NTNU | Norwegian University of Science and Technology

RANDOMNESS 3: BREAKING ECDSA

TTM4205 – Lecture 4

Caroline Sandsbråten

30.08.2024

Contents

Contents

Elliptic Curves

- Let $\mathbb{F}_p$ be a finite field of prime order p.
- $E_{a,b} = \{(x, y) \in \mathbb{F}_p \quad | \quad y^2 = x^3 + ax + b\} \cup \{\mathcal{O}\}$
- Given two points $P = (x_1, y_1)$ and $Q = (x_2, y_2)$ on $E_{a,b}$ we can compute $P + Q$ as follows
  - If $P = \mathcal{O}$, then $P + Q = Q$.
  - If $x_1 = x_2$ and $y_1 = -y_2$, then $P + Q = \mathcal{O}$.
  - Otherwise, let $x_3 = \lambda^2 - x_1 - x_2$ and $y_3 = -y_1 - \lambda \mod (x_3 - x_1)$, where

$$\lambda = \begin{cases} \frac{3x_1^2 + a}{2y_1} & \text{if } P = Q \\ \frac{y_1 - y_2}{x_1 - x_2} & \text{otherwise,} \end{cases}$$

    and output $R = (x_3, y_3)$.
- Scalar multiplication of points is denoted as $Q = [x]P$ where $2P = P + P$.

Why Elliptic Curves?

Hard problems

- (DLP) Let p be a prime, and let a, b be integers such that $a \mod p \neq 0$ and $b \mod p \neq 0$. Assume there exists an integer x such that $a^x \equiv b \mod p$ The DLP is then to find x such that $a^x \equiv b \mod p$. More generally, we have the following.

- Using Elliptic Curves, the same problems becomes the ECDLP:

- (ECDLP) Let $P, Q \in E(\mathbb{F}_p)$, where $E(\mathbb{F}_p)$ is an ellitpic curve and p is prime. P and Q is points on $E(\mathbb{F}_p)$. The ECDLP is then to find an integer x satisfying the equation $[x]P = Q$.

Contents

ECDSA Signature Algorithm

(Input): Message m, private key sk, the elliptic curve $E_{a,b}$, and the domain parameters, G, and n.

(Output): Digital signature r, s.

(Algorithm):

$h \leftarrow H(m)$
$k \leftarrow_{\$} \{0, \ldots, n\}$
$Q = (x, y) \leftarrow kG$
$r \leftarrow x \mod n$
$s \leftarrow k^{-1} \cdot (h + r \cdot \mathsf{sk}) \mod n$
return r, s

- What would happen if k is not random?

ECDSA Signature Verification

(Input): Message m, public key Q, the elliptic curve E, and domain parameters of the elliptic curve G, and n.

(Output): Boolean value. True if the signature is verified as being correct, False if not.

ECDSA Signature Verification

if $Q = O$ or Q is not on E **then**
  return False
end if
$h \leftarrow H(m)$
$u_1 := h \cdot s^{-1} \mod n$
$u_2 := r \cdot s^{-1} \mod n$
$(x, y) := u_1 \cdot G + u_2 \cdot Q$
if (x, y) = $\mathcal{O}$ **then**
  return False
end if
if $r \equiv x \mod n$ **then**
  return True
end if
return False

Contents

Reused randomness

- If the same k is used to sign two different messages, the private key can be recovered.
- This is because the signature is (r, s) where $r = x_1 \mod n$ and $s = k^{-1}(z + r\mathrm{sk}) \mod n$.
- If k is reused, then $s_1 = k^{-1}(z_1 + r_1 \cdot \mathrm{sk})$ and $s_2 = k^{-1}(z_2 + r_2 \cdot \mathrm{sk})$.

$$
\begin{aligned}
s_1 - s_2 &= k^{-1}(z_1 + r_1 \cdot \mathrm{sk}) - k^{-1}(z_2 + r_2 \cdot \mathrm{sk}) \\
s_1 - s_2 &= k^{-1}(z_1 - z_2 + r_1 \cdot \mathrm{sk} - r_2 \cdot \mathrm{sk}) \\
s_1 - s_2 &= k^{-1}(z_1 - z_2 + (r_1 - r_2) \cdot \mathrm{sk}) \\
s_1 - s_2 &= k^{-1}(z_1 - z_2) \\
k &= \frac{z_1 - z_2}{s_1 - s_2}
\end{aligned}
$$

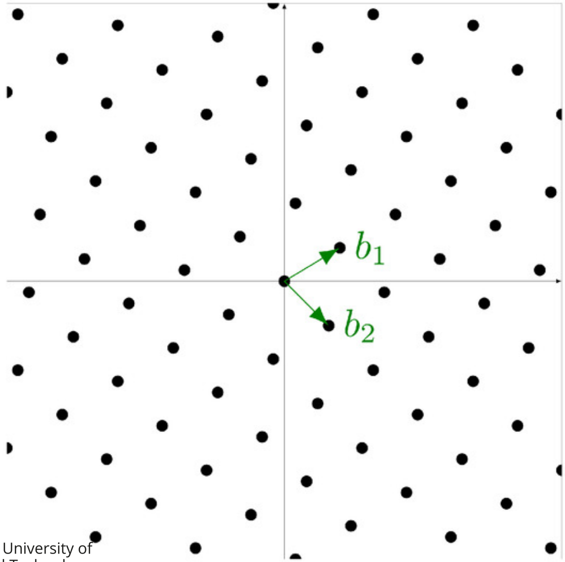
Lattices

Definition

Let $B = [b_1, \ldots, b_k] \in \mathbb{R}^{n \cdot k}$ be a linearly independent set in $\mathbb{R}^n$. A lattice, denoted $\Lambda(B)$, that is generated by matrix B is the set of all linear combinations of the columns of B with integer coefficients. B is thus a basis for lattice $\Lambda(B)$.

$$\Lambda(B) = \left\{ Bx : x \in \mathbb{Z}^k \right\} = \left\{ \sum_{i=1}^{k} x_i \cdot b_i : x_i \in \mathbb{Z} \right\}$$

Lattices (intuition)



Lattice Problems

Definition (Shortest Vector Problem.)
Given a lattice Λ, find a vector $v \in \Lambda \setminus \{0\}$ such that $||v|| \leq ||u_i|| \forall u_i \in \Lambda \setminus \{0\}$

Lattice Problems

Definition (Shortest Vector Problem.)
Given a lattice Λ, find a vector $v \in \Lambda \setminus \{0\}$ such that $||v|| \leq ||u_i|| \forall u_i \in \Lambda \setminus \{0\}$

Definition (Closest Vector Problem.)
Given a lattice Λ, and a vector u, find the lattice vector v such that $||u - v|| \leq ||u - v_i||, \forall v_i \in \Lambda$.

Solving Lattice Problems

Algorithm 1 LLL (Simplified)

Input: A basis $B = \{b_1, b_2, \ldots, b_n\}$ for a lattice Λ
Output: A reduced basis $B' = \{b'_1, b'_2, \ldots, b'_n\}$ where vectors are shorter and nearly orthogonal

Step 1: Gram-Schmidt Orthogonalization
Compute the Gram-Schmidt orthogonalization $\tilde{B} = \{\tilde{b}_1, \tilde{b}_2, \ldots, \tilde{b}_n\}$ of the basis B.
Step 2: Size Reduction
for $k = 2$ to n **do**
  for $j = k - 1$ to 1 **do**
    Set $\mu_{k,j} = \frac{b_k \cdot \tilde{b}_j}{\|\tilde{b}_j\|^2}$
    if $|\mu_{k,j}| > \frac{1}{2}$ **then**
      Update $b_k = b_k - \lfloor \mu_{k,j} \rceil b_j$
    end if
  end for
end for
Step 3: Lovász Condition Check
for $k = 2$ to n **do**
  if $\|\tilde{b}_k\|^2 < \left(\delta - \mu_{k,k-1}^2\right) \|\tilde{b}_{k-1}\|^2$ **then** ▷ $\delta \in (1/4, 1)$ is a constant
    Swap b_k and b_{k-1}
    Recompute Gram-Schmidt orthogonalization for the updated basis
    Go back to Step 2
  end if
end for
Step 4: Return the reduced basis
return $B' = \{b'_1, b'_2, \ldots, b'_n\}$

How is this helping us?

- With *luck* the shortest vector in the new basis is the shortest vector in the lattice.
- It should at least be closer to the shortest vector than the original basis.
- So how will we use this? Let us look at ECDSA signatures that use short randomness.

Short Randomness

Prerequisites

We have m number of signatures on the form

$$s_i \equiv k_i^{-1}(h_i + r_i \cdot \mathsf{sk}) \mod p, \quad \text{for } i \in [m]$$

If the randomness is “too short” we can assume the rest of the MSB are 0. In our first example, let us have $m = 3$ signatures signed using 128-bit randomness, while in reality it should be 256 bit randomness for security.

What can we do?

First, we now know that each randomness k_i is on the form

$$k_i = 2^{128}a + b_i, \quad b_i < 2^{128}$$

Short randomness: What can we do?

$$s_i \equiv k_i^{-1}(h_i + r_i \cdot \mathsf{sk}) \mod p$$
$$s_i \equiv (2^{128}a + b_i)^{-1}(h_i + r_i \cdot \mathsf{sk}) \mod p$$
$$b_i + 2^{128}a \equiv s_i^{-1}(h_i + r_i \cdot \mathsf{sk}) \mod p$$
$$b_i + 2^{128}a \equiv s_i^{-1}h_i + s_i^{-1}r_i \cdot \mathsf{sk} \mod p$$

Now we have some equation describing the randomness k_i. But how can we actually use this to recover sk?

- First, we know that $a = 0$ because of our short randomness.

$$\color{red}{b_i} \equiv s_i^{-1}h_i + s_i^{-1}r_i \cdot \color{red}{\mathsf{sk}} \mod p$$

Answer: The Hidden Number problem

- Our problem: Recovering an unknown scalar sk, knowing only partial information about multiples of the scalar.
- What we know: Some partial information about the randomness k_i
- We also know: s_i, r_i, h_i, p.
- So we can reformulate our problem a bit to make it easier to deal with by letting $t_i = s_i^{-1} r_i$ and $u_i = s_i^{-1} h_i$. We also have that $a = 0$ because our randomness is short. We then have

$$b_i \equiv t_i \cdot \mathsf{sk} + u_i \mod p$$

Formalizing the Hidden Number Problem (HNP)

$$
\begin{aligned}
&\text{Adversary is given } m \text{ pairs of integers } \{(t_i, u_i)\}_{i=1}^{m} \\
&\text{Such that } t_i x - u_i \mod p = b_i \\
&\text{Where } |b_i| < B, \text{ for some } B < p
\end{aligned}
\tag{1}
$$

Solving the Hidden Number Problem

Let us set up our problem as a system of linear equations, assuming b_i is 128 bit long (128 0-bits preceding it to form k_i), and $m = 3$ is our amount of signatures:

$$\begin{aligned}
b_1 &\equiv t_1 \cdot \mathsf{sk} + u_1 \mod p \\
b_2 &\equiv t_2 \cdot \mathsf{sk} + u_2 \mod p \\
&\vdots \\
b_m &\equiv t_m \cdot \mathsf{sk} + u_m \mod p
\end{aligned}$$

We know that b_i should be relatively short, so this should be able to be formed as an instance of the shortest vector problem, and (hopefully) solved using LLL.

Solving the Hidden Number Problem

Our goal is now to construct a lattice where the shortest vector in the lattice is our solution. Setting up our system of equations as a matrix equation yields us:

$$\begin{bmatrix} j_1 & j_2 & j_3 & \mathsf{sk} & 1 \end{bmatrix} \begin{bmatrix} p & 0 & 0 & 0 & 0 \\ 0 & p & 0 & 0 & 0 \\ 0 & 0 & p & 0 & 0 \\ t_1 & t_2 & t_3 & B/p & 0 \\ u_1 & u_2 & u_3 & 0 & B \end{bmatrix} = \begin{bmatrix} b_1 & b_2 & b_3 & \mathsf{sk} \cdot B/p & B \end{bmatrix}$$

Because all the b_i's are short, we can assume that the shortest vector in the lattice is the solution to our problem, and calculating our secret key sk should after this just be a problem of simple arithmatic.

Partially equal randomness

But what if $a \neq 0$? We are not generating a too short randomness, but instead our PRF is broken making each k_i partially equal, but not entirely. Unfortunatly we don't know what this shared randomness is. Can we still recover our secret key sk?

Partially equal randomness

Recall:

$$s_i \equiv k_i^{-1}(h_i + r_i \cdot \mathrm{sk}) \mod p$$

and

$$k_i = 2^{128}a + b_i, \quad b_i < 2^{128}, a \neq 0$$

$$2^{128}a + b_i \equiv s_i^{-1}h_i + s_i^{-1}r_i \cdot \mathrm{sk}) \mod p$$

Partially equal randomness

Recall the equations describing $k_i = 2^{128}a + b_i$:

$$\begin{aligned}
2^{128}a + b_1 &\equiv s_1^{-1}h_1 + s_1^{-1}r_1 \cdot \mathsf{sk} \mod p \\
2^{128}a + b_2 &\equiv s_2^{-1}h_2 + s_2^{-1}r_2 \cdot \mathsf{sk} \mod p \\
2^{128}a + b_3 &\equiv s_3^{-1}h_3 + s_3^{-1}r_3 \cdot \mathsf{sk} \mod p
\end{aligned}$$

What is the problem with solving this? How can we fix it?

Partially equal randomness

Subtracting equation 3 from 1 and 2 yields us:

$$
\begin{aligned}
b_1 - b_3 &\equiv (s_1^{-1} h_1 - s_3^{-1} h_3) + (s_1^{-1} r_1 - s_3^{-1} r_3) \cdot \mathsf{sk} \mod p \\
b_2 - b_3 &\equiv (s_2^{-1} h_2 - s_3^{-1} h_3) + (s_2^{-1} r_2 - s_3^{-1} r_3) \cdot \mathsf{sk} \mod p
\end{aligned}
$$

And $b_i - b_3$ is still short. Every other factor is big, and finding the shortest vector in a lattice constructed as before should solve our problem.

$$\begin{bmatrix} j_1 & j_2 & \mathsf{sk} & 1 \end{bmatrix} \begin{bmatrix} p & 0 & 0 & 0 \\ 0 & p & 0 & 0 \\ 0 & 0 & 0 & 0 \\ t_1 - t_3 & t_2 - t_3 & B/p & 0 \\ u_1 - u_3 & u_2 - u_3 & 0 & B \end{bmatrix} = \begin{bmatrix} b_1 - b_3 & b_2 - b_3 & \mathsf{sk} \cdot B/p & B \end{bmatrix}$$

should, when LLL-reduced give us a new basis containing the shortest vector in the lattice, which contains our secret key sk.

Contents

Setting up our parameters (secp256k1)

```
p = 0xFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFEFFFFFC2F
E = EllipticCurve(GF(p), [0, 7])
G = E([0x79BE667EF9DCBBAC55A06295CE870B07029BFCDB2DCE28D959F2815B16F81798, 0
    x483ADA7726A3C4655DA4FBFC0E1108A8FD17B448A68554199C47D08FFB10D4B8])
n = G.order()
nl = int(n).bit_length()
## Create private key
d = randrange(1, n-1)
## Create public key
Q = d*G
## Function to reduce mod n
N = Zmod(n)
```

Signing messages

```
# Number of messages we capture
m = 3
messages = [f"message {i}".encode() for i in range(m)]

## Length of randomness used
T = 2^128
K = [randrange(1, T) for _ in range(m)]
print(K)
H = [int.from_bytes(sha256(m).digest()[:nl//8], "big") for m in messages]

Points = [int(K[i])*G for i in range(m)]

X = [P[0] for P in Points]
R = [N(x) for x in X]
S = [(H[i] + d*R[i])/N(K[i]) for i in range(m)]
```

Recovering this

- I will not show how to here, because this is very similar to one of the CryptoHack tasks.
- But you can use the LLL algorithm to solve this.
- I would recommend using Sagemath or the python library fpylll or C library fplll, depending on your preference.
- You need to construct the lattice basis described earlier, and then reduce it using LLL and lastly do simple arithmetic to compute the secret key.

Recovering the key

For small lattices where the randomness used is very short or have big chunks in common with each other, this is a very fast attack.

```
Curve: Elliptic Curve defined by y^2 = x^3 + 7 over Finite Field of size 115792089237316195423570985008687907853269984665640564039457584007908834671663
p: 115792089237316195423570985008687907853269984665640564039457584007908834671663 0xfffffffffffffffffffffffffffffffffffffffffffffffffffffffefffffc2f
n: 115792089237316195423570985008687907852837564279074904382605163141518161494337 0xfffffffffffffffffffffffffffffffebaaedce6af48a03bbfd25e8cd0364141
d: 93014717667518195997201708971372419465881181548562368794432875239154275681886 0xcda476ecc4a3cf5012534f99e6b85f1852442c71fd1741ec308db39e591d865e
G: (0x79be667ef9dcbbac55a06295ce870b07029bfcdb2dce28d959f2815b16f81798, 0x483ada7726a3c4655da4fbfc0e1108a8fd17b448a68554199c47d08ffb10d4b8)
Q: (0x130e99394b6358f01bf55cfb7daf10faaa1de4552b8b719db89f4c1aa56de88, 0x7ee04c1622da77d9868574eee8b812c2d7be6a519eea0a6a28f3719201277618)
[338312366969595278585886726037755088320, 118131619054077270500463840160131183903, 137658558047925316977704416948783073297]
------ Attack ------
--------------------


Found d: 93014717667518195997201708971372419465881181548562368794432875239154275681886
sage lattice attack demo.sage  0.85s user 0.15s system 102% cpu 0.980 total
```

The curious case of the half-half Bitcoin ECDSA nonces

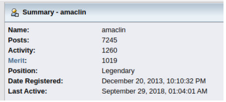
Dylan Rowe[1], Joachim Breitner[2][0000-0003-3753-6821], and Nadia Heninger[1][0000-0002-7904-7295]

[1] University of California, San Diego
drowe@ucsd.edu,nadiah@cs.ucsd.edu
[2] Unaffiliated
mail@joachim-breitner.de

Figure: https://eprint.iacr.org/2023/841

“Amaclin”

- Screenname of a user on multiple forums
- They have tricket a lot of people into using bad nonces, and have most likely stolen a lot of money.
- One of the things they tricket people to do was using part of their secret key as a nonce, combined with actual randomness, leading them to be able to run an attack very similar to the ones described in this lecture.



Contents

Fast Practical Lattice Reduction through Iterated Compression

Keegan Ryan and Nadia Heninger

University of California, San Diego, USA
kryan@ucsd.edu,nadiah@cs.ucsd.edu

Figure: https://eprint.iacr.org/2023/237

On Bounded Distance Decoding with Predicate: Breaking the "Lattice Barrier" for the Hidden Number Problem

Martin R. Albrecht[1] and Nadia Heninger[2⋆]

[1] Information Security Group, Royal Holloway, University of London
[2] University of California, San Diego

Figure: `https://eprint.iacr.org/2020/1540`

Biased Nonce Sense: Lattice Attacks against Weak ECDSA Signatures in Cryptocurrencies

Joachim Breitner[1][0000−0003−3753−6821] and Nadia Heninger[2]

[1] DFINITY Foundation, Zug, joachim@dfinity.org
[2] University of California, San Diego, nadiah@cs.ucsd.edu

Figure: https://eprint.iacr.org/2019/023

Questions?