide-Rotational Analysis of the SM3 Hash Function." 2014. https://eprint.iacr.org/2012/274.pdf. 3 12 2014 <https://eprint.iacr.org/2012/274.pdf>.