Klausuren

1 Klausur

1 Restklassengleichnugssysteme

Bestimme eine natürliche Zahl z die das Resstklassengleichungssystem

- $z = 164 \ (mod \ 267)$
- $z = 205 \ (mod \ 287)$
- $z = 163 \ (mod \ 391)$

```
import si
z = si.chinese_remainder([267,287,391],[164,205,163])
```

$$z = 11.111$$

Wie viele natürliche Zahlen mit höchstens 8 Dezimalstellen sind Lösungen des Resklassengleichungssystemes:

- $z = 1 \ (mod \ 2)$
- $z = 1 \ (mod \ 3)$
- $z = 1 \ (mod \ 5)$
- $z = 1 \ (mod \ 7)$
- $z = 1 \ (mod \ 11)$
- $z = 1 \ (mod \ 13)$
- $z = 1 \ (mod \ 17)$
- $z = 1 \ (mod \ 19)$

$(2, 3, 5, 7, 11, 13, 17, 19) = 2 * 3 * 5 * 7 * 11 * 13 * 17 * 19 = 9,699,690$
$Anzahl \ der \ Lösungen = 99,999,999 // 9,699,690 = 10$

2 RSA

Du verwendest RSA-Signaturen mit einer Schlüssellänge von 2048 Bit.
Aus Sicherheitsgründen soll die Schlüssellänge auf 8192 Bit erhöht werden.
Welche auswirkungen auf die Performance erwartest du?

$\Longrightarrow$ **Länge 2*2** $\Longrightarrow$ ***4**

a)

Die Schlüsselerzeugung wird um den Faktor **16*16**= **256** länger dauern.

b)

Das Signieren wird etwa um den Faktor **8*8= 64** länger dauern.

c)

Das Verifizieren wird etwa um den Faktor **4*4**= **16** länger dauer.

3 Elliptische Kurven - Domainparameter

Du erzeugst für ECDH Key Agreement eigene Domain-Parameter, um dich nicht auf die vom NIST zur Verfügung gestellten Kurven verlassen zu müssen. Du möchtest ein Sicherheitsniveau von mindestens 128 Bit erreichen.

a)

Welche der folgenden Angriffe musst du bei der Wahl der Domain-Parameter berücksichtingen?

- → Pholig-Hellman ←
- → Baby-Step-Giant-Step ←
- Index-Calculus Attacke
- → Wiener-Attacke ←

b)

Besonders relevant sind bei den Domain-Parametern die Primzahl p (über welchem Körper Z_p wird die elliptiche Kurve definiert) und die Primzahl ω (die Ordnung des Punktes auf der Kurve, mit dem gerechnet werden soll). Wie groß müssen diese beiden Zahlen sein, um das gewünschte Sicherheitsniveau von 128 Bit zu erreichen?

$\Longrightarrow$ **Elliptische Kurven**

- p muss mindestens **256 Bit** lang sein
- ω muss mindestens **256 Bit** lang sein

$\Longrightarrow$ **Gruppe**

- p muss mindestens **3072 Bit** lang sein
- ω muss mindestens **256 Bit** lang sein

4 Diffie-Hellman

Alice und Bob führen einen DH-Schlüsselaustausch (gem. Algorithmus 2.7 im Begleitheft) durch, um einen gemeinsamen Schlüssel zu erzeugen. Sie wählen eine Gruppe $(Z_n, +, [0]_n, -)$ mit $n = 76543$ und $g = 34567$ als Domainparameter.

Leider haben die beiden eine Gruppe gewählt, in der das DLP nicht schwierig genug ist.
Du fängst die DH-Nachrichten $A = 2390$ von Alice und $B = 37674$ von Bob an Alice ab. Berechne damit den Schlüssel, den die beiden vereinbart haben.

$k = \alpha * B$
$B \, mod(n) = b * g$
$b = B * 34567^{-1}{}_n$
$b = (si.inversemod(g, n) * B) \, mod(n)$
$b = 54321$

$check = (54321 * 34567) \, mod(n) = 37674$

$k = (b \ * \ A) \, mod \, (n)$
$k = 10262$

5 Diskrete Logarithmen

Die elliptische Kurve $\epsilon : y^2 = x^3 + x + 4 \; mod(32771)$ hat die Ordnung 32492. Du kennst drei Punkte:

- $P = (20624, 19557)$
- $Q = (11794, 3282)$
- $R = (0, 2)$

auf der elliptischen Kurve. In dieser Aufgabe zeigst du, dass man diskrete Logarithmen zu einer beliebigen Basis berechnen kann, wenn man diskrete Logarithmen wenigstens zu einer Basis berechnen kann.

Der diskrete Logarithmus von P zur Basis $G = (0, 2)$ auf der elliptischen Kurve ist 13421.
Der diskrete Logarithmus von Q zur Basis $G = (0, 2)$ auf der elliptischen Kurve ist 599.

Berechne den diskreten Logartihmus von P zur Basis Q auf der elliptischen Kurve.

$P = G^{13421}$ $\qquad$ $Q = G^{599}$
$P = Q^b$ $\qquad$ $P = (G^{599})^b$

$$P = Q^{20531}$$

$$599 * b \; mod \; 32492 = 13421$$

$$G^{599 * \alpha} = G^{13421}$$

$$b = (si.inverse\_mod(599, 32492) * 13421$$

$$b = 29531 \; mod \; \alpha$$

6 Merkle Trees

Du erzeugst einen Merkle Tree für 65536 Datensätze. Als Hashfunktion verwendest du SHA-256.

a)

Wie lang ist die Merkle Root für deine Datensätze (in Bit) in diesem Fall?
→ **= 256 Bit**

b)

Du Möchtest die Integrität eines Datensatzes überprüfbar machen. Aus wie vielen Hashwerten besteht der Authentication Path in diesem Fall?
→ **log2(65536)=16 Hashwerte**

```
import math
math.log2(65536)
```

c)

Die Integrität des von dir gesendeten Datensatzes (samt Authentication Path) soll überprüft werden. Wie viele SHA-256-Hashwerte müssen dafür berechnet werden?
→ **Knoten + Wurzel = 16 + 1 = 17**

7 Endliche Körper

Berechne im endlichen Körper $\mathbb{Z}_2[x]/(x^3+x^2+1)$.

```
import finitefield as ff

f = ff.FiniteField(2,[1,0,1])
a = FiniteFieldElt(f,[1,0,1])
b = FiniteFieldElt(f,[1,1])
```

a)

$(\alpha^2+1)(\alpha+1)$

```
print(a*b)
```

b)

$\frac{\alpha^2+1}{\alpha+1}$

```
print(a*b.inv())
```

c)

$\alpha^{1001}=1$

Klausur 2. Termin

1)

Für den DSA-Signaturalgorithmus werden Domainparameter benötigt. Dies sind: eine große Pirmzahl p und eine weitere (viel kleinere) Primzahl q, welche $p-1$ teilt.
Schließlich wird ein Element g der Ordnung q in der Gruppe $\mathbb{Z}_p^**$ benötigt.

Angenommen, eine 15360 Bit lange Primzahl p und eine geeignete, 512 Bit lange Primzahl q sind bereits gefunden.

1. Berschreibe wie dann effizient das Element g gefunden werden kann.
    - Berechne h = (p-1)/q. Da q ein Teiler von p-1 ist, ist h eine ganze Zahl.
    - Wähle ein zufälliges Element a aus der Menge {2, 3, ..., p-2}.
    - Berechne g = a^h mod p.
    - Überprüfe, ob g != 1. Wenn g == 1, wiederhole Schritte 2-4 mit einem anderen zufälligen Element a.
2. Begründe auch warum das das vorgeschlagene Vorgehen effizient ist.

- Da g = a^h mod p berechnet wird, hat g die Ordnung q in der Gruppe Z_p^*. Wenn g != 1 ist, haben wir ein geeignetes Element g der Ordnung q gefunden.
- Das vorgeschlagene Vorgehen ist effizient, weil:
  1) Die Berechnung von h = (p-1)/q ist eine einfache Division, die schnell durchgeführt werden kann.
  2) Die Wahl eines zufälligen Elements a ist schnell und einfach.
  3) Die Berechnung von g = a^h mod p kann effizient durch exponentielle Potenzierung (wie beispielsweise Square-and-

Multiply-Algorithmus) durchgeführt werden. Da h deutlich kleiner ist als p (15360 Bit gegenüber 512 Bit), ist diese Berechnung relativ schnell.
4) Die Überprüfung von g != 1 ist eine einfache Vergleichsoperation.

2)

Ein reicher Mann war schwer erkrankt und so wollte er sein Vermögen unter seinen Freunden aufteilen. Er besaß so viele Goldstücke, dass er sie ganz gerecht unter den 15 Freunden aufteilen konnte. Als nun einer seiner Freunde überraschend schon vor ihm starb, war dies nicht mehr möglich, beim Aufteilen wären 2 Goldstücke übrig geblieben. Es wurde noch schlimmer, ein weiterer der Freunde starb, und nun blieben beim gerechten Verteilen gar 5 Goldstücke übrig. Schließlich durfte sich der treue Diener des Mannes über die 5 Goldstücke freuen, als das Vermögen unter den 13 Freunden gerecht aufgeteilt wurde.

1. Wie viele Goldstücke erhielöt jeder der 13 noch lebenden Freunde des Mannes noch mindestens?

```
import si
a = si.chinese_remainder([15,14,13],[0,2,5])
print(a/13)
```

2. Gib eine zweite mögliche Lösung an.
   Bruteforce:

```
i=1
while True:
        if i % 15 == 0 and i % 14 == 2 and i % 15 == 5:
                print(i)
        else:
                i += 1
```

3)

Du verwendest RSA-Signaturen mit einer Schlüssellänge von 2048 Bit. Aus Sicherheitsgründen soll die Schlüssellänge auf 3072 Bit erhöht werden. Welche Auswirkungen auf die Performance erwartest du?

* 5.0625
(1.5^4)

1. Die Schlüsselerzeugung wird etwa um den Faktor **16/2=8** länger dauern.
2. Das Signieren wird etwa um den Faktor **4/2=8** länger dauern.
3. Das Verifizieren einer Signatur wird etwa um den Faktor **4/2=2** länger dauern.

5)

Du weißt, wie man die Ordnung eines Punktes auf einer elliptischen Kurve bestimmen kann, wenn man ie Ordnung der Kurve kennt. In dieser Aufagbe sollst du den umgekehrten Weg gehen.
Du kennst die Ordnung $\omega = 1073741827$ eines Punktes auf einer elliptischen Kurve

$$x^2 = x^3 + ax + b \; mod \; 8589934621$$

Bestimme die Ordnung der elliptischen Kurve.

```python
import math

def hasse_interval(q):
    lower_bound = q + 1 - 2 * math.sqrt(q)
    upper_bound = q + 1 + 2 * math.sqrt(q)
    return lower_bound, upper_bound

def find_curve_order(point_order, field_size):
    lower_bound, upper_bound = hasse_interval(field_size)

    possible_orders = []
    for i in range(int(lower_bound), int(upper_bound) + 1):
        if i % point_order == 0:
            possible_orders.append(i)
    return possible_orders

point_order = 1073741827
field_size = 8589934621
curve_orders = find_curve_order(point_order, field_size)
print("Possible orders of the elliptic curve:", curve_orders)
```

6)

Die elliptische Kurve $\epsilon : y^2 = x^3 + x + 4 \; mod \; 32771$ hat eine Ordnung 32492. Du kennst 2 Punkte

- $G = (0, 2)$
- $Q = (11794, 3282)$
  auf einer ellptischen Kurve.
  Der diskrete Logarithmus von Q zur Basis $G = (0, 2)$ auf einer elliptischen Kurve ist 599.

1. Berechne den diskreten Logarithmus von $17 * Q$ zur Basis G.
2. 2. Berechne den diskreten Logarithmus von G zur Basis Q.
   (Hinweis: Für diese Aufgabe sind keine Punktadditionen und keine Punktmultiplikationen auf der elliptischen Kurve erforderlich)

```
import si
ec = si.EC(1,4,32771) # curve ax + b
g = si.Point(ec, (0,2)) # here G(0,2)
q = si.Point(ec, (11794,3282)) # here Q(11794,3282)

# Teil 1
k = 599 * 17 % 32492 # val to calc
if g.mult(k) == q.mult(17):
    # proof that it works
    print(f"True, k= {k}")
else:
    print("False")

# Teil 2
x = si.inverse_mod(k,32492)
if g == q.mult(17*x):
    print("True")
```

```
else:
    print("False")
```

7)

Die 256 Bit lange Nachricht
0x 101102a1 101102a1 101102a1 101102a1 101102a1 101102a1 101102a1 101102a1
soll mit dem WOTS-Verfahren signiert werden.

$0x\ 101102a1 \implies biniär = 0001\ 0000\ 0001\ 0001\ 0000\ 0010\ 1010\ 0001$

1. Wie viele Hashwerte müssen für das Signieren berechnet werden?

   $$64 * 15 - (m_0, m_1, m_2, \dots)$$

   Where m je 4 Bit Werte (siehe b) $\implies 16 * 8 = 128$

   $$64 * 15 - (sum(m)) \implies 64 * 15 - 128 = 835\ Hashwerte$$

2. Wie viele Hashwerte müssen für das verifizieren der Signatur berechnet werden?
   $0001 = 1 \implies 15 - 1 = 14\ Hashwerte$
   $0000 = 0 \implies 15 - 0 = 15\ Hashwerte$
   $0001 = 1 \implies 15 - 1 = 14\ Hashwerte$
   $0001 = 1 \implies 15 - 1 = 14\ Hashwerte$
   $0000 = 0 \implies 15 - 0 = 15\ Hashwerte$
   $0010 = 2 \implies 15 - 2 = 13\ Hashwerte$
   $1010 = 10 \implies 15 - 10 = 5\ Hashwerte$
   $0001 = 1 \implies 15 - 1 = 14\ Hashwerte$

   $14 + 15 + 14 + 15 + 13 + 5 + 14 = 90 * 8 = 720\ Hashwerte$

Altklausuren

Probeklausur 11/12

1)

(5 Punkte) Alice und Bob führen einen DH-Schlüsselaustausch mit folgenden Parametern durch: Sie benutzen die elliptische Kurve ϵ über $\mathbb{Z}_7$ mit der Gleichung

$$y^2 = x^3 + 2x + 1$$

mit dem Punkt $G = (0, 1)$. Welche Nachrichten senden die beiden, wenn Alice im Pro tokoll $a = 4$ und Bob $b = 5$ wählt? Was ist der Schlüssel, den die beiden vereinbaren?

$\Longrightarrow$ **Code von FTK.ipynb → Thx Joe**

```
# bob key: 18; alice key: 8; curve: y^2 = x^3 + 13x + 13 in
Z_23; Punkt G: (1,2) - omega: 29

curve = si.EC( 13, 13, 23 )
G = si.Point(curve, (1,2))
alpha = 8
beta = 18

# public keys
A = G.mult(alpha)
B = G.mult(beta)
print(f"A = {A}\nB = {B}")

# shared key
K = B.mult(alpha)
print(f"Shared Key B^alpha: {K}")
K = A.mult(beta)
print(f"Shared Key A^Beta:  {K}")
```

2)

(5 Punkte) Sie haben einen 1024-Bit-DSA-Schlüssel.
(a) Wie viele Bits lang ist der private key?
$\rightarrow 160\ Bit$
(b) Wie viele Bits lang ist der public key?
$\rightarrow 160\ Bit$
(c) Wie viele Bits lang ist eine Signatur? (ohne die Nachricht)
$\rightarrow 320\ Bit$

3)

(5 Punkte) Der diskrete Logarithmus von 919 zur Basis 19 modulo 1223 ist 12.
Der diskrete Logarithmus von 395 zur Basis 919 modulo 1223 ist 19.
Bestimmen Sie den diskreten Logarithmus von 395 zur Basis 19 modulo 1223.
(Hinweis: 1223 ist eine Primzahl.)

4)

(5 Punkte) Es sei $p = x^3 + x + 1$. Berechnen Sie im endlichen Körper $Z_2[x]/p$
$[x^2 + x]^4{}_p * [x^2 + x + 1]_p$

Klausur 2014

1)

Berechnen Sie den größten ggT t der Polynome

$$p = x^9 + x^8 + x^6 + x + 1$$

$$q = x^9 + x^6 + x^4 + x^2 + 1$$

über dem Körper Z_2. Finden Sie die Polynome $a, b\ ElemZ_2[x]$, so dass gilt $ap + bq = t$.

2)

Ein RSA-Modul ist balanciert, wenn seine beiden Primfaktoren die gleiche Bitlänge haben. Wie viele Primzahlen mit 512 Bitlänge gibt es mindestens? Wie viele blancierte RSA-Moduln mit genau 1024 Bit länge gibt es mindestens? Verwenden Sie den Satz von Hadamard und de la Vallee Poussin. Wie viel Prozent der (genau) 1024 Bit großen Zahlen sind mindestens balancierte RSA-Moduln?

Satz 3.9 (Hadamard (1896) und de la Vallée Poussin (1896)). *Es sei $\pi(x)$ die Anzahl der Primzahlen, die kleiner sind als $x \in \mathbb{N}$. Dann gilt für $x \geq 17$*

$$\frac{x}{\ln x} < \pi(x) < 1.25506 \cdot \frac{x}{\ln x}.$$

3)

Der Diskrete Logarithmus von 28131 zur Basis 9988 modulo 55579 ist 10001, der diskrete Logarithmus von 24031 zur Basis 28131 modulo 55579 ist 8023. Berechnen Sie daraus den diskreten Logarithmus von 9988 zur Basis 24031 modulo 55579.

$\varphi(55579) = 55578$
$28131_{55579} \equiv 9988^{10001}$

$$24031_{55579} \equiv 28131^{8023}$$
$$9988_{55579} \equiv 24031^{x}$$

$$9988_{55579} \equiv 28131^{8023^{x}}$$
$$9988_{55579} \equiv 9988^{10001^{8023^{x}}}$$
$$9988_{55579} \equiv 9988^{80238023*x}$$

$$9988_{55579} \equiv 9988^{80238023*x \ mod \ 55578}$$
$$9988_{55579} \equiv 9988^{38969*x}$$

```
import si
si.extended_gcd(si.euler_phi(55579),38969, verbose=1)
```

$$\Downarrow$$

$$x = 19763$$

$$9988^{1}_{55579} \equiv 9988^{(38969*19763) \ mod \ 55578}$$
$$9988_{55579} \equiv 9988^{1}$$

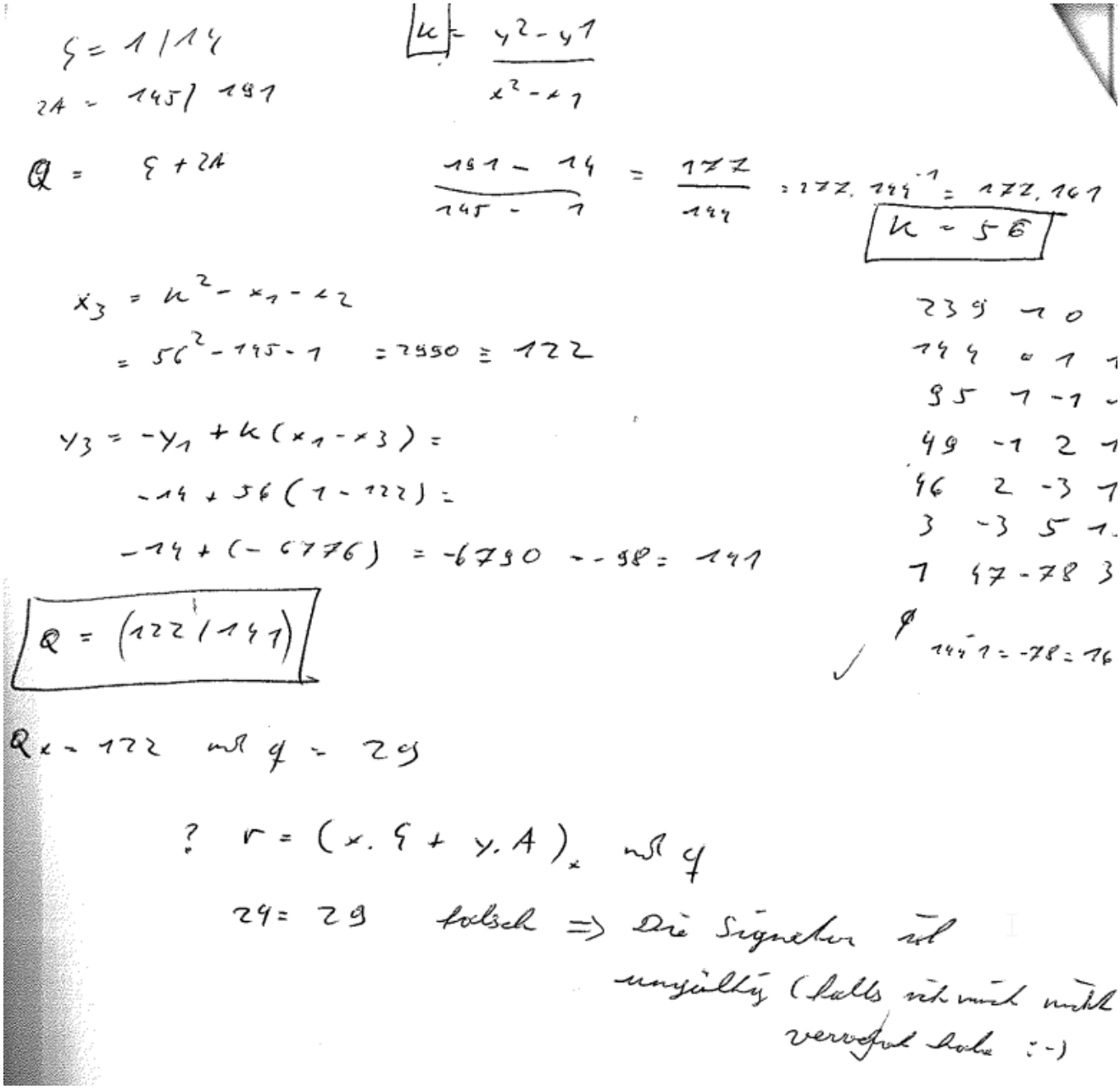
4)

Der Punkt $P = (244, 203)$ auf der Kurve $y^2 = x^3 + x + 243$ über $\mathbb{Z}_{257}$ hat die Ordnung 118. Bestimmen Sie damit unter Zuhilfename Ihres Wissens über die Ordnung von Kurven und Punkten und des Satzes von Hasse die Ordnung der Kurve exakt.

```
import math

def hasse_interval(q):
    lower_bound = q + 1 - 2 * math.sqrt(q)
    upper_bound = q + 1 + 2 * math.sqrt(q)
    return lower_bound, upper_bound
(hasse_interval(257))
```

$1188 * 2 = 236 \implies Ordnung\ der\ Kurve$



5)

Eine Nachricht wurde mit ECDSA signiert (in diesem Beispiel werden Parameter gewählt, die zu klein sind, um wirklich verwendet werden zu dürfen (ignorieren Sie an dieser Stelle diese Tatsache). Als Parameter wurden die Kurve $\epsilon : y^2 = x^3 + x + 194$ über Z_{239} und der Punkt G = (1,14) auf der elliptischen Kurve mit der Ordnung 31 verwendet. Verifizieren Sie die ECDSA-Signatur (24,12) einer Nachricht mit Hashwert 12 mit dem Public Key (ϵ, $A = (182, 226), G = 1, 14)$).

$q = (1, 14)$

$E = y^2 + x^3 + x + 194$

$\mathbb{Z}_{239}$

$q = 31$

$r = R_4$

$s = 12$

$A = (182, 226)$

$SHA1(m) = 12$

a) $1 \leq r \leq q$ o.k.

$1 \leq s \leq q$ o.k.

$w = s^{-1} \bmod q$

$x = w \cdot (SHA1(m)) \bmod q$

$y = w \cdot r \bmod q$

$? \; r = (x \cdot G + y \cdot A)_x \bmod q$

31	1	0	
12	0	1	2
7	1	-2	1
5	-1	3	1
2	2	-5	2
1	-5	13	2

$w = 13$ ✓

1. $G = G$

2. A: $K = \frac{3x_1^2 + a}{2y_1} = \frac{3 \cdot 182^2 + 1}{2 \cdot 226} = \frac{99373}{452} = \frac{189}{213}$

$2a = (145 / 191)$

$= 189 \cdot 213^{-1} = 189 \cdot (-17) = 189 \cdot 227 = 42876$

$K = 134$

$x = 13 \cdot 12 \bmod q$

$x = 1$ ✓

$y = 13 \cdot 24 \bmod q$

$y = 2$ ✓

$x_3 = k^2 - 2x_1 = 134^2 - 2 \cdot 182 = 17532 \equiv 145$

$y_3 = -y_1 + k(x_1 - x_3) = -226 + 134(182 - 145) = 4732 \equiv 191$

239	1	0	
219	0	1	1
20	1	-1	10
19	-10	11	1
1	11	-12	19

Q

Hilfreiche Formeln:

$(x_1, y_1) + (x_2, y_2)$	$(x_1, y_1) + (x_1, y_1)$
$k = \frac{y_1 - y_2}{x_1 - x_2}$	$k = \frac{3x_1^2 + a}{2y_1}$
$x_3 = k^2 - x_1 - x_2$	$x_3 = k^2 - 2x_1$
$y_3 = -y_1 + k(x_1 - x_3)$	$y_3 = -y_1 + k(x_1 - x_3)$

Krypto Prüfung

1)

Welche der folgenden Zahlen ist/sind nach dem Satz von Hasse und dem Satz von Lagrange sicher nicht die Ordnung eines Punktes auf der Kurve $y^2 = x^3 + 157x + 211$ über $Z_{1235467}$?

- 61751
- 123502
- $\rightarrow 125013 \leftarrow$
- $\rightarrow 17859 \leftarrow$

2)

Alice und Bob signieren mit ECDSA und den folgenden Domain-Parametern:
Sie benutzen die elliptische Kurve ϵ über $\mathbb{Z}_{29}$ mit der Gleichung $y^2 = x^3 + 4x + 20$ mit dem Punkt $G = (1, 5)$ der Ordnung 37

Zwei Nachrichten m_1 und m_2 mit den Hashwerten $h(m_1) = 25$ und $h(m_1) = 17$ wurden mit dem gleichen Schlüssel ECDSA-signiert. Die Signaturen lauten:

$$(r_1, s_1) = (24, 36)$$

$$(r_2, s_2) = (24, 19)$$

Berechnen Sie den Private Key.

3)

Es sei $p = x^9 + x^7 + x^4 + x^2 + 1$ Element $Z_2[x]$. Berechnen Sie im endlichen Körper $Z_2[x]/p$.

$$\frac{[x^6+1]_p}{[x^8+x^6+x+1]_p}$$

```
import finitefield as ff
# alles Seitenverkehrt
# x^6 + 1 => 1+0x+0x^2+0x^3+0x^4+0x^5+x^6 => [1,0,0,0,0,0,1]
f = ff.FiniteField(2,[1,0,1,0,1,0,0,1,0,1]) #p
a = ff.FiniteFieldElt(f,[1,0,0,0,0,0,1]) # Zähler | frac oben
b = ff.FiniteFieldElt(f,[1,1,0,0,0,0,1,0,1]) # Nenner | frac
unten

print(f)
print(a)
print(b)
print(a*b.inv())
```

4)

Der diskrete Logarithmus von919 zur Basis 19 modulo 1223 ist 12.
Der diskrete Logarithmus von 395 zur Basis 919 modulo 1223 ist 19.
Bestimmen Sie den diskreten Logarithmus von 395 zur Basis 19 modulo 1223.
(1223 ist eine Primzahl)

$919^{19}\ mod\ 1223 = 395$
$19^{12}\ mod\ 1223 = 919$
$19^{28}\ mod\ 1123 = 395$
$19 * 12 = 228$

6)

Skizzieren Sie static-ephemeral DH

- Gibt es Public Keys? Wenn ja, wie sehen diese aus?
- Welche Nachrichten werden bei der Erstellung des Session Keys gesendet?

- Was wird durch static-ephemeral DH erreicht?

AKY3 Klausur

1)

Sie haben einen DSA-Public Key (p, q, g, A) bestehend aus einer L Bit langen Primzahl p, einer N Bit langen Primzahl q, einem Element g aus ${Z^*}_p$ mit der Ordnung $q$und einem weiterem Element A aus ${Z^*}_p$. Der dazugehörige Private Key ist (p, q, g, a), wobei

$$A = g^{\alpha} \; mod \, p$$

1. Eine Empfehlung im DSA Standard ost das L = 2048, N = 256.
    1. Wie viele Bits lang ist der geheime Exponent a?
        1. 256???
    2. Wie viele Bits lang ist die Signatur? (ohne nachricht)
        1. 512 Bits
2. Eine andere Empfehlung ist L = 3072, N = 256. Die Erstellung einer Signatur mit L = 3072 dauert länger als L = 2048. Um wie viel Prozent länger?
    1. → 125% ←
    2. 50%
    3. 225%
    4. 150%

3)

Sie verwenden das Elliptic Curve Integrated Encryption Scheme (ECTES) zur Verschlüsselung. Sie verwenden eine Kurve über $\mathbb{Z}_p$, wobei p eine 512 Bit lange Primzahl ist. Schließlich verwenden Sie einen Punkt G mit Cofaktor 1.

1. Wie lange (in Bits) sind die Private Keys und Public Keys in diesem Fall ungefähr?

    1. ECTES Sicherheit $256Bit$
2. Weiterhin benötigen Sie eine Blockchiffre und einen sicheren MAC für das Verfahren. Welche Verfahren (mit welcher Schlüssellänge, in welchenm Mode of Operation) könnten Sie verwenden, die dem Sicherheitsniveau der gewählten elliptischen Kurve in etwa entsprechen? (Es reicht, jeweils ein Verfahren anzugeben)
3. Sie verschlüsseln nun eine 30 Byte lange Nachricht mit ECTES unter Verwendung der in Teil 2) genannten Verfahren. Wie lang in Bytes ist das Ergebnis?

4)

Alice und Bob führen einen static-ephemeral DH-Schlüsselaistausch mit folgenden Parametern durch: Sie benutzen die elliptische Kurve ε über $\mathbb{Z}_{47}$ mit der Gleichung

$$y^2 = x^3 + 2x - 9$$

mit dem Punkt G = (11,13). Alices static Public Key ist A = (36,1).

1. Berechnen sie den Schlüssel, den Bob erhält, wenn er als ephemeral Private Key b = 3 wählt.
2. Alices static Public Key ist $a = 12$. Berechnen Sie daraus und aus dem Ergebnis aus Teil 1) die Ordnung von G.
3. Welche Zahlen kommen nach dem Satz von Hasse als Ordnung von ε in Frage?
4. Bestimmen sie aus den Antworten aus den Teilen 2) und 3) die Ordnung der elliptischen Kurve

AKY WS12/13 3 Termin

1)

- Ist der diskrete Logarithmus von 18 zur Basis 13 in $\mathbb{Z}_{31}^*$
  - 14 oder 16 oder 20?
- Erklären Sie den Unterschied zwischen Private Key und Public Key bei asymmetrischer Verschlüsselung. Wer hat welchen Schlüssel, welcher Schlüsselwird wozu verwendet?
- Erklären Sie das Diffie-Hellman Problem. Nennen Sie ein kryptografisches Verfahren, dessen Sicherheit auf der Schwierigkeit des Diffie-Hellman-Problems beruht.

3)

Alice und Bob signieren mit ECDSA und den Domain-Parametern. Sie bnutzen die elliptische Kurve ε über $mathbbZ_{13}$ mit der Gleichung:

$$y^2 = x^3 - 4x + 4$$

mit dem Punkt G = (0,2) der Ordnung 7. Berechnen Sie eine ECDSA-Signatur für eine Nachricht m mit dem Hashwert $h(m) = 2$.
(Berechner zuerst den s-Teil der Signatur dann den r-Teil)

4)

Kreuze die richtigen Aussagen an.

256 Bit Schlüssellänge gilt für ECDSA als ausreichend sicher.	ja ○	nein ○
DSA-Signaturen (Schlüssellänge 1024 Bit) sind länger als RSA-Signaturen (Schlüssellänge 1024 Bit).	ja ○	nein ○
256 Bit lange Primzahlen sind für sichere ECDSA-Schlüssel ausreichend.	ja ○	nein ○
Baby-Step-Giant-Step ist ein Verfahren zur Berechnung diskreter Logarithmen.	ja ○	nein ○
RSA-2048, DSA (Schlüssellänge 2048 Bit) sind sichere Signaturverfahren	ja ○	nein ○

AKY WS 14/15

1)

Alice und Bob führen einen DH-Schlüsselaustausch mit folgenden Parametern durch. Sie benutzen die elliptische Kurve ε über $\mathbb{Z}_7$ mit der Gleichung:

$$y^2 = x^3 + 2x + 1$$

mit dem Punkt G = (0,1). Welche Nachrichten senden die beiden, wenn Alice im Protokoll a = 4 und Bob b = 5 wählt? Was ist der Schlüssel, den die beiden vereinbaren?

Sie senden sich gegenseitig A(= G *a) und B(*= *G* b)
Schlüssel ist dann B^{α} und A^{β}

$$2G = (0, 1)$$

$$k = \frac{3x_1^2 + a}{2y_1} \implies \frac{3 * 0 + 2}{2} \implies \frac{2}{2} \implies 1$$

$$x_3 = 1^2 - 2 * 0 \implies 1$$

$$y_3 = -1 + 1 * (0 - 1) \implies -2 \implies 5 \; mod \; 7$$

$$2G = (1, 5)$$

..
..
..
..

2)

Sie haben einen DSA-Public-Key (p,q,g,A) bestehend aus einer 2048 Bit langen Primzahl p, einer 256 Bit langenPrimzahl q, einem Element

g aus $\mathbb{Z}_p^*$. Der dazugehörige Private Key ist (p,q,g,a), wobei $A = g^{\alpha}\ mod\ p$.

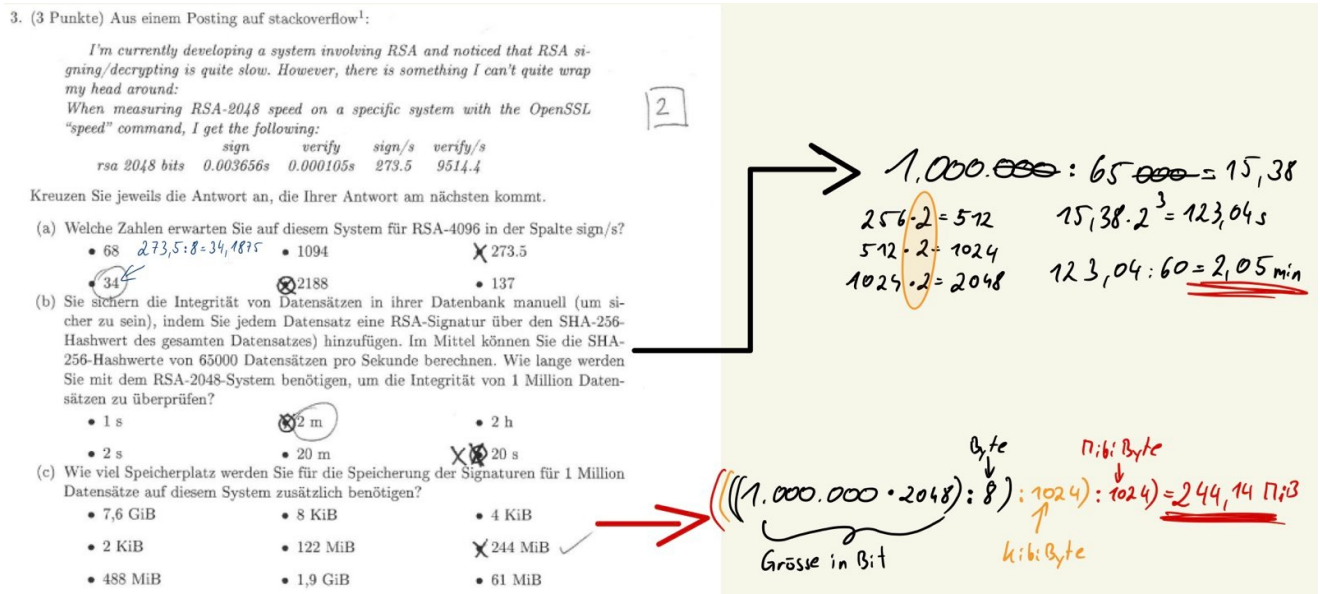
- Wie viele Bits lang ist der geheime Exponent a?
- Wie viele Bits lang ist eine Signatur?
  - → 512 ←

$$r\ mod\ q \implies max\ 256$$

$$s\ mod\ q - 1 \implies max\ 256$$

- Welche Hashfunktion wird sinnvollerweise für die Erstellung von Signaturen mit diesem Private Key verwendet?
  - SHA-256/ SHA512

3)



3. (3 Punkte) Aus einem Posting auf stackoverflow[1]:

I'm currently developing a system involving RSA and noticed that RSA signing/decrypting is quite slow. However, there is something I can't quite wrap my head around:
When measuring RSA-2048 speed on a specific system with the OpenSSL "speed" command, I get the following:

	sign	verify	sign/s	verify/s
rsa 2048 bits	0.003656s	0.000105s	273.5	9514.4

Kreuzen Sie jeweils die Antwort an, die Ihrer Antwort am nächsten kommt.

(a) Welche Zahlen erwarten Sie auf diesem System für RSA-4096 in der Spalte sign/s?
- 68 (273,5 : 8 = 34,1875) • 1094 • 273.5
- 34 • 2188 • 137

(b) Sie sichern die Integrität von Datensätzen in ihrer Datenbank manuell (um sicher zu sein), indem Sie jedem Datensatz eine RSA-Signatur über den SHA-256-Hashwert des gesamten Datensatzes) hinzufügen. Im Mittel können Sie die SHA-256-Hashwerte von 65000 Datensätzen pro Sekunde berechnen. Wie lange werden Sie mit dem RSA-2048-System benötigen, um die Integrität von 1 Million Datensätzen zu überprüfen?
- 1 s • 2 m • 2 h
- 2 s • 20 m • 20 s

(c) Wie viel Speicherplatz werden Sie für die Speicherung der Signaturen für 1 Million Datensätze auf diesem System zusätzlich benötigen?
- 7,6 GiB • 8 KiB • 4 KiB
- 2 KiB • 122 MiB • 244 MiB
- 488 MiB • 1,9 GiB • 61 MiB

2

1.000.000 : 65.000 = 15,38
256 · 2 = 512
512 · 2 = 1024
1024 · 2 = 2048
15,38 · 2³ = 123,04 s
123,04 : 60 = 2,05 min

((1.000.000 · 2048) : 8) : 1024) : 1024) = 244,14 MiB
Grösse in Bit
Byte
kibiByte
MibiByte

5)

Es sei $p = x^3 + x + 1$. Berechnen Sie im endlichen Körper $\mathbb{Z}_2[x]/p$

$$[x^2 + x]_p^4 * [x^2 + x + 1]_p$$

$$[x^2 + x]_p^4 \implies (x^2 + x)^2 \implies (x^4 + x^2)^2 \implies x^8 + x^4$$

$$(x^8 + x^4) * (x^2 + x + 1) \implies x^{10} + x^9 + x^8 + x^6 + x^4$$

$$x^{10} + x^9 + x^8 + x^6 + x^4 \; mod \; x^3 + x + 1 \implies -x^7 + x^6 - 2x^3$$

$$\Downarrow$$

$$Rest \implies 1$$

AKY3 - 2017

Die Zahl p = 1237 ist eine Primzahl. Der diskrete Logarithmus von 739 zur Basis 760 modulo p ist 437. Berechnen Sie den diskreten Logarithmus von 760 zur Basis 739 modulo p.

$$\varphi(1237) = 1236$$

$$760^{437} = 739^{mod\ 1237} \implies 739 = 760^{437}$$

$$739^{x} = 760\ mod\ 1237 \implies 760 = 739^{x}$$

$$760 = 739^{x}$$

$$760 = (760^{437})^{x}$$

$$760^{1=760^{437*x}}\ mod\ 1237$$

$$\Downarrow$$

```
import si
si.extended_gcd(si.euler_phi(1237),437, verbose= 1)
```

$$\Downarrow$$

$$x = -379$$

2)

Betrachten Sie das RSA-Schlüsselpaar bestehend aus dem Public Key (n, e) und dem Private Key (n, d), wobei
n =
0x1a6299f8219c18bce4107c6db1178cede7baf9f094c88322a04bb758975c18864ce875e2f67b2e1d23b3154ea2dab9788926dec4db79268dfa5c59af16e99cd7f6
£725fc9a5abf0dc13fd555286fa6a16a032e34218fa26717a0a7688684acb2f96
69bd5e5e2ff34fe8dd647f91c5f4eca482eb9927a9d3d9ca5730787c713c479f9

481551dd7add32dd854231b8940a413b3c35bd9cb6b3409e189378d6
dca35efa32d6071fae999a977d557a04984d3258df868c3627e23f444f
e66a2b1350837d77
d69ff48b5462290de484fdd2a8676b5158dbe4da2e71980c13ad86ed3
251c8fb0
b9189af9750188d8ad4a5ff5f7b089a0cb447abe97580b73d95e4c9069
e29b91f
6564ad2b81b13a74f769c2b7b73484bdcf94d356028c892180891dd92
7f98f250
c0478ffdc59d9fbb5f7e6ac42df735d3fd6e2e54aae087dd0fc3b3a0b6e0
4de96
63ff9cbe2eb0a24df5a2abb1b6a7b3eb300de0909f66d5088dcb108a6b
b49e2ed
1a7255566c0b138309de8dde1b01ea3cba5

e =
0x1410db3a1f672845b46580a750ac501d34859274845c63045eb6cb1
d13717054cdbb96936c85f129ccc90f4e3755c500b162b4a3e49ababd
57e305886028fa3d1ce633e26a0ba9244720035edc0e784604278e4a
2c5b59a70c740d0e70eb5135672d7140a458bf0699aa4962448808c5
7ce529614c4d584ddc901eca277e0e1c93f4
df709c24aef9e93b30b38ad6da45cb5f5f0c3dcd8fa65b5b9a6fc117bdf5
7b52ae4f871560ab0bb7c25f8d3db914fdd05a12557c44843bf4072365
5151fd631e4035332b2776bda6233fd91fb84bd4d15f4d562e1adcebd3
0a7c90607c68558f60bcb7804da3256ba4654bc934b4a1208b4bf6898
2b72e172526845cd605515cc2354e36dfccb658b0115df1c6fe3201a95
e46147a471d99d22df322b8e29de58be38405ca325214c4e6575536a
bf6d5deadca8c04eb8da84a06ba1a6e8f34950d25d0c86d34035f3672
16e0c3d0ed7574f6ae4dcb5268016faba3b10f3f4bf889afc9752531b51
6037d80d6fd5e53c27cb6634a5d

d = 0xc43aab26a937f854eac764ce8c7036670970bb9f482c42b199

(Alle Werte sind hexadezimal angegeben.) Welches Problem oder welche Probleme sehen Sie beim Einsatz dieses Schlüsselpaars?

$\Longrightarrow$

3)

Die Zahl 1031 ist eine Primzahl. Der diskrete Logarithmus von 473 zur Basis 37 modulo 1031 ist 14. Die Ordnung von 37 in $\mathbb{Z}_{1031}$ ist 1030.

(a) (2 Punkte) Bestimmen Sie die Ordnung von 473 in $\mathbb{Z}_{1031}$.

$$37^{14} = 473\ mod 1031$$

$$Erzeugendes\ Element\ \varphi(1031) = 1030$$

$$473^{x} =\perp 1\ mod\ 1031$$

$$1 = 473^{x} \mid 1 = 37^{14*x}$$

$$\Downarrow$$

```
import si
print(si.extended_gcd(si.euler_phi(1031),14, verbose=1))
# Ergebnis = -147
# 1031-147 = 884 --> verwenden? statt -147???
```

$$\Downarrow$$

$$1 = 37^{14*-147}$$

$$x = -147 \implies Ordnung$$

(b) (2 Punkte) Wie viele Elemente der Ordnung 103 gibt es in $\mathbb{Z}_{1031}$?

```
import si
si.euler_phi(103)
```

$$\varphi(103) = 102$$

(c) (2 Punkte) Berechnen Sie ein Element der Ordnung 103 in $\mathbb{Z}_{1031}$.

$$t = \frac{1030}{ggT(1030, 103)} \implies \frac{1030}{103} \implies 103$$

$$t = 10$$

$$37^{10} \implies Elem\ der\ Ordnung\ 103$$

$$\Downarrow$$

$$37^{10}\ mod\ 1031 = 29$$

5)

Eine Nachricht wurde mit ECDSA signiert (in diesem Beispiel werden Para- meter gewählt, die zu klein sind, um wirklich verwendet werden zu dürfen; ignorieren Sie an dieser Stelle diese Tatsache). Als Parameter wurden die Kurve $\epsilon : y^2 = x^3 + x + 194$ über $\mathbb{Z}_{239}$ und der Punkt $G(1, 14)$ auf der elliptischen Kurve mit der Ordnung 31 verwendet. Verifizieren Sie die ECDSA-Signatur $(24, 12)$ einer Nachricht mit Hashwert 12 mit dem Public Key $(\epsilon, A = (182, 226), G = (1, 14))$.

$$\omega = s^{-1}\ mod\ q$$

$$x = \omega * h(m)\ mod q$$

$$g = \omega * r\ mod\ q$$

$$r = x * G + g * A\ mod\ q$$

$$\omega = 12^{-1}\ mod\ 31 = 13\ mod\ 31$$

$$y = 13 * 12\ mod\ 31 = 156 = 1\ mod\ 31$$

$$g = 13 * 24\ mod\ 31 = 2$$

....

....

...

...

7)

Ein Benchmark Ihrer RSA-Signatur-Funktion zeigt, dass Sie pro Sekunde ungefähr 35 RSA-Signaturen bei einer Schlüssellänge von 4096 Bit erstellen können. Bei dieser Schlüssellänge können Sie 9230 RSA-Signaturen pro Sekunde prüfen. Sie haben in AKY gelernt, dass sich RSA mit dem chinesischen Restsatz beschleunigen lässt. Welche Geschwindigkeiten für das Signieren und Verifizieren erwarten Sie (ungefähr) durch den Einsatz von RSA-CRS zu erreichen, wenn sonst keine Veränderungen vorgenommen werden? Erklären Sie auch, wie Sie zu Ihrer Schätzung kommen.

$$dp = d \; mod \; (p-1)$$

$$dq = d \; mod \; (q-1)$$

$$mp = c^{dp} \; mod \; p \implies 2x$$

$$m1 = c^{d1} \; mod \; 1 \implies 2x$$

$$\Downarrow$$

$$4 \; mal \; so \; schnell$$

$$m = mp + q * (mp - mq) \; mod \; n$$

$$\Downarrow$$

$$2 \; Rechnungen \; statt \; einer$$

$Sign \implies 4 \; so \; schnell \sim 140/s$
$Verify \implies ???$

AKY Klausur 2017

1)

Ihr RSA Private Key enthalt alle Parameter fiir RSA-CRS. Diese sind
• Modul: n = 588139
• Faktor 1: p = 701
• Faktor 2: q = 839
• Exponent 1: dp = l 73
• Exponent 2: dq = 683
• Koeffizient: 'Y = 481587
Entschlüsseln Sie das Chiffrat c = 150770 mit diesem Private Key.

2)

Der Punkt H = (28, 80) auf der Kurve c: $y^2 = x^3 + x + 194$ über $\mathbb{Z}_{239}$ hat die Ordnung

62. (a) (2 Punkte) Bestimmen Sie die Ordnung des Punkts 13 · H.
    (b) (1 Punkt) In welchem Bereich liegt nach dem Satz von Hasse die Ordnung von c?
    (c) (2 Punkte) Bestimmen Sie die Ordnung von c.

3)

(4 Punkte) Eine Nachricht wurde mit ECDSA signiert (in diesem Beispiel werden Parameter gewahlt, die zu klein sind, um wirklich verwendet werden zu dürfen, ignorieren Sie an dieser Stelle diese Tatsache). Als Parameter wurden die Kurve $c : y^2 = x^3 + x + 194$ über $\mathbb{Z}_{239}$ und der Punkt $G = (1, 14)$ auf der elliptischen Kurve mit der Ordnung 31 verwendet. Verifizieren Sie die ECDSA Signatur $(24, 12)$ einer Nachricht mit Hashwert
12 mit dem Public Key $(\varepsilon, A = (47, 92), G = (1, 14))$.

6)

Die Zahl $p = 2237$ ist eine Primzahl. Betrachten Sie die Gruppe
$G := (\mathbb{Z}_p^*, \cdot, ^{-1}, 1)$.
(a) (1 Punkt) Bestimmen Sie die Ordnung der Gruppe G.
(b) (1 Punkt) Wie viele erzeugende Elemente gibt es in der Gruppe G?
(c) (3 Punkte) Ist 2179 ein erzeugendes Element in der Gruppe G?
(d) (2 Punkte) 122 ist ein erzeugendes Element in der Gruppe G.
Bestimmen Sie ein Element der Ordnung 559 in der Gruppe G.