ECDSA - Application and Implementation Failures

Markus Schmid

Abstract— **Elliptic Curve Cryptography (ECC) is the newest member of public-key algorithms with practical relevance. It is based on the algebraic structure of elliptic curves over finite fields. Compared to RSA and Discrete Logarithm (DL) schemes, in many cases ECC has performance advantages with respect to fewer computations, and bandwidth advantages due to shorter signatures and keys. In addition, ECC provides the same level of security but with significantly shorter operands.[11] The Elliptic Curve Digital Signature Algorithm (ECDSA) is the elliptic analogue of the Digital Signature Algorithm (DSA). It uses the advantages of elliptic curves and was standardized in the US in 1999 by the American National Standards Institute (ANSI).[7]**

After a short introduction of ECC and the comparison in terms of security of RSA and ECC, the main purpose of this paper is to compare their security levels and to present the ECDSA and its applications.[13], [4] Furthermore, implementation failures like in the case of the ECDSA based code authentication of the Playstation 3 in 2010 will be analyzed.

I. Introduction

In 1976, a new type of cryptographic algorithms was introduced by Whitfield Diffie, Martin Hellman and Ralph Merkle. These kind of algorithms are based on mathematical problems that admit no efficient solution. In contrast to symmetric algorithms where a secret key is shared between two parties that is used for both, encryption and decryption, in public-key cryptography a user possesses a secret key but also a public key. The RSA algorithm was one of the first public-key cryptosystems and is still the most prominent and widely used crypto scheme in practice.

With the introduction of elliptic curve cryptography by Koblitz and Miller, a new type of public-key algorithms was introduced that is used in many new applications with security needs such as mobile devices.[11] After a short introduction to ECC and RSA and their comparison in terms of security, ECDSA as the elliptic curve variant of the Digital Signature Algorithm (DSA) is presented and implementation failures in two examples are analyzed.

II. ECC - Overview

Elliptic Curve Cryptography (ECC) is based on the generalized discrete logarithm problem and can be used for encryption, for key exchange and for digital signatures. ECC is steadily gaining popularity in applications due to its performance advantages over other public-key algorithms in many cases. For cryptographic use, we consider the curve over a finite field, where the most popular choice are prime fields $GF(p)$ and where all arithmetic is performed modulo a prime p. An elliptic curve over Z_p, $p > 3$ is the set of all pairs $(x, y) \in Z_p$ which fulfill

$$y^2 \equiv x^3 + a \cdot x + b \pmod{p}$$

together with an imaginary point of infinity $\mathcal{O}$, where

$$a, b \in Z_p$$

and the condition $4 \cdot a^3 + 27 \cdot b^2 \neq 0 \pmod{p}$

According to the definition of elliptic curves, the curve has to be nonsingular, which means that the plot has no self-intersections or vertices. This can be ensured if the discriminant of the curve -16($4a^3$+$27b^2$) is nonzero. Although, elliptic curves over prime fields $GF(p)$ than over other finite fields are currently more widely used in practice, curves over binary Galois fields $GF(2^m)$ are also popular. A particular type of elliptic curve over $GF(2^m)$ with the values 0 and 1 for the coefficients, is the Koblitz curve, which allows fast point multiplication.[11]

III. RSA - Overview

The RSA cryptosystem is named after its inventors R. Rivest, A. Shamir, and L. Adleman and its security is based on the intractability of the integer factorization problem.[9] Even though elliptic curves and discrete logarithm schemes are becoming more popular, it is currently the most widely used asymmetric cryptographic scheme. RSA has many application areas but in practice it is most often used for the encryption of small data items, mainly for key transport, and for digital signatures. However, RSA encryption is not meant to replace symmetric ciphers. Since performing RSA involves many computations, it is multiple times slower than ciphers such as AES. Therefore, the main use of its encryption is to securely exchange a key for a symmetric cipher. The key generation, encryption and decryption works as follows [11] :

RSA Key Generation
Output: public key: k_{pub} = (n,e) and private key: k_{pr}=(d)
1. Choose two large primes p and q.
2. Compute $n = p \cdot q$.
3. Compute $\Phi(n) = (p-1)(q-1)$.
4. Select the public exp. e $\in \{1, 2, ..., \Phi(n) - 1\}$ such that

$$gcd(e, \Phi(n)) = 1.$$

5. Compute the private key d such that

$$d \cdot e \equiv 1 \bmod \Phi(n)$$

RSA Encryption Given the public key (n,e) = k_{pub} and the plaintext x, the encryption function is:

$$y = e_{k_{pub}}(x) \equiv x^e \bmod n$$

where x,y $\in Z_n$.

RSA Decryption Given the private key $d = k_{\mathrm{pr}}$ and the ciphertext y, the decryption function is:

$$x = d_{\mathrm{k_{pr}}}(y) \equiv y^d \mathrm{mod}\ n$$

where x,y $\in Z_n$.

IV. Security of ECC and RSA

In 'The Case for Elliptic Curve Cryptography' the U.S. National Security Agency recommend that the industry should move on from the first generation of public key algorithms to elliptic curves due to both, the relative security offered by these kind of curves and their relative performance.[14] Elliptic curves offer more efficient implementations and bandwidth advantages with respect to shorter signatures and keys at the same security level as e.g. RSA. In terms of numbers, about 160-256 bit vs. 1024-3072 bit. For elliptic curves, a security level of 80 bit provides medium-term security but bit lengths up to 256 bit are usually used, which corresponds to a security level of up to 128 bit.[3], [11]

A comparison of key sizes and their bit security between ECC and RSA is shown by [15]:

Bit Security

	80	112	128	192	256
ECC	160	224	256	384	512
RSA	1024	2048	3072	8192	15360

However, the stated security is only achieved in case that cryptographically strong elliptic curves are used. The National Institute of Standards and Technology (NIST) often propose standardized curves for the application in practice since several families of curves have cryptographic weaknesses, e.g., supersingular curves.[11]

According to FIPS 186-2, examples for recommended curves over prime fields F_{p}, defined by a generalized Mersenne prime, are:

$\mathrm{p}_{192} = 2^{192} - 2^{64} - 1$
$\mathrm{p}_{224} = 2^{224} - 2^{96} + 1$
$\mathrm{p}_{256} = 2^{256} - 2^{224} + 2^{192} + 2^{96} - 1$
$\mathrm{p}_{384} = 2^{384} - 2^{128} - 2^{96} + 2^{32} - 1$
$\mathrm{p}_{521} = 2^{521} - 1$

Their group orders are all prime and have the same coefficient $a = -3$.[15], [3]

In case the elliptic curve is chosen with care, the best known attacks against the ECDLP are notably weaker than the best factoring algorithms to attack the RSA, and the best algorithms to solve the discrete logarithm problem modulo p. Powerful attacks against the DLP modulo p like the index-calculus algorithms are not applicable against elliptic curves. Therefore, the only remaining attacks are Pollard's rho method and Shanks' baby-step giant-step method.[11]

Since the RSA cryptosystem was introduced, the effectiveness of brute-force attacks got stronger and stronger. This also had an considerable impact on the choice of RSA modulus sizes. The Number Field Sieve (NFS), which is already more than 20 years old, is the most effective method that has been published to attack the RSA. In terms of integer factorization, quite a number of small improvements were achieved over time that slightly influenced NFS' effectiveness but there was no major breakthrough. However, worth mentioning is the polynomial time factoring on a quantum computer.[4]

V. Ecdsa and examples of implementation failures

Elliptic curves and the digital signature algorithm that is based on it, offer a great level of security when implemented correctly. In [13], Vaudenay listed four necessary security conditions for DSA and ECDSA respectively. First, the discrete logarithm in the subgroup spanned by G has to be hard. Otherwise it is possible to compute the discrete logarithm of the public key to receive the secret key. Furthermore, SHA-1 has to be a one-way and at the same time, a collision-resistant hash function to withstand forgery attacks. Last, the generator for k has to be unpredictable. In the following, two examples are presented where the last condition didn't hold, which led to an intrusion into the system.

A. Console Hacking - Playstation 3

On the 27th Chaos Communication Congress in 2010, a hacker group called fail0verflow presented a way to sign software for Sony's game console, the Playstation 3. This could be achieved by finding out the private key that is used within the ECDSA. As a result, copied and unlicensed games could be downloaded on any Playstation 3 and even malicious software could be deployed since it was officially signed by Sony. ECDSA is used for the code authentication that is checking and verifying the digital signature of a binary file before it is allowed to be run on a processor. For the ECC version of the digital signature algorithm defined over a prime finite field ($\mathrm{Z_p}$), a large prime p should be selected. In addition, the parameters a and b for the curve have to be chosen and a base point G of high order n, meaning that $n \times G = O$ for a large n. After randomly selecting X, $1 \leq x \leq n-1$, which serves as the private key, the public key Y can be calculated by

$$Y = X \times G.$$

According to [1], an approved hash function shall be used to generate the ECDSA curve parameters. Furthermore, for a high security, n shall be chosen in adherence to the recommendations in the NIST document FIPS 186-3.

The parameters p, a, b, G, n and Y will be publicly available whereas X serves as the private key. Constructing a digital signature of a document requires a one-time random number K such that $0 < K < n - 1$. Each digital signature has to be created with a different value for K. M is an integer that represents a hash of the document, which has to be signed. The digital signature that is constructed for M consists of two parts, sig_1 and sig_2 where sig_1 is

constructed by calculating the point $K \times G$ on the elliptic curve first and retaining the modulo n of its x-coordinate. In case the modulo operation generates a zero value for sig_1, a different K has to be tried.

$$sig_1 = (K \times G)_{\mathrm{x}} \mathrm{mod}\ n$$

$$sig_2 = K^{-1} \cdot (M + X \cdot sig_1) \mathrm{mod}\ n$$

The recipient of the document can verify the authenticity by: [8], [5]

(a) calculating the hash M of the document
(b) calculating $w = {sig_2}^{-1} \mathrm{mod}\ n$, $u_1 = M \cdot w \mod n$, $u_2 = sig_1 \cdot w \mod n$
(c) computing the point (x, y) on the curve:
$(x, y) = u_1 \times G + u_2 \times Y$
(d) authenticating the signature by checking whether the equivalence holds: $sig_1 \equiv x \mod n$

In cryptography, generating random numbers is a fundamental task. These random numbers are needed in steps of cryptographic algorithms or protocols as well as for generating cryptographic keys. Even if [6] claim that a certain robustness and therefore perfect randomness is not guaranteed by the two Linux PRNGs (pseudo-random number generators), /dev/random and /dev/urandom, the hacker group fail0verflow presented a common way to generate a random K:

$$m = open(\text{"}/dev/random\text{"}, \text{"}rb\text{"}).read(30)$$

Sony didn't generate a K with the help of a PRNG but used 4 constantly as its random number. The accompanying risk of using the same K for two different documents is that an adversary can find out the private key and proceed to counterfeit the signature. Given the hashes of two different documents M and M', which are signed with the same K, the signatures for these two documents will look like the following (primed signatures symbolize the second document):

$sig_1 = (K \times G)_{\mathrm{x}} \mathrm{mod}\ n$
$sig_2 = K^{-1} \cdot (M - X \cdot sig_1) \mathrm{mod}\ n$
$sig'_1 = (K \times G)_{\mathrm{x}} \mathrm{mod}\ n$
$sig'_2 = K^{-1} \cdot (M' - X \cdot {sig_1}') \mathrm{mod}\ n$

Sig_1 and sig'_1 are independent of the document and stay the same. By calculating the difference of $sig_2 - sig'_2$, the adversary would receive

$$sig_2 - sig'_2 = K^{-1}(M - M')$$

and is able to instantly calculate K. Finally, the private key X can be computed by

$$sig_2 = K^{-1} \cdot (M - X \cdot sig_1) \mathrm{mod}\ n$$

[8], [5]

B. Bitcoin

Another example of a flawed implementation of ECDSA happened in the case of Bitcoin. Bitcoin is a digital currency, which is based on a distributed peer-to-peer system. User can transact money directly without the need of a financial institution as an intermediary. As of today, it is a popular way of online payment: According to [2], the total amount of bitcoins in circulation on the 4th of December 2015 was 14.913.800 at a market price of 361.11USD. Bitcoins are transferred between two users A and B by attaching a digital signature (which uses A's private key) of the hash of the previous transaction. In addition, information about B's public key are attached at the end of a new transaction. The verification of the signature can be achieved by A's public key from the previous transaction.[3] [12] shows an exemplary transaction that has one output and two inputs:

transaction:
9ec4bc49e828d924af1d1029cacf709431abbde46d59554b62

input 1:
30440220d47ce4c025c35ec440bc81d99834a624875161a26bf
56ef7fdc0f5d52f843ad1022044e1ff...

input 2:
30440220d47ce4c025c35ec440bc81d99834a624875161a26bf
56ef7fdc0f5d52f843ad102209a5f1c...

By comparing the inputs, it can be noticed that bytes at the start and at the end are equal. The starting bytes represent the actual signature (r, s):

r1: d47ce4c025c35ec440bc81d99834a624875161a26bf56ef...
r2: d47ce4c025c35ec440bc81d99834a624875161a26bf56ef...

s1: 44e1ff2dfd8102cf7a47c21d5c9fd5701610d04953c68365...
s2: 9a5f1c75e461d7ceb1cf3cab9013eb2dc85b6d0da8c3c6e...

The flaw was the same as in the case of the Playstation 3: a random number is needed for each signature but the random number was used twice with the same private key. Therefore the private key could be calculated and attackers, who were able to sign any transaction, could steal bitcoins from affected Bitcoin wallets. This was due to a vulnerability in the Android implementation of the Java class SecureRandom that prevented the generation of safe random numbers to protect the wallet applications.[10]

[3] made an effort by extracting 47.093.121 elliptic curve points from the signatures and verifying their correctness. In addition, they looked for duplicated nonces in the signature. They found that 158 unique public keys used the same signature nonce's value for more than one signature. One address could be traced back to have stolen bitcoins from 10 addresses with a value of approximately $12.000 USD.

VI. Conclusion

Elliptic curves and ECDSA as its digital signature algorithm offer a secure, fast and efficient alternative to the RSA signature scheme and DSA. This evaluation only holds true when the algorithm is implemented correctly. In this work, two cases were analyzed, in which Vaudenay's fourth security condition didn't hold: the generator for k was predictable and thus, the private key could be calculated. In case the elliptic curve is chosen with care and the security conditions are met, the best known attacks against the ECDLP are notably weaker than the best factoring algorithms to attack the RSA. Another advantage is that powerful attacks like the index-calculus algorithms are not applicable against elliptic curves. Therefore only Pollard's rho method and Shanks' baby-step giant-step method remain for attacking, which have a comparable slow runtime.

References

[1] FIPS 186-3. *Digital Signature Standard (DSS)*. Federal Information Processing Standards Publication. National Institute of Standards and Technology, Gaithersburg, 2009.

[2] "Bitcoin". `http://blockchain.info`, 2015.

[3] Joppe W. Bos, J. Alex Halderman, Nadia Heninger, Jonathan Moore, Michael Naehrig, and Eric Wustrow. Elliptic curve cryptography in practice. Cryptology ePrint Archive, Report 2013/734, 2013.

[4] Joppe W. Bos, Marcelo E. Kaihara, Thorsten Kleinjung, Arjen K. Lenstra, and Peter L. Montgomery. On the security of 1024-bit rsa and 160-bit elliptic curve cryptography. 2009.

[5] "Bushing", "Marcan", "Segher", and "Sven". Ps3 epic fail. `https://events.ccc.de/congress/2010/Fahrplan/attachments/1780_27c3_console_hacking_2010.pdf`, 2010.

[6] Yevgeniy Dodis, David Pointcheval, Sylvain Ruhault, Damien Vergnaud, and Daniel Wichs. Security analysis of pseudo-random number generators with input: /dev/random is not robust. *Proceedings of the 2013 ACM SIGSAC conference on Computer and communications security*, pages 647–658, 2013.

[7] Don Johnson, Alfred Menezes, and Scott Vanstone. The elliptic curve digital signature algorithm (ecdsa). *International Journal of Information Security*, pages 36–63, 2001.

[8] Avinash Kak. *Elliptic Curve Cryptography and Digital Rights Management*. Lecture Notes on Computer and Network Security. Purdue University, 2015.

[9] Alfred Menezes, Paul van Oorschot, and Scott Vanstone. *Handbook of Applied Cryptography*. Discrete Mathematics and Its Applications. CRC Press, Boca Raton, 1997.

[10] AVG Now Official Blog Online Security News and Tips. The android bitcoin vulnerability explained. `http://now.avg.com/android-bitcoin-vulnerability-explained/`, 2013.

[11] Christof Paar and Jan Pelzl. *Understanding Cryptography*. A Textbook for Students and Practitioners. Springer-Verlag, Berlin Heidelberg, second edition, 2010.

[12] Nils Schneider. Recovering bitcoin private keys using weak signatures from the blockchain. `https://www.nilsschneider.net/2013/01/28/recovering-bitcoin-private-keys.html`, 2013.

[13] Serge Vaudenay. The security of dsa and ecdsa - public key cryptography - pkc 2003. *Lecture Notes in Computer Science*, pages 309–323, 2003.

[14] Ann Hibner Koblitz, Neal Koblitz, and Alfred Menezes. Elliptic curve cryptography: The serpentine course of a paradigm shift. *Journal of Number Theory*, 131(5):781–814, 2011.

[15] Marc Joye. Elliptic curves and fault attacks. *University of California, Santa Barbara*, 2015.