Attacks on Two-Key Elliptic Curve Digital Signature Algorithm

N. Anil Kumar and M. Gopi Chand

Abstract In general, digital signature schemes have one parameter as secret and the remaining parameters as public. But in two-key signature scheme, we have two parameters as secret and the remaining as public. In this paper, we analyse the security of the two-key signature scheme and proposed an algorithm to attack two-key signature scheme and named it as Modified Shanks algorithm whose time complexity is $O(n)$.

Keywords Signature schemes · Elliptic curves · Shanks algorithm · ECDLP

1 Introduction

The elliptic curve digital signature algorithm (ECDSA) is a popular signing Algorithm. ECDSA is standardized elliptic curve-based signature scheme, appearing in ANSI X9.62 [1], ISO/IEC 19790:2012, etc. [4] has proposed a two-key signature scheme/algorithm. We will briefly discuss the scheme in Sect. 2. In this paper, we proposed the attack on the two-key signature scheme which will be described in Sect. 3.

2 Two-Key Elliptic Curve Digital Signature Scheme

Elliptic curve digital signature scheme/algorithm (ECDSA) is a signature where the signer uses his private key (one parameter) and hash of the message and generates

N. Anil Kumar (✉) · M. Gopi Chand
Department of IT, Vardhaman College of Engineering, Shamshabad, Kacharam, Hyderabad 500018, Telangana, India
e-mail: anil230@gmail.com

M. Gopi Chand
e-mail: gopi_merugu@yahoo.com

R. S. Bapi et al. (eds.), *First International Conference on Artificial Intelligence and Cognitive Computing* , Advances in Intelligent Systems and Computing 815,
https://doi.org/10.1007/978-981-13-1580-0_21

signature, which is a bit stream. Only the person who has the private key can only generate the valid signature. The person who wants to verify the signature uses the public parameters of the signer and hash of the message to determine whether the signature is valid or not. The security of the ECDSA is based on the assumption that the elliptic curve discrete log problem (ECDLP) is hard. ECDLP can be stated as finding d given P, R of the equation $dP = R$ where P and R are points on elliptic curve and d is a integer [2, 3].

Two-key elliptic curve digital signature algorithm (two-key ECDSA) is a signature scheme [4] where the signer uses his private keys, i.e. two keys (two parameters) and hash of the message, and generates the signature, which is a bit stream. Verification is same as ECDSA. The security is based on the hard problem. Let $dP + d'P' = S$ where P and P' are two base points on the elliptic curve. Given P, P' and S finding d and d' is hard problem.

We define the domain parameters as follows

Let the elliptic curve be $y^2 + x^3 + ax + b$. a, b are the coefficients that define the curve. The curve be defined on the group$\mathbb{F}_q$. The number of points on the elliptic curve is called the order of the curve. Let P and P' are two base point. Let the order of P be n and P' be n. Let i and d' be chosen at random be the private keys and $dP + d'P' = S$. So P, P' and S are the public parameters. H denotes a hash function. We take the hash of the message, if the bits in output of hash is greater than the number of bits of $n \times n'$ we truncate the output of hash so that it will be equal to the number of bits of $n \times n'$. We call the resulting as h.

2.1 Two-Key ECDSA Signature Generation

We use the private key and hash of the message to generate the signature for the given message. The signature generation procedure is as follows:

Input: Domain parameters, private key, hash of message h .
Output: Signature (x_1, s, s')
begin
1 Select r_1 at random in the interval $[1, n-1]$ and r_2 at random in the interval $[1, n'-1]$;
2 Compute $r_1P + r_2P'$ and let it be the point (x_1, y_1) ;
3 **if** $(x_1, y_1) = \mathcal{O}$ **then**
goto step 1
end
4 $s = h\,r_1 + x_1\,d$ and $s' = h\,r_2 + x_1\,d'$;
5 **return** (x_1, s, s');
end

Algorithm 1: Two key signature generation

2.2 Two-Key ECDSA Signature Verification

The signature can be generated by the person who has the private key but for verification private key is not required. The message along with the signature is placed in the public domain. To verify that the message is accepted by the person who has signed, we have to use the signature verification algorithm. As the public parameters are made public, any person can verify that the signature is generated by the person who has the private key.

Input: Domain parameters, public key, hash of message h, signature (x_1, s, s')
Output: validate or invalidate the signature
begin
  Compute $u = sP + s'P'$;
  Substitute x_1 in curve equation and generate two points W_1 and W_2 ;
  Then $W_1 = (x_1, y_1)$ and $W_2 = (x_1, y_1')$;
  Compute $u' = h\,W_1 + x_1 S$;
  if $u = u'$ **then**
    return *(The signature is valid)*
  else if $u' = h\,W_2 + x_1 S$ **then**
    OUTPUT (signature is valid)
  else
    OUTPUT (signature is not valid)
  end
end

Algorithm 2: Two key Signature Verification

3 Attack on Two-Key Signature Scheme

In order to attack the cryptosystem, we have to find d, d' from (x_1, s, s'). The equations which have the secret parameters are $s = hr_1 + x_1 d$ and $s' = hr_2 + x_1 d'$. The known variables are h, s, x_1 and s'. The unknown variables are r_1, r_2, d, d'. From two equations, we cannot solve for four unknown variables. This is an impossible case.

So in order to attack a public key cryptosystem, we have to attack the underlining hard problem. To attack the two-key ECDSA, we have to attack the underlining hard problem. The underlying hard problem is $d\,P + d'\,P' = S$.

We try to solve the underlying problem; we write the equation as

$$d \times P = S - d' \times P'$$

We calculate $S - i \times P'$ for each i in $[0, n_2]$ where n' is the order of P' and store the point and i in the List. The list looks like (point, number), i.e. $(S-1 \times P', 1)$, $(S-2 \times P', 2)$, $(S-3 \times P', 3), \ldots, (S - n \times P', n)$.

Then we compute $P, 2 \times P \ldots$ keep on checking the list for matching point. On finding a match, compute d and d'.

$S - y'P' = y'' \times P$ which implies

$$S = y'' \times P + y'P'.$$

So $d = y''$ and $d' = y'$.

3.1 Analysis

Let us analyse the time complexity. The time complexity of Algorithm 3 is $O(n) + O(n') = O(n+n') = O(2*n) = O(n)(\text{let } n > n')$. As the time complexity is of the order of n, we can conclude that two-key signature is a safe algorithm to use.

Input: P, P', S, elliptic curve.
Output: d, d' satisfying the underlying hard problem.
begin
  let LIST be empty ;
  $S' = S$;
  for i *in* 1 *to* n_2 **do**
    $S' = S' - \times P'$;
    add to the LIST $\{(S', i)\}$;
  end
  The elements on the LIST look like [(x,y),i];
  sort the LIST based on (x,y);
  $T = \mathcal{O}$;
  for j *from* 1 *to* n' **do**
    $T = T + P$;
    search the LIST for T using binary search;
    if T *found in LIST* **then**
      $d = j$;
      $d' = i$;
      return d, d' ;
    end
  end
end

Algorithm 3: Algorithm to solve Two Key problem

4 Conclusion

In this paper, we have briefly explained about two-key ECDSA signature. We proposed an algorithm to attack two-key problem. We analysed its time complexity and found that two-key signature scheme is secure to use.

References

1. E. Barker, W. Barker, W. Burr, W. Polk, and M. Smid. Recommendation for Key Management –Part 1: General. NIST Special Publication, pages 800–857, 2005.
2. Joppe W Bos, J Alex Halderman, Nadia Heninger, Jonathan Moore, Michael Naehrig, and Eric Wustrow. Elliptic curve cryptography in practice. Microsoft Research. November, 2013.
3. Darrel Hankerson, Affred Menezes, and Scott Vanstone. Guide to Elliptic Curve Cryptography. Springer Verlag, 2004.
4. N.Anil Kumar and Chakravarthy Bhagvati. Two key signature scheme with application to digital certificates. In Recent Advances in Information Technology (RAIT), pages 19–22, ISM, Dhanbad, 2012.