8. Criptografia de corbes el·líptiques

En capítols anteriors s'ha vist com la criptografia de clau pública permetia solucionar alguns dels problemes presentats per la criptografia simètrica (com ara la distribució de claus) i alhora oferia propietats addicionals més enllà del xifratge (com ara el no repudi a través de signatures digitals). Ara bé, la criptografia de clau pública requereix de recursos computacionals més elevats que la criptografia simètrica, cosa que en dificulta la seva execució en dispositius amb poc recursos i en limita el nivell de seguretat com a conseqüència del compromís amb el temps d'execució. De fet, en el cas del xifratge, sovint es combina l'ús de la criptografia de clau pública amb criptografia simètrica a través de la tècnica del sobre digital, cosa que permet xifrar continguts de gran mida amb claus públiques més petites.

En aquest capítol es presenta la criptografia de corbes el·líptiques. Entre els seus principals avantatges hi trobem que la criptografia de corbes el·líptiques ofereix el mateix nivell de seguretat que la criptografia de clau pública tradicional però amb claus més petites. D'aquesta manera, s'aconsegueix també que les operacions criptogràfiques siguin més ràpides d'executar i requereixin de menys recursos computacionals, fent-les possibles en dispositius amb recursos limitats. A més, la criptografia de corbes el·líptiques permet la definició de *pairings*, amb els quals es poden crear construccions criptogràfiques amb propietats addicionals a les que ens oferia la criptografia de clau pública bàsica.

Així doncs, en primer lloc aquest capítol detalla els beneficis de la criptografia de corbes el·líptiques. A continuació, es presenten les corbes el·líptiques, la seva aritmètica, i es descriu com es fan servir les corbes el·líptiques en criptografia. Després s'explica el problema del logaritme discret sobre corbes el·líptiques i es detallen els algorismes criptogràfics més populars basats en aquest problema.

8.1 L'origen de la criptografia de corbes el·líptiques

Com hem vist, la criptografia de clau pública es va donar a conèixer a la dècada dels 70, amb el protocol d'intercanvi de claus de Diffie-Hellman, que permet a dues parts establir un secret compartit sense necessitat d'haver-se intercanviat cap informació prèviament. La seguretat del protocol de Diffie-Hellman es basa en la dificultat de calcular el logaritme discret en el grup dels enters mòdul un primer. Poc després, Rivest, Shamir, i Adleman van proposar l'RSA, un sistema de xifratge de clau pública que es basa en un altre problema, la factorització d'enters. Així doncs, aquests primers algorismes de clau pública es basaven en problemes computacionalment difícils de resoldre que es definien sobre els enters o sobre grups d'enters mòdul un

primer.

L'ús de corbes el·líptiques en el disseny de criptosistemes de clau pública va ser proposat per primera vegada l'any 1985 per Neal Koblitz i Victor Miller, de manera independent. Ambdós van proposar fer servir el grup de punts d'una corba el·líptica definida sobre un cos finit en criptosistemes basats en el problema del logaritme discret, donant llum així a la criptografia de corbes el·líptiques (o ECC, de les seves sigles en anglès, *Elliptic Curve Cryptography*).

Més d'una dècada després, els primers estàndards que descrivien algorismes de criptografia de corbes el·líptiques i paràmetres per a les corbes sobre les quals construir-los es van començar a publicar.

Primers estàndards d'ECC

El primer estàndard publicat sobre corbes el·líptiques va ser l'*ANSI X9.62: The Elliptic Curve Digital Signature Algorithm (ECDSA)* l'any 1999. Un any després, al 2000, el NIST també incloïa l'ECDSA a l'ara obsolet *NIST FIPS PUB 186-2: Digital Signature Standard (DSS)*.

L'adopció de la criptografia de corbes el·líptiques no ha estat, però, lliure de polèmica. L'algorisme Dual_EC_DRBG (*Dual Elliptic Curve Deterministic Random Bit Generator*) va ser estandarditzat pel NIST l'any 2006, juntament amb uns altres tres algorismes, per a la generació de números pseudoaleatoris. L'estàndard explicitava no només l'algorisme de generació de números pseudoaleatoris, sinó també la corba el·líptica concreta i els punts de la corba a fer servir per l'algorisme. Altres organismes d'estandardització també van incloure aquests mateixos paràmetres als seus estàndards. Ja durant el procés d'estandardització, alguns investigadors van començar a mostrar preocupacions per les possibles vulnerabilitats de l'algorisme. En particular, els investigadors se'n van adonar que el coneixement d'un cert secret podia permetre recuperar l'estat intern del generador a partir de només 256 bits de la sortida. A més, també van notar que era possible generar els paràmetres de l'algorisme de manera que qui ho fes conegués aquest secret. Dit d'una altra manera, els investigadors van advertir que l'algorisme podia incorporar una porta del darrere (o *backdoor*). Tot i així, diverses implementacions de llibreries criptogràfiques comercials van incorporar l'algorisme amb els paràmetres recomanats pel NIST i, en alguns casos, fins i tot van configurar-lo com a algorisme per defecte. Anys després, les sospites dels investigadors van quedar confirmades quan les revelacions d'Edward Snowden apuntaven que l'NSA havia introduït intencionadament una porta del darrere a l'algorisme Dual_EC_DRBG. Aquests fets van precipitar que l'algorisme fos retirat de l'estàndard del NIST l'any 2014.

La història del Dual EC DRBG

Per a conèixer amb més detalls els fets que van portar a l'estandardització de l'algorisme Dual_EC_DRBG i el paper que les diferents institucions i investigadors hi van jugar, us recomanem la lectura de l'article *Dual EC: A standardized back door*, de Daniel Bernstein, Tanja Lange i Ruben Niederhagen.

Nombres *nothing-up-my-sleeve* (NUMS)

En anglès es fa servir l'expressió *nothing-up-my-sleeve numbers* (literalment, nombres sense res tret de la màniga) per designar els nombres que es troben lliures de sospita de tenir propietats ocultes. Aquests nombres s'utilitzen com a constants en els algorismes criptogràfics, per exemple, per a la inicialització o la definició de paràmetres, amb l'objectiu d'assegurar que no han estat triats intencionadament per a debilitar l'algorisme o incorporar-hi portes del darrere. Per aconseguir-ho, els nombres se seleccionen usant fonts conegudes que deixin poc marge de maniobra per a manipular l'algorisme, com ara els primers dígits decimals de pi.

Tot i aquests daltabaixos amb les agències d'estandardització, la criptografia de corbes el·líptiques s'ha anat fent lloc en la societat, oferint esquemes de signatura digital, de xifratge híbrid, d'intercanvi de claus, i de generació de números pseudoaleatoris, tots ells basats en el problema del logaritme discret sobre corbes el·líptiques.

Actualment, la criptografia de corbes el·líptiques es fa servir àmpliament en protocols com ara l'IPsec o el TLS. L'empresa americana F5 (especialitzada en xarxes de lliurament d'aplicacions, gestió del núvol i seguretat a la xarxa) publica informes anuals sobre l'ús que se n'està fent del protocol TLS a Internet. El seu informe de 2019 (*The 2019 TLS Telemetry report*) reporta que del milió de pàgines web millor situades al

rànquing Alexa, prop d'un 20% estan fent servir claus basades en corbes el·líptiques. En concret, un 18.23% fan servir una clau de 256 bits sobre una corba el·líptica. Tot i així, encara predomina l'ús de l'RSA (un 73.57% de les pàgines fan servir claus RSA de 2048 bits).

La criptografia de corbes el·líptiques també s'ha popularitzat per ser utilitzada en esquemes de signatura digital en algunes criptomonedes basades en cadena de blocs (*blockchain*). Així, per exemple, Bitcoin fa servir ECDSA (*Elliptic Curve Digital Signature Algorithm*) com a algorisme de signatura; i Ethereum 2.0 fa servir signatures BLS (*Boneh–Lynn–Shacham*).

8.2 Beneficis de la criptografia de corbes el·líptiques

El principal avantatge de la criptografia de corbes el·líptiques és que utilitza claus més petites per arribar als mateixos nivells de seguretat que els algorismes de criptografia de clau pública tradicionals. Aquesta disminució en la mida de la clau comporta una millora en la velocitat d'execució d'algunes de les primitives bàsiques (per exemple, en la generació de la clau) i, alhora, un estalvi de recursos, de manera que es pot utilitzar en dispositius amb recursos limitats.

El nivell de seguretat d'un algorisme és una mesura creada per comparar la seguretat que ofereixen diferents algorismes criptogràfics quan es fan servir amb diverses mides de clau.

> El **nivell de seguretat** d'un algorisme és n quan el millor atac conegut contra l'algorisme requereix 2^n passos. El nivell de seguretat d'un algorisme també es coneix com a la **mida efectiva** de la clau i, en conseqüència, és habitual veure'l expressat en bits.

La Taula 8.1 detalla el nivell de seguretat ofert per diferents algorismes criptogràfics en funció de la mida de la clau utilitzada. Tant el nivell de seguretat com les mides de les claus estan expressades en bits. La mida de la clau correspon a la mida del mòdul per als algorismes basats en el problema de la factorització d'enters; a la mida de la clau pública per als algorismes basats en el logaritme discret; i a l'ordre del punt base per als algorismes basats en corbes el·líptiques.

Així, per exemple, l'AES amb una mida de clau de 128 bits ofereix un nivell de seguretat de 128 bits (en general, en els algorismes simètrics la mida de la clau coincideix amb el nivell de seguretat). Per aconseguir aquest mateix nivell de seguretat fent servir RSA, caldrà utilitzar un mòdul de 3072 bits. En canvi, el mateix nivell de seguretat requereix només d'una clau de 256 bits per a l'ECDSA.

Taula 8.1: Comparativa del nivell de seguretat proporcionat per diferents mides de clau (en bits) depenent de l'algorisme criptogràfic.

	Algorismes criptogràfics			
	Clau simètrica	Factorització d'enters	Logaritme discret	Corbes el·líptiques
Nivell de seguretat	AES, 3DES	RSA	DSA, DH, ElGamal	ECDSA, ECDH
80	80	1024	1024	160
112	112	2048	2048	224
128	128	3072	3072	256
192	192	7680	7680	384
256	256	15360	15360	512

És interessant destacar no només la diferència en la mida de la clau, sinó també en el creixement d'aquesta mida en funció del nivell de seguretat. El creixement de la mida de la clau és molt més ràpid per a algorismes basats en la factorització d'enters i el logaritme discret que en algorismes basats en corbes el·líptiques, de manera que les diferències en les mides de clau entre aquests grups d'algorismes s'incrementen amb l'augment del nivell de seguretat.

També cal remarcar que el nivell de seguretat d'un algorisme criptogràfic es calcula en base a la complexitat

del millor algorisme que es coneix en un moment donat per a trencar-lo. Per tant, els nivells de seguretat detallats en aquesta taula reflecteixen el coneixement que es té actualment sobre possibles tècniques de factorització i/o càlcul del logaritme discret. Amb els avenços en la investigació criptogràfica es poden descobrir noves maneres d'atacar aquests problemes, que facin variar aquests nivells de seguretat.

El cost de bullir aigua

La definició que acabem de proporcionar del nivell de seguretat d'un algorisme criptogràfic és poc intuïtiva, en tant que és difícil valorar l'esforç necessari per trencar un criptosistema d'un determinat nivell de seguretat.

Arjen K. Lenstra i altres van proposar fer servir com a mesura més informal de la seguretat d'un algorisme la quantitat d'aigua que s'aconseguiria fer bullir amb l'energia necessària per trencar l'algorisme. Així, l'energia necessària per a trencar una clau RSA de 242 bits és l'equivalent a la necessària per a fer bullir l'aigua que hi cap en una cullereta de cafè. Trencar l'RSA de 745 bits és l'equivalent a fer bullir l'aigua d'una piscina, i per trencar l'RSA de 2380 bits caldria tanta energia com la necessària per fer bullir tota l'aigua del planeta Terra!

Tot i això, l'ús de la criptografia de corbes el·líptiques té també alguns inconvenients en relació als algorismes de clau pública basats en els problemes tradicionals. D'una banda, certs aspectes legals poden dificultar-ne la seva adopció. Algunes empreses tenen patentats diferents aspectes de la criptografia de corbes el·líptiques, cosa que pot dificultar-ne el seu desplegament. D'altra banda, el fet que siguin algorismes més recents fa que els estàndards estiguin menys desenvolupats, i també genera dubtes sobre possibles vulnerabilitats no conegudes. Així, per exemple, encara hi ha poca recerca en aspectes com ara els atacs de canal lateral. Finalment, la complexitat de les matemàtiques rere de les corbes el·líptiques també en dificulta la seva comprensió i accessibilitat, cosa que pot afectar a la seguretat de les implementacions.

8.3 Corbes el·líptiques

La criptografia de corbes el·líptiques proporciona algorismes de clau pública basats en l'estructura algebraica de les corbes el·líptiques definides sobre cossos finits.

Definició 8.1 La **corba el·líptica** $E/\mathbb{Z}_p$ (amb $p > 3$) és el conjunt de tots els parells $(x,y) \in \mathbb{Z}_p$ tals que:

$$y^2 = x^3 + ax + b \mod p$$

juntament amb un punt imaginari a l'infinit $\mathcal{O}$, amb $a,b \in \mathbb{Z}_p$ i $\Delta = -16(4a^3 + 27b^2) \neq 0 \mod p$.

L'expressió que defineix la corba el·líptica tal com l'acabem d'exposar es coneix com la forma curta de Weierstrass.

El valor Δ correspon al discriminant de la corba. Geomètricament, la condició que el discriminant sigui diferent de zero assegura que la corba no té cap vèrtex ni es creua amb sí mateixa, és a dir, no té cap punt que tingui dues o més rectes tangents. Això faria que la corba no fos adequada per al seu ús en els algorismes criptogràfics que descriurem a continuació.

Exemple 8.1 Exemple de punts sobre una corba el·líptica

La corba el·líptica $E/\mathbb{Z}_{11} : y^2 = x^3 - 5x + 5$ té 17 elements:

$$[\mathcal{O}, (0,4), (0,7), (1,1), (1,10), (2,5), (2,6), (4,4), (4,7), (6,2), (6,9), (7,4), (7,7), (8,2), (8,9), (10,3), (10,8)]$$

Podem comprovar que aquests punts efectivament pertanyen a la corba verificant que compleixen l'equació

que la defineix. Així, per exemple, per al punt $(0,4)$, tenim que:

$$\begin{aligned} y^2 &= x^3 - 5x + 5 \mod 11 \\ 4^2 &= 0^3 - 0x + 5 \mod 11 \\ 16 &= 5 \mod 11 \end{aligned}$$

També podem verificar que la corba té discriminant diferent de zero:

$$-16(4a^3 + 27b^2) \mod p = -16(4(-5)^3 + 27(5)^2) \mod 11 = 5 \neq 0 \mod 11$$

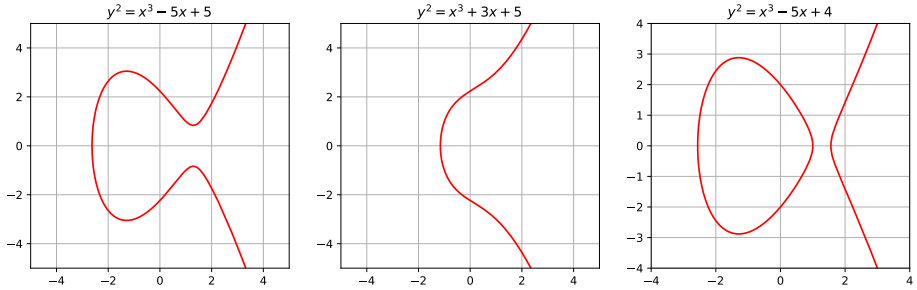
Exercici 8.1 Calculeu tots els punts de la corba el·líptica $E/\mathbb{Z}_5 : y^2 = x^3 - 4x + 1$.

En criptografia es fan servir principalment corbes el·líptiques sobre cossos finits, tal com les acabem de definir. Ara bé, la representació geomètrica de les corbes sobre els reals ens permet obtenir una visualització més intel·ligible d'aquestes. Així, en els propers paràgrafs presentarem les corbes el·líptiques sobre els reals, per tal d'apropar-nos a la seva descripció i les seves propietats.

8.3.1 Corbes el·líptiques sobre els reals

Les figures següents mostren tres exemples de corbes el·líptiques definides sobre els nombres reals, és a dir, corbes de la forma $y^2 = x^3 + ax + b$ on $(x,y) \in \mathbb{R}$:

Figura 8.1: Exemples de corbes el·líptiques sobre $\mathbb{R}$.



Com podem observar, les corbes són simètriques respecte a l'eix de les abscisses. Això és així ja que y és el resultat d'una arrel quadrada, de manera que per cada valor d'x avaluat, obtindrem dos valors per a y, que correspondran al valor positiu i al negatiu de l'arrel (sempre que aquesta sigui diferent de 0).

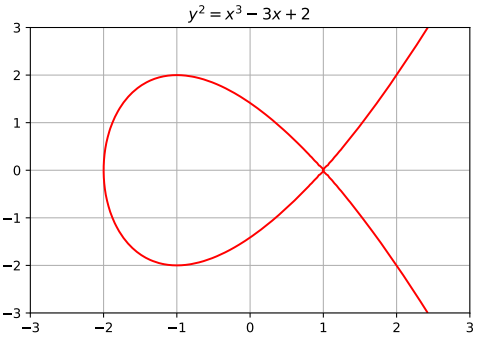
D'altra banda, les corbes de la figura anterior no es creuen amb sí mateixes ni tenen cap vèrtex, ja que el discriminant de totes elles és diferent de zero. En canvi, la corba de la Figura 8.2 té discriminant zero ($a = -3, b = 2$ i, per tant, $\Delta = 4(-3)^3 + 27(2)^2 = 0$) i es creua amb sí mateixa en el punt $(1,0)$:

La criptografia de corbes el·líptiques treballa sobre un grup. Per tant, a més dels punts de la corba, que seran els elements d'aquest grup, necessitem definir una **operació de grup**.

Grup

Tal com s'ha presentat al capítol de Fonaments matemàtics, un **grup** és una estructura algebraica en què l'operació definida compleix la propietat associativa i, a més, el conjunt sobre el qual està definida l'operació conté l'element neutre i l'element invers d'aquesta operació.

Figura 8.2: Exemple de corba el·líptica amb discriminant zero.

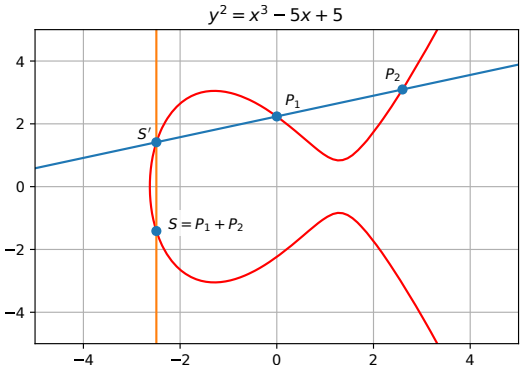


Siguin $P_1 = (x_1, y_1)$ i $P_2 = (x_2, y_2)$ dos punts sobre una corba el·líptica, definirem una operació *suma* de la manera següent.

Si els dos punts són diferents (és a dir, $P_1 \neq P_2$), per calcular el punt resultant de la suma, $S = P_1 + P_2$, traçarem la recta entre P_1 i P_2; trobarem el tercer punt d'intersecció S' d'aquesta recta amb la corba el·líptica; i buscarem el punt simètric d'S' respecte a l'eix de les x, S. Aquest punt simètric S serà el resultat de la suma.

La Figura 8.3 mostra un exemple d'una suma de dos punts diferents, P_1 i P_2. La línia blava correspon a la recta que passa pels dos punts. El tercer punt d'intersecció de la recta amb la corba el·líptica és el punt S'. La línia taronja és una recta vertical que passa pel punt S', i que ens permet calcular el punt simètric S respecte a l'eix de les x. Aquest punt simètric S és el resultat de la suma $P_1 + P_2$.

Figura 8.3: Suma de dos punts diferents.



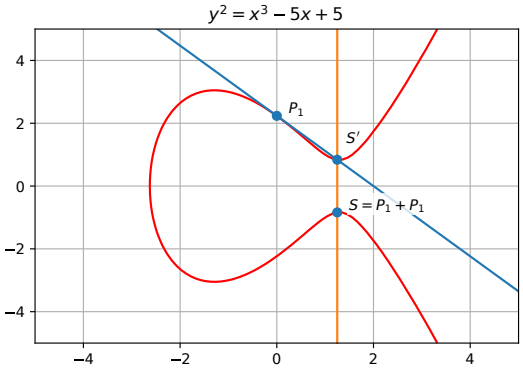
Si els dos punts són iguals (és a dir, $P_1 = P_2$), per calcular el punt suma, $S = P_1 + P_1$, caldrà fer una petita modificació al procediment: la línia a traçar en el primer pas del procediment serà la recta tangent a la corba el·líptica en el punt P_1. Després, es procedeix anàlogament a l'operació de suma de punts diferents: es troba el punt d'intersecció S' de la recta tangent amb la corba el·líptica i, de nou, es busca el punt simètric S respecte a l'eix de les x.

La Figura 8.4 mostra un exemple d'una suma d'un punt P_1 amb ell mateix. La línia blava correspon a la recta tangent a la corba el·líptica en el punt P_1. El punt d'intersecció de la recta amb la corba és el punt S'. De nou, la línia taronja és una recta vertical que passa pel punt S' i que permet calcular el punt simètric S, que és el resultat de la suma $P_1 + P_1$.

El mètode que acabem de descriure per a sumar punts d'una corba és coneix amb el nom de **mètode de la corda i la tangent**.

Ja tenim doncs el conjunt d'elements del grup (els punts de la corba) i una operació de grup (la *suma* que

Figura 8.4: Suma d'un punt amb ell mateix.



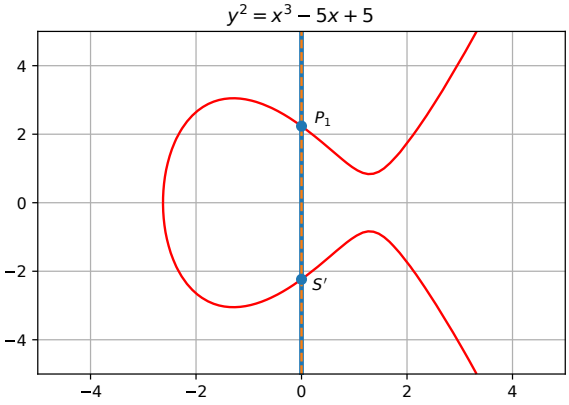
acabem de definir). Per tal d'acabar de definir el grup, caldrà disposar d'un element neutre respecte a l'operació *suma*, és a dir, un element tal que:

$$P_1 + \mathscr{O} = P_1$$

Doncs bé, aquest element neutre és precisament el punt imaginari a l'infinit $\mathscr{O}$ que afegíem com a element de la corba en la definició de l'inici del capítol. Podem imaginar aquest punt com a un punt situat a l'infinit als finals de l'eix de les y. Aquest element és necessari, ja que no hi ha cap altre punt sobre la corba que compleixi que sumat a un altre punt P_1 obtinguem com a resultat el mateix P_1.

Podem fer servir la mateixa estratègia que hem descrit anteriorment per sumar un punt P_1 amb l'element $\mathscr{O}$, i comprovar com efectivament el resultat és el mateix P_1. Així, per calcular $S = P_1 + \mathscr{O}$, en primer lloc es traça una recta entre el punt P_1 i $\mathscr{O}$. Aquesta recta serà la recta vertical que passi pel punt P_1 (en l'exemple de la Figura 8.5, correspon a la línia blava). A continuació, es troba el segon punt d'intersecció S' d'aquesta recta amb la corba el·líptica. Finalment, es busca el punt simètric d'S' respecte a l'eix de les x. Aquest punt simètric S serà el resultat de la suma, i serà precisament el mateix punt P_1.

Figura 8.5: Suma d'un punt amb l'element neutre $\mathscr{O}$.

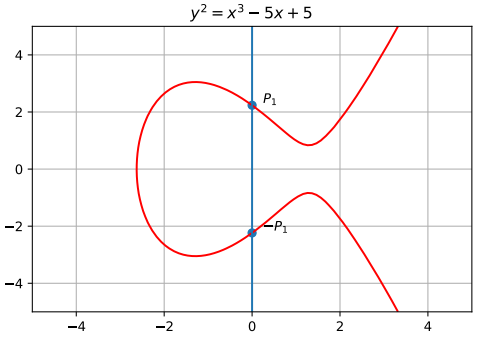


En un grup cal que els elements tinguin inversos. Una vegada tenim el punt neutre definit, podem definir l'invers additiu de qualsevol punt P_1 de la corba com l'element $-P_1$ tal que:

$$P_1 + (-P_1) = \mathscr{O}$$

Donat un punt $P_1 = (x_1, y_1)$, el seu invers additiu $-P_1$ és simplement $(x_1, -y_1)$. Efectivament, si sumem P_1 i $-P_1$ fent servir l'operació suma que hem definit anteriorment, obtenim com a resultat el punt a l'infinit $\mathcal{O}$.

Figura 8.6: L'invers d'un punt.



Això fa que calcular l'invers d'un punt a la corba sigui molt eficient, ja que per a un punt $P_1 = (x_1, y_1)$, el seu invers additiu $-P_1$ és simplement $(x_1, -y_1)$.

Semàntica

En aquest capítol fem servir els termes *suma*, *neutre* i *invers additiu* (o simplement *invers*) per a referir-nos a l'operació que definim sobre els punts de la corba el·líptica, l'element tal que sumat a un punt dona el mateix punt, i el punt tal que sumat a un altre punt dona l'element neutre. Aquests termes són, però, una mica arbitraris. Podríem haver utilitzat algun altre nom per a descriure l'operació (per exemple, multiplicació) i altres termes com ara element identitat i element negatiu per referir-nos a l'element neutre i a l'invers d'un punt.

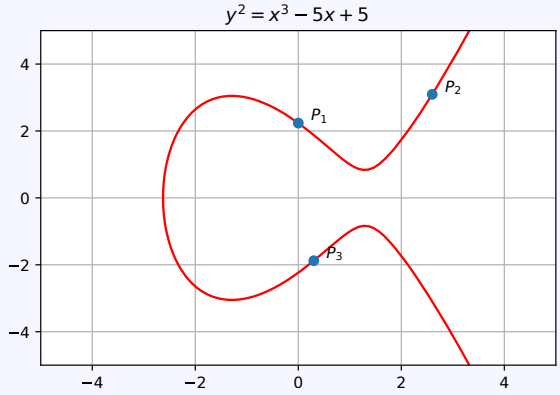
Per últim, per a tenir estructura de grup l'operació *suma* ha de ser associativa, és a dir, donats tres punts P_1, P_2 i $P_3 \in E$, aquests han de complir que:

$$(P_1 + P_2) + P_3 = P_1 + (P_2 + P_3)$$

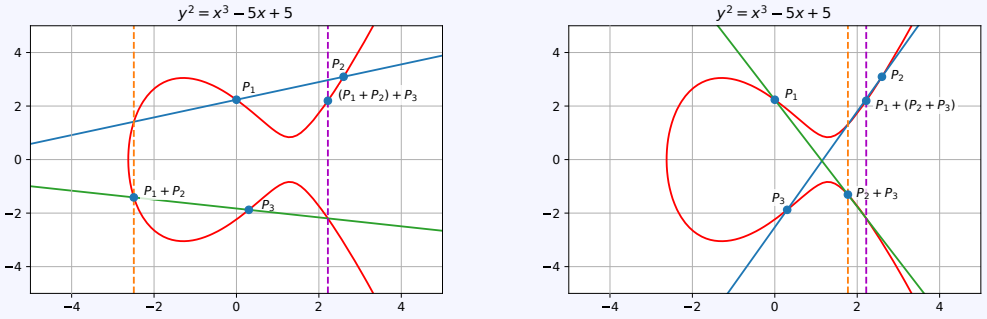
La demostració d'aquesta propietat queda fora de l'abast d'aquest document, però en veurem un exemple.

Exemple 8.2 Exemple de propietat associativa de la suma

Donats tres punts P_1, P_2 i $P_3 \in E/\mathbb{R} : y^2 = x^3 - 5x + 5$, calcularem el resultat de la suma $P_1 + P_2 + P_3$ executant les sumes entre dos punts en ordres diferents, i comprovarem que el resultat és el mateix.



Les dues imatges següents mostren gràficament la suma dels tres punts. A la imatge de l'esquerra s'efectua la suma $(P_1+P_2)+P_3$; a la imatge de la dreta es calcula $P_1+(P_2+P_3)$.



Per a calcular $(P_1+P_2)+P_3$ (imatge de l'esquerra) se suma primer P_1+P_2 (la línia blava mostra la recta entre els dos punts i la línia taronja el punt simètric al tercer punt d'intersecció amb la corba). A continuació se suma el resultat amb P_3: la línia verda mostra la recta entre els dos punts a sumar i la línia violeta el punt simètric al tercer punt d'intersecció amb la corba, que és el resultat de la suma dels tres punts.

Anàlogament, per calcular $P_1+(P_2+P_3)$ (imatge de la dreta) se suma P_2+P_3 (línies blava i taronja) i al resultat se li suma P_1 (línies verda i violeta).

En efecte, el resultat d'ambdues operacions és el mateix punt.

Així doncs, els punts sobre una corba el·líptica (juntament amb el punt a l'infinit) i l'operació suma que acabem de definir (amb el punt a l'infinit com a element neutre que permet definir els inversos dels elements) formen un grup. Ara bé, per tal que aquest grup pugui ser usat en criptografia, caldrà deixar enrere la representació sobre els reals i tornar als cossos finits.

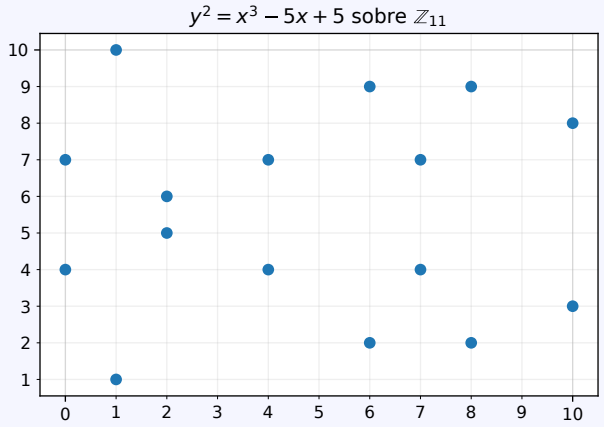
8.3.2 Corbes el·líptiques sobre cossos finits

Deixant enrere la representació de les corbes el·líptiques sobre els reals, que ajuda a comprendre'n les seves característiques però que no és gaire útil per a la criptografia, reprenem ara les corbes el·líptiques sobre cossos finits, tal com s'han definit a l'inici del capítol.

Es poden representar gràficament els punts que conformen una corba el·líptica sobre un cos finit de manera similar a com es fa sobre els reals. En aquest cas, però, es deixa de visualitzar la forma de la corba, i simplement es podrà observar el conjunt de punts i algunes propietats de la seva estructura. En particular, se segueix mantenint la simetria respecte l'eix de les x.

Exemple 8.3 Exemple de representació gràfica d'una corba el·líptica sobre un cos finit

A continuació es representa gràficament la corba el·líptica de l'Exemple 8.1, $E/\mathbb{Z}_{11}: y^2 = x^3 - 5x + 5$, que com s'ha vist té 17 elements. Val a dir que a la figura s'observen només 16 punts, ja que el 17è element correspon al punt a l'infinit $\mathcal{O}$.

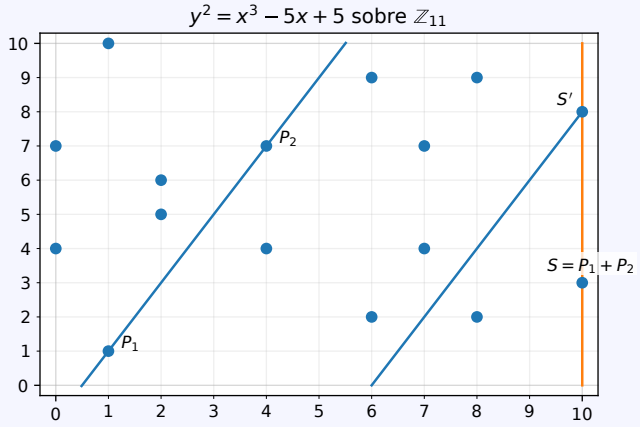


La llei de grup

Podem mantenir la definició de l'operació *suma* que s'ha presentat a la secció anterior, treballant ara sobre el cos finit, com a operació de grup. Gràficament, es pot seguir aplicant el mateix procediment per tal de calcular el resultat d'una suma de dos punts, considerant ara però que les rectes que es tracen són mòdul el primer.

Exemple 8.4 Exemple de suma de punts d'una corba el·líptica sobre un cos finit

Seguint amb la corba dels exemples anteriors, $E/\mathbb{Z}_{11} : y^2 = x^3 - 5x + 5$, calculem la suma entre els punts $P_1 = (1,1)$ i $P_2 = (4,7)$ gràficament. Per fer-ho, es traça la recta que els uneix, prolongant-la si cal considerant el mòdul, fins a trobar el tercer punt d'intersecció de la recta amb els punts de la corba, $S' = (10,8)$. Finalment, es calcula el punt simètric d'S', S, que correspon al resultat de la suma ($S = P_1 + P_2 = (10,3)$).



A la pràctica, però, quan s'opera amb punts sobre corbes el·líptiques, es fan servir expressions analítiques per tal de calcular els resultats de les operacions. A continuació es detallen les expressions que permeten

calcular la suma de dos punts sobre una corba el·líptica mòdul un primer.

Donats dos punts, $P_1 = (x_1, y_1)$ i $P_2 = (x_2, y_2)$, que pertanyen a una corba el·líptica $E/\mathbb{Z}_p$, el punt P_3 resultant de la suma, $P_3 = P_1 + P_2 = (x_3, y_3)$, es pot calcular com:

Si $P_1 \neq P_2$:

$$m = \frac{y_2 - y_1}{x_2 - x_1} \mod p$$

Si $P_1 = P_2$:

$$m = \frac{3x_1^2 + a}{2y_1} \mod p$$

i en ambdós casos:

$$x_3 = m^2 - x_1 - x_2 \mod p$$
$$y_3 = m(x_1 - x_3) - y_1 \mod p$$

Convé esmentar que el valor m que es calcula en el primer pas de la suma correspon al pendent de la recta entre els dos punts (quan els punts a sumar són diferents entre ells) o bé de la recta tangent a la corba que passa pel punt (quan els punts a sumar són iguals).

D'altra banda, pel que fa a la terminologia, a vegades es distingeix entre la *suma de punts* i el *doblat d'un punt* per referir-se, respectivament, a la suma de punts diferents (és a dir, el cas $P_1 \neq P_2$) i de punts iguals (és a dir, el cas $P_1 = P_2$).

Exemple 8.5 Suma de punts

Donats els punts $P_1 = (x_1, y_1) = (1, 10)$ i $P_2 = (x_2, y_2) = (4, 7)$ de la corba $E/\mathbb{Z}_{11} : y^2 = x^3 - 5x + 5$, podem calcular el punt $P_3 = P_1 + P_2$ de la manera següent.

Com que $P_1 \neq P_2$, aleshores el pendent m és:

$$m = \frac{y_2 - y_1}{x_2 - x_1} \mod p = \frac{7 - 10}{4 - 1} \mod 11 = \frac{8}{3} \mod 11 = 10$$

Després, es calculen les coordenades del punt:

$$x_3 = m^2 - x_1 - x_2 \mod p = 10^2 - 1 - 4 \mod 11 = 7$$

$$y_3 = m(x_1 - x_3) - y_1 \mod p = 10(1 - 7) - 10 \mod 11 = 7$$

Finalment, es pot comprovar com $P_3 = (x_3, y_3) = (7, 7)$ es troba efectivament a la corba E:

$$y^2 = x^3 - 5x + 5 \mod 11$$
$$7^2 = 7^3 - 5 \cdot 7 + 5 \mod 11$$
$$49 = 313 \mod 11$$
$$5 = 5 \mod 11$$

Exercici 8.2 Donats els punts $P_1 = (x_1, y_1) = (1,1)$ i $P_2 = (x_2, y_2) = (4,7)$ de la corba $E/\mathbb{Z}_{11} : y^2 = x^3 - 5x + 5$, calculeu analíticament el punt resultant de la suma $P_3 = P_1 + P_2$, i comproveu que el resultat coincideix amb el que hem calculat gràficament a l'Exemple 8.4.

Doncs bé, els punts de la corba sobre un cos finit, amb l'operació suma que acabem de definir, poden formar un grup cíclic, sobre el qual es poden construir algorismes criptogràfics basats en el problema del logaritme discret.

Grup cíclic

Tal com s'ha presentat al capítol de Fonaments matemàtics, un **grup cíclic** és un grup que conté un element g (que s'anomena generador) tal que les seves potències generen tots els elements del grup, llevat del zero.

Exemple 8.6 Exemple de grup cíclic sobre una corba el·líptica

Com hem vist a l'exemple anterior, la corba el·líptica $E/\mathbb{Z}_{11} : y^2 = x^3 - 5x + 5$ té 17 elements:

$$[\mathcal{O}, (0,4), (0,7), (1,1), (1,10), (2,5), (2,6), (4,4), (4,7), (6,2), (6,9), (7,4), (7,7), (8,2), (8,9), (10,3), (10,8)]$$

Aquests punts formen un grup cíclic d'ordre 17. Com ja hem vist al capítol de Fonaments Matemàtics, com que 17 és primer, tots els elements són primitius i, per tant, podem generar tots els punts de la corba sumant un punt amb sí mateix iterativament. Per exemple, per a $P = (0,4)$:

$$\begin{aligned}
P &= (0,4) \\
2P &= P + P = (4,4) \\
3P &= P + P + P = (7,7) \\
4P &= (8,2) \\
5P &= (1,10) \\
6P &= (2,6) \\
7P &= (10,8)
\end{aligned}
\qquad
\begin{aligned}
8P &= (6,9) \\
9P &= (6,2) \\
10P &= (10,3) \\
11P &= (2,5) \\
12P &= (1,1) \\
13P &= (8,9) \\
14P &= (7,4)
\end{aligned}
\qquad
\begin{aligned}
15P &= (4,7) \\
16P &= (0,7) \\
17P &= \mathcal{O} \\
18P &= (0,4) \\
19P &= \cdots
\end{aligned}$$

Exercici 8.3 Genereu tots els punts de la corba el·líptica $E/\mathbb{Z}_{11} : y^2 = x^3 - 5x + 5$ fent servir el punt $P = (8,9)$ com a generador.

Exercici 8.4 Calculeu quin és l'invers del punt $P_1 = (4,7)$ de la corba $E/\mathbb{Z}_{11} : y^2 = x^3 - 5x + 5$.

Exercici 8.5 La corba el·líptica $E/\mathbb{Z}_{11} : y^2 = x^3 - 3x + 6$ té 9 elements:

$$[\mathcal{O}, (1,2), (1,9), (4,5), (4,6), (7,3), (7,8), (9,2), (9,9)]$$

Podem generar tots els punts de la corba a partir del punt $(4,5)$?

Multiplicació escalar

Cal destacar que a l'exemple anterior implícitament acabem de definir l'operació de **multiplicació escalar** en una corba el·líptica com a l'operació de suma reiterada d'un element amb ell mateix (escrivíem $2P$ per a representar $P+P$; $3P$ per a representar $P+P+P$, etc.).

Per tal de calcular de manera eficient operacions de multiplicació escalar, es pot utilitzar una generalització de l'algorisme de multiplicar i elevar. Com que s'ha fet servir notació additiva per a definir l'operació de grup en corbes el·líptiques però, en canvi, la versió tradicional de l'algorisme de multiplicar i elevar fa servir notació multiplicativa, caldrà adaptar l'algorisme. En concret, caldrà substituir les operacions de multiplicació per l'operació *suma* de punts que s'ha definit. A més, com que en corbes el·líptiques sovint es fa servir notació additiva (tal com estem fent en aquest capítol), l'algorisme per a corbes el·líptiques es coneix habitualment amb el nom d'**algorisme de doblar i sumar**.

Així, per tal de calcular la multiplicació escalar entre un punt (`point`) d'una corba (`ec`) i un enter (`scalar`), podem fer servir l'algorisme següent:

```
def double_and_add(point, scalar, ec):
    """
    Donat un punt (point) que pertany a una corba (ec) i un enter
        (scalar), retorna la multiplicació escalar del punt per
         l'enter.
    """

    # Obtenim la representació binària de l'enter
    bin_scalar = bin(scalar)[2:]

    # Inicialitzem el resultat amb l'element neutre
    result = ec.neutre

    # Recorrem la representació binària des del dígit
    # menys significatiu al més significatiu
    for i, e in enumerate(bin_scalar[::-1]):
        if e == "1":
            result = (result + point) # sumar
        if i != len(bin_scalar) - 1: # si no és l'última iteració
            point = (point + point) # doblar
    return result
```

Per a un escalar s, l'algorisme requereix $\log_2 s$ iteracions del bucle, cadascuna de les quals pot comportar una o dues sumes de punts, en funció del valor del bit de l'enter que s'està processant a cada iteració. Per tant, en el pitjor cas l'algorisme requereix de $2\log_2 s$ sumes.

A més de permetre calcular la multiplicació de manera més eficient que sumant repetidament el punt amb sí mateix, l'algorisme de doblar i sumar té un altre avanatge. Quan el punt P a multiplicar és un punt fixat (per exemple, si es fa servir una corba estandarditzada com les que veurem més endavant), es pot accelerar el temps de computació precalculant i emmagatzemant alguns valors que es reutilitzen sovint. En concret, s'emmagatzemen els resultats d'anar doblant el punt P (és a dir, $2P, 4P, 8P, \ldots$), de manera que el pitjor cas només requereixi de $\log_2 s$ sumes. Aquesta reducció en el temps d'execució ve, però, a canvi d'un increment en l'espai necessari per córrer l'algorisme, doncs cal desar els resultats precalculats.

Exemple 8.7 Exemple de multiplicació escalar

Procedim a calcular $10P$ per a $P = (0,4) \in E/\mathbb{Z}_{11} : y^2 = x^3 - 5x + 5$ fent servir l'algorisme de doblar i sumar.

La representació binària de l'enter 10 és 1010. Per tant, es faran quatre iteracions del bucle.

En primer lloc s'inicialitza `result` a $\mathscr{O}$. A continuació s'executen les iteracions:

Iteració	e	Còmput
1	$e=0$	`point = point + point` $=(0,4)+(0,4)=(4,4)$
2	$e=1$	`result = result + point` $=\mathscr{O}+(4,4)=(4,4)$
		`point = point + point` $=(4,4)+(4,4)=(8,2)$
3	$e=0$	`point = point + point` $=(8,2)+(8,2)=(6,9)$
4	$e=1$	`result = result + point` $=(4,4)+(6,9)=(\mathbf{10},\mathbf{3})$

Per tant, $10P=(10,3)$. Cal remarcar, d'una banda, que el càlcul ha requerit únicament de quatre operacions de suma entre punts (en comptes de les nou que caldrien si haguéssim anat sumant repetidament P amb sí mateix). D'altra banda, és interessant notar com s'ha arribat al resultat: la variable `point` conté, a cada iteració, el resultat de doblar P (els valors $2P=(4,4)$, $4P=(8,2)$ i $8P=(6,9)$); i la variable `result` acumula els valors que permeten calcular el resultat i que vénen indicats per la representació binària de l'escalar (en aquest cas, $10P=8P+2P$).

Exercici 8.6 Calculeu $12P$ per a $P=(7,2)\in E/\mathbb{Z}_{23}: y^2=x^3+3x+8$ fent servir l'algorisme de doblar i sumar.

El nombre de punts d'una corba el·líptica

Comptar el nombre de punts d'una corba el·líptica sobre un cos finit $\#E/\mathbb{Z}_p$ no és senzill. El teorema de Hasse, provat l'any 1933, proporciona uns llindars que permeten acotar aquest valor.

Teorema 8.1 Donada una corba el·líptica $E/\mathbb{Z}_p$ el nombre de punts de la corba $\#E$ compleix que:

$$|\#E-(p+1)|\leq 2\sqrt{p}$$

Exercici 8.7 Proporcioneu una estimació del nombre de punts de la corba el·líptica $E/\mathbb{Z}_{11}: y^2=x^3-3x+6$ fent servir el teorema de Hasse.

Però el teorema de Hasse no ens permet calcular el nombre exacte de punts i, com hem vist, aquest és important per caracteritzar el grup que es genera. El mètode més simple per trobar el nombre exacte de punts d'una corba consisteix a calcular per cadascun dels possibles valors d'$x\in\mathbb{Z}_p$, el nombre de solucions que té l'equació de la corba (tal com es proposa a la solució de l'Exercici 8.1). Ara bé, aquest mètode no és computacionalment viable per a les corbes que són útils en criptografia.

Durant anys no es coneixia cap algorisme eficient per al càlcul exacte del nombre de punts d'una corba el·líptica (els algorismes existents eren exponencials). Això va canviar l'any 1985 amb la publicació de l'algorisme de Schoof, el primer algorisme amb temps d'execució polinomial que permetia comptar els punts d'una corba el·líptica. Aquesta versió de l'algorisme seguia sent ineficient per a corbes amb l'ordre necessari per a aplicacions criptogràfiques, però una versió millorada d'aquest, l'algorisme de Schoof-Elkies-Atkin (SEA), sí que es pot fer servir a la pràctica per a aquestes corbes. L'algorisme de SEA és actualment el millor algorisme genèric conegut per a comptar punts de corbes el·líptiques.

Estructura dels grups generats per corbes el·líptiques

Sabem doncs que els punts d'una corba el·líptica sobre un cos finit amb l'operació suma poden formar un grup cíclic. Ara bé, ens podem preguntar si això és sempre així. La resposta és negativa. Si l'ordre de la corba, $\#E/\mathbb{Z}_p$, es pot factoritzar en el producte de primers diferents, aleshores el grup $E/\mathbb{Z}_p$ és cíclic. En cas contrari, el grup és isomorf al producte directe de dos grups cíclics.

Grups isomorfs i producte directe

Informalment, diem que dos grups són isomorfs si tenen la mateixa estructura, és a dir, si les diferències entre els dos grups són només *cosmètiques* (per exemple, en el nom dels elements). D'altra banda, el producte directe de dos grups és un grup que té com a elements els membres del producte Cartesià entre els dos grups. L'operació de grup opera component a component, utilitzant l'operació definida en cada grup d'origen. Per a una introducció més formal i completa a aquests termes, us recomanem la lectura de *Further pure mathematics: Group theory* de *The Open University* (2016).

El cas particular en què el nombre d'elements és primer és el que hem anat veient en la majoria d'exemples d'aquest capítol. En aquest cas, el grup $E/\mathbb{Z}_p$ és cíclic i, a més, tots els elements (excepte el punt a l'infinit $\mathcal{O}$) en són generadors. A continuació estudiarem l'estructura dels grups formats per corbes el·líptiques amb nombre d'elements no primer i, per fer-ho, veurem algunes definicions i teoremes que descriuen l'estructura de grups finits (no necessàriament definits per corbes el·líptiques).

Definició 8.2 Donat un grup $(G,\cdot)$, un subconjunt d'elements $H \subseteq G$ és un **subgrup** de G si $(H,\cdot)$ és un grup. Per denotar que H és un subgrup de G, escrivim $H \leq G$.

Aquesta definició segueix el que entendríem intuïtivament com a un subgrup, és a dir, un subgrup no és res més que un subconjunt d'elements d'un grup que manté les propietats de grup amb la mateixa operació (associativitat, element neutre i element invers).

Definició 8.3 Donat un grup $(G,\cdot)$ i un element $e \in G$, el subgrup H format per les potències de l'element e és un **subgrup cíclic** de G.

Això es deriva directament de les definicions de subgrup i grup cíclic. L'element e les potències del qual generen el subgrup cíclic és doncs el generador del subgrup.

Teorema 8.2 L'ordre d'un element $e \in G$ és igual a l'ordre del subgrup cíclic que genera.

Donat un grup qualsevol, ens podem preguntar com són els subgrups que conté. El teorema de Lagrange ens descriu l'ordre d'aquests subgrups.

Teorema 8.3 El teorema de Lagrange estableix que per tot grup finit G, l'ordre de cada subgrup de G és un divisor de l'ordre de G.

Fixeu-vos que el teorema de Lagrange ens diu que l'ordre dels subgrups de G (i, per tant, l'ordre dels elements de G) és un divisor de l'ordre de G, però no ens descriu si per a cada divisor de l'ordre de G hi ha un element que té aquell ordre.

Si el grup G és cíclic, podem determinar exactament quants elements amb cada ordre possible hi ha.

Teorema 8.4 Sigui G un grup cíclic d'ordre n, si $d|n$ aleshores hi ha $\phi(d)$ elements $g \in G$ d'ordre d.

Exemple 8.8 Exemple de grup cíclic amb ordre no primer amb enters

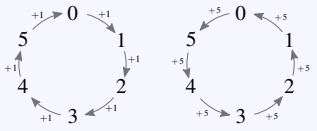
Sigui $G = (\mathbb{Z}_6,+)$ el grup format pels enters mòdul 6 amb l'operació suma modular. Veiem alguns exemples de les propietats de $(\mathbb{Z}_6,+)$ com a grup:

- L'operació suma és associativa, per exemple, $2+(3-5) \mod 6 = (2+3)-5 \mod 6$.
- Té element neutre respecte a la suma modular, el 0, ja que qualsevol element sumat a 0 dona el mateix element.
- Per a qualsevol element e del grup, l'element $-e \mod 6$ és l'invers additiu, ja que $e-e=0$

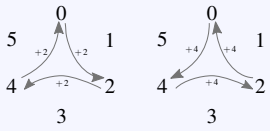
mod 6.

El grup $(\mathbb{Z}_6,+)$ és un grup cíclic, ja que es pot generar a partir de les *potències* dels elements 1 i 5. Noteu que en aquest cas, com que estem fent servir notació additiva, les *potències* són la suma repetida (i no pas la multiplicació).

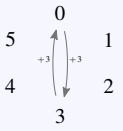
El grup $(\mathbb{Z}_6,+)$ té dos subgrups (més enllà del subgrup amb l'element neutre i d'ell mateix). L'element 2 genera un subgrup cíclic d'ordre 3, format pels elements $\{2,4,0\}$. L'element 4 també genera aquest mateix subgrup. D'altra banda, l'element 3 genera el subgrup cíclic d'ordre 2: $\{3,0\}$. Noteu com tots els subgrups de $(\mathbb{Z}_6,+)$ contenen l'element 0.



Subgrup d'ordre 6
Generadors: 1, 5



Subgrup d'ordre 3
Generadors: 2, 4



Subgrup d'ordre 2
Generadors: 3

Tal com ens diu el teorema de Lagrange, l'ordre dels subgrups és un divisor de l'ordre de G ($2|6$ i $3|6$).

A més, com que G és cíclic, sabem que hi ha $\phi(2)=1$ element d'ordre 2 (l'element 3) i $\phi(3)=2$ elements d'ordre 3 (els elements 2 i 4).

Ordre	Núm. d'elements	Elements	Subgrup
1	$\phi(1)=1$	0	$\{0\}$
2	$\phi(2)=1$	3	$\{0,3\}$
3	$\phi(3)=2$	2, 4	$\{0,2,4\}$
6	$\phi(6)=2$	1, 5	$\{0,1,2,3,4,5\}$

Exemple 8.9 Exemple de grup cíclic amb ordre no primer amb corba el·líptica

La corba el·líptica $E/\mathbb{Z}_{11}: y^2 = x^3+10x+4$ té 15 elements:

$$[\mathcal{O},(0,2),(0,9),(1,2),(1,9),(4,3),(4,8),(5,5),(5,6),(6,4),(6,7),(9,3),(9,8),(10,2),(10,9)]$$

El grup format pels punts de la corba el·líptica amb l'operació suma tal com l'hem definida és un grup cíclic ja que 15 es pot factoritzar en el producte de primers diferents (3 i 5). L'element $(0,2)$ n'és un dels generadors.

Seguint el teorema de Lagrange, l'ordre dels subgrups són els divisors de 15, és a dir, 3 i 5. Com que G és cíclic, sabem que hi ha $\phi(3)=2$ elements d'ordre 3 i $\phi(5)=4$ elements d'ordre 5.

Ordre	Núm. d'elem.	Elements	Subgrup
1	$\phi(1)=1$	$\mathcal{O}$	$\{\mathcal{O}\}$
3	$\phi(3)=2$	(1, 2), (1, 9)	$\{\mathcal{O},(1,2),(1,9)\}$
5	$\phi(5)=4$	(5, 5), (5, 6), (10, 2), (19, 9)	$\{\mathcal{O},(5,5),(5,6),(10,2),(19,9)\}$
15	$\phi(15)=8$	(0, 2), (0, 9), (4, 3), (4, 8), (6, 4), (6, 7), (9, 3), (9, 8)	$E/\mathbb{Z}_{11}: y^2=x^3+10x+4$

És molt habitual que les corbes que es fan servir en criptografia tinguin ordre primer (i, per tant, tots els elements generin tot el grup cíclic). En algunes construccions, però, es triaran corbes que no tinguin aquesta propietat. Donat un punt base G d'ordre n, es defineix el cofactor h d'una corba el·líptica, que mesura la proporció de punts útils de la corba:

Definició 8.4 El cofactor h d'una corba el·líptica $E/\mathbb{Z}_p$ d'ordre $\#E$ per a un punt base $G \in E$ d'ordre n és:

$$h = \frac{\#E}{n}$$

A les corbes amb ordre $\#E$ primer, tenim que $\#E = n$ i, per tant, el cofactor és sempre 1.

Exemple 8.10 Exemple de cofactor diferent d'1

Recuperem la corba el·líptica de l'exemple 8.9 (els subgrups de la corba es troben detallats al propi exemple). Com hem vist, la corba $E/\mathbb{Z}_{11}: y^2=x^3+10x+4$ té ordre 15 i l'element $(1,2)$ té ordre 3.

Per tant, el cofactor h de la corba E per al punt base $G=(1,2)\in E$ és:

$$h = \frac{\#E}{n} = \frac{15}{3} = 5$$

8.4 Corbes el·líptiques per a usos criptogràfics

Com s'ha comentat anteriorment, en criptografia es fan servir corbes el·líptiques sobre cossos finits, i no pas corbes definides sobre els reals. Ara bé, són totes les corbes el·líptiques sobre cossos finits adequades per a usos criptogràfics? La resposta és, de nou, negativa. No totes les corbes el·líptiques permeten construir criptosistemes segurs: algunes d'elles són vulnerables a atacs coneguts. Anomenem corbes criptogràficament fortes a les corbes que són adequades per a usos criptogràfics.

Abans de descriure les propietats de les corbes criptogràficament fortes, detallarem els paràmetres que defineixen els criptosistemes basats en corbes el·líptiques:

Definició 8.5 Els **paràmetres de domini** d'un criptosistema basat en corbes el·líptiques són els paràmetres que determinen la corba el·líptica E i el punt base G:

- Un primer p que especifica el cos finit sobre el qual es defineix la corba.
- Els dos coeficients $a,b \in \mathbb{Z}_p$ que defineixen la corba $E/\mathbb{Z}_p: y^2=x^3+ax+b$.
- Un punt base $G \in E/\mathbb{Z}_p$ que genera el subgrup cíclic sobre el que es construeix el problema del logaritme discret.
- L'ordre n primer del punt base G.
- El cofactor $h = \#E/n$.

Habitualment els paràmetres de domini s'especifiquen com una tupla (p, a, b, G, n, h).

Considerant els atacs que es coneixen actualment, es requereix que els paràmetres de domini de les corbes el·líptiques per a usos criptogràfics compleixin les condicions següents:

1. És necessari que el nombre de punts de la corba $\#E$ sigui divisible per un primer n suficientment gran (en general, com a mínim major que 2^{160}). Això s'aconsegueix habitualment seleccionant corbes amb $\#E$ primer o bé amb un cofactor petit (normalment, el cofactor h és $1, 2, 3$ o 4).
2. Cal assegurar que l'ordre de la corba no coincideixi amb el del cos finit sobre el qual es defineix, és a dir, per a una corba $E/\mathbb{Z}_p$, cal assegurar que $\#E \neq p$.
3. Cal assegurar que n no divideix $p^k - 1$ per a tots els enters k entre 1 i 20.

Ara bé, a partir d'aquestes condicions, com es poden generar corbes el·líptiques per a usos criptogràfics? Una de les alternatives que es fan servir és seleccionar corbes aleatòriament, descartant aquelles corbes que no compleixen els tres requisits especificats al paràgraf anterior. El procediment consisteix doncs en repetir el procés de seleccionar aleatòriament una corba i verificar que aquesta compleixi els requisits fins a trobar-ne una que els compleixi. La selecció aleatòria de la corba proporciona un cert nivell de seguretat, ja que la probabilitat de generar una corba que pertanyi a una classe especial que sigui vulnerable a un atac específic (potencialment encara per descobrir) és molt baixa.

Tanmateix, com que algunes de les condicions que cal que les corbes compleixin per a ser segures per a usos criptogràfics no són trivials de verificar (sobretot per part de desenvolupadors sense coneixements específics en aquest tipus de criptografia) i, a més, el procediment de generar les corbes és computacionalment costós (implica comptar el nombre de punts de la corba, que com ja hem vist, no és trivial), sovint es fan servir corbes estandarditzades que han estat validades per experts i per a les quals el nombre de punts és conegut. Existeixen diferents organismes que han estandarditzat corbes el·líptiques per a usos criptogràfics, com ara el NIST, el consorci alemany Brainpool o Certicom Research. Aquestes corbes han estat suposadament generades per a evitar els atacs coneguts i són suposadament segures per a usos criptogràfics. Però fer ús d'una corba dissenyada per un tercer implica confiar que aquest tercer no l'ha dissenyada malintencionadament i, com es descriu a l'inici d'aquest capítol (Secció 8.1), això històricament no sempre ha estat així. Per aquest motiu, alguns estàndards inclouen corbes seleccionades pseudoaleatòriament utilitzant un algorisme que permet a tercers verificar que realment s'ha seguit aquest algorisme per a generar-les. D'aquesta manera, hom pot verificar que la corba ha estat generada pseudoaleatòriament (cosa que en dificulta la possible inclusió de portes del darrere) i alhora confiar que la corba compleix els requisits de seguretat mínims (doncs ha passat per un escrutini públic estens abans de ser incorporada a l'estàndard).

L'ús de corbes estandarditzades pot simplificar els algorismes de generació de claus (que veurem més endavant en aquest mateix capítol) i també permet en certes ocasions precalcular valors necessaris per a la creació de signatures digitals. Així, es poden aprofitar moments en què el processador no està ocupat per precalcular aquests valors (basats en els paràmetres del domini especificats pels estàndards) i fer-los servir quan es requereixi generar una signatura, reduint així el temps necessari per calcular-la.

8.4.1 Selecció verificablement pseudoaleatòria de corbes

Alguns estàndards recomanen l'ús de corbes el·líptiques que s'han generat pseudoaleatòriament, seguint un procediment que permet a terceres parts comprovar que efectivament s'han generat seguint aquest procediment. En aquesta secció presentarem el procediment de generació de corbes pseudoaleatòries verificable que es descriu a l'estàndard l'ANSI X9.62 i a l'estàndard NIST.FIPS.186-4.

El procediment consta de dos algorismes: l'algorisme de selecció, que selecciona una corba pseudoaleatòria, i l'algorisme de verificació, que permet a un tercer verificar que la corba s'ha triat amb l'algorisme de selecció.

L'algorisme de **selecció verificablement aleatòria** de corbes el·líptiques rep com a entrada un primer p que definirà el cos finit i una funció hash H de mida l bits. A partir d'aquests valors, l'algorisme retorna una corba el·líptica $E/\mathbb{Z}_p : y^2 = x^3 + ax + b$ i una llavor S que s'utilitza en el procés de verificació.

L'algorisme de selecció verificablement aleatòria de corbes el·líptiques segueix els passos següents:

1. Es calcula $t = \lceil \log_2 p \rceil$, $s = \lfloor \frac{t-1}{l} \rfloor$ i $v = t - sl$.
2. Es genera una cadena binària aleatòria S de g bits, amb $g \geq l$.
3. Es calcula $c_0 = H(S)_{[l-v...l]}$, és a dir, c_0 correspon als v últims bits del hash de la llavor S.
4. Es calcula $W_0 = 0 \,||\, c_{0[1...v]}$, és a dir, W_0 és el resultat de fixar el primer bit de c_0 a 0.
5. Per i des de 1 fins a s:
   - Es calcula $s_i = (S + i) \mod 2^g$.
   - Es calcula $W_i = H(s_i)$. A l'hora de calcular el hash, el valor s_i es representa com una cadena binària de g bits.
6. Es calcula $r = W_0 \,||\, W_1 \,||\, \ldots \,||\, W_s$.
7. Si $r = 0$ o bé $4r + 27 = 0 \mod p$, es torna al pas 2.
8. Es trien valors a i $b \in \mathbb{Z}_p$ arbitraris, de manera que com a mínim un d'ells sigui diferent de 0 i tals que $r \cdot b^2 = a^3 \mod p$.
9. L'algorisme retorna els coeficients a i b seleccionats i la llavor S generada al pas 2.

Notació

La notació $X_{[a...b]}$ indica els bits des de la posició a fins a la posició b del valor X. El símbol $||$ expressa la concatenació.

Així, per exemple, per al valor $X = 101100$ tenim que $X_{[0...2]} = 10$, $X_{[3...6]} = 100$ i $0 \,||\, X_{[1...6]} = 001100$.

La sortida de l'algorisme són els coeficients de la corba $E/\mathbb{Z}_p$ seleccionada (amb el valor p especificat a l'entrada) i la cadena binària S que s'ha fet servir com a llavor del procés pseudoaleatori. Aquesta llavor S es farà servir posteriorment en el procés de verificació, i permet assegurar que els coeficients de la corba no s'han dissenyat manualment.

Els paràmetres que defineixen la corba (els coeficients a i b) queden gairebé determinats pel valor r. En concret, per a un valor d'r fixat, hi ha essencialment dos possibles valors a triar per al parell a, b. Com que r es deriva de l'aplicació d'una funció hash, no és computacionalment factible per a un atacant trobar una llavor S que generi uns paràmetres a, b seleccionats manualment per l'atacant.

L'elecció d'a i b

El lector interessat a entendre perquè r determina els valors a i b pot consultar la *Guide to Elliptic Curve Cryptography* de Darrel Hankerson, Alfred Menezes i Scott Vanstone.

És interessant notar que les condicions validades al pas 7 ($r \neq 0$ i $r \neq -\frac{27}{4} \mod p$) permeten assegurar que el discriminant de la corba és diferent de zero, ja que $r = \frac{a^3}{b^2} \mod p$ per construcció (pas 8).

L'algorisme que acabem de descriure selecciona una corba pseudoaleatòria de manera que després es pugui verificar que s'ha seleccionat així. Ara bé, l'algorisme de selecció no té en compte les condicions necessàries per a que la corba sigui segura per a usos criptogràfics. Per tal de generar uns paràmetres de domini segurs, es procedeix a executar l'algorisme anterior i, a continuació, es comprova que la corba generada compleixi les tres condicions que la fan criptogràficament forta (especificades a la secció anterior). Si no les compleix, es torna a executar l'algorisme tantes vegades com calgui, fins a aconseguir seleccionar uns paràmetres adequats.

A més a més de la corba el·líptica E, la generació de paràmetres de domini per a criptografia de corbes el·líptiques requereix d'un punt base G. Per tal de calcular aquest punt, es tria un punt $G' \in E$ arbitrari i es

calcula $G = hG'$, on $h = \#E/n$ (recordeu que n és el primer gran que divideix $\#E$). L'única condició que cal que G satisfaci és que sigui diferent de $\mathcal{O}$ (si no ho és, simplement es torna a repetir el procediment fins a trobar un $G \neq \mathcal{O}$). L'ordre del punt G és n.

Donada una corba E (especificada pels paràmetres a, b i p) i la llavor S generades per l'algorisme de generació de corbes verificablement aleatòries, juntament amb la funció hash H utilitzada, l'algorisme de **verificació de la selecció aleatòria** permet comprovar que una corba ha estat creada seguint l'algorisme anteriorment.

L'algorisme consta dels passos següents:

1. S'executen els passos 1 i de 3 a 6 de l'algorisme de selecció. El pas 2 no és necessari, ja que el valor S que s'utilitza és el que es rep a l'entrada.
2. Es comprova que $r \cdot b^2 = a^3 \mod p$. Si la comprovació és satisfactòria, l'algorisme dona per vàlida la corba. En cas contrari, la verificació falla.

Noteu com l'ús de l'algorisme de selecció verificablement aleatòria en la creació de corbes dificulta la creació de corbes especialment vulnerables, ja que els paràmetres d'aquestes es deriven d'una funció hash i, per tant, no es poden construir deliberadament vulnerables. Ara bé, ningú no impedeix a un possible generador de corbes malintencionat d'executar l'algorisme repetidament fins a trobar una corba que compleixi alguns requisits que li siguin d'interès. És per això que l'ús d'aquest algorisme en la creació de corbes estandarditzades no està totalment lliure de sospita.

8.4.2 Corbes estandarditzades

Les corbes generades i publicades per organismes d'estandardització es coneixen com a corbes estàndard o bé com a corbes amb nom.

La Taula 8.2 anomena les corbes amb nom més conegudes així com l'estàndard que les defineix:

Taula 8.2: Estàndards que inclouen la definició de corbes el·líptiques.

Organisme	Estàndard	Corbes sobre $\mathbb{Z}_p$
NIST	FIPS PUB 186-4: Digital Signature Standard (DSS)	P-192, P-224, P-256, P-384, P-521
Certicom Research	SEC 2: Recommended Elliptic Curve Domain Parameters	secp192k1, secp192r1, secp224k1, secp224r1, secp256k1, secp256r1, secp384r1, secp521r1
Brainpool	ECC Brainpool Standard Curves and Curve Generation	brainpoolP160r1, brainpoolP192r1, brainpoolP224r1, brainpoolP256r1, brainpoolP320r1, brainpoolP384r1, brainpoolP512r1

El NIST, en el seu estàndard per a signatures digitals (*FIPS PUB 186-4: Digital Signature Standard (DSS)*) publicat al juliol de 2013 defineix cinc corbes el·líptiques sobre $\mathbb{Z}_p$, fent servir primers de diferents mides (192, 224, 256, 384 i 521 bits). Les corbes s'anomenen anteposant el prefix *P-* a la mida. Aquest prefix indica que les corbes estan definides sobre cossos finits d'ordre primer. Les corbes han estat generades amb l'algorisme de generació de corbes verificablement aleatòries descrit a la secció anterior, fent servir SHA-1 com a funció hash.

Totes elles tenen en comú que fan servir el coeficient $a = -3$, el que permet optimitzar la suma de punts en una de les representacions habituals. A més, els primers p seleccionats són primers de quasi-Mersenne, que permeten optimitzar la reducció modular. Les cinc corbes també tenen en comú el cofactor, $h = 1$.

Primers de quasi-Mersenne

Els nombres primers de quasi-Mersenne són primers de la forma $2^n - c$, amb $c \ll 2^n$. Per exemple, $2^{192} - (2^{64} + 1)$ és un primer de quasi-Mersenne, ja que $(2^{64} + 1) \ll 2^{192}$.

A tall d'exemple, a continuació es detallen els paràmetres de la corba P-256, una de les 5 corbes sobre $\mathbb{Z}_p$ definides a l'estàndard del NIST. L'especificació de la corba consta dels paràmetres següents: el mòdul primer p, l'ordre primer n de la corba, la llavor S que s'ha utilitzat en l'algorisme de generació de corbes verificablement aleatòries per tal de generar-la, el valor c que correspon al valor r calculat en l'algorisme de generació, el coeficient b de l'equació de la corba (el coeficient a és -3 per a totes elles), i les coordenades x i y del punt base G.

La corba P-256 del NIST queda definida pels paràmetres següents:

$$p = 2^{256} - 2^{224} + 2^{192} + 2^{96} - 1$$

$$n = 115792089210356248762697446949407573529996955224135760342422259061068512044369$$

$$SEED = \texttt{(0x) c49d3608 86e70493 6a6678e1 139d26b7 819f7e90}$$

$$c = \texttt{(0x) 7efba166 2985be94 03cb055c 75d4f7e0 ce8d84a9 c5114abc af317768 0104fa0d}$$

$$b = \texttt{(0x) 5ac635d8 aa3a93e7 b3ebbd55 769886bc 651d06b0 cc53b0f6 3bce3c3e 27d2604b}$$

$$G_x = \texttt{(0x) 6b17d1f2 e12c4247 f8bce6e5 63a440f2 77037d81 2deb33a0 f4a13945 d898c296}$$

$$G_y = \texttt{(0x) 4fe342e2 fe1a7f9b 8ee7eb4a 7c0f9e16 2bce3357 6b315ece cbb64068 37bf51f5}$$

Exemple 8.11 Exemple de verificació de la generació aleatòria d'una corba el·líptica

A continuació procedirem a verificar que la corba del NIST P-256 ha estat efectivament generada pseudoaleatòriament amb l'algorisme descrit anteriorment.

Com que la generació ha fet servir SHA-1 com a funció hash, aleshores $l = 160$.

1. Es calcula $t = \lceil \log_2 p \rceil = 256$, $s = \lfloor \frac{t-1}{l} \rfloor = \lfloor \frac{256-1}{160} \rfloor = 1$ i $v = t - sl = 96$.
2. La llavor S és $\texttt{0xc49d360886e704936a6678e1139d26b7819f7e90}$, amb $g = 160 \geq l$.
3. Es calcula:

$$\begin{aligned} c_0 &= H(S)_{[l-v \ldots l]} = \\ &= \texttt{0x3f07c5eac96ecc0bfefba1662985be9403cb055c}_{[64 \ldots 160]} = \\ &= \texttt{0xfefba1662985be9403cb055c} \end{aligned}$$

4. Es calcula $W_0 = 0 \,||\, c_{0[1 \ldots v]} = \texttt{0x7efba1662985be9403cb055c}$.
5. Per i des de 1 fins a s:

$$s_1 = (S + 1) \mod 2^{160} = \texttt{0xc49d360886e704936a6678e1139d26b7819f7e91}$$

$$\begin{aligned} W_1 = H(s_1) &= H(\texttt{0xc49d360886e704936a6678e1139d26b7819f7e91}) = \\ &= \texttt{0x75d4f7e0ce8d84a9c5114abcaf3177680104fa0d} \end{aligned}$$

6. Es calcula:

$$r = W_0||W_1 = \\ = \texttt{0x7efba1662985be9403cb055c75d4f7e0ce8d84a9c5114abcaf3177680104fa0d}$$

7. Es comprova que $r \cdot b^2 = a^3 \mod p$:

$$\texttt{0x7efb...fa0d} \cdot \texttt{0x5ac6...604b}^2 = (-3)^3 \mod 2^{256} - 2^{224} + 2^{192} + 2^{96} - 1$$

Com que la igualtat es compleix, podem verificar que la corba ha estat generada pseudoaleatòriament seguint l'algorisme descrit.

Fixeu-vos que com que $s = 1$, només s'executa una única iteració del bucle que calcula les W_i. A més, cal tenir en compte que al calcular els hashos (pas 3 i pas 5), cal representar les entrades de la funció hash (els valors S i s_1) com a cadenes binàries.

Exercici 8.8 Verifiqueu que la corba del NIST P-192 ha estat generada pseudoaleatòriament amb l'algorisme verificable de generació de corbes.

La corba P-192 del NIST queda definida pels paràmetres següents:

$$\begin{aligned} p &= 2^{192} - 2^{64} - 1 \\ n &= 6277101735386680763835789423176059013767194773182842284081 \\ SEED &= \texttt{(0x) 3045ae6f c8422f64 ed579528 d38120ea e12196d5} \\ c &= \texttt{(0x) 3099d2bb bfcb2538 542dcd5f b078b6ef 5f3d6fe2 c745de65} \\ b &= \texttt{(0x) 64210519 e59c80e7 0fa7e9ab 72243049 feb8deec c146b9b1} \\ G_x &= \texttt{(0x) 188da80e b03090f6 7cbf20eb 43a18800 f4ff0afd 82ff1012} \\ G_y &= \texttt{(0x) 07192b95 ffc8da78 631011ed 6b24cdd5 73f977a1 1e794811} \end{aligned}$$

Certicom Research, a l'estàndard *SEC 2: Recommended Elliptic Curve Domain Parameters* publicat el gener de 2010 (versió 2.0), descriu vuit corbes el·líptiques sobre $\mathbb{Z}_p$, fent servir primers de les mateixes mides que l'estàndard del NIST (192, 224, 256, 384 i 521 bits). En general, per a cada mida del primer, l'estàndard descriu dos tipus de corbes: una corba generada pseudoaleatòriament de manera verificable i una corba de Koblitz. L'excepció són les dues mides superiors (384 i 521 bits), per a les quals només es proporciona la corba pseudoaleatòria.

Els noms de les corbes del SEC 2 segueixen un patró que permet identificar-ne les característiques fàcilment: els primers tres caràcters són *sec*, que denota *Standards for Efficient Cryptography*; a continuació s'inclou una p, que descriu corbes sobre cossos finits $\mathbb{Z}_p$; després hi ha un número, que descriu la mida del primer utilitzat; a continuació hi ha una lletra, k o r, especificant si es tracta d'una corba aleatòria o de Koblitz; i finalment hi ha un número de seqüència.

Les corbes aleatòries (corbes r) especificades al SEC2 són equivalents a les corbes del NIST de la mateixa mida especificades a la Taula 8.2 (la corba P-192 és equivalent a la secp192r1, la corba P-224 a la secp224r1, etc.). Per tant, les corbes aleatòries que especifica el SEC 2 han estat generades amb l'algorisme verificable de generació de corbes fent servir SHA-1 com a funció hash.

Les corbes de Koblitz sobre cossos finits $\mathbb{Z}_p$ tenen la particularitat d'estar definides per uns paràmetres que

permeten una implementació especialment eficient (el càlcul del doblat d'un punt en aquest tipus de corbes és molt eficient). Les corbes de Koblitz especificades al SEC 2 han estat seleccionades repetint el procés de seleccionar paràmetres eficients fins a aconseguir trobar uns paràmetres que descriguin una corba d'ordre primer i tenen $a = 0$, de manera que l'equació que les defineix és de la forma $E/\mathbb{Z}_p : y^2 = x^3 + b$.

Corbes de Koblitz

És important tenir en compte que el mot *corbes de Koblitz* és també utilitzat (i, de fet, més comunament) per a referir-se a corbes sobre $\mathbb{F}_{2^m}$. A l'estàndard de Certicom, es generalitza el terme i es fa servir per referir-se també a les corbes sobre un cos finit d'ordre primer que tenen una propietat concreta que permet la implementació eficient del càlcul del doblat d'un punt.

La corba *secp256k1*

La corba *secp256k1* és actualment una corba molt coneguda, ja que és la que fa servir la criptomoneda Bitcoin per a les signatures digitals que autoritzen les transaccions. Anteriorment al seu ús a Bitcoin, la corba era poc utilitzada. Els valors especialment petits dels coeficients de la corba semblen indicar que és una corba *nothing-up-my-sleeve*.

La corba *secp256k1* queda definida pels paràmetres següents:

$$
\begin{aligned}
p &= 2^{256} - 2^{32} - 2^9 - 2^8 - 2^7 - 2^6 - 2^4 - 1 \\
n &= \texttt{(0x) FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFE BAAEDCE6 AF48A03B BFD25E8C} \\
&\quad\ \texttt{D0364141} \\
a &= 0 \\
b &= 7 \\
G_x &= \texttt{(0x) 79BE667E F9DCBBAC 55A06295 CE870B07 029BFCDB 2DCE28D9 59F2815B} \\
&\quad\ \texttt{16F81798} \\
G_y &= \texttt{(0x) 483ADA77 26A3C465 5DA4FBFC 0E1108A8 FD17B448 A6855419 9C47D08F} \\
&\quad\ \texttt{FB10D4B8} \\
h &= 1
\end{aligned}
$$

L'estàndard *ECC Brainpool Standard Curves and Curve Generation* publicat el 2005 inclou set corbes el·líptiques de set mides diferents, quatre de les quals coincideixen amb les del NIST i Certicom. Les altres tres mides són 160, 320, i 512 bits (aquesta última substitueix les de 521 del NIST i Certicom). Totes elles han estat generades de manera verificablement pseudoaleatòria, amb un mètode similar al que s'ha explicat anteriorment.

A diferència de les corbes del NIST, les de Brainpool no fan servir primers de quasi-Mersenne, de manera que en general són menys eficients. Aquesta decisió va ser presa per tal d'evitar problemes de patents amb els algorismes de càlculs aritmètics ràpids.

De manera similar als altres estàndards, els noms de les corbes de Brainpool segueixen un patró que en descriu les seves propietats: els primers caràcters són *brainpool*, que denota l'organisme de certificació; a continuació s'inclou una *P*, que descriu corbes sobre cossos finits $\mathbb{Z}_p$; després hi ha un número, que descriu la mida del primer utilitzat; a continuació hi ha una lletra, *r*, que especifica que es tracta d'una corba aleatòria; i finalment hi ha un número de seqüència.

L'informe de 2017 sobre l'ús del TLS de F5 (*The 2017 TLS Telemetry report*) resumeix les observacions sobre més de 20 milions de *hosts* TLS repartits arreu del món. L'informe reporta que un 74% dels intercanvis de clau de Diffie-Hellman sobre corbes el·líptiques que es porten a terme fan servir la corba del NIST P-256, sent clarament per tant la més popular de totes les corbes estandarditzades. Part d'aquesta popularitat ve donada pel fet de ser la corba per defecte a l'OpenSSL en aquell moment i perquè el TLS 1.3 requereix que totes les implementacions suportin aquesta corba per a l'intercanvi de claus de Diffie-Hellman. La segona corba més popular és la *Curve25519*, una corba proposada pel criptògraf Daniel J. Bernstein sobre un primer de 256 bits (el primer $2^{255} - 19$, que li dona nom). Aquesta corba no està coberta per cap patent i

la implementació de referència és codi lliure. El 2017 el NIST va anunciar que inclouria aquesta corba al seu estàndard de signatura digital i, de fet, la propera versió d'aquest, encara en format d'esborrany (*FIPS PUB 186-5 (Draft)*), ja la contempla.

8.4.3 Funcions hash que retornen punts de corbes el·líptiques

Alguns dels esquemes criptogràfics basats en corbes el·líptiques requereixen d'una funció hash que rebi una entrada arbitrària (per exemple, la cadena de caràcters a signar) i retorni un punt d'una corba el·líptica concreta. En aquesta secció descriurem com es poden construir aquestes funcions hash.

Donada una corba $E/\mathbb{Z}_p : y^2 = x^3 + ax + b$ i un punt $G \in E/\mathbb{Z}_p$ d'ordre n, una construcció ingènua d'una funció hash H que rebi com a entrada cadenes arbitràries i retorni punts de la corba $E/\mathbb{Z}_p$ és la següent:

$$H(m) = H'(m) \cdot G = P \in E/\mathbb{Z}_p$$

on H' és una funció hash que rep entrades arbitràries i retorna un valor a $\mathbb{Z}_n$.

És a dir, la funció hash H es construeix multiplicant un enter (resultant d'una altra funció hash) per un punt de la corba, de manera que s'obté un punt de la corba desitjada. Noteu que la funció H' és fàcil de construir. Per exemple, podríem definir H' com el resultat de la funció hash SHA-1 mòdul n (és a dir, $H'(m) = SHA\text{-}1(m) \mod n$).

Tanmateix, aquesta construcció presenta problemes de seguretat a diferents nivells, motiu pel qual no es fa servir a la pràctica. Més endavant (a l'Exercici 9.4) veurem un exemple concret d'aquests problemes de seguretat en l'ús d'aquesta construcció en les signatures BLS.

Una construcció més elaborada per a aconseguir funcions hash que retornin punts de corbes el·líptiques és el mètode d'intentar-i-incrementar (en anglès, es coneix amb el nom de *try-and-increment*). El mètode consisteix a interpretar el missatge m com a la coordenada x del punt de la corba i calcular el valor d'y corresponent, tenint en compte però dos detalls que podrien ser problemàtics. D'una banda, pot ser que un missatge donat m no correspongui a la coordenada x de cap punt de la corba. D'altra banda, donada una coordenada x, ja hem vist que existiran dos possibles valors d'y.

L'**algorisme de triar-i-incrementar** permet convertir un missatge m en un punt d'una corba el·líptica $E/\mathbb{Z}_p : y^2 = x^3 + ax + b$ seguint els passos següents:

1. Inicialitzar el comptador a zero: $c = 0$.
2. Calcular $(x, s) = H''(c||m)$.
3. Calcular $t = x^3 + ax + b$.
4. Si t és un residu quadràtic mòdul p:
   (a) Calcular l'arrel quadrada de t: $y = (-1)^s \cdot t^{1/2} \mod p$.
   (b) Retornar el punt (x, y).

   En cas contrari:
   (a) Incrementar c: $c = c + 1$.
   (b) Tornar al pas 2.

La funció hash auxiliar H'' rep el valor d'un comptador concatenat amb el missatge i retorna dos valors, un valor a $\mathbb{Z}_p$, que correspon a la coordenada x del punt, i un bit s que es farà servir per decidir quin dels dos possibles valors per a la coordenada y se selecciona (en cas que efectivament existeixin). D'aquesta manera, si el primer valor x no correspon a cap punt de la corba, es pot incrementar el comptador i tornar-ho a intentar, esquivant així el primer dels detalls problemàtics que esmentàvem. D'aquest procediment en prové el nom de l'algorisme. Un cop calculat el valor x, calculem el valor y com l'arrel quadrada d'x, triant sempre el valor més petit dels dos possibles (es tria una ordenació qualsevol, ja que no és important per l'algorisme). Després, el bit s es fa servir per decidir si es retorna el valor més petit o es canvia per l'altre valor.

Exemple 8.12 Exemple d'execució de l'algorisme de triar-i-incrementar

Calculem un punt de la corba $E/\mathbb{Z}_{11} : y^2 = x^3 - 5x + 5$ que representa el missatge *CRIPTOGRAFIA*. Per fer-ho, farem servir la següent funció hash auxiliar:

$$H''(m) = (SHA\text{-}1(m) \mod p, \quad SHA\text{-}1(m) \mod 2)$$

i triarem com a resultat de l'arrel quadrada el valor més petit considerat sobre els enters.

1. Inicialitzem el comptador a zero: $c = 0$.
2. Calculem la funció hash auxiliar:

$$\begin{aligned}(x,s) = H''(c||m) &= H''(0\,||\,\texttt{CRIPTOGRAFIA}) = \\ &= (SHA\text{-}1(\texttt{0CRIPTOGRAFIA}) \mod 11, \quad SHA\text{-}1(\texttt{0CRIPTOGRAFIA}) \mod 2) \\ &= (\texttt{0x2f48}\cdots\texttt{8086} \mod 11, \texttt{0x2f48}\cdots\texttt{8086} \mod 2) = \\ &= (8,0)\end{aligned}$$

3. Calculem $t = x^3 + ax + b = 8^3 - 5\cdot 8 + 5 = 477 \mod 11 = 4$.
4. El valor $t = 4$ és un residu quadràtic a $\mathbb{Z}_{11}$ (ja que $2^2 = 4 \mod 11$ i $9^2 = 4 \mod 11$).
   (a) Les dues arrels de 4 a $\mathbb{Z}_{11}$ són 2 i 9. Com que $2 < 9 \in \mathbb{Z}$, aleshores $y = (-1)^s \cdot t^{1/2} \mod p = (-1)^0 \cdot 2 \mod 11 = 2$.
   (b) Es retorna el punt $(8,2)$.

El punt $(8,2)$ pertany, per tant, a la corba E, i es pot fer servir com a representació del missatge *CRIPTOGRAFIA* en qualsevol algorisme criptogràfic que la faci servir en els seus paràmetres de domini.

8.5 El problema del logaritme discret sobre corbes el·líptiques

Utilitzant la multiplicació escalar sobre corbes el·líptiques i de manera anàloga al problema del logaritme discret a $\mathbb{Z}_p$, podem definir ara el problema del logaritme discret sobre una corba el·líptica (ECDLP, de l'anglès, *Eliptic Curve Discrete Logarithm Problem*).

Donada una corba el·líptica $E/\mathbb{Z}_p$ amb ordre $\#E$, un element primitiu $G \in E$ i un altre element $T \in E$, el problema del **logaritme discret sobre corbes el·líptiques** consisteix a trobar un enter $1 \leq i \leq \#E$ tal que:

$$i \cdot G = T$$

Notació

Convé esmentar que la formulació que es fa servir s'allunya una mica de la formulació tradicional del problema ja que s'ha triat la notació additiva per a expressar l'operació de grup entre els punts de la corba el·líptica. En canvi, si s'hagués fet servir la notació multiplicativa, la definició del logaritme discret seria més similar a la definició clàssica. Tot i això, el problema és equivalent.

Exercici 8.9 Calculeu el logaritme discret del punt $T = (0,7)$ en base $G = (8,9)$ per a la corba $E/\mathbb{Z}_{11} : y^2 = x^3 - 5x + 5$. Podeu fer servir el resultat de l'Exercici 8.3 per a resoldre aquesta activitat.

Per quantificar el nivell de dificultat del problema és, per tant, necessari conèixer el número de punts de la corba, $\#E$.

El problema del logaritme discret sobre corbes el·líptiques es pot generalitzar eliminant la restricció que el punt G hagi de ser primitiu. En aquest cas, donat un punt base G d'ordre n primer, cal trobar l'enter $1 \leq i \leq n$ tal que $i \cdot G = T$. El logaritme discret es calcula doncs sobre el subgrup cíclic de E generat per G, al qual el punt T també pertany. L'ordre d'aquest punt, n, passa a tenir importància en la seguretat dels protocols que se'n deriven.

El millor algorisme que es coneix actualment (2021) per solucionar el problema del logaritme discret sobre corbes el·líptiques (excepte per casos molt específics) és l'algorisme ro de Pollard. Aquest algorisme permet calcular el logaritme discret amb un temps d'execució $\mathcal{O}(\sqrt{n})$, on n és l'ordre del punt. D'aquí se'n deriven els nivells de seguretat que ofereixen les corbes el·líptiques per a diferents mides de la clau que s'han detallat a la comparativa de la Taula 8.1.

Rècord de còmput	El rècord de còmput de l'ECDLP amb l'algorisme ro de Pollard és sobre una corba definida sobre un cos primer de 112 bits de l'any 2012.

8.6 Criptografia basada en el problema del logaritme discret sobre corbes

En aquesta secció es descriuen algorismes criptogràfics que es basen en el problema del càlcul del logaritme discret sobre corbes el·líptiques. En primer lloc, veurem la versió de l'intercanvi de claus de Diffie-Hellman sobre corbes. A continuació, presentarem l'algorisme de signatura ECDSA. Finalment, descriurem l'algorisme de xifratge ECIES.

8.6.1 Intercanvi de claus de Diffie-Hellman amb corbes el·líptiques

L'algorisme d'intercanvi de claus de Diffie-Hellman sobre corbes el·líptiques (ECDH, de l'anglès, *Elliptic Curve Diffie-Hellman Key Exchange*) és una variant de l'algorisme d'intercanvi de claus de Diffie-Hellman (que ja hem presentat al Capítol 6) que treballa sobre corbes el·líptiques fent servir les operacions de *suma* i *multiplicació* que hem definit en aquest capítol. De la mateixa manera que la variant clàssica, l'objectiu de l'ECDH és aconseguir que dos usuaris que es comuniquen per un canal insegur derivin una clau compartida.

El procés d'inicialització de l'algorisme consisteix en la selecció dels paràmetres de domini: els elements que defineixen la corba el·líptica E (un primer p i els coeficients a i $b \in \mathbb{Z}_p$) i un element $G = (x_g, y_g) \in E$ d'ordre n.

Una vegada fixats els paràmetres de domini, l'algorisme d'intercanvi de claus s'executa de manera anàloga a la variant clàssica:

L'algorisme d'**intercanvi de claus de Diffie-Hellman sobre corbes el·líptiques** entre dos usuaris, A i B, consta dels passos següents:

1. A tria un valor aleatori $k_{privA} = a \in (1, \ldots, n)$ i calcula $k_{pubA} = a \cdot G = (x_a, y_a)$.
2. B tria un valor aleatori $k_{privB} = b \in (1, \ldots, n)$ i calcula $k_{pubB} = b \cdot G = (x_b, y_b)$.
3. A i B intercanvien els seus valors k_{pub}, és a dir, A envia a B el valor k_{pubA} i B envia a A el valor k_{pubB}.
4. A deriva la clau compartida $k_{AB} = k_{pubB} \cdot k_{privA} = (x_{AB}, y_{AB})$.
5. B deriva la clau compartida $k_{AB} = k_{pubA} \cdot k_{privB} = (x_{AB}, y_{AB})$.

Les claus privades (els valors a i b) són enters, mentre que tant les claus públiques (k_{pubA} i k_{pubB}) com la clau compartida (k_{AB}) són punts de la corba el·líptica. Per tant, a cada pas (a excepció del pas 3) es calcula una multiplicació escalar.

Així doncs, efectivament, el punt k_{AB} derivat per les dues parts participants en el protocol és el mateix, ja que, d'una banda, l'usuari A calcula:

$$k_{AB} = k_{pubB} \cdot k_{privA} = (bG)a$$

i, d'altra banda, l'usuari B calcula:

$$k_{AB} = k_{pubA} \cdot k_{privB} = (aG)b$$

Com que la suma de punts és associativa, les dues parts calculen el mateix valor.

Pel que fa a la seguretat davant d'un atacant que escolti el canal, l'atacant coneixerà els dos valors intercanviats pel canal, k_{pubA} i k_{pubB}, a més dels paràmetres públics que defineixen E (a, b i p) i el punt G, però calcular k_{AB} a partir d'aquests valors és un procés computacionalment difícil.

Exemple 8.13 Exemple d'intercanvi de claus de Diffie-Hellman amb corbes el·líptiques

Els usuaris A i B disposen d'un canal insegur amb el qual comunicar-se, i volen aconseguir crear una clau compartida k_{AB}:

Se seleccionen els paràmetres de domini $a = 3, b = 8, p = 23$ (i, per tant, $E/\mathbb{Z}_{23} : y^2 = x^3 + 3x + 8$) i $G = (19,1) \in E/\mathbb{Z}_{23}$ (de manera que $\#E = n = 29$).

1. A tria el valor aleatori $k_{privA} = a = 15$ i calcula:
   $k_{pubA} = a \cdot G = 15(19,1) = (6,14)$.
2. B tria un valor aleatori $k_{privB} = b = 18$ i calcula:
   $k_{pubB} = b \cdot G = 18(19,1) = (13,17)$.
3. A i B intercanvien els seus valors k_{pub}.
4. A deriva la clau compartida, calculant:
   $k_{AB} = k_{pubB} \cdot k_{privA} = 15(13,17) = (17,2)$.
5. B deriva la clau compartida, calculant:
   $k_{AB} = k_{pubA} \cdot k_{privB} = 18(6,14) = (17,2)$.

El protocol finalitza amb A i B compartint el mateix valor secret $k_{AB} = (17,2)$.

8.6.2 L'esquema de signatura ECDSA

L'ECDSA (per les seves sigles en anglès, *Elliptic Curve Digital Signature Algorithm*) és una variant de l'algorisme de signatura DSA (Digital Signature Algorithm) que es basa en el problema del logaritme discret sobre corbes el·líptiques. L'ECDSA és anàleg al DSA, però operant sobre corbes el·líptiques en comptes de sobre els enters. L'ECDSA va ser proposat l'any 1992 per Scott Vanstone, i des d'aleshores ha estat inclòs en diversos estàndards.

L'algorisme de generació de claus de l'ECDSA coincideix amb els primers passos de l'intercanvi de claus de Diffie-Hellman amb corbes el·líptiques.

En primer lloc, s'executa el procés d'inicialització, en què se seleccionen els paràmetres de domini: la corba $E/\mathbb{Z}_p$ (amb mòdul p i coeficients a i b) i un punt G que genera un grup cíclic d'ordre primer n. Els paràmetres de domini són doncs la tupla (p,a,b,G,n). Si es fan servir corbes estandarditzades, no cal executar el procés d'inicialització, ja que els paràmetres de domini ja venen definits.

Una vegada generats els paràmetres de domini, es procedeix a la generació de claus:

L'algorisme de **generació de claus ECDSA** consta dels passos següents:

1. Es tria un enter aleatori $k_{priv} = d \in_R (1, n)$.
2. Es calcula $k_{pub} = B = d \cdot G$.
3. La clau pública k_{pub} és el punt B, mentre que la clau privada k_{priv} és el valor d.

Les claus generades per l'algorisme anterior es poden fer servir per a generar i validar signatures digitals fent servir els algorismes següents:

A partir d'un missatge en clar m, la clau privada de l'emissor $k_{priv} = d$, i els paràmetres de domini (p, a, b, G, n), es calcula la **signatura digital ECDSA** del missatge:

1. Es tria una clau efímera $k_e \in_R (0, n)$.
2. Es calcula $R = k_e \cdot G$. Com que R és un punt de la corba el·líptica, tenim que $R = (x_R, y_R)$.
3. Es calcula $r = x_R \mod n$. Si $r = 0$, es torna al pas 1.
4. Es calcula $e = H(m)$.
5. Es calcula $s = (e + d \cdot r) \cdot k_e^{-1} \mod n$. Si $s = 0$, es torna al pas 1.
6. La signatura és la tupla (r, s).

L'algorisme de signatura ECDSA requereix de l'ús d'una funció hash, al pas 4, que s'aplica al missatge abans de signar-lo, com és habitual en les signatures digitals. Per a l'algorisme ECDSA la funció hash ha de retornar un enter i, per tant, no és necessari fer servir un esquema que permeti obtenir punts a partir de funcions hash (a diferència d'altres esquemes criptogràfics que veurem més endavant).

En els passos 3 i 5 es fan comprovacions sobre els valors que formen part de la signatura digital, i que asseguren que la signatura resultant de l'execució no contingui zeros. Això permet evitar certs atacs.

Com que els tres primers passos de la signatura no requereixen l'ús del missatge m a signar, si es fan servir corbes estandarditzades (i, per tant, els paràmetres de domini són coneguts anticipadament) es poden precalcular valors r aprofitant moments en què la càrrega del sistema sigui baixa i el processador estigui disponible. Aquests valors poden ser utilitzats després, quan es requereixi fer una signatura digital, reduint el temps de còmput necessari per calcular-la.

La variant de l'algorisme de signatura ECDSA que acabem de descriure és probabilística. Per a un mateix missatge i una mateixa clau privada, existeixen diverses signatures vàlides. Aquesta variabilitat s'introdueix en la generació d'una clau efímera (k_e), que se selecciona aleatòriament i que serà diferent per a cada signatura. De fet, és indispensable per a la seguretat de l'esquema que la clau efímera sigui única, doncs múltiples signatures fetes amb una mateixa clau privada i una mateixa clau efímera rebel·len la clau privada. Quan no es pot assegurar que es disposa d'una font d'aleatorietat prou bona per a complir aquest requisit, s'utilitzen versions deterministes de l'ECDSA, que deriven la clau efímera del missatge a signar.

A partir d'un missatge en clar m, la clau pública k_{pub}, els paràmetres de domini $((p,a,b,G,n))$ i una signatura del missatge (r,s), els passos següents permeten **verificar una signatura ECDSA**:

1. Es verifica que r i s es trobin a l'interval $(0,n)$.
2. Es calcula $e = H(m)$.
3. Es calcula $w = s^{-1} \mod n$.
4. Es calcula $u_1 = w \cdot e \mod n$ i $u_2 = w \cdot r \mod n$.
5. Es calcula $P = u_1 \cdot G + u_2 \cdot B$. Com que P és un punt de la corba el·líptica, tenim que $P = (x_P, y_P)$.
6. Si $x_P = r \mod n$, aleshores la signatura és vàlida i la verificació finalitza correctament. En cas contrari, la signatura es considera invàlida i la verificació fracassa.

Noteu com l'algorisme de verificació de signatures ECDSA és correcte. Si la signatura (r,s) ha estat creada per un signant legítim, aleshores $s = (e + d \cdot r) \cdot k_e^{-1} \mod n$ (pas 5 de l'algorisme de signatura) i, per tant:

$$k_e = s^{-1}(e + d \cdot r) \mod n = s^{-1} \cdot e + s^{-1} \cdot d \cdot r \mod n = w \cdot e + w \cdot d \cdot r \mod n = \\ = u_1 + u_2 \cdot d \mod n$$

Aleshores, tenim que:

$$P = u_1 \cdot G + u_2 \cdot B = u_1 \cdot G + u_2 \cdot d \cdot G = G(u_1 + u_2 \cdot d) = G \cdot k_e$$

i, per tant, $x_P = r$.

Exemple 8.14 Exemple de signatura i verificació amb ECDSA

L'usuari A vol enviar el missatge $m = 567$ signat amb ECDSA a un altre usuari.

En primer lloc, se seleccionen els valors públics $a = -5, b = 5, p = 11$ $(E/\mathbb{Z}_{11} : x^3 - 5x + 5)$ i $G = (2,6) \in E$ (de manera que $\#E = n = 17$).

A continuació, cal que A disposi d'un parell de claus ECDSA. L'algorisme de creació de claus és el mateix que en l'ECDH:

1. A tria el valor aleatori $k_{privA} = d = 5$.
2. A calcula $k_{pub} = B = a \cdot G = 5(2,6) = (8,9)$.
3. La clau pública k_{pub} és el punt $B = (8,9)$, mentre que la clau privada k_{priv} és el valor $d = 5$.

Per tal de signar el missatge $m = 567$, A executa l'algorisme de signatura. Per simplicitat, a l'exemple es farà sevir la funció identitat (que retorna a la sortida el valor que rep a l'entrada) com a funció hash H.

1. A tria una clau efímera $k_e = 10 \in_R (0,17)$.
2. A calcula $R = k_e \cdot G = 10(2,6) = (6,2) = (x_R, y_R)$.
3. A calcula $r = x_R \mod n = 6$. Com que $r \neq 0$, segueix l'execució de l'algorisme al pas següent.
4. A calcula $e = H(m) = 567$.
5. A calcula $s = (e + d \cdot r) \cdot k_e^{-1} \mod n = (567 + 5 \cdot 6) \cdot 10^{-1} \mod 17 = 597 \cdot 12 \mod 17 = 7$. Com que $s \neq 0$, segueix l'execució de l'algorisme al pas següent.
6. La signatura és la tupla $(r,s) = (6,7)$.

A partir del missatge en clar $m = 567$, la clau pública k_{pub}, els paràmetres de domini, i la signatura del missatge $(r,s) = (6,7)$, el receptor del missatge pot verificar-ne la signatura:

1. Verifica que $6 \in (0,17)$ i $7 \in (0,17)$.

2. Calcula $e = H(m) = 567$.
3. Calcula $w = s^{-1} \mod n = 7^{-1} \mod 17 = 5$.
4. Calcula $u_1 = w \cdot e \mod n = 5 \cdot 567 \mod 17 = 13$ i $u_2 = w \cdot r \mod q = 5 \cdot 6 \mod 17 = 13$.
5. Calcula $P = u_1 \cdot G + u_2 \cdot B = 13(2,6) + 13(8,9) = (6,2) = (x_P, y_P)$.
6. Comprova si $x_P = r \mod q$. Com que efectivament $6 = 6 \mod 17$, la signatura és vàlida.

Exercici 8.10 Donada la corba i claus generades per l'algorisme de generació de claus de l'Exemple 8.14 i la mateixa funció hash que en aquest exemple, verifiqueu la validesa de la signatura ECDSA $(10,3)$ per al missatge $m = 876$.

Exercici 8.11 Donada la corba $E/\mathbb{Z}_{23} : y^2 = x^3 + 3x + 8$ i el punt base $G = (13,6)$ d'ordre 29:

1. genereu un parell de claus ECDSA fent servir aquests paràmetres de domini,
2. proporcioneu una signatura amb ECDSA del missatge $m = 25504446$ considerant com a funció hash la funció $H(x) = x \mod 10$, i
3. valideu la signatura que acabeu de generar.

8.6.3 L'esquema de xifratge integrat de corbes el·líptiques (ECIES)

L'esquema de xifratge integrat de corbes el·líptiques (ECIES) (de l'anglès, *Elliptic Curve Integrated Encryption Scheme*) va ser proposat per Mihir Bellare, Michel Abdalla i Phillip Rogaway a finals dels noranta i és l'algorisme de xifratge basat en corbes el·líptiques més estès actualment.

Diferents versions de l'ECIES (similars però no totalment compatibles entre elles) es troben estandarditzades per diversos organismes als estàndards ANSI X9.63, SEC 1, ISO/IEC 15946-3, i IEEE P1363a.

L'ECIES és un algorisme de xifratge híbrid, que fa ús de la criptografia de clau pública per a generar una clau que s'utilitza en un algorisme de xifratge simètric. D'aquesta manera, s'aprofita l'eficiència del xifratge simètric i les propietats que ofereix la criptografia de clau pública.

A grans trets, el funcionament de l'ECIES es basa en generar una clau secreta compartida entre emissor i receptor de manera anàloga a l'algorisme de Diffie-Hellman i a partir d'aquesta clau se'n deriven dues claus simètriques. La primera d'aquestes claus simètriques es fa servir per a xifrar el missatge fent servir un criptosistema de clau simètrica, i la segona es fa servir per a autenticar el text xifrat.

L'ECIES fa servir tres primitives criptogràfiques: un algorisme de xifratge de clau simètrica (tal que $m = D_k(E_k(m))$, una funció de derivació de clau (KDF) i un codi d'autenticació de missatges (MAC).

A partir d'un missatge en clar m, la clau pública $k_{pub} = B$ (generada amb el mateix algorisme de generació de claus que per a l'ECDSA) i els paràmetres de domini (p,a,b,G,n,h), els passos següents permeten **xifrar el missatge amb ECIES** (l'esquema de xifratge integrat de corbes el·líptiques):

1. Es tria aleatòriament una clau efímera $k_e \in_R (0,n)$.
2. Es calcula $R = k_e \cdot G$ i $Z = h \cdot k_e \cdot B$. Si $Z = \mathcal{O}$, es torna al pas 1. Com que Z és un punt de la corba el·líptica, tenim que $Z = (x_Z, y_Z)$.
3. Es calcula $(k_1, k_2) = KDF(x_Z, R)$.
4. Es calcula $C = E_{k_1}(m)$.
5. Es calcula $t = MAC_{k_2}(C)$.
6. Es retornen els valors (R,C,t).

Noteu com del secret compartit Z se'n deriva la clau simètrica k_1, que es fa servir per a xifrar el missatge, i la clau simètrica k_2, que es fa servir per autenticar-lo.

A partir d'un missatge xifrat (R,C,t), la clau privada $k_{priv} = (d)$, i els paràmetres de domini (p,a,b,G,n), els passos següents permeten **desxifrar el missatge amb ECIES** (l'esquema de xifratge integrat de corbes el·líptiques):

1. Es valida que R sigui un punt vàlid de la corba diferent de $\mathscr{O}$.
2. Es calcula $Z = h \cdot d \cdot R$. Si $Z = \mathscr{O}$, es rebutja el text xifrat. Com que Z és un punt de la corba el·líptica, tenim que $Z = (x_Z, y_Z)$.
3. Es calcula $(k_1,k_2) = KDF(x_Z,R)$.
4. Es calcula $t' = MAC_{k_2}(C)$. Si $t \neq t'$, es rebutja el text xifrat.
5. Es calcula $m = D_{k_1}(C)$.
6. Es retorna el text en clar m.

És interessant notar com el punt Z es deriva del secret de Diffie-Hellman compartit entre emissor i receptor (l'emissor calcula $k_e \cdot B$ fent servir la clau pública del receptor B i el receptor calcula $d \cdot R$ fent servir la seva clau privada).

Si el text xifrat (R,C,t) és realment el resultat de xifrar m seguint l'algorisme de xifratge, aleshores:

$$Z = h \cdot d \cdot R = h \cdot d \cdot k_e \cdot G = h \cdot k_e \cdot d \cdot G = h \cdot k_e \cdot B$$

i, per tant, el receptor deriva les mateixes dues claus simètriques k_1 i k_2 que l'emissor ha utilitzat per xifrar, i pot validar i desxifrar el missatge.

8.7 Resum

En aquest capítol s'ha presentat la criptografia de corbes el·líptiques. En primer lloc, s'ha explicat el seu origen i els seus avantatges en relació a la criptografia de clau pública tradicional. A continuació, s'ha descrit com es fan servir les corbes el·líptiques per crear grups d'interès per a la criptografia, s'han presentat els principals estàndards que cobreixen la criptografia de corbes el·líptiques, i s'ha exposat la variant del problema del logaritme discret sobre corbes el·líptiques. Per últim, s'han detallat alguns dels esquemes criptogràfics més populars que fan servir aquest tipus de criptografia: l'intercanvi de claus de Diffie-Hellman sobre corbes el·líptiques, l'esquema de signatura ECDSA, i l'esquema de xifratge ECIES.

8.8 Solucions dels exercicis

Exercici 8.1:

A continuació es proposa una aproximació que permet resoldre el problema plantejat amb els coneixements exposats fins a la secció en què es proposa l'exercici. Més endavant s'expliquen altres alternatives més eficients per al càlcul dels punts d'una corba.

Els punts de la corba seran tots aquells que compleixin l'equació $y^2 = x^3 - 4x + 1$ a $\mathbb{Z}_5$. Per tant, en primer lloc avaluem l'expressió per a tots els possibles valors $x \in \mathbb{Z}_5$:

x	y^2
0	1
1	3
2	1
3	1
4	4

Ara, ens cal saber quins dels valors d'y^2 són residus quadràtics a $\mathbb{Z}_5$. Com que el cos és petit, els podem calcular per força bruta, provant tots els possibles casos:

y	y^2
0	0
1	1
2	4
3	4
4	1

Veiem doncs que els valors 1 i 4 són residus quadràtics, mentre que el 3 no ho és.

Per tant, trobem que els punts de la corba són:

$$[(0,1),(0,4),(2,1),(2,4),(3,1),(3,4),(4,2),(4,3),\mathcal{O}]$$

Exercici 8.2:

Com que $P_1 \neq P_2$, aleshores el pendent m és:

$$m = \frac{y_2 - y_1}{x_2 - x_1} \mod p = \frac{7-1}{4-1} \mod 11 = \frac{6}{3} \mod 11 = 2$$

Després, es calculen les coordenades del punt:

$$x_3 = m^2 - x_1 - x_2 \mod p = 2^2 - 1 - 4 \mod 11 = 10$$

$$y_3 = m(x_1 - x_3) - y_1 \mod p = 2(1-10) - 1 \mod 11 = 3$$

Per tant, el resultat de la suma és $P_3 = (10,3)$. Efectivament, el resultat coincideix amb el càlcul fet gràficament a l'Exemple 8.4.

Exercici 8.3:

En primer lloc, calculem $P + P$. Per fer-ho, es calcula el valor del pendent m:

$$m = \frac{3x_1^2 + a}{2y_1} \mod p$$

$$m = \frac{3 \cdot 8^2 - 5}{2 \cdot 9} \mod 11 = \frac{187}{18} \mod 11 = 0$$

i després, es calculen les coordenades del punt:

$$x_3 = m^2 - x_1 - x_2 \mod p = 0 - 8 - 8 \mod 11 = 6$$

$$y_3 = m(x_1 - x_3) - y_1 \mod p = 0(8-6) - 9 \mod 11 = 2$$

Per tant, $2P = P + P = (6,2)$.

Seguint el mateix procediment, podem calcular la resta de punts:

$P = (8,9)$
$2P = P + P = (6,2)$
$3P = P + P + P = (1,10)$
$4P = (0,4)$
$5P = (7,4)$
$6P = (10,3)$
$7P = (2,6)$
$8P = (4,4)$
$9P = (4,7)$
$10P = (2,5)$
$11P = (10,8)$
$12P = (7,7)$
$13P = (0,7)$
$14P = (1,1)$
$15P = (6,9)$
$16P = (8,2)$
$17P = \mathcal{O}$
$18P = (8,9)$
$19P = \cdots$

Exercici 8.4:

L'invers del punt $P_1 = (x_1, y_1)$ serà el punt P_2 tal que $P_1 + P_2 = \mathcal{O}$.

D'una banda, sabem que aquest punt P_2 és precisament $P_2 = -P_1 = (x_1, -y_1) = (4,-7) = (4,4)$.

Alternativament, si prenem com a referència la solució de l'exercici anterior, veiem que:

$$P_1 = (4,7) = 9P$$

$$9P + 8P = 17P = \mathcal{O}$$

$$P_2 = 8P = (4,4)$$

Exercici 8.5:

Com que 9 no és primer, no tots els elements del grup són generadors. En particular, el punt $(4,5)$ no ho és, ja que no genera tots els 9 elements del grup:

$P = (4,5)$
$2P = P + P = (4,6)$
$3P = P + P + P = \mathcal{O}$

Exercici 8.6:

La representació binària de l'enter 12 és 1100. En primer lloc s'inicialitza `result` a $\mathcal{O}$. A continuació s'executen les iteracions:

Taula 8.3: Execució de l'algorisme de sumar i doblar

Iteració	e	Còmput
1	$e=0$	`point = point + point` $= (7,2)+(7,2) = (18,11)$
2	$e=0$	`point = point + point` $= (18,11)+(18,11) = (22,2)$
3	$e=1$	`result = result + point` $= \mathscr{O}+(22,2) = (22,2)$
		`point = point + point` $= (22,2)+(22,2) = (10,16)$
4	$e=1$	`result = result + point` $= (22,2)+(10,16) = (\mathbf{16},\mathbf{14})$

Per tant, $12P = (16,14)$.

Exercici 8.7:

Substituint els valors al teorema de Hasse tenim que:

$$|\#E - 12| \leq 2\sqrt{11}$$
$$5.36 \leq \#E \leq 18.63$$

Per tant, les corbes el·líptiques definides sobre $\mathbb{Z}_{11}$ tindran entre 6 i 18 punts.

Exercici 8.8:

1. Es calcula $t = \lceil \log_2 p \rceil = 192$, $s = \lfloor \frac{t-1}{l} \rfloor = \lfloor \frac{192-1}{160} \rfloor = 1$ i $v = t - sl = 32$.
2. La llavor S és `0x3045ae6fc8422f64ed579528d38120eae12196d5`, amb $g = 160 \geq l$.
3. Es calcula:

$$\begin{aligned} c_0 &= H(S)_{[l-v\ldots l]} = \\ &= \texttt{0x7f6da10026c7ff92c5ac2e890bd59b44b099d2bb}_{[128\ldots160]} = \\ &= \texttt{0xb099d2bb} \end{aligned}$$

4. Es calcula $W_0 = 0 \,||\, c_{0[1\ldots v]} = \texttt{0x3099d2bb}$.
5. Per i des de 1 fins a s:

$$s_1 = (S+1) \mod 2^{160} = \texttt{0x3045ae6fc8422f64ed579528d38120eae12196d6}$$

$$\begin{aligned} W_1 = H(s_1) &= H(\texttt{0x3045ae6fc8422f64ed579528d38120eae12196d6}) = \\ &= \texttt{0xbfcb2538542dcd5fb078b6ef5f3d6fe2c745de65} \end{aligned}$$

6. Es calcula:

$$\begin{aligned} r &= W_0 || W_1 = \\ &= \texttt{0x3099d2bbbfcb2538542dcd5fb078b6ef5f3d6fe2c745de65} \end{aligned}$$

7. Es comprova que $r \cdot b^2 = a^3 \mod p$:

$$\texttt{0x3099...de65} \cdot \texttt{0x6421...b9b1}^2 = (-3)^3 \mod 2^{192} - 2^{64} - 1$$

Com que la igualtat es compleix, podem verificar que la corba ha estat generada aleatòriament seguint l'algorisme descrit.

Exercici 8.9:

Com que la corba ($E : y^2 = x^3 - 5x + 5$ a $\mathbb{Z}_{11}$) coincideix amb la de l'exercici 8.3 i la base $G = (8,9)$ és precisament el punt P, aleshores podem observar directament de la solució de l'exercici 8.3 que:

$$13P = (0,7) = T$$

i, per tant, el logaritme és 13.

Exercici 8.10:

1. Es verifica que $10 \in (0,17)$ i $3 \in (0,17)$.
2. $e = H(m) = 876$.
3. $w = s^{-1} \mod q = 3^{-1} \mod 17 = 6$.
4. $u_1 = w \cdot e \mod q = 6 \cdot 876 \mod 17 = 3$ i $u_2 = w \cdot r \mod q = 6 \cdot 10 \mod 17 = 9$.
5. $P = u_1 \cdot G + u_2 \cdot B = 3(2,6) + 9(8,9) = (0,7) = (x_P, y_P)$.
6. Es comprova si $x_P = r \mod q$. Com que $0 \neq 10 \mod 17$, la signatura és invàlida.

Exercici 8.11:

1. La generació del parell de claus consta dels passos següents:

1. Es tria el valor aleatori $k_{privA} = d = 6$.
2. Es calcula $B = aG = 6(13,6) = (18,11)$.
3. La clau pública k_{pub} és el punt $B = (18,11)$, mentre que la clau privada k_{priv} és el valor $d = 6$.

2. La signatura del missatge $m = 25504446$ consta dels passos següents:

1. Es tria una clau efímera $k_e = 19 \in_R (0,29)$.
2. Es calcula $R = k_e \cdot G = 19(13,6) = (12,1) = (x_R, y_R)$.
3. Es calcula $r = x_R \mod n = 12$. Com que $r \neq 0$, segueix l'execució de l'algorisme al pas següent.
4. Es calcula $e = H(m) = H(25504446) = 6$.
5. Es calcula $s = (e + d \cdot r)k_e^{-1} \mod n = (6 + 6 \cdot 12)19^{-1} \mod 29 = 20 \cdot 26 \mod 29 = 27$. Com que $s \neq 0$, segueix l'execució de l'algorisme al pas següent.
6. La signatura és la tupla $(r,s) = (12,27)$.

3. La verificació de la signatura $(r,s) = (12,27)$ per al missatge $m = 25504446$ consta dels passos següents:

1. Es verifica que $12 \in (0,29)$ i $27 \in (0,29)$.
2. Es calcula $e = H(25504446) = 6$.
3. Es calcula $w = s^{-1} \mod q = 27^{-1} \mod 29 = 14$.
4. Es calcula $u_1 = w \cdot e \mod q = 14 \cdot 6 \mod 29 = 26$ i $u_2 = w \cdot r \mod q = 14 \cdot 12 \mod 29 = 23$.
5. Es calcula $P = u_1 \cdot G + u_2 \cdot B = 26(13,6) + 23(18,11) = (12,1) = (x_P, y_P)$.
6. Es comprova si $x_P = r \mod q$. Com que efectivament $12 = 12 \mod 29$, la signatura és vàlida.

8.9 Bibliografia

American Bankers Association (1998). *ANSI X9.62: The Elliptic Curve Digital Signature Algorithm (ECDSA)*.

Antonopoulos, Andreas M. (2014). *Mastering Bitcoin: unlocking digital cryptocurrencies*. Capítol 4: Keys, addresses. O'Reilly Media, Inc..

Arjen, Lenstra; Thorsten, Kleinjung; i Thomé, Emmanuel (2013). *Universal security: from bits and mips to pools, lakes - and beyond*. Number Theory and Cryptography, Nov 2013, Darmstadt, Germany, pp.121-124.

Bernstein, Daniel. J.; Lange, Tanja; i Niederhagen, Ruben (2015). *Dual EC: a standardized back door*. The New Codebreakers, 256-281, Springer.

Boneh, Dan; i Victor Shoup (2020). *A graduate course in applied cryptography*.

Hales, Thomas C.(2013). *The NSA back door to NIST*. Notices of the AMS 61.2: 190-192.

Hankerson, Darrel; Menezes, Alfred J.; i Vanstone, Scott (2006). *Guide to elliptic curve cryptography*. Springer Science & Business Media.

Holmes, David (2018). *The 2017 TLS Telemetry Report*. F5.

Kerry, Cameron F.; i Charles Romine (2013). *NIST FIPS PUB 186-4: Digital Signature Standard (DSS)*.

Open University, The (2016). *Further pure mathematics: Group theory*.

Paar, Christof, and Jan Pelzl (2009). *Understanding cryptography: a textbook for students and practitioners*. Springer Science & Business Media.

Perlroth, Nicole (2013). *Government announces steps to restore confidence on encryption standards*. The New York Times.

Schwennesen, Ben; i Hubert Bray (2016). *Elliptic curve cryptography and government backdoors*.

Warburton, David (2019). *The 2019 TLS Telemetry Report*. F5.