Kryptografia i bezpieczeństwo systemów informatycznych

Andrzej M. Borzyszkowski

Instytut Informatyki
Uniwersytet Gdański

sem. letni 2023/2024

inf.ug.edu.pl/~amb/

Teoria liczb:

źródło funkcji jednokierunkowych

podstawa kryptografii asymetrycznej

Dlaczego teoria liczb?

- Podstawą kryptografii są funkcje jednokierunkowe
  – DES i inne szyfry blokowe są praktyczną implementacją
  – ale nie ma żadnych dowodów jednokierunkowości
- Problemy z teorii liczb
  – znajdywanie dzielników
  – logarytm dyskretny
  – pierwiastek kwadratowy
  – są dobrze znane i istnieje uzasadnione podejrzenie, że są to funkcje jednokierunkowe
- Kryptografia asymetryczna opiera się w całości na algorytmach teorioliczbowych

Arytmetyka

- Podstawy liczenia
  – układ pozycyjny (schemat Hornera):
  $wart_b(Napis\ Cyfra) = wart_b(Napis) \cdot b + wart_b(Cyfra)$
- Zamiana podstaw liczenia
  – $x = b \cdot x/b + x\%b$ (dzielenie całkowite i reszta)
  – czyli $zapis_b(x) = zapis_b(x/b)\, zapis_b(x\%b)$
  (zapis ilorazu i cyfra $[0 \ldots b-1]$ oznaczająca resztę na końcu)
  – części ułamkowe też można zapisywać
- Liczba cyfr: jeśli $k =$ długość$(zapis_b(x))$, to $b^{(k-1)} \leqslant x < b^k$
  – czyli $k \approx log_b(x)$, różne podstawy tylko zmieniają stałą
  – większa podstawa ma mniej cyfr ale zapis każdej cyfr wymaga odpowiednio więcej bitów

Działania arytmetyczne

- Dodawanie: złożoność = liczba cyfr większej liczby
- Mnożenie (algorytm szkolny): mnoży się każdą parę cyfr, dodatkowo występuje dodawanie
  – złożoność = iloczyn liczb cyfr
  – istnieją lepsze algorytmy (ostatnie osiągnięcia $\ell \log \ell$, ℓ–długość)
  – długość$(m \cdot n)$=długość(m)+długość(n) lub o 1 mniej
  – przykład: długość$(n!) \leqslant n \cdot$ długość(n),
  obliczenie silni: n mnożeń liczb o długościach długość(n) oraz długość$(n!)$ czyli $(n \cdot log(n))^2$
- Dzielenie: podobna złożoność
- Zamiana podstawy liczenia (np. z dwójkowego na $\approx 2^\ell$) $\log(n)/\ell$ dzieleń, każde ma złożoność $\log(n) \cdot \ell$, razem $\log(n)^2$

Dzielniki

- $a|b$ oznacza, że a dzieli b, tzn. $\exists x. a \cdot x = b$
  – a jest dzielnikiem, b jest wielokrotnością, można czasami żądać, by $b \neq 0$
  – oczywiście $a|0$, $a|a$, $1|a$, jeśli $a|b$ oraz $b|c$ to $a|c$, jeśli $a|b$ oraz $a|c$, to $a|(\beta \cdot b + \gamma \cdot c)$ – kombinacje liniowe
- $p > 1$ jest liczbą pierwszą gdy jedynymi dzielnikami są 1 i p
  – fakt: liczba liczb pierwszych $< m \approx m/\ln(m)$
  np. liczb pierwszych dokładnie 100 cyfrowych jest $\geqslant 10^{97}$
  – fakt: każda liczba naturalna ma jednoznaczny rozkład na czynniki pierwsze

Algorytm Euklidesa

- $NWD(a, b) = c$ jeśli $c|a$, $c|b$ oraz c jest największe j.w.
  – def: jeśli $NWD(a, b) = 1$ to a i b są względnie pierwsze
  – fakt: jeśli $NWD(a, b) = 1$ to w ich rozkładach na czynniki pierwsze występują zupełnie inne liczby
- Niech $a = \beta \cdot b + r$, wówczas dzielniki (a, b) oraz (b, r) są te same
  – w szczególności $NWD(a, b) = NWD(b, r)$
  – zaczynając od $a \geqslant b \geqslant 0$ otrzymujemy algorytm Euklidesa, koniec jest dla $NWD(a, 0) = a$
  – np. w języku python

```
def nwd(a,b):
while b:
  a,b=b,a%b
  return a
```


Algorytm Euklidesa, złożoność, przykład

- Fakt: liczba kroków jest rzędu $\log(a)$
  – dw.: po dwóch krokach wielkość zadania jest $<$ połowy, $a\%b < a/2$ (najgorszy przypadek to „złoty podział": $a = b + r$ ma te same proporcje co $b = r + s$)
- Złożoność algorytmu: liczba kroków · złożoność dzieleń czyli $\log(a)^3$ (dokładniejsza analiza pokazuje $\log(a)^2$, bo liczby są szybko coraz mniejsze)

nwd(987654321,6543210)

987654321	6543210
6543210	6172821
6172821	370389
370389	246597
246597	123792
123792	122805
122805	987
987	417
417	153
153	111
111	742
42	27
27	15
15	12
12	3

Algorytm Euklidesa, rozszerzony

- Tw.: jeśli $NWD(a, b) = c$, to $\exists \alpha, \beta . \alpha \cdot a + \beta \cdot b = c$
- Dw.: każda reszta w algorytmie jest kombinacją liniową argumentów, $NWD(a, b)$ jest ostatnią niezerową resztą.
- Wniosek z dowodu: współczynniki można obliczyć efektywnie podczas obliczania NWD (rozszerzony algorytm Euklidesa)

```
#!/usr/bin/env python
#Tadeusz Andrzej Kadlubowski, marzec 2008
a,b=map(int,file("dane.txt").readlines()[:2])
p,q,r,s = 1,0,0,1
while b: a,b,p,q,r,s = b,a%b,r,s,p-a/b*r,q-a/b*s
file("nwd.txt","w").write("%d\n%d\n%d\n"%(a,p,q))
```

- dw.: jeśli A i B są pierwotnymi wartościami a i b to w każdym przebiegu pętli zachodzi $p \cdot A + q \cdot B = a$ oraz $r \cdot A + s \cdot B = b$.

Kongruencje (arytmetyka modularna)

- $a = b \mod n$ jeśli $n | a - b$ ($n \neq 0$, w praktyce $n > 0$)
  – czyli $\exists \lambda . a = b + \lambda \cdot n$
- Relacja równoważności (zwrotność, symetria, przechodniość)
  – $a \equiv a$, $a \equiv b$ implikuje $b \equiv a$, $a \equiv b$, $b \equiv c$ implikuje $a \equiv c$
  – reprezentantami są $0, 1, \ldots, n-1$, oznaczenie $Z_n = \{0, \ldots, n-1\}$
  – dla każdej liczby istnieje reszta, $a = r \mod n$, $r \in Z_n$
- Można wykonywać działania dodawania i mnożenia
  – tzn. $a = b \mod n$ oraz $c = d \mod n$ implikuje $a + c = b + d \mod n$, $a \cdot c = b \cdot d \mod n$
- Tw.: jeśli $ab = ac \mod n$ oraz $NWD(a, n) = 1$, to $b = c \mod n$ (tzn. dzielenie ma sens)
- dw. $\exists \alpha, \beta . \alpha \cdot a + \beta \cdot n = 1$ czyli $\alpha \cdot a \cdot (b - c) + \beta \cdot n \cdot (b - c) = b - c$, ponieważ $n | \beta \cdot n \cdot (b - c)$ i $n | \alpha \cdot a \cdot (b - c)$ (założenie) więc $n | b - c$
  – np. $5 \cdot x = 7 = 18 = 29 = 40 = 5 \cdot 8 \mod 11$ i $NWD(5, 11) = 1$ więc $x = 8 \mod 11$
  – ale $4 \cdot 4 = 4 \cdot 1 \mod 12$, a jednak $4 \neq 1 \mod 12$

Złożoność działań w arytmetyce modularnej

- Zasada: najpierw redukcja modulo, potem działanie
  – $a \cdot b \mod N = ((a \mod N) \cdot (b \mod N)) \mod N$
  – złożoność: $\log(a) + \log(b) + \log(N)$ (obliczenie reszt mod N) $+ \log(N)^2$ (obliczenia dla reszt)
- Przykład
  – $10986576198654210 43 \cdot 87397655445287387347 \mod 100$
  $= 43 \cdot 47 = 2021 = 21 \mod 100$

Szybkie potęgowanie

Obliczyć

$$
\begin{aligned}
2^{100} &\mod 123 \\
= 4^{50} \cdot 1 &\mod 123 \\
16^{25} \cdot 1 &\mod 123 \\
16^{24} \cdot 16 &\mod 123 \\
(16^2)^{12} \cdot 16 &\mod 123 \\
10^{12} \cdot 16 &\mod 123 \\
(10^2)^6 \cdot 16 &\mod 123 \\
(100^2)^3 \cdot 16 &\mod 123 \\
37^3 \cdot 16 &\mod 123 \\
37^2 \cdot 16 \cdot 37 &\mod 123 \\
37^2 \cdot 100 &\mod 123 \\
16^1 \cdot 100 &\mod 123 \\
16^0 \cdot 100 \cdot 16 &\mod 123 \\
1 &\mod 123
\end{aligned}
$$

```
r=1;
while(n) /* inv:r*b^n mod m */{
 if(n%2){n--; r=(b*r)%m;}
 else{n=n/2; b=(b*b)%m;}
}
```

```
def potm(b,n,m):
if n==0: return 1
if n%2==1: return (b*potm(b,n-1,m))%m
else: return potm((b*b)%m,n/2,m)%m
```

– liczba kroków $\approx$ długość wykładnika
– przy każdej okazji dokonuje się redukcji modulo
– nigdy nie operuje się na liczbach większych niż kwadrat modułu

„Odwrotności" liczb w arytmetyce modularnej

- $5 \cdot x = 7 \mod 11$ można rozwiązać znajdując odwrotność 5:
  – $9 \cdot 5 = 1 \mod 11$ więc $x = 9 \cdot 7 = 8 \mod 11$
- Jeśli $NWD(a, n) = 1$, to $\exists \alpha, \beta.\ \alpha \cdot a + \beta \cdot n = 1$, czyli $\alpha \cdot a = 1 \mod n$
  – $11111 \cdot x = 4 \mod 12345$, rozwiązanie $x = 2471 \cdot 4 = 9884$
  – ponieważ $2471 \cdot 11111 = 1 \mod 12345$ zostało obliczone w rozszerzonym algorytmie Euklidesa
- Jeśli $\exists \alpha.\ \alpha \cdot a = 1 \mod n$, to $\exists \beta.\ \alpha \cdot a + \beta \cdot n = 1$ czyli $NWD(a, n) = 1$
- Problem: rozwiązać równanie $a \cdot x = b \mod n$
  – jeśli istnieje rozwiązanie, to $NWD(a, n) | b$
  – jeśli $NWD(a, b) = c$, $c|b$, to można rozwiązać równoważny problem $(a/c) \cdot x = b/c \mod n/c$
  – np. $12 \cdot x = 21 \mod 39$ daje $4 \cdot x = 7 \mod 13$, $x = 5$ w oryginalnym problemie są jeszcze rozwiązania 18 i 31

Kryptografia klucza publicznego Implementacja idei

Teoria grup

- Zbiór G z działaniem
  – zapis np. $a + b$, albo $a \cdot b$, albo $a \bullet b$
  – istnieje „jedynka" e: $a \bullet e = e \bullet a = a$
  – łączność: $a \bullet (b \bullet c) = (a \bullet b) \bullet c$
  – istnieją elementy odwrotne: $\forall a.\ \exists b.\ a \bullet b = e$, zapis a^{-1}
  – dla nas ważne będą wyłącznie grupy przemienne: $a \bullet b = b \bullet a$
- Grupa skończona: rząd grupy = liczebność grupy $|G|$
- Przykłady:
  – liczby całkowite i dodawanie, rząd nieskończony
  – reszty mod N i dodawanie: grupa addytywna $\mathbb{Z}_N = \{0, .., N-1\}$, $a \bullet b = a + b \mod N$, rząd $|Z_N| = N$
  – niezerowe reszty mod p i mnożenie: grupa multiplikatywna $\mathbb{Z}_p^* = \{1, .., p-1\}$, $a \bullet b = a \cdot b \mod p$, rząd $|Z_p^*| = p - 1$
  – również grupa multiplikatywna $\mathbb{Z}_N^*$ – elementy odwracalne mod N, rząd grupy – funkcja Eulera $\varphi(N)$

Teoria grup, c.d.

- Potęgowanie
  – $a \in G$, n – liczba, $a^n = a \bullet a \ldots \bullet a$ (n krotnie)
- Tw.: $a^{|G|} = e$
  – dw.: mnożymy wszystkie elementy grupy $a_1 \bullet a_2 \bullet \ldots \bullet a_n$, $n = |G|$, a potem jeszcze raz pomnożone przez a:
  $(a \bullet a_1) \bullet (a \bullet a_2) \bullet \ldots \bullet (a \bullet a_n)$
  – To są te same elementy, tylko kolejność inna,
  – po skróceniu $a^n = e$
- Wn.: potęgowanie w grupie można wykonywać modularnie:
  $a^j = a^{j \mod |G|}$
  – jeśli $NWD(j, |G|) = 1$, to $f(a) = a^j$ jest bijekcją, odwrotną jest $g(a) = a^i$, gdzie $j \cdot i = 1 \mod |G|$

Twierdzenie Fermata (małe)

- Tw.: jeśli p jest liczbą pierwszą i nie dzieli a, to $a^{p-1} = 1 \mod p$
- Dw.: $a \mod p$ jest elementem $\mathbb{Z}_p^*$ (tzn. $\{1, 2, \ldots, p-1\}$), $|\mathbb{Z}_p| = p - 1$ i stosujemy poprzednie twierdzenie
- przykład: $2^{10} = 1 \mod 11$
  stąd $2^{123456789} = 2^9 = 6 \mod 11$
- $a^{560} = a^0 = 1 \mod 3$ ponieważ $560 = 0 \mod 2$
  $a^{560} = a^0 = 1 \mod 11$ ponieważ $560 = 0 \mod 10$
  $a^{560} = a^0 = 1 \mod 17$ ponieważ $560 = 0 \mod 16$
  więc $a^{560} = 1 \mod 561$ (chińskie tw. o resztach, $561 = 3 \cdot 11 \cdot 17$)
- prawie zawsze $2^{n-1} = 1 \mod n$ implikuje, że n jest liczbą pierwszą
  – 561 jest wyjątkiem, dowolne a spełnia $a^{n-1} = 1 \mod n$
  – test pierwszości: szybkie potęgowanie w celu sprawdzenia czy $a^{n-1} = 1 \mod n$, dla różnych wartości a zaczynając od $a = 2$
  – jeśli nie ma równości, to liczba nie jest pierwsza

Funkcja Eulera φ

- $\varphi(n) =$ liczba liczb $a \in \{1, .., n-1\}$ takich że $NWD(a, n) = 1$
- tw. Euklidesa: istnieją współczynniki k, ℓ takie, że $k \cdot a + \ell \cdot n = NWD(a, n)$
- czyli $\varphi(n)$ jest rzędem grupy multiplikatywnej $\mathbb{Z}_n^*$
  – jeśli $n = p$, to $\varphi(p) = p - 1$ (tw. Fermata)
  – jeśli $n = p^k$, to $\varphi(p^k) = p^k - p^{k-1} = (1 - \frac{1}{p}) \cdot n$
  (nie są względnie pierwsze wielokrotności p: $\left\{p, 2 \cdot p, \ldots, p^k - p\right\}$)
  – ogólnie $\varphi(n) = n \cdot$ iloczyn $(1 - \frac{1}{p})$ po wszystkich dzielnikach pierwszych n (chińskie tw. o resztach)
  – ważny przypadek $\varphi(p \cdot q) = (p - 1) \cdot (q - 1)$
  – np. $\varphi(26) = \varphi(2 \cdot 13) = 12$

Twierdzenie Eulera

- Tw.: Jeśli $NWD(a, n) = 1$, to $a^{\varphi(n)} = 1 \mod n$
- małe tw. Fermata jest przypadkiem szczególnym
- Wniosek: jeśli $NWD(a, n) = 1$, $x = y \mod \varphi(n)$, to $a^x = a^y \mod n$
  – tzn. działania mod n można zastąpić działaniami mod $\varphi(n)$ w wykładniku
  – przykład: $2^{43210} = 2^{10} = 10 \mod 26$, bo $\varphi(26) = 12$, $43210 = 3600 \cdot 12 + 10$ oraz $2^{10} = 1024 = 39 * 26 + 10$
  – albo: $3^{456789} = 3^9 = 1 \mod 26$, bo $\varphi(26) = 12$, $456789 = 38065 \cdot 12 + 9$ oraz $3^9 = 19683 = 757 * 26 + 1$

RSA

Rozkład na czynniki

- Problem rozkładu na czynniki
  – Tadeusz losuje dwie liczby i podaje ich iloczyn N
  – Ewa wygrywa, jeśli znajdzie jakikolwiek rozkład N na czynniki
- Powyższy problem nie jest trudny (trudny = praktycznie niemożliwy do wykonania algorytmem wielomianowym)
  – $\frac{3}{4}$ przypadków daje parzyste N
  – liczby nieparzyste, ale z małymi dzielnikami, też powinny być wykluczone
  – w zasadzie trzeba zacząć od losowania liczb pierwszych
- Hipoteza: rozkład na czynniki jest zadaniem trudnym, jeśli losowane są liczby pierwsze (N jest iloczynem dwóch liczb pierwszych)

Kryptografia klucza publicznego: RSA

- Historia
  – pierwsza opublikowana implementacja: Rivest, Shamir, Adleman (USA) 1977
  – dziś wiadomo, że wywiad brytyjski znał ideę w 1970, implementację w 1973 (w zasadzie też RSA)
- Duże liczby pierwsze p oraz q (co najmniej 500 bitowe)
  – $N = p \cdot q$, $\varphi(N) = (p-1) \cdot (q-1)$
  – wybieramy $e \in \mathbb{Z}_N^*$, w zasadzie losowe $< N$ wystarczy, szansa, że jest wielokrotnością p lub q jest zaniedbywalnie mała
  – dla e znajdujemy d takie, że $e \cdot d = 1 \mod \varphi(N)$
  w tym celu wykonujemy algorytm Euklidesa dla e oraz $\varphi(N)$
  – klucz publiczny (N, e), klucz prywatny (N, d)
  – $E((N,e),m) = m^e \mod N$, $D((N,d),c) = c^d \mod N$, $m \in \mathbb{Z}_N^*$, znowu szansa, że m nie będzie spełniać warunku jest zaniedbywalnie mała

Kryptografia klucza publicznego: RSA

- Uzasadnienie:

$$D((N,d),E((N,e),m)) = D((N,d),m^e \mod N) = (m^e)^d \mod N =$$
$$= m^{(e \cdot d) \mod \varphi(N)} \mod N = m^1 \mod N = m$$

- Złożoność
  – przygotowanie: znalezienie liczb pierwszych czyli testowanie pierwszości kolejnych liczb
  – użycie: podnoszenie do potęgi w arytmetyce modularnej
  – dla liczb n-bitowych trzeba $O(n^3)$ operacji

Złożoność używania RSA vs. łamania RSA czyli n^3 vs. $e^{2 \cdot \sqrt{n}}$

n	n**3	2*sqrt(n)	exp(2*sqrt(n))
2	8	3	17
4	64	4	55
8	512	6	286
16	4096	8	2981
32	32768	11	81937
64	262144	16	8886111
128	2097152	23	6713706353
256	16777216	32	78962960182681
512	134217728	45	4.507385299E+0019
1024	1073741824	64	6.235149081E+0027
2048	8589934592	91	2.031652223E+0039
4096	68719476736	128	3.887708406E+0055
8192	549755813888	181	4.127610756E+0078
16384	4398046511104	256	1.511427665E+0111
32768	35184372088832	362	1.703717056E+0157

RSA, możliwe ataki

- Mały klucz prywatny: wyczerpujące przeszukiwanie
- Mały klucz publiczny, np. $e = 3$
  – jeśli $m \leqslant N^{\frac{1}{3}}$, to $m^e \leqslant N$ i łatwo obliczyć pierwiastek (nie ma w ogóle działań modularnych)
  – trzy różne klucze publiczne $(N_1, 3)$, $(N_2, 3)$, $(N_3, 3)$, ta sama wiadomość m zaszyfrowana trzy razy: $c_i = m^3 \mod N_i$
  – z chińskiego twierdzenia o resztach obliczamy $m^3 \mod N_1 \cdot N_2 \cdot N_3$, ale $m^3 \leqslant N_1 \cdot N_2 \cdot N_3$, a więc m^3 znane jest dokładnie, i m też
- Mały tekst jawny m (np. szyfruje się klucz symetryczny DES)
  – Ewa zna $c = m^e \mod N$
  – sporządza listę $c \cdot x^{-e} \mod N$ oraz listę $y^e \mod N$ dla $x, y \leqslant \sqrt{N}$ i szuka wspólnego elementu, wówczas $m = x \cdot y$
  – jeśli m jest l.pierwszą, to nie znajdziemy tak rozwiązania

RSA, możliwe ataki c.d.

- Wspólna liczba N dla kilku par kluczy
  – znajomość jednej pary e, d pozwala znaleźć rozkład (R–M), funkcję Eulera oraz odwrotności innych kluczy publicznych
  – jeśli $NWD(e_1, e_2) = 1$, to $\beta \cdot e_1 + \gamma \cdot e_2 = 1$
  jeśli szyfrowana jest ta sama wiadomość, to $(m^{e_1})^\beta \cdot (m^{e_2})^\gamma = m \mod N$
- Dalsze ataki
  – Jeśli znanych jest pierwsze lub ostatnie ćwierć bitów liczby N, to rozkład jest możliwy
  – Jeśli znanych jest ostanie ćwierć bitów klucza prywatnego d, to złożoność znalezienia d jest proporcjonalna do wartości e
  – Jeśli p i q są prawie równe i $\log d < (\log N)/4$, to d można łatwo obliczyć znając klucz publiczny (N, e)

Randomizacja RSA

- Brak niedeterminizmu RSA przyczyną skutecznych ataków
- Idea:
  – niezależnie od randomizacji trzeba uzupełniać jakoś zbyt krótkie wiadomości
  – przed zaszyfrowaniem wiadomości dodać losowy element
  – trzeba umieć usunąć ten element po odszyfrowaniu
- Standard PKCS#1 v1.5:
  – wiadomość m poprzedzona jest co najmniej 11 bajtami (88 bitów) z czego trzy są ustalone, pozostałe losowe
  – po odszyfrowaniu jest nadzieja, że rozpozna się wzorzec dodany
  – ostatnio opracowano nowy, lepszy sposób uzupełniania wiadomości

Logarytm dyskretny

Grupy cykliczne, generatory

- Każdy element $g \in G$ generuje podgrupę $\langle g \rangle = \{e, g, g \cdot g, \ldots, g^{r-1}\}$
  - rząd elementu: najmniejsze takie r, że $g^r = e$
  - tw.: jeśli $g^x = g^y$, to $x = y \mod r$
  - na pewno $g^m = e$, dla $m = |G|$ rząd grupy
  - wn.: rząd elementu jest dzielnikiem rzędu grupy
- Generator: element g, taki, że $\langle g \rangle = G$, tzn. rząd g = rząd G
  - jeśli rząd jest l. pierwszą, to każdy element $\neq e$ jest generatorem
- Grupa cykliczna: grupa, która ma generator
  - nie musi wcale istnieć generator grupy
  - jeśli istnieje, to raczej nie jest jednoznaczny
  - np. $\langle g \rangle$ jest grupą cykliczną, generatorem jest g

Pierwiastki pierwotne (generatory grupy multiplikatywnej)

- $\mathbb{Z}_p^* = \{1, \ldots, p-1\}$ elementy odwracalne mnożenia mod p, grupa multiplikatywna
  - Tw.: $\mathbb{Z}_p^*$ jest grupą cykliczną, ma $\varphi(p-1)$ generatorów
  - nazwa: generator $\mathbb{Z}_p^*$ = pierwiastek pierwotny
- Przykłady:
  - 3 generuje $\mathbb{Z}_7^*$:

3^1	3^2	3^3	3^4	3^5	3^6
3	2	6	4	5	1

  - ale 2 nie:

2^1	2^2	2^3	2^4	2^5	2^6
2	4	1	2	4	1

  - 2 generuje $\mathbb{Z}_{13}^*$:

2^1	2^2	2^3	2^4	2^5	2^6	2^7	2^8	2^9	2^{10}	2^{11}	2^{12}
2	4	8	3	6	12	11	9	5	10	7	1

  - ale 3 nie:

3^1	3^2	3^3	3^4	3^5	3^6	3^7	3^8	3^9	3^{10}	3^{11}	3^{12}
3	9	1	3	9	1	3	9	1	3	9	1

Pierwiastki pierwotne c.d.

- Rząd elementu dzieli rząd grupy, największe możliwe dzielniki to $(p-1)/q$ dla $q|p-1$
  - wystarczy sprawdzić tylko te największe potęgi
  - np. $p = 601$, $p-1$ ma dzielniki pierwsze $2, 3, 5$
  - sprawdzamy jedynie $7^{300} = 600 \neq 1$, $7^{200} = 576 \neq 1$, $7^{120} = 423 \neq 1$
  - wniosek: 7 jest generatorem $\mathbb{Z}_{601}^*$
  - można sprawdzić, że nie jest generatorem żadna z mniejszych liczb, jest np. 11

$\mathbb{Z}_{601}^*$ – potęgi 7

7	49	343	598	580	454	173	9	63	441	82	574	412	480	355	81	567	363	137	358
102	113	190	128	295	262	31	217	317	416	508	551	251	555	279	150	449	138	365	151
456	187	107	148	435	40	280	157	498	481	362	130	309	360	116	211	275	122	253	569
377	235	443	96	71	497	474	313	388	312	381	263	38	266	59	413	487	404	424	564
342	591	531	111	176	30	210	268	73	511	572	398	382	270	87	8	56	392	340	577
433	26	182	72	504	523	55	385	291	234	436	47	329	500	495	460	215	303	318	423
557	293	248	534	132	323	458	201	205	233	429	599	587	503	516	6	42	294	255	583
475	320	437	54	378	242	492	439	68	476	327	486	397	375	221	345	11	77	539	167
568	370	186	100	99	92	43	301	304	325	472	299	290	227	387	305	332	521	41	287
206	240	478	341	584	482	369	179	51	357	95	64	448	131	316	409	459	208	254	576
426	578	440	75	525	69	483	376	228	394	354	74	518	20	140	379	249	541	181	65
455	180	58	406	438	61	427	585	489	418	522	48	336	549	237	457	194	156	491	432
19	133	330	507	544	202	212	282	171	596	566	356	88	15	105	134	337	556	286	199
191	135	344	4	28	196	170	589	517	13	91	36	252	562	328	493	446	117	218	324
465	250	548	230	408	452	159	512	579	447	124	267	66	462	229	401	403	417	515	600
594	552	258	3	21	147	428	592	538	160	519	27	189	121	246	520	34	238	464	243
499	488	411	473	306	339	570	384	284	185	93	50	350	46	322	451	152	463	236	450
145	414	494	453	166	561	321	444	103	120	239	471	292	241	485	390	326	479	348	32
224	366	158	505	530	104	127	288	213	289	220	338	563	335	542	188	114	197	177	37
259	10	70	490	425	571	391	333	528	90	29	203	219	331	514	593	545	209	261	24
168	575	419	529	97	78	546	216	310	367	165	554	272	101	106	141	386	298	283	178
44	308	353	67	469	278	143	400	396	368	172	2	14	98	85	595	559	307	346	18
126	281	164	547	223	359	109	162	533	125	274	115	204	226	380	256	590	524	62	434
33	231	415	501	502	509	558	300	297	276	129	302	311	374	214	296	269	80	560	314
395	361	123	260	17	119	232	422	550	244	506	537	153	470	285	192	142	393	347	25
175	23	161	526	76	532	118	225	373	207	247	527	83	581	461	222	352	60	420	536
146	421	543	195	163	540	174	16	112	183	79	553	265	52	364	144	407	445	110	169
582	468	271	94	57	399	389	319	430	5	35	245	513	586	496	467	264	45	315	402
410	466	257	597	573	405	431	12	84	588	510	565	349	39	273	108	155	484	383	277
136	351	53	371	193	149	442	89	22	154	477	334	535	139	372	200	198	184	86	1

$\mathbb{Z}_{601}^*$ – potęgi 11

11	121	129	217	584	414	347	211	518	289	174	111	19	209	496	47	517	278	53	583
403	226	82	301	306	361	365	409	292	207	474	406	259	445	87	356	310	405	248	324
559	139	327	592	502	113	41	451	153	481	483	505	146	404	237	203	430	523	344	178
155	503	124	162	580	370	464	296	251	357	321	526	377	541	542	553	73	202	419	402
215	562	172	89	378	552	62	81	290	185	232	148	426	479	461	263	489	571	271	577
337	101	510	201	408	281	86	345	189	276	31	341	145	393	116	74	213	540	531	432
545	586	436	589	469	351	255	401	204	441	43	473	395	138	316	471	373	497	58	37
407	270	566	216	573	293	218	595	535	476	428	501	102	521	322	537	498	69	158	536
487	549	29	319	504	135	283	108	587	447	109	598	568	238	214	551	51	561	161	569
249	335	79	268	544	575	315	460	252	368	442	54	594	524	355	299	284	119	107	576
326	581	381	585	425	468	340	134	272	588	458	230	126	184	221	27	297	262	478	450
142	360	354	288	163	591	491	593	513	234	170	67	136	294	229	115	63	92	411	314
449	131	239	225	71	180	177	144	382	596	546	597	557	117	85	334	68	147	415	358
332	46	506	157	525	366	420	413	336	90	389	72	191	298	273	599	579	359	343	167
34	374	508	179	166	23	253	379	563	183	210	507	168	45	495	36	396	149	437	600
590	480	472	384	17	187	254	390	83	312	427	490	582	392	105	554	84	323	548	18
198	375	519	300	295	240	236	192	309	394	127	195	342	156	514	245	291	196	353	277
42	462	274	9	99	488	560	150	448	120	118	96	455	197	364	398	171	78	257	423
446	98	477	439	21	231	137	305	350	244	280	75	224	60	59	48	528	399	182	199
386	39	429	512	223	49	539	520	311	416	369	453	175	122	140	338	112	30	330	24
264	500	91	400	193	320	515	256	412	325	570	260	456	208	485	527	388	61	70	169
56	15	165	12	132	250	346	200	397	160	558	128	206	463	285	130	228	104	543	564
194	331	35	385	28	308	383	6	66	125	173	100	499	80	279	64	103	532	443	65
114	52	572	282	97	466	318	493	14	154	492	3	33	363	387	50	550	40	440	32
352	266	522	333	57	26	286	141	349	233	159	547	7	77	246	302	317	482	494	25
275	20	220	16	176	133	261	467	329	13	143	371	475	417	380	574	304	339	123	151
459	241	247	313	438	10	110	8	88	367	431	534	465	307	372	486	538	509	190	287
152	470	362	376	530	421	424	457	219	5	55	4	44	484	516	267	533	454	186	243
269	555	95	444	76	235	181	188	265	511	212	529	410	303	328	2	22	242	258	434
567	227	93	422	435	578	348	222	38	418	391	94	433	556	106	565	205	452	164	1

Logarytm dyskretny

- Problem dotyczy grup cyklicznych
  – może być $\mathbb{Z}_p^*$, albo podgrupa cykliczna w $\mathbb{Z}_N^*$
  – albo zupełnie inna grupa (krzywe eliptyczne, ...)
- Dana grupa cykliczna G i jej generator g:
  – dla każdego elementu h, istnieje wykładnik $g^x = h$
  – nazwa: logarytm dyskretny z h przy podstawie g
  – grupa G jest izomorficzna z grupą addytywną $\mathbb{Z}_m$, $m = |G|$
  – ale izomorfizm nie musi być łatwy obliczeniowo
- Problem logarytmu dyskretnego:
  – znane są grupa G, jej rząd m oraz generator g
  – Tadeusz podaje element $h(= g^x)$
  – Ewa ma znaleźć x
- Hipoteza: dla odpowiednich wyborów danych, problem jest trudny

„Wymiana" klucza w/g Diffie'go—Hellmana

- Alicja i Bolek uzgadniają p oraz generator g
  – każde generuje prywatną liczbę a oraz $b < p - 1$
  – i ogłasza $g^a \mod p$ oraz $g^b \mod p$
  – każde potrafi obliczyć $(g^b)^a = (g^a)^b \mod p$
- Problem DH (obliczeniowy):
  – obliczyć $g^{a \cdot b} \mod p$ znając g^a i g^b
  – problem DH (decyzyjny): mając g^a, g^b, h sprawdzić czy $h = g^{a \cdot b}$
- Rozwiązanie problemu logarytmu dyskretnego rozwiązuje problem obliczeniowy DH
  – rozwiązanie problemu obliczeniowego DH rozwiązuje problem decyzyjny DH
  – nie wiadomo, czy ogólnie odwrotne implikacje zachodzą
  – dla niektórych grup problem decyzyjny DH jest łatwy, ale obliczeniowy jest trudny

Kryptografia klucza publicznego: problem Diffie'go—Hellmana

- Tw.: Jeśli G jest grupą, $m \in G$, to element $g \cdot m$ jest tak samo losowy jak g, dla $g \in G$
  – dw.: $Prawd[g \cdot m = g_0] = Prawd[g = g_0 \cdot m^{-1}] = 1/|G|$
  – wniosek: szyfrowanie może polegać na zaburzeniu wiadomości czynnikiem losowym
  – szyfr jednorazowy jest dokładnie tym schematem (działaniem grupowym jest $\oplus$)
  – zamiast czynnika losowego, można użyć pseudolosowego
- Szyfr ElGamala
  – w oryginale dla grupy multiplikatywnej $\mathbb{Z}_p^*$
  – można stosować np. dla krzywych eliptycznych czy hipereliptycznych
- Kluczy publicznego i prywatnego nie można zamienić rolami !

Szyfr ElGamala

- Klucz publiczny Bolka: p, generator g, potęga $\beta = g^b \mod p$
  klucz prywatny Bolka: wykładnik b
- Alicja szyfruje $m < p$: losuje y, oblicza $r = g^y \mod p$ oraz $t = \beta^y \cdot m \mod p$, szyfrogramem jest para (r, t)
- Bolek odszyfrowuje: $t \cdot r^{-b} \mod p$
- Uzasadnienie: $\beta^y \cdot m \cdot (g^y)^{-b} = m \cdot g^{e \cdot (b \cdot y - y \cdot b)} = m \mod p$
  – analogia do protokołu Diffie-Hellmana:
  – udział Bolka w kluczu wspólnym – klucz publiczny,
  – udział Alicji – część r
  – klucz wspólny $\beta^y = r^b = g^{b \cdot y}$
  – Bolek potrafi obliczyć klucz wspólny i odszyfrować m

Szyfr ElGamala, własności

- Złamanie szyfru ElGamala jest problemem obliczeniowym DH
  – jeśli (β, r) prowadzi do klucza, to $m = t \cdot \text{klucz}^{-1}$
  – jeśli mając (β, r) i dowolne t można znaleźć m, to klucz $= t \cdot m^{-1}$
- Tw.: przy założeniu trudności problemu decyzyjnego Diffie'go—Hellmana, szyfr ElGamala jest odporny na atak z wybranym tekstem jawnym
  – dw.: dane są g^b, g^y, g^z i chcemy sprawdzić, czy trzecia wartość jest wspólnym kluczem
  – to jest to samo co znać klucz publiczny g^b, czynnik zaburzający, g^y, i pytać, czy g^z jest kryptogramem $m = 1$

Szyfr ElGamala, porównanie z RSA

- Użycie szyfru ElGamala wymaga (w zasadzie) znalezienia liczby silnie pierwszej
  – ale $\frac{p-1}{2}$ nie musi być pierwsza, wystarczy bardzo duży dzielnik pierwszy
  – grupa i generator mogą być ustalone raz na zawsze
  – cała tajność zawarta jest w prywatnym wykładniku
  – w RSA dla każdego klucza trzeba szukać nowych liczb pierwszych
- Szyfr ElGamala jest niedeterministyczny z samej definicji
  – dla RSA (bez randomizacji) łatwo sprawdzić, czy dany szyfrogram szyfruje tekst dany jawny
  – niedeterminizm jest konieczny dla bezpieczeństwa szyfru z kluczem publicznym

Atak z wybranym kryptogramem

- RSA: nie jest odporny na ten atak
  – Mariola nie potrafi odszyfrować $c = m^e \mod N$
  – ale żąda odszyfrowania $r^e \cdot c \mod N$, dla dowolnego r
  – wynik będzie równy $r \cdot m$, da się obliczyć m
- ElGamal: też nie jest odporny na ten atak
  – Mariola nie potrafi odszyfrować pary $(g^y \mod p, \beta^b \cdot m \mod p)$
  – ale żąda odszyfrowania $(g^y \mod p, \beta^b \cdot m \cdot r \mod p)$ z wybranym r
  – wynik będzie równy $m \cdot r$, da się obliczyć m
  – może też łatwo wprowadzić zaburzenie pierwszego składnika kryptogramu, by nie budzić podejrzeń
- Znane są szyfry odporne na atak z wybranym kryptogramem
  – są wersją szyfru ElGamala
  – nieznane są podobne szyfry klasy RSA