Eötvös Loránd Tudományegyetem

Természettudományi Kar

Rusznák Demeter

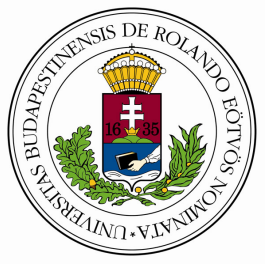
Az elliptikus görbékkel való titkosítás néhány matematikai kérdése

BSc Szakdolgozat

Témavezető:

Szabó István

Valószínűségelméleti és Statisztika Tanszék



Budapest, 2016

Tartalomjegyzék

Bevezetés

Korunkban egyre fontosabbá válik, hogy biztonságosan kommunikáljunk embertársainkkal. Ehhez az szükséges, hogy az üzenet váltásban résztvevőkön kívül harmadik fél ne tudja elolvasni a beszélgetést. Ezt a különféle titkosítási és adatrejtési eljárások biztosítják.

Szakdolgozatom témájául is egy ilyen titkosítási rendszer, nevezetesen az Elliptikus Görbéken Alapuló rendszer(ECC), bemutatását tűztem ki célomul. Ez egy úgynevezett nyilvános-kulcsú titkosítási rendszer amelyben minden felhasználó két kulccsal rendelkezik, egy titkossal, amit az üzenetek dekódolására használ, és egy nyilvánossal aminek segítségével mások küldhetnek neki titkosított üzenetet.

A dolgozatban elöszőr a matematikai háttéret fogom bemutatni, ezen belül is a következő témákat dolgozom fel: mi az az elliptikus görbék, mi teszi alkalmassá őket a titkosításra, majd néhány hozzájuk igazított algoritmus leírása következik (kulcscsere, üzenettitkosítás, aláírás). Majd ezen rendszerek támadhatóságáról lesz szó. Terjedelmi korlátok miatt a teljesség igénye nélkül.

1. fejezet

Matematikai alapok

1.1. Az elliptikus görbékről általában

1.1.1. Néhány fontos definíció

1.1.1. Definíció. *(Csoport)[1] Egy G nem üres halmaz csoport ha értelmezett rajta egy művelet, a következő tulajdonságokkal:*

- *A $(+)$ művelet asszociatív*
- *A G a $+$ műveletre zárt (Ha a és b $\in G$ akkor $a+b$ is $\in G$)*
- *Van neutrális elem*
- *Van minden elemnek inverze*

1.1.2. Definíció. *(Részcsoport)[1] Legyen G csoport, és $g \in G$.Ekkor a g elem hatványaiból álló részcsoportot a g által generált részcsoportnak nevezzük és $\langle g \rangle$-vel jelöljük.*

1.1.3. Definíció. *(Ciklikus csoport)[1] Egy G csoportot ciklikusnak nevezünk ha egy elemmel generálható.*

1.1.4. Definíció. *(Abel-csoport)[1] Egy G csoportot Abel-csoportnak nevezünk ha kommutatív.*

1.1.5. Definíció. *(Karakterisztika)[10] Legyen T tetszőleges test és $0, 1 \in T$ a zérus-, illetve egységelem. Azt a legkisebb k pozitív egész számot, amelyre $k * 1 = 0$ a test karakterisztikájának nevezzük. Ha nem létezik ilyen pozitív egész, akkor a test karakterisztikája 0. Ezt char(T)-vel szokták jelölni.*

1.1.2. Az elliptikus görbék definíciója és aritmetikája

1.1.6. Definíció. *(Elliptikus görbe)[2] Egy F test feletti E elliptikus görbét az*

$$y^2 + a_1xy + a_3y = x^3 + a_2x^2 + a_4x + a_6 \quad a_i \in F \tag{1.1}$$

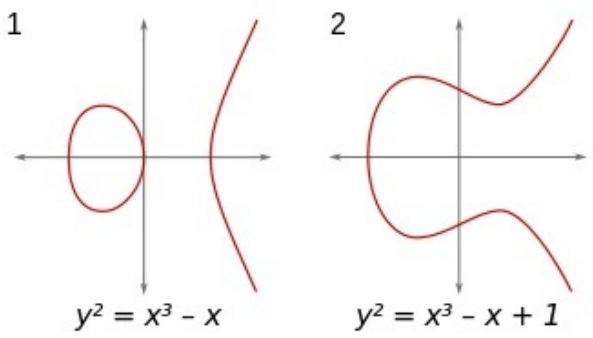
egyenlet definiálja

Az $E(F)$ görbét azon $(x, y) \in F^2$ pontok alkotják, amelyek kielégítik a 1.1 egyenletet, valamint a görbéhez tartozik egy absztrakt pont, a "végtelenben" fekvő $\mathcal{O}$ pont.Továbbá a 1.1 által meghatározott görbének "simának" kell lennie.

1.1.7. Definíció. *(Simaság)[2] Az 1.1 szerint definiált görbe sima ha nincs olyan pontja amely egyszerre kielégíti az*

$$\begin{aligned} a_1y &= 3x^2 + 2a_2x + a_4 \\ 2y + a_1x + a_3 &= 0 \end{aligned}$$

parciális deriváltakat.



1.1. ábra. Elliptikus görbék[12]

A fenti egyenlet $x \to x - (1/3)a_2$ változó cserével, $y^2 = x^3 + ax + b$ alakra hozható, ezt a görbe Weierstrass alakjának is nevezik, ha F karakterisztikája nem egyenlő 2 vagy 3-al. A simasági feltétel akkor teljesül ha az $x^3 + ax + b$ polinomnak csak egyszeres gyökei vannak vagyis a $D = -(4a^3 + 27b^2)$ determináns nem zérus.

1.1.8. Megjegyzés. Jelen dolgozatban, hely hiány miatt nem foglalkozunk, azon esettel amikor F karakterisztikája 2 ill 3, ezért bár kriptográfiailag fontos, kihagyásra kerül későbbiekben tárgyalt módszerek F_{2^m} feletti megvalósításainak tárgyalása.

Legyen $E(F)$ az F test felett értelmezett E elliptikus görbe.

1.1.9. állítás. *Az $E(F)$ pontjai, egy alkalmasan meghatározott művelettel Abel–csoportot alkotnak.*

- A görbe pontjai a csoport elemei
- A neutrális elem a "végtelen távoli" $\mathcal{O}$ pont
- Tetszőleges $P \in E(F)$ inverze, a P pont x-tengelyre vetített tükörképe.

A műveletet, melyet jelöljünk $(+)$-al, a következő képen írhatjuk le:

Vegyünk az $E(F)$ görbén három olyan pontot (P, Q, R) amelyek egy egyenesbe esnek. Ekkor

$$P + Q + R = \mathcal{O}$$

1.1.10. Megjegyzés. A művelethez három darab egy egyenesbe eső pont szükséges, és a három pont sorrendtől függetlenül egy egyenesre esik tehát: $P+(Q+R) = Q+(P+R) = R+(P+Q) = \ldots = \mathcal{O}$, ezt figyelembe véve beláttuk hogy az így definiált művelet asszociatív és kommutatív, vagyis egy Abel–csoportban vagyunk.

Továbiakban a műveleti szabályok, valamint végrehajtásainak szabályait részletezzük.

1.1.3. Geometriai megközelítés

Tudjuk, hogy az $E(F)$ elliptikus görbe pontjai kiegészítve a $\mathcal{O}$-val, Abel–csoportot alkotnak. Mint azt fentebb már írtam ha $P, Q, R \in E(F)$ és ezen pontok egy egyenesre esnek, akkor $P+Q+R = \mathcal{O}$. Mivel Abel–csoportban vagyunk ezért ezt az egyenletet átírhatjuk $P + Q = -R$ alakra.

Két pont összeadása:

Legyen P és $Q \in E(F)$ és $P \neq Q$, $P \neq -Q$ és $Q \neq -P$. Húzzuk meg a két pontot,P-t és Q-t összekötő egyenest. Ez a görbét egy harmadik pontban metszi. Ez a pont $-R$, tükrözzük $-R$-t az x-tengelyre, így megkapjuk R-t.

- Ha $P = \mathcal{O}$, vagy $Q = \mathcal{O}$? Ebben az esetben nem húzhatjuk meg a görbét metsző egyenest, mivel a $\mathcal{O}$ a "végtelenben" van, viszont mivel neutrális elemnek defíniáltuk, ezért a definícióból következik, hogy $P+\mathcal{O} = P$, illetve $Q+\mathcal{O} = Q \quad \forall\ P, Q \in E(F)$.

- Ha $P = -Q$, ebben az esetben a húzott egyenes merőleges, nem metszi harmadik pontban a görbét. Viszont az inverz definíciójából következik: $P + (-P) = \mathcal{O}$, ha $Q = -P$ akkor ugyan ez a helyzet.

- Ha $P = Q$, vagyis mennyi $2 * P$? Akkor érintőt húzunk P-ben majd ahol az érintő metszi a görbét az lesz $-2P$ ezt tükrözve megkapjuk $2P$-t.

- Ha $P \neq Q$, de nincs harmadik pont? Ebben az esetben a Q-n és P-n átmenő egyenes a görbe érintője lesz. Ha P az érintő, akkor az előző esetből: $P + P = Q$ ezt most átírhatjuk $P + Q = -P$ és ha Q az érintési pont, akkor $P + Q = -Q$

1.1.4. Algebrai alak

Eddig a görbén használt (+) műveletet geometriai módszerekkel mutattam be, most jöjjön az algebrai alakja, amely következtethető a geometriai alakból, ugyanis egy görbe és egyenes metszéspontjai számolhatóak.

- Előjelváltás *(Inverz képzés)*: Legyen $P \in E(F)$ és $P(x_P; y_P)$ ekkor $-P = (x_P; -y_P)$

- Két különböző pont $(P, Q, P \neq Q)$ összege: $(x_P; y_P) + (x_Q; -y_Q) = (x_R; y_R)$ ahol

$$\begin{aligned} x_R &= \lambda^2 - x_P - x_Q \\ y_R &= \lambda(x_P - x_R) - y_P, \text{ ahol} \\ \lambda &= \frac{y_Q - y_P}{x_Q - x_P} \end{aligned}$$

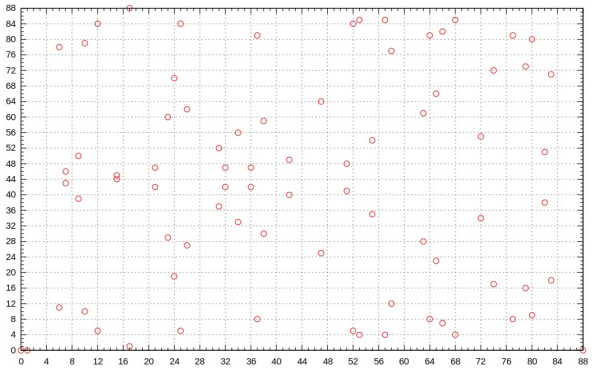
- Pont duplázás: $P + P = R$

  $P(x_P; y_P) \in E(F)$ és $y_P \neq \mathcal{O}$, ekkor:$(x_P; y_P) + (x_P; y_P) = (x_R; y_R)$

$$\begin{aligned} x_R &= \lambda^2 - 2x_P \\ y_R &= \lambda(x_P - x_R) - y_P, \end{aligned}$$

  ahol a $\lambda = \frac{3x_P^2 + a}{2y_P}$ és a az E görbe megfelelő együtthatója.

Eddig láthattuk, hogy mi az elliptikus görbe, valamint, hogyan végezhetünk műveleteket a pontjaival. De ahhoz, hogy a kriptográfiában is felhasználhassuk őket, a valós számok testéről át kell térnünk F_p-re (ahol p prím), vagyis a véges prím testekre.



1.2. ábra. Az $y^2 = x^3 - x$ F_{89} felett[12]

1.2. Elliptikus görbék véges testek felett: F_p

1.2.1. Definíció. *(Véges test)[10] Legyen F_p a p elemet tartalmazó véges test, ahol p-prím. F_p elemei a $\{0, 1, 2, \ldots, p-1\}$ egészek.*

Ekkor az E elliptikus görbét F_p felett a következő egyenlet határozza meg, amennyiben feltesszük, hogy char(F)$\neq 2; 3$:

$$y^2 = x^3 + ax + b \mod p \quad \{x, y | y^2 = x^3 + ax + b \mod p\} \cup \{\infty\}$$

vagyis ha az egyenlet mindkét oldala ugyanazt a maradékot adja p -vel való osztás után akkor Q a görbe pontja. A $\mathcal{O}$ még mindíg a végtelen távoli pont, valamint a és $b \in F_p$. Továbbra is igaz, hogy a görbe nem lehet szinguláris, azaz $D = 4a^3 + 27b^2$ diszkrimináns nem 0. Csak most $\mod (p)$ vizsgáljuk. Valamint az, hogy minden pont koordinátája 0 és $p-1$ közé esik a következő előnyökkel jár:

- A számolás gyorsabb lesz, az eredmény pontos, mivel csak egész számokkal dolgozunk.
- A koordináták 0 és $p-1$ közé szorítása lecsökkenti a pontok számát.
- A moduláris aritmetika megnöveli a megoldások számát.

Az elöbb definiált számolási szabályok érvényben maradnak, csak $\mod (p)$ kell őket tekinteni.

A görbéket ábrázolva megfigyelhetjük, hogy a görbe pontjai viszonylag véletlenszerűen helyezkednek el, ugyanakkor még mindíg tartják a szimmetriát, csak nem feltétlenül az x-tengelyre.

A pontok száma:

Legyen F_p egy véges test és legyen $E(F_p)$ egy elliptikus görbe ($a, b \in F_p$).Az $E(F_p)$ görbe pontjainak számát a görbe rendjének nevezzük és $\#E(F_p)$-vel jelöljük. Tudjuk, hogy egy ilyen görbe szimmetrikus, tehát minden x-hez legfeljebb két y érték tartozik, ezen kívül a végteln távoli pont. Tehát $E(F_p)$-nek maximum $2p + 1$ pontja lehet.
Hasse tétele ennél pontosabb becslést is ad:

1.2.2. Tétel. *(Hasse)[5] Legyen N az F_p felett értelmezett E görbe pontjainak száma. Ekkor:* $p + 1 - \sqrt{p} < N < p + 1 + \sqrt{p}$

Pontos megállapítására azomban Schoof[7] algoritmusát használhatjuk, ami polinomiális időben végzi el ennek számítását.

Skalárral való szorzás:

Ahogy $\mathbb{R}$ felett itt is definiálhatjuk a skalárral való szorzást. Ha n skalár és $P \in E(F_p)$, akkor

$$nP = \overbrace{P + P + P + \ldots + P}^{n\ db}$$

Ez nem új művelet mivel pontok összeadások és pont duplázások egymásután fűzésével megkapható és a megfelelő módszer használatával $O(\log(n))$ lépésben végrehajtható.

Az F_p-ben egy P és skalárszorosai ciklikus csoportot alkotnak. P neve bázis vagy generátor pont. Azt a legkisebb n számot melyre $nP \equiv 0 \mod (p)$ a $\langle p \rangle$ csoport rendjének nevezzük, ez megegyezik az alcsoport elem számával. Ezen csoport rendjének meghatározásában segít Lagrange tétele, amely azt mondja ki, hogy a részcsoport rendje, osztja az "eredeti" csoport rendjét. Más szóval, ha adott egy csoportunk amelynek a rendje N és azon belül adott egy részcsoport melynek rendje n akkor $n|N$.

Tehát:

1. Kiszámítjuk N-t
2. Meghatározzuk N osztóit.
3. Minden n-re amely osztja N-t kiszámoljuk nP-t
4. A legkisebb n amelyre $nP = 0$ lesz az részcsoport rendje.

Bázis pont keresése:
Ahhoz hogy, az ECC algoritmusai megfelelően működjenek, nagy elemszámú csoportok kellenek. Ezek generálásához kell egy alkalmas generátor elem, amihez keresünk egy görbét, kiszámítjuk a rendjét, majd az osztóit, választunk egy megfelelően nagy osztót, majd keresünk egy olyan pontot amelynek az a rendje.

Ehhez szükségünk lesz még egy fogalom bevezetésére, Lagrange tétel szerint a $h = N/n$ mindíg egész szám lesz, mivel n osztója N-nek.A részcsoport kofaktorát h-nak nevezzük. Tudjuk,hogy $\forall P \in E(F)$ re igaz, hogy $NP = 0$, mert N mindegyik n-nek többszöröse. A kofaktor definícióját felhasználva, írhatjuk hogy $n(hP) = 0$, feltehetjük, hogy n prím, az egyenlet ezen alakjából következik, hogy a $G = hP$ pont egy n-ed rendű alcsoportot generál, kivéve ha $G = hP = 0$ mert ebben az esetben P rendje 1.

Ennek fényében felírhatjuk a következő bázis pont kereső algoritmust:

1. Kiszámítjuk a görbe rendjét: N
2. Kiválasztjuk a célnak megfelelő n-et, ahhoz, hogy az algoritmus működjön, az kell, hogy n prim legyen.
3. Kiszámítjuk a kofaktort $h = N/n$
4. Válasszunk ki egy tetszőleges $P \in E(F)$
5. Számítsuk ki $G = hP$-t
6. Ha $G = 0$ akkor visszatérünk a 4. lépéshez, egyébként megtaláltuk a bázis pontot.

1.2.3. Megjegyzés. Vegyük észre, hogy az algoritmus csak akkor működik ha n prím, ha ez nem így lenne, akkor G rendje n bármelyik osztója is lehetne.

1.2.1. Elliptikus görbe feletti Diszkrét Logaritmus Probléma (ECDLP)

Adott $P, Q \in E(F_p)$, keressük azt a k-t melyre $kP = Q$.
Ezt a problémát elliptikus görbék felett értelmezett diszkrét logaritmus problémának nevezik, az ECDLP rövidítés az angol Elliptic Curve Discrate Logarythm Problem-ból származik. Diszkrét, mert véges testek felett keressük k-t, elliptikus, mert egy görbén vagyunk, és logaritmus, mert ebben az esetben analóg a "hagyományos" logaritmussal. Amely a mai ismereteink szerint ebben a formájában alkalmas egy a nyilvános kulcsú rendszerek alapjául szolgáló "egyirányú függvény" definiálásához. Mint említettük, pontot skalárral megszorozni, gyorsan tudunk ($O(\log(n))$), viszont a "törésére" használt leggyorsabb algoritmus is exponenciális idejű. Így elég "nehéz"-nek vélt, ahhoz, hogy titkosításra alkalmasnak minősítsük. Arra azonban, hogy valóban "nehéz"-e, nincs matematikai bizonyíték. Az általunk használt nyilvános kulcsú rendszerek alapjait képző hasonló problémák közül ez a legnehezebbnek vélt. Ez megmutatkozik a felhasznált k-kulcsok méreteiben is.

"Nehéz" probléma: a megoldására nem ismert általános esetben működő, legfeljebb polinom idejű algoritmus.

1.2.4. Definíció. *(Elliptikus görbék felett értelemzett diszkrét logaritmus probléma)[5] Adott egy elliptikus görbe E, F_p véges test felett, feltesszük hogy p prím, és egy $P \in E(F_p)$ amelynek rendje n, és egy $Q \in \langle P \rangle$. Keressük azt az $l \in [0, n-1]$ számot amelyre $lP = Q$. Ekkor l-t Q, p alapú diszkrét logaritmusának nevezzük és $log_p Q = l$-ként jelöljük.*

2. fejezet

Kriptográfiai felhasználás

[1]

Most már, megvannak az elliptikus görbéink, tudunk rajtuk számolni és adott egy "nehéz" probléma is. A most kövekezőkben néhány olyan kriptográfiai algoritmust tekintünk át amelyek felhasználják az előző fejezetben bemutatott matematikai módszereket.

2.1. Görbe és kulcs generálás

Az itt bemutatott algoritmusok két legfontosabb elemével kezdjük, a görbék és a kulcsok előállításának algoritmusaival. Ezek nem konkrét megvalósítások, csak azon általános szempontok gyűjteményei amiket szem elött kell tartanunk ahhoz, hogy a titkosított információk ezen két összetevőn keresztül nehezen támadhatók legyenek.

2.1.1. A görbe paraméterei és érvényességük

Az E elliptikus görbét, és a bázispontot egyértelműen kijelölik a következő paraméterek:

$$T = (p, a, b, G, n, h)$$

Ahol p az F_p test p-je, azaz itt p meghatározza azt a véges testet amely felett a görbét értelmezzük. a és b az $y^2 = x^3 + ax + b$ egyenlet megfelelő paraméterei, pontosan meghatározzák a használt görbét, G a görbén használt bázispont, n a G rendje és h a kofaktor.

[1] Ebben a fejezetben szereplő algoritmusok megtalálhatóak a [6]-ban

Ez a paraméter 6-os egyértelműen meghatároz minden szükséges információt ahhoz, hogy a felhasználásukkal létrehozott rendszer biztonságosnak nevezhető legyen.

Ha a görbe paramétereit véletlenszerűen állítjuk elő, akkor az előző 6 adathoz még egy hetediket is hozzá kell vennünk. Ez az S érték (Seed).

Az elliptikus görbe paramétereinek előállításának lépései:

Bemenet: A paraméterektől elvárt biztonsági szint a bitek számában kifejezve, ez egy $t \in \{80, 112, 128, 192, 256\}$[2] egész, és egy tetszés szeinti S érték.

Kimenet: Egy olyan F_p feletti paraméter "csomag" $T = (p, a, b, G, n, h)$ amelyek egy olyan rendszert adnak meg amiben a diszkrét logaritmus kiszámítása nagyjából 2^t-en műveletet igényel.

Az előállítás lépései:

1. Válasszunk egy p prímet úgy, hogy $\lceil \log_2(p) \rceil = 2t$ ha $80 < t < 256$
2. Válasszuk meg $a, b \in F_p$, hogy megkapjuk $E(F_p)$ -t amelyet az

$$y^2 \equiv x^3 + ax + b \mod (p)$$

3. egy G bázis pontot melynek rendje n
4. számítsuk ki $h = \#E(F_p)/n$,

úgy hogy a következő szempontoknak megfeleljenek:

- $4a^3 + 27b^2 \not\equiv 0 \mod (p)$
- $\#E \neq 0$
- $P^B \equiv 0 \mod (p) \forall\ 1 < B < 100$
- $h < 2^{t/8}$
- $n - 1$ és $n + 1$ -nek legyen egy $log_n(r) > \frac{19}{20}$ nagyságrendű prím osztója.

[2]azért ezek a bithosszak mert a szakirodalom amiből az algoritmus származik ezeket használja[6]

Ha adott S akkor vagy az együtthatók (a, b), vagy G bázispont, vagy akár mindegyik kiválasztásánál fel kell használnunk.

Gyakran szükséges, vagy legalábbis ajánlott a felhasználónak megbizonyosodni arról, hogy adott $T = (p, a, b, G, n, h)$ paraméterek érvényesek-e. Az általuk meghatározott test, görbe és bázis pont valóban alkalmas-e egy megbízható titkosítási rendszer felépítésére. Ezt azért célszerű megtenni, mert így kiszűrhető a megbízhatatlan paraméterek rosszindulatú beillesztése, észlelhetővé válnak a figyelmetlen kódolásból, vagy továbbításból származó hibák. Ennek eldöntésére az U felhasználó a következő módszerek közül választhat. Legalább az egyiket végre kell hajtani, de a nagyobb bizonyosság érdekében javasolt többnek is a végrehajtása.

- U maga hajtja végre a $T = (p, a, b, G, n, h)$ paramétereken az ellenőrzésükre szolgáló lépéseket.
- U által megbízhatónak ítélt rendszer állította elő T -t, betartva a fentebb ismertetett szempontokat.
- Egy az U felhasználó által megbízható harmadik fél tanúsítja, hogy az ellenőrző algoritmus végrehajtásra került.
- Egy az U felhasználó által megbízható harmadik fél tanúsítja, hogy az általa használt, a T előállítására használt rendszer, a feltételeknek megfelelő paramétereket állít elő.

A T érvényességének ellenőrzésére szolgáló lépések:

1. Ellenőrizzük, hogy (a, b) valamint x_G és $y_G \in [0, p-1]$
2. p -re igaz-e, hogy $\log_2 p = 2t$, amennyiben $80 < t < 256$ valamint, hogy p valóban prím e
3. $4a^3 + 27b^2 \neq 0 \mod p$ igaz-e?
4. G valóban rajta van-e az (a, b) által meghatározott görbén.
   Vagyis $y_G^2 = x_G^3 + ax_G + b \mod p$
5. n prím-e?
6. $h < 2^{t/8}$ és $h = \lfloor (\sqrt{p} + 1)^2 / n \rfloor$

7. $nG = 0$

8. $p^B \neq 1(\mod n) \forall 1 \leq B \leq 100$ és $p \neq n$

Ha a fenti feltételek közül legalább az egyik nem teljesül akkor a görbe "érvénytelen" (invalid), egyébként "érvényes" (valid) amennyiben "érvényes" a felhasználó megbízik a T paraméter csomagban. A 8. lépés azért szükséges, hogy kizárjuk a görbéknek egy olyan csoportját amelyek gyengék a Menezes-Okamoto-Vanstone, a Frey-Rück vagy a Samaev-Smart-Sato-Araki támadásokkal szemben.

2.1.2. Kulcs generálás és érvényesítés

Az ECC algoritmusaiban felhasznált építő elemek közül már csak a kulcsok hiányoznak. Mint minden nyílt kulcsú rendszernél, itt is minden felhasználóhoz egy kulcspár tartozik. Az ECC-nél használt párokat egy Q és egy d pár alkotja amelyből Q a nyilvános, és d a titkos. A d számot egy arra alkalmas véletlenszám-generátorral állítjuk elő, ez a pár titkos része, míg a nyilvános Q kiszámítása $d * G = Q$ módon történik ahol G az adott görbe bázis pontja. Minden kulcspár csak a hozzátartozó $T = (p, a, b, G, n, h)$ paraméterekkel együtt érvényes.

Kulcsok generálásának lépései

Bemenet: a felhasználók által használt "környezetet" meghatározó érvényes $T = (p, a, b, G, n, h)$ paraméter lista

Kimenet: Egy a T hez tartozó (Q, d) elliptikus kulcs-pár.

Lépések:

1. Véletlen vagy pszeudo véletlen módszerekkel kiválasztjuk d egészet.

2. Elvégezzük a $dG = Q$ számítást

3. Visszatérünk a (d, Q) kulcs-párral.

Kulcs-párok érvényességének vizsgálata:
Ahogy a görbe paramétereknél, úgy a kulcsoknál is szükséges megbizonyosodni arról, hogy a felhasználni kívánt kulcsok megfelelnek-e a kulcsoktól elvárt biztonsági feltételeknek. Ennek megállapítására az U felhaszálónak ugyanazok a módszerek állnak rendelkezésünkre mint amikor a T-k érvényességéről akartunk meggyőződni.

- U maga hajtja végre a kulcson az ellenőrzésükre szolgáló lépéseket.
- U által megbízhatónak ítélt rendszer állította elő Q -t, betartva a fentebb ismertetett szempontokat.
- Egy az U felhasználó által megbízható harmadik fél tanúsítja hogy az ellenőrző algoritmus végrehajtásra került.
- Egy az U felhasználó által megbízható harmadik fél tanúsítja hogy az általa használt, a Q előállítására használt rendszer, a feltételeknek megfelelő kulcsokat állít elő.

A folyamat lépései:
Bemenet: a $Q = (x_Q, y_Q)$ pont és a hozzá tartozó $T = (p, a, b, G, n, h)$

Kimenet: "érvényes" (valid) vagy "érvénytelen" (invalid).

1. Ellenőrizzük, hogy $Q \neq \mathcal{O}$
2. Amennyiben T ténylegesen egy F_p feletti görbe paraméterei, akkor ellenőrizzük, hogy x_Q és $y_Q \in [0, p-1]$ ha igen akkor kielégítik-e az

$$y_Q^2 = x_Q^3 + ax_Q + b(\mod p)$$

   egyenletet.
3. Ellenőrizzük, hogy $nQ = \mathcal{O}$

Ha a fenti feltételek közül bármelyik nem teljesül akkor a kimenet "érvénytelen" (invalid).

Az 1. és 2. feltételek azt hivatottak eldönteni, hogy Q pontja-e E-nek és ha igen, akkor megegyezik-e $\mathcal{O}$-val. A 3. feltétel annak viszgálatára szolgál, hogy Q valóban G többszöröse-e? Erre általában nincs szükség, például, ha $h = 1$, akkor $nQ = \mathcal{O}$ ez következik a 2. feltételből mivel az a tulajdonság minden $Q \in E$-re igaz, mivel a pontok rendjei a görbe rendjének osztói, így a görbe rendje bármely pont rendjének egész szám szorosa.

Részleges érvényességű kulcsok:

Van olyan szituáció amikor az üzenet küldéséhez elég, ha a kulcsnak csak részleges "érvényességéről" (partial-validation) meggyőződnünk. Ezalatt azt értjük, hogy a Q pont eleme ugyan az $E(F_p)$ görbének, de nem biztos, hogy $Q = dG$ valamely d-re. Az MQV, valamint a kofaktor Diffie-Hellman kulcs csere protokolloknak meg van az a tulajdonsága, hogy akkor is megfelelően működnek, ha csak "részlegesen érvényes" kulcsokkal használjuk őket.

Ezen tulajdonság azért jó, mert ezáltal pl.: a kofaktor Diffie-Hellman kulcscsere gyorsabb lesz mint a hagyományos megfelelője. Mivel azt ellenőrizni hogy egy kulcs részlegesen "érvényes" gyorsabb, mint a teljes érvényességének a levizsgálása.

Részleges érvényesség vizsgálata:

Bemenet: a $Q = (x_Q, y_Q)$ pont és a hozzá tartozó $T = (p, a, b, G, n, h)$

Kimenet: "érvényes" (valid) vagy "érvénytelen" (invalid).

Lépések

1. Ellenőrizzük, hogy $Q \neq \mathcal{O}$
2. Amennyiben T ténylegesen egy F_p feletti görbe paraméterei, akkor ellenőrizzük, hogy x_Q és $y_Q \in [0, p-1]$, ha igen, akkor kielégítik e az

$$y_Q^2 = x_Q^3 + ax_Q + b \mod p$$

   egyenletet.

Ha bármelyik feltétel nem teljesül, a kimenet "érvénytelen" (invalid).

Ezen feltételekkel csak azt ellenőriztük, hogy a Q valóban eleme $E(F_p)$-nek, és Q nem egyezik meg $\mathcal{O}$-val.

2.1.3. Kulcscsere, titkosítás, aláírás

Most, hogy minden fontosabb ECC-hez szükséges elem rendelkezésünkre áll, áttekintjük néhány kriptográfiai algoritmus elliptikus görbéken értelmezett változatát.

ECDH - Elliptic Diffie-Hellman - Elliptikus Diffie-Hellman

Az ECDH, vagy Elliptic Curve Diffie-Hellman, az eredeti Diffie-Hellman elliptikus görbéken értelemzett változata. Arra szolgál, hogy két felhasználó, akik előzetesen nem tudtak kulcsot egyeztetni, képesek legynek egy ilyen közös kulcs létrehozására, és lehetőségük legyen az egymásközti titkos kommunikációra.

Kétféle megvalósítását nézzük meg:
Az egyik az alap ECDH, a másik a kofaktor ECDH, amely a $T = (p, a, b, G, n, h)$ paraméterlistából a h-t vagyis a kofaktort is beépíti a kulcsgenerálási eljárásba, azzal a céllal, hogy magasabb fokú védelmet nyújtson bizonyos támadásokkal szemben (pl: "small subgroup"). Valamint a kofaktor ECDH képes részben érvényes kulcsok felhasználására.

ECDH primitív:

Ha U és V szeretnének titkos üzeneteket küldeni egymásnak, de nincs közös titkos kulcsuk, akkor a következő lépések végrehajtásával készíthetnek egyet.

Adott: $U(d_U, Q_U)$ kulcspárja $V(d_V, Q_V)$ kulcspárja és mindkét felhasználó közös $T = (p, a, b, G, n, h)$ paraméterei.

Lépések

1. U kiszámítja Q_v-nak felhasználásával a $P = (x_p, y_p) = d_u Q_v$ pontot.
2. Ellenőrizzük, hogy $P \neq \mathcal{O}$, ha igen, akkor az eljárás érvénytelen.
3. Kimenet $Z = P(x_p, y_p)$ pont.

Ha ezen lépéseket mindkét felhasználó végrehajtja akkor ugyanahhoz a P ponthoz jutnak, azért mert a nyilvános kulcs $d_u G$ vagy $d_v G$ alakú és ezt megszorozva a megfelelő paraméterrel a $d_u d_v G$ pontot kapjuk. Az eljárás elött, vagy után a felek megegyeznek abban, hogy pl a P pont x koordinátáját fogják kulcsként használni. Ezzel megvan a közös kulcs.

Kofaktor ECDH primitív:

Adott: $U(d_U, Q_U)$ kulcspárja $V(d_V, Q_V)$ kulcspárja, lehetnek "valid", vagy "részben valid" kulcsok, és mindkét felhasználó közös $T = (p, a, b, G, n, h)$ paraméterei.

Lépések

1. U kiszámítja Q_v felhasználásával a $P = (x_p, y_p) = hd_uQ_v$ pontot.
2. Ellenőrizzük, hogy $P \neq \mathcal{O}$, ha igen akkor az eljárás érvénytelen.
3. Kimenet $Z = P(x_p, y_p)$ pont, ami nem más mint hd_ud_vG.

2.1.1. Megjegyzés. Annak eldöntésére, hogy a felhasznált Q_v, Q_u kulcsok valóban az adott felhasználókhoz tartoznak-e, egy mindkét résztvevő által megbízhatónak ítélt, haramdik felet szoktak felkérni.

EC-ELGamal - Elliptic Curve ElGamal - Elliptikus ElGamal

Az EC-ELGamal, csakúgy mint a ECDH, az eredeti algoritmus görbék felett történő alkalmazása. Segítségével a felhasználók képesek titkosított üzeneteket küldeni egymásnak.

Az üzenet beágyazása a görbébe[4],[11]
Ehhez az eljáráshoz szükséges, hogy az adott üzenetet le tudjuk képezni a görbe egy, esetleg több pontjába, azért, hogy alkalmazható legyen rá az algoritmus. Az ilyen leképzéseknél feltétel:

- A leképzés legyen kölcsönösen egyértelmű
- A leképzett blokk legyen kisebb mint a test modulusa.

Például: az üzenet karaktereit az ASCII kódjukkal helyetesítjük, így minden egyes betű ASCII kodja fogja jelképezni, legyen ez a keresett pontok x koordinátáit, és ezekhez keressük a megfelelő y koordinátákat úgy, hogy az adott számokat beírjuk a görbe egyenletébe, az y-ból való gyökvonás során, közös megegyezés szerint használhatjuk csak a pozitív gyököket, csak a negatívakat esetleg, bizonyos szabály szerint mindkettőt. Ha nem élétezne az adott görbén megfelelő y akkor vagy keresünk egy olyan görbét amelyen minden karakterhez létezik hozzátartozó y, vagy az ASCII kóddá alakítás után nem azonnal a kapott kódokat használjuk fel, hanem közbe iktatunk egy megfelelő függvényt amely lehetővé teszi a koordináta "csúsztatását". Ezen köztes függvényeket úgy htatározzák meg, hogy a csusztatásokkal együtt minden karakterhez legyen megfelelő y ugyan akkor ne kelljen "túl sokáig" keresni a visszafejtés során. Így az üzenet küldés lépései a következőképen alakulnak:

1. Az üzenet betüinek ASCII kódjaival történő helyettesítése.
2. Az így kapott értékekhez megkeresni a korrekt y -okat, (beágyazás a görbébe)
3. A kiválasztott pontok kódolása és elküldése.
4. Az üzenet megérkezése után az ASCII kódok kinyerése.
5. Az ASCII kódok visszaalakítása szöveggé.

vagy Tegyük fel, hogy üzenetünk a következő karaktereket tartalmazhatja: $0, 1, 2, \ldots, 9$ és $A, B, C, \ldots, Z$ (az angol abc betüi) és ezeket a $0, 1, 2, \ldots, 35$ számokkal reprezentáljuk. Ekkor:

1. A fenti feltételekkel, az üzenet számok sorozatává alakítható aminek minden eleme 0 és 35 közé esik.
2. Ezután választunk egy k segéd paramétert, például $k = 30$ (ez közös megegyezés alapján történik)
3. Minden mk-hoz megkeressük $x = mk + 1$ és ezt tekintjük x-nek és ehhez keresünk y-t
4. Ha nem találunk akkor vesszük az $x = mk + 2$-t és azzal keresünk y-t. ha ilyen sincs akkor $x = mk + 3$, stb.
5. A gyakorlatban minden x-hez találunk y mielött elérnénk $x = mk + k - 1$. Ha találtunk y-t akor vesszük az (x, y) pontot.

Ezzel a módszerrel az üzenet pontok sorozatává alakítható.

A szeretne üzenetet küldeni B-nek.B titkos kulcsa b,A titkos kulcsa k. Ekkor a következő képpen tudnak egymásnak üzeneteket küldeni: A valamilyen módszer segítségével leképzi az üzenetet a görbe egy, vagy több pontjába (M), majd generál egy véletlen számot (k), kiszámolja kG-t, és elküldi B-nek a $(kG, M + k(bG) = Q_B)$ pontpárt. Ahhoz, hogy B visszanyerje az üzenetet a következő lépéseket kell tennie.

A kapott pontpár első tagját (kG) megszorozza a saját titkos kulcsával így megkapja a $(b(kG))$ pontot, ezt kivonja a pontpár második feléből, így megkapja M-et.

A támadónak ebben az esetben (bG)-ből kéne kiszámolnia b-t, vagy kG-ből k-t, de jelen tudásunk szerint, ezt nem lehet megtenni, mielött az információ elévülne.

ECDSA - Elliptic Curve Digital Signature Algorithm - Elliptikus Digitális Aláírás Algoritmus

Mint minden aláírásnak az elektronikus aláírásnak is a hitelesítés a szerepe, hogy az adott üzenet ténylegesen az adott felhaszálótól származik-e? A ECDSA ezt a feladatot valósítja meg, az elliptikus görbékre épülő titkosítás ezközeivel.

Az algoritmus:
Bemenet: $T = (p, a, b, G, n, h)$ görbe paraméterek és egy (d_A, Q_A) az A felhasználóhoz tartozó kulcspár.

Kimenet: $S = (r, s)$ az üzenethez tartozó aláírás.

Lépések:

1. Választunk egy $k \in 1 \leq k \leq n-1$
2. Kiszámoljuk $kG = (x_r, y_r)$, $r = x_r \mod n$. Ha $r = 0$ akkor másik k-t választunk. Ezzel meg van az aláírás egyik fele r.
3. Kiszámoljuk k multiplikatív inverzét n-re nézve. $(k^{-1} \mod n)$
4. Kiszámoljuk az üzenet pecsétjét egy $H(m)$ hash függvényt használva. $e = H(m)$
5. Az aláírás másik fele $s = k^{-1}(e + d_A r) \mod n$.
6. Az A által küldött M üzenet aláírása: (r, s)

2.1.2. Megjegyzés. Ha $s = 0$ akkor kezdhetjük előlről, valamint ha $r = 0$ lett volna, akkor az aláírás nem tartalmazta volna a titkos kulcsot.

Az aláírás ellenőrzése:
Annak eldöntésére, hogy az üzenet sértetlen, eredeti, módosítatlan formájában jutott el a B-hez, a következő ellenőrző lépéseket hajtja végre a címzett. Feltételezzük, hogy B-nek megvan A hiteles nyilvános kulcsa $d_A G$ és a közös rendszer paraméterek $T = (p, a, b, G, n, h)$.

Lépések:

1. Ellenőrizzük, hogy az (r, s) pár tagjai valóban elemei a $[0, n-1]$ intervallumnak.

2. Kiszámoljuk az üzenet pecsétjét $e = H(M)$.

3. Kiszámoljuk s multiplikatív inverzét n-re nézve. ($w = s^{-1} \mod n$), ezért nem lehetett $s = 0$.

4. Kiszámoljuk $u_1 = ew \mod n$ és $u_2 = rw \mod n$ számokat.

5. Kiszámoljuk a $P(x_P, y_P) = u_1G + u_2G$ pontot. Ha $P = 0$, akkor biztosan hibás az aláírás, egyébként $v = x_P$

6. Az aláírás akkor fogadható el ha $v = r$.

Miért működik?

Az aláírás ellenőrzéséhez azt kell belátnunk, hogy az 5. pontban kiszámított P megegyezik A nyilvános kulcsával, tehát $P = Q_A$.

Az aláírás egyik fele:

$$s = k^{-1}(e + dr) \mod n,$$

ha az egyenlet mindkét oldalát megszorozzuk $s^{-1}k$-val akkor,

$$k = s^{-1}(e + dr) = s^{-1}e + s^{-1}dr = we + wdr = u_1 + u_2d \mod n.$$

Ezután:

$$P(x_P, y_P) = u_1G + u_2Q = u_1G + u_2dG = (u_1 + u_2d)G = kG.$$

Ha B az A által kiválasztott pontot kapja vissza, akkor az üzenet hiteles, tényleg A-tól származik.

3. fejezet

Az ECC biztonsága:

Eddig láthattuk, hogy mi az ECC matematikai alapja, elliptikus görbék és matematikájuk. Az előző fejezetben megismerhettük néhány kriptográfiai algoritmus elliptikus görbék feletti átiratát: kulcscsere, titkosítás, aláírás. A dolgozat utolsó részében az ECC támadhatóságáról lesz szó. A rengeteg támadási mód közül, a matematikai támadásokra helyezem a hangsúlyt, tehát az implementálási hibáktól, "social engineering" stb. most csak említés szintjén lesz szó. Azok közül is a Pohlig-Helman féle támadásról valamint Pollard ρ algoritmusáról lesz szó bővebben, mint a jelenleg ismert leghatékonyabb, a rendszer alapját adó ECDLP-t megoldó algoritmusokról. Érintőlegesen szó lesz még a rendszer szabványosításáról valamint törésének figyeléséről.

3.1. ECDLP

Az Elliptikus Diszkrét Logaritmus Probléma, mint már korábban említettem, létfontosságú az ECC alapú rednszerek biztonságának kérdésében.

Ha egy ilyen problémát "kimerítő" támadással próbálnánk megoldani, tehát végig probálni P összes skalárszorosát $(P, 2P, 3P, \ldots, (n-1)P)$, amíg meg nem találjuk Q-t, akkor legrosszabb esetben n lépést kéne megtennünk. Átlagosan is $\frac{n}{2}$-t. Ha úgy választjuk meg $\langle P \rangle$-t, hogy a rendje legalább 2^{80} nagyságrendű legyen, máris olyan mértékü lesz a művelet igény, hogy nem érdemes ezzel a módszerrel próbálkozni. A ma ismert leggyorsabb, általános célú, tehát sem a görbének, sem a testnek nincs semmilyen "speciális" tulajdonsága, ECDLP megoldásra használt módszerek a Pohlig-Hellman féle módszer, valamint Pollard ρ algoritmusa, és ezek keverékei. Ezen módszerek futási ideje exponenciális $(\sqrt{p})$,

ahol p itt n legnagyobb prím osztója. Ahhoz hogy ezen módszerek ellen védve legyünk, elég ha olyan P bázispontot választunk, amely rendjének legnagyobb p prímosztója olyan nagy legyen, hogy ne tehessenek meg $(\sqrt{p})$ lépést elfogadható időn belül. Ez körülbelül 2^{160} nagyságrendet jelent. Mindebből az következik, hogy ha jól választjuk meg a görbe praramétereket $(T = (p, a, b, G, n, h))$, akkor az ECDLP-t a mai ismereteink szerint nem lehet időben feltörni.

3.2. Pohlig-Hellman féle támadás

A Pohlig-Hellman féle támadás azt használja ki, hogy diszkrét logaritmust megoldani F_p csoport felett, melyben a P bázispont által generált alcsoport rendje n, nem nehezebb mint abban a csoportban amelynek rendje p, ahol p az n legnagyobb prím osztója. Ez által jelentősen meggyorsítva a probléma megoldását az ilyen rendű csoportokban. Ezért, hogy a görbénk ellenálló legyen ezzel a támadással szemben, olyan generátor pontot kell választanunk melynek rendje osztható egy "nagy" prímmel.

Az eljárás leírása:
Legyen $\log_P Q = l$, és P rendje n. Feltesszük, hogy az n prímtényezős felbontása $n = p_1^{e_1} p_2^{e_2} \dots p_r^{e_r}$. Az eljárás lényege, hogy kiszámoljuk $l_i = l \mod p_i^{e_i}$-ket minden $1 < i < r$-re, majd ezután megoldjuk a keletkezett kongruencia rendszert:

$$\begin{aligned} l &= l_1 \mod p_1^{e_1} \\ l &= l_2 \mod p_2^{e_2} \\ &\vdots \\ l &= l_r \mod p_r^{e_r} \end{aligned}$$

minden $l \in [0, n-1]$-re. A kínai maradék tétel miatt a megoldás egyértelmű lesz.

Most megmutatjuk, hogy az l_i-k kiszámítása miért egyszerűsödik ez e_i-k kiszámítására azon részcsoportokban amelyeknek a rendje osztja n-t. Az egyszerűség kedvéért l_i felírásában e-t fogunk használni az e_i helyett, valamint p-t p_i helyett.
Fejezzük ki l_i-t p-k segítségével:

$$l_i = z_0 + z_1 p + z_3 p^2 + \ldots + z_{e-1} p^{e-1}$$

ahol $z_i \in [0, p-1]$ és kiszámításuk a következőképpen történik:

Először meghatározzuk $P_0 = (n/p)P$ és $Q_0 = (n/p)Q$-t. P_0 rendje p ezért:

$$Q_0 = \frac{n}{p}Q = l(\frac{n}{p}P) = lP_0 = z_0P_0.$$

Ebből tudjuk,hogy $z_0 = \log_{P_0} Q_0$-t ki tudjuk számolni egy $\langle P_0 \rangle$-beli ECDLP megoldása révén.

Következik $Q_1 = (n/p^2)(Q - z_0P)$ kiszámítása:

$$Q_1 = \frac{n}{p^2}(Q - z_0P) = \frac{n}{p^2}(l - z_0)P = (l - z_0)(\frac{n}{p^2}P) = (z_0 - z_1p - z_0)(\frac{n}{p^2}P) = z_1(\frac{n}{p}P) = z_1P_0$$

Ebből látszik, hogy $Z_1 = \log_{P_0} Q_1$ megkapható egy $\langle P_0 \rangle$-beli ECDLP megoldásaként.

Általában: ha $z_0, z_1, \ldots z_{t-1}$ számokat kiszámoltuk akkor z_t megkapható az $z_t = \log_{p_0Q_t}$ logaritmus megoldásából, ahol

$$Q_t = \frac{n}{p^{t-1}}(Q - z_0P - z_1pP - z_2p^2P - \ldots - z_{t-1}p^{t-1}P)$$

3.3. Pollard ρ algoritmusa:

Emlékeztető:
A diszkrét logaritmus probléma a következő: Adott P és Q esetén keressük azt az x-et amelyre $Q = xP$. Az algoritmus azon az elgondoláson alapszik, hogy ezt az egyenletet $aP + bQ = AP + BQ$ alakban is fel lehet írni, így a keresett értékek a, b, A és B egészek.

Ha ezeket megtaláltuk a következőt tehetjük:

$$\begin{aligned} aP + bQ &= AP + BQ \\ aP + bxP &= AP + BxP \\ (a + bx)P &= (A + Bx)P \\ (a - A)P &= (B - b)xP \end{aligned}$$

Ebből P-t ehagyva:

$$\begin{aligned} a - A &= (B - b)x \mod n \\ x &= (a - A)(B - b)^{-1} \mod n \end{aligned}$$

Az algorimus eddigi működése hasonlít Shank "kislépés-nagy lépés" módszerére, csak ott a logaritmust $Q = (am+b)P$ alakra írtuk át amit végül $Q - amP = bP$-vé alakítottunk és kerestük a megfelelő (a, b) párokat amelyekre az egyenlet teljesül. Az m itt $\lceil\sqrt{n}\rceil$-t jelöli.

Egy lényeges különbség a "kislépés-nagylépés" és a ρ algoritmus között, hogy az utóbbinál az (a, b) párok meghatározására egy pszeudo-véletlen függvényt használunk, és az így kapott sorozatból állítjuk elő az $aP+bQ$ pontokat. Ezt azért tehetjük meg, mert mind P mind Q ugyanannak a ciklikus csoportnak az elemei, tehát az $aP + bQ$ pont sorozat is ciklikus lesz.

Ebből következik, hogy elöbb-utóbb körbeérünk, találunk (a, b) és (A, B) párokat, amelyre $aP + bQ = AP + BQ$, de $(a, b) \neq (A, B)$.

Ahhoz, hogy megtaláljuk ezen párokat, szükségünk van egy algoritmusra, amely megmutatja, ha ciklusba futottunk. Erre jó Floyd algoritmusa. Állítsuk párba az (a, b)-ket a hozzájuk tartozó $aP+bQ$ párokkal. Majd haladjunk végig a az így kapott párokon. Lehet, hogy az (a, b) párok nem ciklikusak, viszont $aP + bQ$-k biztosan azok, mivel ugyanazon bázispontból generáltuk őket, egy ciklikus csoportban. Így biztosan találunk olyat, aminél az $aP + bQ$ megegyzik, viszont az (a, b) nem.

Ezen a sorozaton elindítva két iterátort, az egyiket egyesével léptetve, a másikat úgy, hogy minden lépés után, a következő pontot kihagyja. Így az algoritmus nagyjából $\sqrt{n}$ lépésben talál ilyen párt.

3.3.1. Párhuzamosított ρ

Tegyük fel, hogy rendelkezésünkre áll M darab gép, és ezeken futtatjuk az algoritmust. Ebben az esetben jelentős gyorsítás érhető el. A naív elgondolás az lenne, hogy minden gépen külön külön futtatjuk (különböző kezdőpontokkal.), ameddig valamelyik meg nem találja a keresett megoldást. Ha így csinálnánk a becsült lépésszám, amíg egy processzor végez $3\sqrt{n/M}$. Ebből következik, hogy ez csak $\sqrt{M}$-szeres gyorsítása lenne az eredeti eljárásnak. Van Oorschot és Wiener azonban, találtak egy olyan módszert amelynek segítségével M szeres gyorsítás is elérhető. Ezt úgy tehetjük meg, hogy "megengedjük" hogy a processzorok ciklusai "összeütközzenek". Minden processzor maga választja meg a kezdőpontját, de ugyanazt a véletlen generátort használják a pontok előállítására. Ha két ilyen sorozat ütközik akkor az ütközést követően már ugyanazon a sorozaton haladnak.

A következő stratégia megmutatja, hogyan lehet hatékonyan megtalálni az ütközéseket a processzorok által generált véletlen sorozatokban. Először, válasszunk olyan pontokat

amelyek valamely könnyen ellenőrizhető különleges tulajdonsággal rendelkeznek. Például az x koordinátájukat jelentő bitsorozat első l tagja 0. Legyen θ azon pontok halmaza $\langle P \rangle$-ben melyek rendelkeznek ezen tulajdonsággal. Ha egy processzor elér egy ilyen pontot akkor elküldi egy központi szervernek, amely eltárolja egy listában. Amennyiben a szerver másodszorra is megkapja ugyanazt a pontot, kiszámolja a keresett logaritmust és megszakítja a további számításokat. A várható lépésszám processzoronként mielőtt ütközést találunk ($\sqrt{\pi n/2}/M$).A speciális tulajdonságú pontok nagyjából $\frac{1}{\theta}$ lépésenként helyezkednek el. Tehát a teljes művelet igény ($\sqrt{\pi n/2}/M) + \frac{1}{\theta}$ Ebből láthatjuk, hogy ha több processzoron futtatjuk párhuzamosan a ρ algoritmust, ezzel a módszerrel lineáris nagyságrendű gyorsítást érhetünk el.

3.3.2. Átvitel

Ennek a támadásnak a lényege, hogy amennyiben nem megfelelően választottunk meg bizonyos paramétereket, akkor az ECDLP, átvihető egy lineáris algebrai csoport feletti DLP-be, amit már lényegesen könnyebb megoldani, titkosításra alkalmatlanná válik. Legyen l az alappont által generált pontok száma a görbén.

- Additív átvitelről beszélünk, ha az alappontunk rendje megegyezik p-vel (a test p-jével), ebben az esetben elérhető, hogy az adott ECDLP helyett egy olyan DLP-t oldjunk meg, amit már nem az elliptikus görbe felett értelmezünk (így nehéz megoldani), hanem értelmezhetjük F_p additív csoportján is, ahol viszont ennek megoldása már könnyű.

- Első szintű multiplikatív átvitel hajtható végre, ha l osztója $p - 1$-nek. Ebben az esetben F_p^* multiplikatív csoportjába vihető át a probléma, ahol az Index-kalkulus segítségével szubexponenciális idő alatt oldható meg a DLP. A jelenlegi módszerekkel az Index-kalkulus a fenti csoportban 2^{128} költséggel oldja meg diszkrét logaritmus problémát 2^{3000} nagyságrendű p-k esetén.

- Másod szintű multiplikatív átvitel hajtható végre, amennyiben l osztója $p^2 - 1$-nek. Ekkor $F_{p^2}^*$ multiplikatív csoportját használhatjuk, ahol az index-kalkulus szintén szubexponenciális idő alatt oldja meg a problémát. Jelenlegi állapot szerint 2^{128} költséggel egészen 2^{1500} nagyságrendű p-k esetén is.

- Harmad szintű multiplikatív átvitel hajtható végre, ha l osztja $p^3 - 1$-t. Ilyenkor $F^*_{p^3}$ multiplikatív csoportját használhatjuk, ahol szintén szubexponenciális idő alatt oldhatjuk meg a feladatot. A mostani módszerekkel 2^{128} költséggel, ha $p < 2^{1000}$

Azt a legkisebb szintű átvitelt, amelyet még el lehet végezni az adott görbén, a görbe **beágyazhatósági-szintjének** nevezik. Ez minnél magasabb annál biztonságosabb lesz a görbére épített rendszer ezen támadás ellen. Ezt a támadási típust szokták MOV támadásnak is nevezni.

3.4. Nem a ECDLP-t támadó támadások.

3.4.1. Kis Alcsoport Támadás (Small Subgroup Attack)

Ezt a fajta támadás a DH-kulcs csere közben alkalmazható. A támadó választ egy alcsony rendű pontot, Q , ezt elküldi a címzettnek, aki a titkos kulcsát felhasználva (n) elkészíti nQ-t.

Ezt használva titkosít egy üzenetet, amit visszaküld a támadónak. Mivel Q egy alacsony rendű pont, ezért nem sok lehetőség adódik nQ-ra, így ezeket végigpróbálgatva, Q-nak melyik többszöröse dekódolja az üzenetet, gyorsan megkaphatja n-t $mod\ Q$ rendje.

Általában a használt görbék rendje felírható hl alakban, ahol l a bázispontnak használt pontunk rendje (tehát egy nagy prím), h meg valami kicsi együttható. Ilyen esetben a támadó által használható pontok rendje lehet:

1. l, amiről tudjuk, hogy nagy, ezért a támadó nem megy vele sokra, szorozva h valamely osztójával,

2. vagy h illetve valamelyik osztója.

Ez alapján a Támadó legcélravezetőbb lehetősége az, ha egy h rendű pontot választ, akkor az előzőekben leírt módon tud következtetni n-re. Amennyiben a Küldő úgy választotta meg a kulcsát (n), hogy az egy véletlen egész legyen $\mod (hl)$, akkor n-re még mindíg l azonos valószínűségű lehetőség adódik. Ha n-t úgy választja meg, hogy hs alakú legyen, ahol s egy egyenletesen véletlen egész $\mod l$, akkor a támadás eredménytelen marad, mert még mindíg l külömböző lehetősége lesz n-re, így mint legjobb megoldás még

mindíg csak a ρ használata. Viszont, ha n-t úgy választotta meg a küldő, hogy egyenletesen véletlen legyen $\mod(l)$, akkor n-re l/h lehetőség adódik. Ez valamennyiben gyorsítja a ρ lefutását.

Az implementáló úgy védekezhet ezen támadás ellen, ha nem fogad el olyan pontokat amelyek rendje h, tehát: $hQ = 0$. Már a tervezésnél is lehet védekezni, amennyiben h-t 1-nek választjuk.

3.4.2. Támadás érvénytelen görbék felhasználásával (invalid-curve attack)

Ez alkalommal a támadó egy olyan alacsony rendű pontot (Q) küld a címzettnek, ami nem az "eredeti" görbén található. Például: az eredeti görbe az $y^2 = x^3 + ax + b$, de a támadó a $y^2 = x^3 + ax + c$ egyenletű görbéről választ pontot. Ezt azért teheti meg, mert ha a görbét ebben a alakban adjuk meg (Weierstrass), akkor az erre definiált számolásokban nem kerül felhasználásra a b, illetve a c, tehát a címzett számára nem derül ki, hogy a kapott pont nem az eredeti görbéről származik. Ha ezt a támadó több ponttal is végrehajtja, például: Q_2 ami egy másod rendű pont az adott görbén, valamint Q_3-al ami egy harmad rendű pont, majd Q_5-tel stb., akkor a támadó megkapja n-t $\mod 2$, $\mod 3$, $\mod 5$, stb. és ezeket felhasználva a *Kínai-maradék* tétellel kiszámíthatja n-t.

Ez ellen a támadás ellen legegyszerűbben úgy lehet védekezni, ha a kapott pontokról ellenőrizzük, hogy azok valóban az "eredeti" görbén vannak-e.

3.5. Szabványok és "törési tesztek"

Szabványokról

Láthattuk, hogy ahhoz, hogy egy elliptikus görbéken alapuló rendszert biztonságosan használni tudjunk rengetek paraméter helyes megválasztása szükséges. Lévén, hogy a feladat nem egyszerű, az évek során többféle szabványt is kidolgoztak, a kívánt cél, a biztonságos kommunikáció elérésének érdekében. Ezen szabványok olyan paraméter intervalumokat illetve generálásukhoz szükséges algoritmusokat tartalmaznak, amelyek "garantálják", hogy az így megválasztott paraméterekkel felépített rendszer biztonságos üzenetküldésre alkalmas lesz. A legtöbb szabványt úgy alkották meg, hogy biztosítsa a rendszer magját alkotó ECDLP megfelelően "nehéz" legyen. Ugyanakkor a való életben különbség adó-

dik a teljes rendszer és az ECDLP biztonsága között. Több olyan támadási forma is létezik, amely annélkül töri fel a rendszert, hogy hozzányúlna az ECDLP-hez. Lásd: kis részcsoport-támadás, érvénytelen görbe, stb. Valamint a megvalósítás során is hibák kerülhetnek a rendszerbe:

- Számolás közben hibás eredmények a görbe egyes – feltehetően ritka – pontjaira.
- Adatszivárgás történhet ha a bemenetben szereplő adatok egy, nem a görbén található pontot definiálnak.

A támadók ezeket a réseket kihasználva értékes információkhoz juthatnak. Ebből is látszik, hogy nem elég csak az ECDLP biztonságát garantálni.

- Egy valós rendszernek tudnia kell kezelni egy, a támadó szándékai szerint manipulált bemenetet.
- Papíron a rendszernek csak nP a kimenete, valós esetben viszont megfigyelhető a hardware viselkedése, így további hasznos információhoz jut a támadó.

Ezen problémák által felmerülő veszélyek csökkenthetőek, amennyiben a paramétereket és a számolási módszereket úgy választjuk meg, hogy egyszerűen és biztonságosan végrehajthatóak legyenek.

Valamint mint mindíg amikor titkosításról van szó és másoktól veszünk át szabványt, felmerül a megbízhatóság kérdése. Előfordulhat ugyanis, hogy a szabványban szereplő paraméter intervallumot, esetleg a generáláshoz felhasznált algoritmust szándékosan úgy adták meg, hogy olyan görbékhez veszessen amelyek gyengék egy vagy több, általában csak a készítő számára ismert, támadási móddal szemben. Lásd: NSA által készített Suite B szabványba beépített "hátsóajtót", amelyet a paraméterek "véletlenszerű" generálásához használt algoritmusban helyeztek el. Más szabványok bizonyos görbeparamétereket rögzítenek, állításuk szerint a görbéken való gyorsabb számolások érdekében, ugyanakkor ezen rögzitett értékek vizsgálatakor kiderült, hogy a választott értékek nem optimálisak, sőt némelyik a rendszer biztonságát is befolyásolja. Lásd: NIST P-256; IEEE P1363 szabványok[1].

[1] lásd [9]

3.5.1. Törés figyelés

Az ECC feltörhetőségének figyelésére, és ezen rendszerek jobb megértésének érdekében a Certicom 1997-ben meghírdette az "ECC Challenge" nevű "versenyt". A feladat a következő: adottak meghatározott nyilvános kulcsok a hozzájuk tartozó görbe paraméterekkel és ezekből kell meghatározni a titkos kulcsokat. Az adott nyilvános kulcsokat három szintben különítették el:

- Az első szint 79, 89 és 97 bites kulcsokat tartalmazott ezeket 1997, 1998 és 1999 folyamán fel is törték.
- A második szint egy 109 bites kulcsot és egy 131 bites kulcsot tartalmaz melyekből a 109 biteset 2002-ben törték fel, a 131 bites még feltöretlen.
- A harmadik szint 4 kulcsot tartalmaz 163, 191, 239 és 359 bites hosszakkal, melyek mindezidáig feltöretlenek.

Minden résztvevő két féle test felett próbálkozhat feltörni az adott kulcsokat: az első F_{2m} (ahol a testnek 2m eleme van) a második F_p test felett.

Köszönetnyilvánítás

Köszönetet mondok mindazoknak akik a munkámat segítették, tanárom Szabó István és barátaim.

Irodalomjegyzék

[1] Kiss Emil, *Bevezetés az algebrába*, Typotex, 2007

[2] Buttyán Levente, Vajda István, *Kriptográfia és alkalmazásai*, Typotex, 2004

[3] Liptai Kálmán, *Kriptográfia*, Eszterházy Károly Főiskola Matematikai és Informatikai Intézet, 2011

[4] Virrasztó Tamás, *Titkosítás és Adatrejtés*, NetAkadémia kft., 2004

[5] Darrel Hankerson, Alfred Menezes, Scott Vanstone, *Guide to Ellipitc Curve cryptography*,Springer-Verlag New York, Inc., 2004.

[6] Daniel R. L Brown, *Standards for Efficient Cryptography 1 (SEC 1)*, Certicom Corp., 2009

[7] Gregg Musiker, *Schoof's Algorithm for Counting Points on* $E(F_q)$, 2005.

[8] Andrea Corbellini, *ECC: a gentle itroduction*, 2015, http://andrea.corbellini.name/2015/05/17/elliptic-curve-cryptography-a-gentle-introduction/, utolsó letöltés: 2016. 12. 20.

[9] Daniel J. Bernstein and Tanja Lange, *SafeCurves: choosing safe curves for elliptic-curve cryptography*, 2014 ,http://safecurves.cr.yp.to, utolsó letöltés: 2016. 12. 20.

[10] Maróti Miklós, *Testek - előadás vázlat*, 2008, http://www.math.u-szeged.hu/ mmaroti/okt/2009t/testek.pdf, utolsó letöltés: 2016. 12. 20

[11] Padma Bh, D.Chandravathi, P.Prapoorna Roja, *Encoding And Decoding of a Message in the Implementation of Elliptic Curve Cryptography using Koblitz's Method* , 2010, http://www.enggjournals.com/ijcse/doc/IJCSE10-02-05-08.pdf, Utolsó letöltés: 2016. 12. 27.

[12] https://en.wikipedia.org/wiki/Elliptic_curve, Utolsó letöltés: 2016. 12. 28.

Név: Rusznák Demeter

ELTE Természettudományi Kar, Szak: Matematika BSc

NEPTUN kód: DK09OE

Szakdolgozat címe: Az elliptikus görbékkel való titkosítás néhány matematikai kérdése

A szakdolgozat szerzőjeként fegyelmi felelősségem tudatában kijelentem, hogy a dolgozatom önálló munkám eredménye, saját szellemi termékem, abban a hivatkozások és idézések standard szabályait következetesen alkalmaztam, mások által írt részeket a megfelelő idézés nélkül nem használtam fel.

Budapest, 2017. január 2. .....

a hallgató aláírása