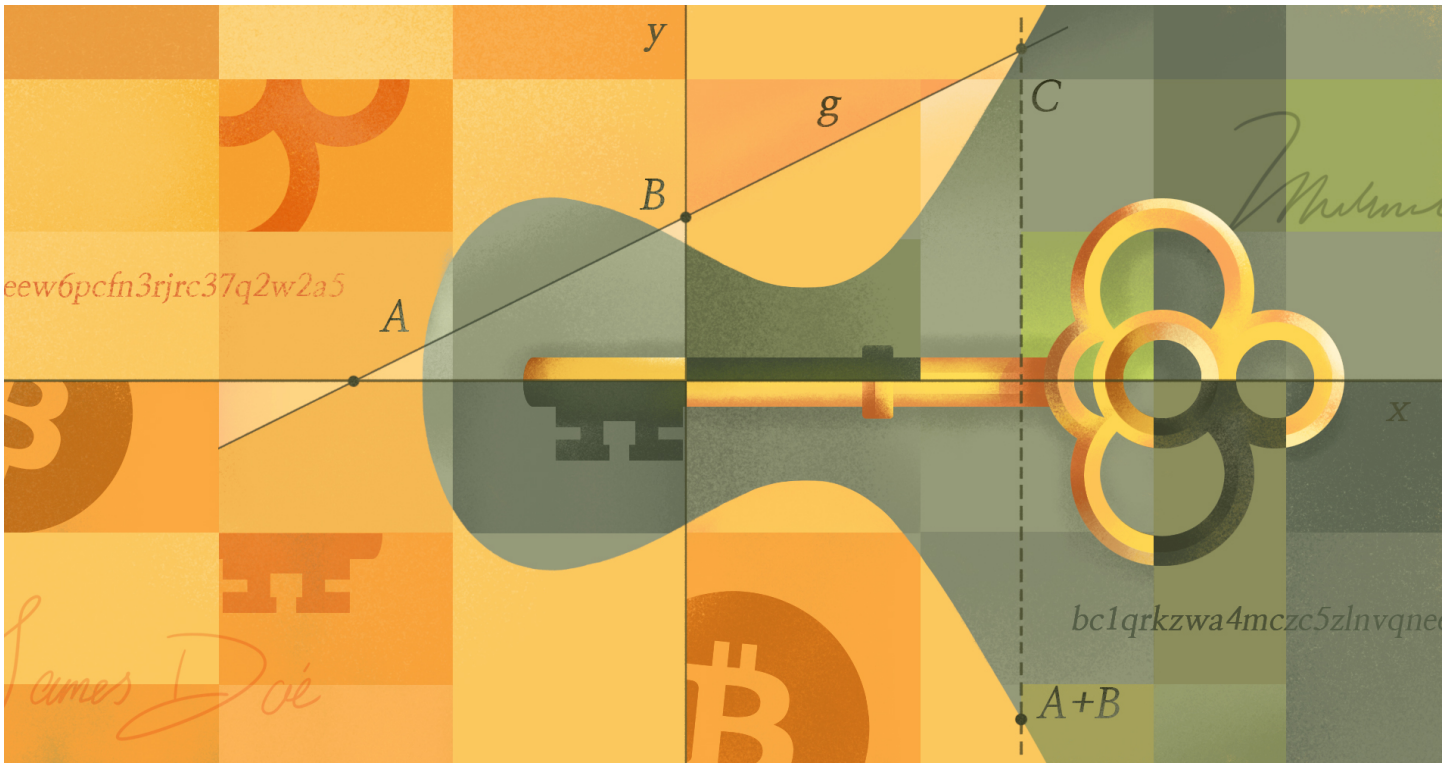
정수론

비트코인 속으로 들어간 타원곡선 [2]: 디지털서명 알고리듬

2022년 11월 18일
이철희



지난 글(클릭 시 1편으로 연결)에서는 비트코인 주소가 만들어지는 과정에 숨겨진 타원곡선의 역할에 대해 이야기했다. 큰 자연수 하나를 비밀키로 선택하면, 그에 대응되는 비트코인 타원곡선의 한 점이 결정된다. 그리고 그 점의 좌표에 해시함수를 적용하면 비트코인 주소가 생성된다. 이 절차는 모두에게 공개되어 있으며, 누구나 실행해 볼 수 있는 것이다. 이런 이야기를 들으면 남몰래 많은 비트코인 주소에 대한 비밀키를 찾아내서 세상에 있는 비트코인을 쓸어 담는 일도 가능하지 않을까 하는 희망찬 생각을 한 번쯤 해보게 마련이다.

내가 생성했다고 주장한 비트코인 주소 bc1qrkzwa4mczc5zlnvqneew6pcfn3rjrc37q2w2a5에 대해 한번 생각해 보자. 이런 형식의 주소는 네이티브 세그윗 주소라 불린다. 가능한 모든 비밀키는 대략 2^{256}개, 네이티브 세그윗 주소는 2^{160}개 존재한다. 가능한 주소의 개수가 비밀키의 수보다 적으므로 이는 각 비트코인 주소가 복수의 비밀키를 가질 수 있음을 의미한다. 랜덤하게 고른 숫자 하나가 이 주소의 비밀키가 될 확률은 $1/2^{160}$이라고 할 수 있다. 이는 동전을 160번 던졌는데 계속해서 앞면만 나오는 일이 일어날 확률과 같은 것이다. 그런 일은 굉장히 일어나기 힘들 것 같지만, 그 어려움의 정도에 대해 좀더 생각을 해보자.

컴퓨터는 지루한 일도 꾸준히 할 수 있으므로, 충분히 오랜 시간 실행한다면 결국엔 특정한 비트코인 주소에 대한 비밀키를 찾을 수 있을 것이라 생각할 수도 있다. 우리가 n개의 비밀키를 랜덤하게 선택해 비트코인 주소를 생성할 때 위에서와 같은 주소가 한 번이라도 얻어질 확률은

$$1-(1-1/2^{160})^n \approx \frac{n}{2^{160}}$$

이다. 넉넉하게 1초에 백경, 즉 1,000,000,000,000,000,000개의 비밀키와 주소를 확인할 수 있는 컴퓨터가 있다고 하면 우주의 나이인 138억 년 동안 계산을 해봤을 때, 위의 주소의 비밀키를 찾을 확률이 3×10^{-13}정도가 된다. 로또 6/45는 1부터 45까지의 숫자 중에서 6개를 잘 고르면 1등으로 당첨이 된다. 로또복권 한 장의 1등 당첨 확률은 8,145,060분의 1, 즉 1.2×10^{-7}이다. 2주 연속으로 로또 1등에 당첨될 확률은 1.5×10^{-14}정도가 된다. 그러니까 특정한 비트코인 주소의 비밀키를 운 좋게 찾는다는 것은, 엄청난 성능의 컴퓨터를 가지고 우주의 나이만큼 계산을 해봤을 때 로또 1등에 2주 연속으로 당첨되는 정도의 행운이 있어야 하는 일이다. 그러니 남의 공개된 주소에 대한 비밀키를 찾는 것은 사실상 불가능하다. 비밀키를 잃어버린 사람이 그걸 되찾고 싶다면 할 수 있는 일이라고는 양자컴퓨터의 등장을 기다려 보는 것 정도라고 할 수 있는데, 이에 대해서는 뒤에서 더 이야기해보도록 하자.

비트코인 주소의 비밀키라는 것은 이렇듯 그 소유자 외에는 접근이 불가능한 것이다. 이 글에서는 비밀키의 소유자가 실제로 비밀키를 소유하고 있다는 사실을 타원곡선을 이용해 어떻게 증명할 수 있는지에 대해 이야기할 것이다. 그리고 내가 실제로 위의 비트코인 주소에 대한 비밀키를 소유하고 있다는 사실을 모두가 확인할 수 있는 방식으로 제시할 것이다.

비트코인 채팅방: 노드 네트워크

우리가 은행을 이용해서 송금할 때는, 어떤 방식을 사용해서든 '계좌 A에서 계좌 B로 얼마만큼의 금액을 보낸다'는 메시지를 은행에 전해야 한다. 이때 은행이 반드시 확인해야 할 것은 이 메시지가 계좌 A의 소유자에게서 왔다는 점일 것이다. 은행을 직접 방문한다면, 신분증과 계좌의 정보를 비교하게 될 것이다. 대리인을 통한다면, 그보다 더 복잡한 확인 절차를 거쳐야 한다. 은행의 앱을 사용한다면 앱에 로그인하는 과정을 통해 계좌의 소유자를 확인할 수 있다. 이러한 확인 절차가 있어야만 거래가 다음 단계로 진행된다.

비트코인처럼 신뢰할 수 있는 중개자를 없애 버린 시스템은 이런 과정을 다른 방식으로 진행해야 한다. 은행이 없는 대신, 여기엔 비트코인 사용자들이 모여서 끊임없이 떠들고 있는 거대한 채팅방과 같은 것이 있다고 할 수 있다. 더 정확히 말하면, 비트코인 노드들이 소통하고 있는 네트워크가 존재한다. 채팅방이라고는 하지만, 이것은 우리에게 익숙한 단체 채팅방 같은 곳과는 또 다르다. 중개자가 없다는 말은 채팅방을 책임지고 운영하는 회사와 같은 주체가 없고, 대화를 진행하는 진행자도 없으며, 모두가 동시에 보고 있는 단일한 대화창도 없다는 것을 의미한다. 이곳은 아무나 신분을 숨기고 들어와서 자기 눈앞에 보이는 사람들에게 아무 말이나 외치고 아무 때나 사라질 수 있는 도떼기시장에 가깝다. 객관적인 판단을 내려줄 더 높은 권한의 참여자도 없으므로, 남의 말을 듣고 그것을 옳다고 받아들일지의 여부도 각자의 주체적인 판단에 맡겨진다. 들은 말을 옮겨 소문을 낼 것인지, 아니면 그냥 무시할 것인지도 각자의 선택에 달렸다.

비트코인 주소가 A인 사람이 자신의 코인을 옮기려면 이 채팅방에서 '주소 A에서 주소 B로 얼마만큼의 비트코인을 보낸다'는 말을 크게 외쳐야 한다. 사람들이 이게 말이 된다고 여긴다면, 이 외침이 소문처럼 계속 퍼질 것이고 머지않아 블록체인 장부에 'A가 B에게 얼마를 주었다'라고 기록하는 사람이 생겨날 것이다. 장부에 소유권의 변경이 기록되면 메시지의 외침에서 시작된 거래가 되돌릴 수 없이 온전히 끝나는 것이다. 그런데 누구나 아무 말이나 외칠 수 있는 곳에서 이런 절차가 제대로 돌아가려면, 메시지가 애초에 주소 A의 주인 즉 비밀키의 소유자에게서 처음 나온 것이라는 사실을 확인할 수단이 있

어야만 한다. 그런 것이 없다면, 이 채팅방에서는 아무나 남의 주소에 있는 비트코인을 자신의 주소로 옮기라고 외치는 말만 난무할 것이기 때문이다. 모두가 자신이 주소 A의 주인이라고 외치기 시작한다면, 이 도떼기시장 같은 곳에서 어떻게 진짜 주인의 외침을 가려낼 수 있을까?

타원곡선 디지털서명 알고리듬

바로 이 장면에서 타원곡선 디지털서명 알고리듬elliptic curve digital signature algorithm이 등장한다. 각 영어 단어의 앞글자를 따서 보통 ECDSA라고 부른다. 이 알고리듬은 비트코인 주소의 비밀키와 비밀키의 소유자가 전파하고자 하는 메시지를 입력받아, 서명이라 불리는 것을 생성한다.

일상에서 우리는 종이 문서에 담긴 내용을 확인한다는 뜻에서 펜으로 서명을 하곤 한다. ECDSA에서의 서명도 이런 익숙한 서명과 비슷한 기능을 한다. 하지만 몇 가지 점에서는 크게 다르다. 펜을 이용한 서명은 다른 사람도 적당한 노력을 통해 만들어 낼 수 있는 것이고, 이 서명이 진짜인지를 가리는 일은 서명 자체만 가지고 할 수 있는 일이 아니다. 박찬호의 사인볼을 다른 여러 사람을 통해 전해받는다면, 공만 가지고서 그게 진짜인지 가짜인지 누가 알겠는가? 하지만 ECDSA를 이용한 서명은 오직 비밀키의 소유자만이 만들어낼 수 있기에 위조가 사실상 불가능하다. 서명을 하지 않았다고 오리발을 내밀 수도 없다. 서명을 가지고 위조 여부를 누구나 투명하게 확인할 수 있다는 점도 크게 다르다. ECDSA에서의 서명은 할 때마다 매번 바뀌며, 그냥 큰 숫자일 뿐이라는 점에서도 다르다.

ECDSA의 핵심에는 타원곡선을 이용해 비밀키의 소유자만이 풀 수 있는 일종의 방정식을 만드는 과정이 있다. 이 방정식을 답과 함께 채팅방에서 외치면, 누구나 이것이 비밀키 소유자의 외침이라는 사실을 의심의 여지없이 검증할 수 있다. ECDSA에서의 서명이란 본질적으로 이 방정식의 답이다. 최종적인 판단을 내려줄 권위의 소유자를 없애버린 비트코인 채팅방에서 각자의 참여자들이 말이 되는 것과 안 되는 것을 판단하기 위해 사용하는 것은 다름 아닌 수학이다.

ECDSA를 좀 더 자세히 살펴보도록 하자. 비트코인은 고정된 소수 $p = 2^{256} - 2^{32} - 977$와 유한체 $\mathbb{F}_p$위의 타원곡선 $y^2 = x^3 + 7$을 사용한다. 점 $G = (x_1, y_1)$는 이 타원곡선의 특별한 점으로, 그 좌표는

$$x_1 = 55066263022277343669578718895168534326250603453777594175500187360389116729240,$$
$$y_1 = 32670510020758816978083085130507043184471273380659243275938904335757337482424$$

이다. 이 점을 반복하여 더하면, 이 타원곡선에 있는

$$n = 115792089237316195423570985008687907852837564279074904382605163141518161494337$$

개의 점을 차례로 얻을 수 있다. 비트코인 주소의 비밀키는 n보다 작은 자연수 k이며, 공개키는 타원곡선의 점 $P = kG$이다. 이 비밀키의 소유자가 서명을 생성하기 위해서는 먼저 다음과 같은 절차를 밟는다:

1. 전파하려는 메시지 m에 약속된 해시함수 h를 적용해 자연수 $z = h(m)$를 계산
(남에게 공개하는 정보)
2. 1부터 $n-1$사이의 임의의 자연수 ℓ를 하나 선택 (남에게 공개하면 안 되는 정보)
3. 선택한 자연수 ℓ을 이용해 타원곡선의 점 $R = \ell G = (r, y)$을 계산 (남에게 공개하는 정보)

여기까지의 과정을 거쳐 타원곡선 위의 세 점 G, P, R과 자연수 z, r을 갖게 되었다. 이제 미지수를 s로 갖는 방정식

$$zG + rP = sR \tag{1}$$

을 생각해 보자. 어떤 자연수 s가 이 방정식을 성립하게 하는 경우, (R, s)를 메시지의 해시 z에 대한 서명이라 한다.

비밀키의 소유자는 매우 쉽게 이 방정식을 풀 수 있다. 왜냐하면, $P = kG$, $R = \ell G$라는 사실을 이용해 이 방정식을

$$zG + r(kG) = s(\ell G)$$

로 쓸 수 있기 때문이다. 따라서

$$s = \ell^{-1}(z + rk) \mod n \tag{2}$$

이어야 한다.

누구나 세 점 G, P, R과 세 자연수 z, r, s가 주어져 있을 때, 식 (1)이 성립하는지 아닌지를 쉽게 확인할 수 있다. 이것은 타원곡선에서의 덧셈만을 필요로 하는 것이기 때문이다. 하지만 비밀키 k와 임의의 수 ℓ을 모른다면, 식 (1)에서 G, P, R, z, r가 주어져 있다고 해도, s를 구할 수 없다. 이는 다름 아닌 타원곡선에 대한 이산 로그 문제이기 때문이다. 그러므로 이런 s의 값이 비트코인 채팅방에서 떠돈다면, 그것은 비밀키의 소유자가 z를 해시값으로 갖는 메시지를 세상에 외쳤음을 의미한다. 그것이 누구의 입을 통해 어떤 경로를 거쳐서 전해지더라도 그 결론에는 변화가 없다. 그러니 비트코인 채팅방에 자신의 말을 대신 전해줄 누군가가 있다면, 비트코인 거래는 모스 부호를 사용해서도 시작될 수 있는 것이다.

서명자가 한 가지 주의할 점은 z, r, s는 모두 남에게 공개되는 정보이기 때문에, ℓ이 공개될 경우 다른 사람들이 (2)를 이용해 비밀키인 k의 값도 알아낼 수 있다는 것이다. 따라서 서명자는 임의로 선택한 자연수 ℓ을 비밀로 유지해야만 한다. 이 점에 대해서는 뒤에서 한번 더 살펴볼 것이다.

비트코인 주소 bc1qrkzwa4mczc5zlnvqneew6pcfn3rjrc37q2w2a5에 대한 서명

이제 위에서 언급한 비트코인 주소 bc1qrkzwa4mczc5zlnvqneew6pcfn3rjrc37q2w2a5에 대한 ECDSA를 이용한 서명을 구체적으로 보도록 하자. 이 주소에 대응되는 공개키 $P = (x_P, y_P)$의 좌표는

$$x_P = 74911881170087217160727624520586491041842985584706548062197841168199893840628,$$
$$y_P = 15150952392829518070294151181456990267246399241330880099141153295825441191914$$

이다. 지난 글에서 링크했던 코드(링크 https://github.com/chlee-0/btc_native_segwit_address)를 적절히 활용하면, 이 공개키로부터 위의 비트코인 주소가 실제로 얻어지는지를 확인할 수 있다. 서명을 하려면, 서명의 대상이 될 메시지가 있어야 하므로 나는 “Don’t Trust, Verify”를 메시지로 선택하였다. 이 메시지에 대한 SHA-256 해시함수의 값은 10진법에 해당하는 숫자로 쓸 때

$$z = 112761469845056919304416565170674990599925418308225593267094103636319905743526$$

이 된다. 방정식 (1)에서 서명자가 임의로 선택하는 점 $R = (r, y)$의 좌표를 나는 다음과 같이 택하였다:

$$\begin{aligned} r &= 363222602425676443275774719148517271610174587059581271709152367154258193330 73, \\ y &= 113817104126258647026551196310596962231430747658282676626153792268748326724326. \end{aligned}$$

이제 자연수 s를 다음과 같이 두자:

$$s = 467103350281427280883664233490704674226026335334709803211541255132991919486 26.$$

이제 방정식 (1)에 등장하는 세 점 G, P, R과 세 자연수 z, r, s이 모두 공개되었다. 이로부터 식(1)이 실제로 성립하는지의 여부를 이제 누구나 확인할 수 있다. 이런 s를 찾아낼 수 있는 사람은 비밀키를 소유한 사람뿐이므로, 이는 내가 비밀키를 실제로 소유하고 있고 “Don’t Trust, Verify”라는 메시지에 서명했음을 증명한다. 이를 확인하는 타원곡선 계산을 실제로 해보고 싶은 사람들은 여기에 있는 코드(링크 https://github.com/chlee-0/btc_native_segwit_address/blob/main/ecdsa_verification.ipynb)를 참고할 수 있다. 나의 모든 주장을 의심한다고 해도, 이것은 누구나 스스로 타당성을 검증할 수 있고 결국엔 받아들일 수밖에 없는 사실이다. 이런 방식으로 수학을 통해 도떼기시장 같은 비트코인 채팅방에 질서가 생겨나기 시작한다.

ECDSA에서 주의할 점

ECDSA의 방정식 (1)에 등장하는 타원곡선의 한 점 $R = \ell G$은 서명자가 임의로 선택하는 것이다. 그런데 이 점을 다음번 서명에 다시 활용해서는 절대로 안된다. 이를 두 차례의 서명을 얻는데 활용하는 경우, 비밀로 지켜져야 하는 두 자연수인 비밀키 k와 임의로 선택된 숫자 ℓ 사이에 성립하는 두 개의 관계식

$$\begin{cases} z_1 + rk = \ell s_1 \mod n \\ z_2 + rk = \ell s_2 \mod n \end{cases}$$

을 (2)로부터 끌어낼 수 있다. 여기서 (z_i, r, s_i)는 모두에게 공개되는 것이기 때문에, 이는 다른 사람들이 비밀키 k를 알아내기에 충분한 정보를 제공한다.

ECDSA의 이런 점과 관련해 2010년에 흥미로운 사건이 일어나기도 했다. 소니의 플레이스테이션 3은 기기 내부에서 ECDSA를 활용하고 있었는데, 위에서처럼 재사용하면 안 되는 난수를 바꾸지 않고 사용하는 치명적인 실수를 저질렀다.[1] 해커들은 이 취약점을 이용해, 기기에서 소니가 허가한 게임 이외에 아무 코드나 실행할 수 있는 권한을 탈취할 수 있었다. 이는 비유하자면 아이폰의 앱스토어나 안드로이드폰의 플레이스토어 밖의 세계에 있는 프로그램들을 가져와서 스마트폰에서 실행하는 일을 가능케 하는 것이다.

ECDSA에 대해 주의를 기울여야 할 또 다른 점은 양자컴퓨터와 관련이 있다. 비트코인의 주소는 공개키로부터 유도되고, 공개키는 비밀키로부터 유도된다. 비트코인을 받을 때 남에게 알려줘야 하는 정보는 비트코인 주소뿐이다. 그러나 위에서 보았듯 가지고 있는 비트코인을 다른 주소로 옮기려면 ECDSA를 이용해 서명을 해야 하고, 공개키를 세상에 드러내야 한다. 그래야만 서명이 올바른 것인지 다른 사람들이 확인할 수 있기 때문이다. 물론 공개키로부터 비밀키를 알아내는 것은 현재의 기술 수준에서는 불가능한 것이다. 하지만 공개키가 알려진 비트코인 주소는 어떤 면에서는 그렇지 않은 주소보다 보안의 측면에서 더 취약해진 것도 사실이다.

미래의 어느 순간 양자컴퓨터가 등장한다면, 이것은 현실적인 문제가 될 수도 있다. 왜냐하면, 타원곡선의 이산 로그 문제에 대한 쇼어 알고리듬이 성능이 좋은 양자컴퓨터에서 실행된다면, 공개키로부터 비밀키를 알아내는 것이 가능할 수도 있기 때문이다. 물론 이는 양자컴퓨터 기술의 상당한 발전을 필요로 하는 것이지만, 언젠가는 일어날지도 모르는 일이다.[2] 따라서 같은 비트코인 주소를 비트코인을 보내는 용도로 재활용하지 않는 것은 좋은 보안 수칙이라고 할 수 있다. 그리고 이는 비교적 간단하게 실천할 수 있는데, 비트코인을 보내고 남게 되는 금액은 모두 새 주소를 만들어 옮기면 되기 때문이다. 이렇게 하면 공개키의 노출도 피할 수 있고, 소니가 저지른 ECDSA에서의 난수 재사용 실수도 원천 차단할 수 있다. 사실 많은 지갑 소프트웨어는 이미 이런 방식으로 작동한다.

공개키가 드러나지 않은 비트코인 주소라면 공개키를 알아낸 이후에야 비밀키를 찾아낼 수 있다. 주소는 공개키에 해시함수를 적용해 얻어진다. 그런데 비트코인에서 활용하는 SHA-256와 RIPEMD-160과 같은 해시함수는 아직까지는 양자컴퓨터의 직접적인 위협에서는 벗어나 있다. 공개키에서 비밀키를 찾는 일은 양자컴퓨터가 기존의 컴퓨터보다 압도적으로 잘할 수 있지만, 해시함수에 대해서 그 압도적인 정도가 덜 한 것으로 알려져 있다.[3] 그렇기 때문에 양자컴퓨터가 나오더라도 공개키가 노출되지 않은 비트코인 주소의 비밀키가 밝혀질 위험은 크게 높지 않다. 그러나 양자컴퓨터 기술이 발전하여 비트코인의 블록체인 업데이트에 걸리는 10분 정도의 시간에 공개키로부터 비밀키를 찾아낼 수 있는 수준까지 도달한다면, 결국엔 비트코인이 활용하고 있는 ECDSA 체계를 적절히 손봐야 할 것이다. 비트코인의 블록체인 장부를 업데이트하는 과정인 채굴에도 해시함수가 중요하게 활용되므로, 해시함수에 대한 양자컴퓨터의 위협 역시 앞으로 많은 관심을 받는 주제일 것이다.

남겨진 이야기들

지난 글에서는 비트코인의 주소를 통해 타원곡선의 기초적인 내용을 소개하였고, 여기서는 비트코인의 거래에서 중요하게 활용되는 타원곡선 디지털서명 알고리듬에 대해 이야기하였다. 비트코인에 대한 타원곡선의 응용과 관련하여 또 다른 흥미로운 개념은 슈노르 서명Schnorr signature이라 불리는 것이다. 타원곡선을 이용해 서명을 생성한다는 면에서는 위에서 다룬 ECDSA와 유사하지만, 슈노르 서명의 선형성과 같은 성질은 여러 명이 하나의 서명을 생성하는 다중 서명에 매우 적합하다. 이것도 여러 측면에서 새로운 가능성을 열어주는 것인데, 이에 대한 흥미로운 이야기들은 나중에 더 다룰 수 있는 기회를 찾아보도록 하자.

참고문헌

1. Casey Johnston, PS3 hacked through poor cryptography implementation, Ars Technica. 2010-12-31. https://arstechnica.com/gaming/2010/12/ps3-hacked-through-poor-implementation-of-cryptography/
2. Roetteler, M., Naehrig, M., Svore, K.M., Lauter, K. (2017). Quantum Resource Estimates for Computing Elliptic Curve Discrete Logarithms. In: Takagi, T., Peyrin, T. (eds) Advances in Cryptology – ASIACRYPT 2017. ASIACRYPT 2017. Lecture Notes in Computer Science, vol 10625. Springer, Cham. https://doi.org/10.1007/978-3-319-70697-9_9
3. Amy, M., Di Matteo, O., Gheorghiu, V., Mosca, M., Parent, A., Schanck, J. (2017). Estimating the Cost of Generic Quantum Pre-image Attacks on SHA-2 and SHA-3. In: Avanzi, R., Heys, H. (eds) Selected Areas in Cryptography – SAC 2016. SAC 2016. Lecture Notes in Computer Science, vol 10532. Springer, Cham. https://doi.org/10.1007/978-3-319-69453-5_18