Provable Security of Elliptic Curve Digital Signature Algorithm (ECDSA)

Haobei Song
20629096
University of Waterloo

January 28, 2018

Abstract

Daniel and Brown proved the security of Elliptic Curve Digital Signature Algorithm (ECDSA) against adaptive chosen-message attacks with four weaker sufficient assumptions compared to previous work. Namely (i) The hash function must be uniform and collision-resistant. (ii) The ephemeral private key generator is a pseudorandom generator. (iii) The underlying elliptic curve group used here is considered to be a generic group so the special group properties per se are not exploited by the adversaries. (iv) The family of functions mapping ephemeral public keys to private keys is further constrained.

1 Introduction

Before this paper was introduced, there was no formal mathematical proof for ECDSA though it had garnered considerable attention as it started to be taken as digital signature standards in many reputable dedicated institutions working on information security. The security of other ELGamal-like digital signature schemes similar to ECDSA (and DSA) was proved by Pointcheval and Stern but the security of ECDSA remained open due to the lack of ephemeral public key in ECDSA which is included in most of other provably secure ELGamal-like digital signature schemes. Daniel and Brown showed the proof of the security of ECDSA by proposing a stronger assumption on being generic of the underlying elliptic curve groups rather than intractability of the elliptic curve discrete logarithm. They claim the strongness of the assumption could be balanced out by assumption (i) which treats the hash function as a uniform and collision-resistant hash function instead of a random oracle.

2 Definitions

2.1 Signature Scheme

A signature scheme is a triple of polytime probabilistic algorithems (PPT) $\Sigma = (K, G, V)$, and they are key generation, signature generation and signature verification respectively. K is given the security parameter n and generates a public key Y and private key x. G is the signer which given private key x and an arbitrary plaintext message m, outputs a signature S. The verifier V can, given public key Y, plaintext message m and signature S, accept or reject the signature. A correct signature scheme should always have that the verifier can accept a signature signed by the true signer with failure rate neglegible and reject a forged signature with success probability neglegible.

2.2 Forger

A forger of a signature scheme $\Sigma = (K, G, V)$ is a PPT F, which given public key Y, a signature S and an internal state X, output a message m, state X and a signature R or a request, also denoted as R for a signature of a message m_i.

A forger game for a specific forger F is a game consisting of counterplay of the following:
Initially, the signer calls K to output a public key which is revealed to the forger and a private key x which it keeps secret.
The forger (which knows the public key Y) starts from an initial state X_0 and output a message m_i, a state X_i and R_i a signture or a request for signature.
For $i \geq 1$. If R_i is a request for a signature, then the signer correctly sign the message m_i and call the forger with input the signature S_i for m_i and the state of the forger X_i. If R_i is a signature, then the forger outputs the signature and the game is over.
An additional constraint is that when the game ends the message for which the forger forge the signature for was not queried to the signer. Otherwise this forger would not be useful.
A signature scheme Σ is (p,Q,t) secure if it does not have a (p,Q,t) forger which could win the forger game in at most Q rounds performing t computations with probability at least p. When Q equals 0 the forger is said to be passive and active otherwise. A selective forger is in addition given a message to be signed before the forger game starts and only succeed until the forger correctly forges a signature for such message which of course is not queried to the signer. In contrast the forger defined previously is called an existential forger.

Signature Security A signature scheme is (p,Q,t)-secure aganist an existential forgery if there exists no (p,Q,t)-forger and is (p,Q,t,U)-secure aganist a selective forgery if there exists no (p,Q,t,U)-selective forger where U is a PPT algorithm to generate the initial selective message(challenge message) to be

forged. Since a selective forger is more harmful in the sense that a signature scheme breakable by a selective forger also fails existential forger test in the straight forward way.

ECDSA   Elliptic Curve Digital Signature Algorithm or ECDSA is a tuple $\Sigma = (K, G, V)$ of Key Generation, Signature Generation and Signature Verification algorithms which are all polytime probabilistic algorithms (PPT). Start from the construction of the underlying group ECDSA is built on. n is a prime number of certain bit-length defined by a security parameter. $\mathbb{A}_n$ is a subgroup of an elliptic curve group defined over a finite field with a generator denoted as G of order n. Such an elliptic curve group can be generated effiently with the above requirements satisfied. The conversion function is just $\mathbf{x}$ which returns the x-coordinate of a point on elliptic curve or more formally:

$$f : \mathbb{A}_n \to \mathbb{Z}_n.$$

A hash function is $h : \{0, 1\}^* \to \mathbb{Z}_n$ with more details in the rest of the paper.

Key Generation   : A private key is an integer $d \in_R \mathbb{Z}_n \backslash \{0\}$ generated by a pseudorandom genrator. The corresponding public key is just $Q = dG$ which could be computed efficiently in the means of double and add.

Signature Generation   For a given message M in the message space $\{0, 1\}^*$, firstly select a ephemeral key $k \in_R \mathbb{Z}_n$ by a pseudorandom generator and compute $R = kG$.
Call the conversion function to compute $r = \mathbf{x}(R)$. If this returns 0 go back to step 1. Since $\mathbf{x}$ returns zero with neglegible probability, this should not be a concern in practice.
Then compute the hash of the plaintext message M to abtain $e = h(M)$. Compute $s = k^{-1}(e + dr) \bmod n$. Similarly, there is a chance that s is zero, Bfut as long as the hash function and pseudorandom generators behave as they are supposed to, this only happens with a neglegible probability. The signature on M is (r, s)

Signature Verification   To verify a signature (r, s) on a plaintext message M with public key Q, group $\mathbb{A}_n$ and generator of the group G. First check the syntax of the signature and that $r, s \in \mathbb{Z}_n$. Then return ACCEPT if

$$r = f(s^{-1}h(M)G + s^{-1}rQ)$$

and REJECT otherwise.

DSA   The Digital Signature Algorithm or DSA differs from ECDSA in the underlying group and the conversion function. Namely

3 Building Blocks

3.1 Discrete Logarithm and Secure Group

The Discrete Logarithm Problem (DLP) is formulated as follows. Let $(\mathbb{G}, *)$ be a group with group operation $*$. A discrete logarithm is, given group elements $G, P \in \mathbb{G}$, the solution (if one exists) x which satisfies that $xG = P$ or equivalently $x = log_G P$. Normally $\mathbb{G}$ is defined in such a way in which a generator always exists and is given beforehand to ensure every group element has a discrete logarithm solution. The Nechaev-Shoup generic model of a group is such a group without any other structural properties to be potentially exploited. In a genric goup, the group operation can only be performed by a oracle and the group operation is in some sense random subject to the given group operations. In a nutshell, a generic group is a group with a back-box like group operation. However this is unachievable for any efficient groups. Such a group as a Nechaev-Shoup generic model is also named a *secure group* of strength $|G|/2$. $|G|/2$ comes from the fact that Pollard-ρ-algorithm could find the logarithm solution with probability $\sqrt{\frac{\pi}{2|G|}}$ by birthday parodox.

The secure group $\mathbb{A}_n$ used in ECDSA is a group of order n generated by points on an elliptic courve over a finite field $\mathbb{F}_q$. Depending on the parity of q, the implementation might differ nontrivially.

3.2 Conversion Functions

In the following statement, define $\mathbb{A}_n$ to be the Elliptic Curve group and $\mathbb{Z}_n$ the exponent group. As an example, $x = log_G P, x \in \mathbb{Z}_n, G, P \in \mathbb{A}_n$.
A conversion function is

$$f : \mathbb{A}_n \to \mathbb{Z}_n \tag{1}$$

3.2.1 Almost-Bijectivity

Almost-Bijectivity is defined in terms of ϵ-*clustering*. A conversion function is ϵ-clustered if given $z_0 \in \mathbb{Z}_n, A \in_R \mathbb{A}_n$ The probability that $f(A) = z_0$ is at least ϵ. And a conversion function is *almost-bijective* of strength at least $log_2(1/\epsilon)$ bits if it is not ϵ-clustered.

3.2.2 Almost-Invertibility

An *almost-inverse* of f is defined as a polytime probabilistic algorithm (PPT) $\mathcal{INV}$

> **Input:** $\mathbb{A}_n, z \in_R \mathbb{Z}_n$
> **Output:** A such that $f(A) = z$ with probability at least 1/10

Fact Almost Invertibility implies almost-bijectivity of strength at least $log_2(1/\epsilon)$ bits where $\epsilon > \frac{10}{n}$.

Note   The contrapositive of the statement above is easy to prove.
The conversion function for ECDSA is just the x-coordinate of points on the elliptic curve.

$$f : G \in \mathbb{A}_n, \to \mathbf{x}(G) \bmod m$$

Given $z \in \mathbb{Z}_n$, the inverse of z can be calculated as following. First find x in $\mathbb{F}_q$ such that $x \equiv z \pmod{m}$ and x is the x-coordinate of some points on the elliptic curve. Then solve the elliptic equation to recover the y-coordinate (there are two y-coordinates correspondin g to the same x-coordinate so one might need to send an extra bit to differentiate them). This method could find the inverse with probability at least 1/8 of the standard implementation in ECDSA. Thus f is almost-inverse which implies f is almost-invertible as well. $x \in$

3.3   Secure Hash Function

A hash function is defined to be a function

$$h : \{0,1\}^* \to \mathcal{H}$$

where the domain $\{0,1\}^* = \{empty\} \cup (\cup_{i=1}^{\infty} \{0,1\}^i)$ is the set of bit strings of arbitrary length.nd the range is $\mathcal{H}$ which is usually of set $\{0,1\}^n$ where n is related to the security parameter.

Preimage Resistant   A PPT algorithm $\mathcal{V}$ is a (ϵ, τ) inverter of hash function h if $\mathcal{V}$ given input $e \in_R \mathcal{H}$ outputs a message M in the domain within running time at most τ. And $h(M) = e$ with probability at least ϵ. If no such inverter exists, h is preimage resistant or one-way of strength (ϵ, τ)

Second-Preimage Resistant   Let S be an $(\epsilon, \tau, \mathscr{F})$ -second-preimage-finder for hash function h if S on input $M \in_R \mathscr{F}$ outputs $M\prime$ such that $h(M) = h(M\prime), M \neq M\prime$ with probability at least ϵ within running time at most τ. A hash function is said to be second-preimage resistant if no such second-image-finder exists

Collision-Resistant   A (ϵ, τ)-collision finder PPT is C such that within running time at most τ S outputs two different messages with the same hash value with probability at least ϵ. And a hash function is collision-resistant if no such collision finders exist.

Zero-Finder-Resistant   A hash function h is said to be (ϵ, τ) zero-finder-resistant if no PPT algorithm can find a message M with $h(M) = 0$ wihin τ and have a success probability greater than ϵ

Zero-Rarity A hash function is zero-rare of strength $(\epsilon, \mathscr{F})$ if for $M \in_R \mathscr{F}$, the probability that $h(M) = 0$ is at most ϵ.
Zero-finder-resistant implies zero-rarity as otherwise one can always try exhaustive search to find enough messages having hash value zero assuming zero-rarity is not true.
I personally don't understant why the auther mentioned zero-finder-resistant and zero-rarity here. The ECDSA or ECD only fails when the hash value equils zero with neglegible probability.

Uniform Hash Function A uniform hash function is that no PPT algorithm could differentiate the hash function from a true uniformly random function defined on the same domain and range with probability significantly greater than one half.

As an important note, uniformity and collision-resistancy together implies preimage resistant but not the opposite way.

Pseudorandom Generators Though pseudorandomness was not mentioned in the original paper in more detail, here is an overview of it. A pseudorandom generator is a random number generator which every time is called with a seed of bit-length n, produces a number x of bit-length m (with $m > n$, or more accurate m is a polynomial of n) such that no PPT algorithms exist which could differentiate it from a random number sampled from a true uniform distribution (the range of the random numbers should agree).

4   Necessary Security Conditions

Intractable Discrete Logarithm Problem If not, then the adversary is able to compute the private key efficiently and undermine the signature scheme by signing any message the same way as a true signer would do (though the ephemeral key might be different).

Almost-Bijective Conversion Function If not, then the conversion function must be ϵ-clustered for a $z_0 \in \mathbb{Z}_n$ and one can construct a selective $(\epsilon, 1, 0, \mathscr{F})$-forger for any message distribution $\mathscr{F}$ without even making any query to the signing oracle.

$$\text{Select a random } s \in \mathbb{Z}_n$$
$$\text{Output } (z_0, s)$$

This would succeed with probability ϵ as the conversion function is ϵ-clustered.

Preimage Resistancy Suppost there exist in inverter $\mathscr{V}$ of strength (ϵ_V, τ_V) of the hash function h. Since with a selected e

$$f(s^{-1}eG + s^{-1}rQ) = f(aG + bQ) \text{ with } a = s^{-1}e, b = s^{-1}r$$
$$r = f(aG + bQ), e = ab^{-1}r, s\prime = rb^{-1}$$
$$\textbf{Output: } (r, s\prime) \text{ on } M = \mathscr{V}(e)$$

The probability that with arbitrarily selected a and b the computed hash value e falls in the range of the hash function is $|\mathscr{H}|/n$. Thus the probability that the above forger succeeds is $\epsilon_V |\mathscr{H}|/n$

Collistion-Resistant Hash If not, invoke the collison-finder to find two message M' and M such that $h(M) = h(M')$. Then make a query to the signing oracle to get a signature on M. This signature is also a valid signature for M'. Therefore, there exists an existential forger which succeeds with probability the same as the success probabiltiy of the second-preimage-finder.

Second-Preimage-Resistant Hash The same forger for the non-Collision-Resistant hash can break the scheme with the difference being that with a second-preimage-finder, one can find a collision for this specific message and produces the signature as the one queried from the signing oracle on the message that has the same hash value. This forger is a selective forger so it is more harmful if one abandons this requirement for a hash function.

Zero-Finder-Resistant Hash The argument for this property of hash function is similar to the argument for preimage resistancy. Set the a in the argument for preimage resistancy to 0 and the same proof works here.

In addition, one should always check if the r-value of a given signature is zero, otherwise the above existential forger turns into a selective-forger.

Arithmetically Unbiased k-Value It happens that same ephemeral key was acturally used more than once in industry, which completely undermine the secrecy of this signature scheme as one can recover the underlying private key from the signatures generated using the same ephemeral key. Now the stronger statement is that even a slightly biased pseudorandom generator (certain kinds of bias) could be exploited to recover the underlying private key.