

Kryptering



1

Kryptering



Scenarium:
Alice ønsker at sende en meddelelse (klartekst) til Bob
Kommunikationskanalen er usikker og kan blive aflyttet
Hvis Alice og Bob er blevet enige om et skema for kryptering,
kan meddelelsen sendes krypteret (som kodetekst)

Spørgsmål:
Hvad er et godt skema til kryptering?
Hvilken kompleksitet har kryptering/afkryptering?
Hvor lang er kodeteksten i forhold til klarteksten?
Hvis Alice og Bob aldrig før har kommunikeret med hinanden,
hvordan kan de så blive enige om et skema for krypteringen?



2

Traditionel kryptografi



Metoder til kryptering (kodeskrift) blev studeret allerede i oldtiden

Cæsars kodeskrift:

erstat *a* med *d*

erstat *b* med *e*

...

erstat *z* med *c*

Cæsars kodeskrift er et eksempel på en **monoalfabetisk erstatningskode**, der permuterer de enkelte tegn i klarteksten

3

Statistiske angreb



Udstyret med simpel statistisk viden kan man let bryde en sådan kode:

- hyppigste bogstaver i engelsk tekst: e, t, o, a, n, i, ...
- hyppigste *digrammer* i engelsk tekst: th, in, er, re, an, ...
- hyppigste *trigrammer* i engelsk tekst: the, ing, and, ion, ...

Den første beskrivelse af dette angreb med frekvensanalyse forekommer i en bog, skrevet i det 9^{nde} århundrede af den arabiske filosof al-Kindi

Eksempel (S. Singh, The Code Book, 1999):

PCQ VMJYPD LBYK LYSO KBXBIXWXV BXV ZCJPO EYPD KBXBJYUXJ
LBJO KCPK. CP LBO LBCMXPV XPV IYJKL PYDBL, QBOP KBO BXV
OPVOV LBO LXRO CI SX'XJMI, KBO JCKO XPV EYKQOV LBO DJCMPV
ZOICJO BYS, KXUYPD: "DJOXL EYPD, ICJ X LBCMXPV XPV CPO PYDBLK
Y BXNO ZOOP JOACMPLYPD LC UCM LBO IXZROK CI FXKL XDOK XPV LBO
RODOPVK CI XPAYOPL EYDK. SXU Y SXEO KC ZCRV XK LC AJXNO X
IXNCMJ CI UCMJ SXGOKLU?" OFYRCMO, LXROK IJCS LBO LBCMXPV
XPV CPO PYDBLK

4

Frekvensanalyse (1)

Vi identificerer den hyppigste tegn, digrammer og trigrammer i klarteksten

Eksempel:

PCQ VMJYPD LBYK LYSO KBXBJXWXV BXV ZCJPO EYPD KBXBJYUXJ
LBJOO KCPK. CP LBO LBCMXXPV XPV IYJKL PYDBL. QBOP KBO BXV
OPVOV LBO LXRO CI SX'XJMI, KBO JCKO XPV EYKKOV LBO DJCMPV
ZOICJO BYS, KXUYPD: "DJOXL EYPD, ICJ X LBCMXXPV XPV CPO PYDBLK
Y BXNO ZOOP JOACMPLYPD LC UCM LBO IXZROK CI FXKL XDOK XPV LBO
RODOPVK CI XPAYOPL EYPDK. SXU Y SXEO KC ZCRV XK LC AJXNO X
IXNCMJ CI UCMJ SXGOKLU?" OFYRCDMO, LXROK IJCS LBO LBCMXXPV
XPV CPO PYDBLK

Første gæt:

LBO er THE

5

Frekvensanalyse (2)

Under antagelse af, at LBO repræsenterer THE, erstatter vi L med T, B med H, og O med E, og får

PCQ VMJYPD THYK TYSE KHXHJXWXV HXV ZCJPE EYPD KHXHJYUXJ
THJEE KCPK. CP THE THCMKXPV XPV IYJKT PYDHT, QHEP KHO HXV
EPVEV THE LXRE CI SX'XJMI, KHE JCKE XPV EYKKEV THE DJCMPV
ZEICJE HYS, KXUYPD: "DJEXT EYPD, ICJ X THCMKXPV XPV CPE PYDHTK
Y HXNE ZEEP JEACMPYTPD TC UCM THE IXZREK CI FXKT XDEK XPV THE
REDEPVK CI XPAYEPT EYPDK. SXU Y SXEE KC ZCRV XK TC AJXNE X
IXNCMJ CI UCMJ SXGEKTU?" EFYRC DME, TXREK IJCS THE THCMKXPV
XPV CPE PYDBTK

6

Afkryptering



Kode:

X Z A V O I D B Y G E R S P C F H J K L M N Q T U W
A B C D E F G H I J K L M N O P Q R S T U V W X Y Z

Kodetekst:

PCQ VMJYPD LBYK LYSO KBXBJXWXV BXV ZCJPO EYPD KBXBJYUXJ
LBJOO KCPK. CP LBO LBCMXXPV XPV IYJKL PYDBL. QBOP KBO BXV
OPVOV LBO LXRO CI SX'XJMI, KBO JCKO XPV EYKKOV LBO DJCMPV
ZOICJO BYS, KXUYPD: "DJOXL EYPD, ICJ X LBCMXXPV XPV CPO PYDBLK Y BXNO
ZOOP JOACMPLYPD LC UCM LBO IXZROK CI FXKL XDOK XPV LBO RODOPVK CI
XPAYOPL EYPDK. SXU Y SXEO KC ZCRV XK LC AJXNO X IXNCMJ CI UCMJ
SXGOKLU?" OFYRCDMO, LXROK IJCS LBO LBCMXXPV XPV CPO PYDBLK

Klartekst:

Now during this time Shahrazad had borne King Shahriyar three sons. On the thousand and first night, when she had ended the tale of Ma'aruf, she rose and kissed the ground before him, saying: "Great King, for a thousand and one nights I have been recounting to you the fables of past ages and the legends of ancient kings. May I make so bold as to crave a favour of your majesty?" Epilogue, Tales from the Thousand and One Nights

7

Kryptering med hemmelig nøgle



En **hemmelig-nøgle** kode bruger en unik nøgle K til kryptering og afkryptering

Cæsars generaliserede kode bruger modulær addition af hvert tegn (betragtet som et heltal) med nøglen:

$$C[i] = (P[i] + K) \bmod m \quad (\text{kryptering})$$
$$P[i] = (C[i] - K) \bmod m \quad (\text{afkryptering})$$

Mere sikre hemmelig-nøgle krypteringskoder er blevet opfundet de seneste 50 år

Eksempler: DES 3DES IDEA BLOWFISH

Med hemmelig-nøgle kryptering må en særskilt nøgle etableres for ethvert par af deltagere

8

Offentlig-nøgle kryptering



Bob bruger et par af nøgler (K_E, K_D),
gør nøglen K_E offentlig og holder
nøglen K_D privat

Alle og enhver kan bruge den offentlige nøgle K_E til at kryptere en klartekst til Bob
Kun Bob kan afkryptere kodedeksten ved brug af sin private nøgle K_D

Det mest populære krypteringsskema er RSA, opkaldt efter sine opfindere:
Rivest, Shamir og Adleman (1978)
RSA-patentet udløb 2000



9

Talleteoretiske algoritmer



10

Fakta om tal



Primtal p :

p er et heltal

$p \geq 2$

De eneste divisorer for p er 1 og p

Eksempler:

2, 7, 19 er primtal

-3, 1, 6 er ikke primtal

Primtalsopløsning af et positivt heltal n :

$$n = p_1^{e_1} \times p_2^{e_2} \times \dots \times p_k^{e_k}$$

Eksempel:

$$200 = 2^3 \times 5^2$$

Fundamental sætning i aritmetik:
Primtalsopløsningen af et positive heltal er unik

11

Største fælles divisor



Den **største fælles divisor** (GCD) for to positive heltal a og b , betegnet $\gcd(a, b)$, er det største positive heltal, der går op i både a og b

Ovenstående definition kan udvides til vilkårlige heltal

Eksempler:

$$\gcd(18, 30) = 6$$

$$\gcd(0, 20) = 20$$

$$\gcd(-21, 49) = 7$$

To heltal a og b siges at være **indbyrdes primiske**, hvis $\gcd(a, b) = 1$

Eksempel:

15 og 28 er indbyrdes primiske

12

Modulær aritmetik



Modulo-operatoren for et positivt heltal n

$$r = a \bmod n$$

er ækvivalent med

$$a = r + kn$$

og

$$r = a - \lfloor a/n \rfloor n$$

Eksempler:

$$29 \bmod 13 = 3$$

$$13 \bmod 13 = 0$$

$$-1 \bmod 13 = 12$$

$$29 = 3 + 2 \cdot 13$$

$$13 = 0 + 1 \cdot 13$$

$$-1 = 12 + (-1) \cdot 13$$

Modulo og GCD:

$$\gcd(a, b) = \gcd(b, a \bmod b)$$

Eksempel:

$$\gcd(21, 12) = 3$$

$$\gcd(12, 21 \bmod 12) = \gcd(12, 9) = \gcd(9, 3) = \gcd(3, 0) = 3$$

13

Euclids GCD-algoritme



300 f. Kr.

Euclids algoritme til beregning GCD

benytter gentagne gange formlen

$$\gcd(a, b) = \gcd(b, a \bmod b)$$

Eksempel:

$$\gcd(412, 260) = 4$$

Algorithm *EuclidGCD*(a, b)

Input integers a and b

Output $\gcd(a, b)$

if $b = 0$

return a

return *EuclidGCD*($b, a \bmod b$)

| | | | | | | | |
|-----|-----|-----|-----|-----|----|----|---|
| a | 412 | 260 | 152 | 108 | 44 | 20 | 4 |
| b | 260 | 152 | 108 | 44 | 20 | 4 | 0 |

14

Analyse

Algorithm *EuclidGCD*(a, b)
Input integers a and b
Output $\gcd(a, b)$
if $b = 0$
 return a
 return *EuclidGCD*($b, a \bmod b$)

Køretiden er proportional med antallet af rekursive kald af *EuclidGCD*

Lad a_i og b_i være argumenterne til det i 'te rekursive kald

Vi har

$$a_{i+2} = b_{i+1} = a_i \bmod b_i = a_i \bmod a_{i+1} < a_{i+1}$$

Følgen a_1, a_2, \dots, a_n aftager eksponentielt, idet vi kan vise, at

$$a_{i+2} < 1/2 a_i \text{ for } i > 1$$

Tilfælde 1: $a_{i+1} \leq 1/2 a_i$; $a_{i+2} < a_{i+1} \leq 1/2 a_i$

Tilfælde 2: $a_{i+1} > 1/2 a_i$; $a_{i+2} = a_i \bmod a_{i+1} \leq a_i - a_{i+1} < 1/2 a_i$

Derfor er det maksimale antal kald af algoritmen *EuclidGCD*(a, b)
 $1 + 2 \log \max(a, b)$

Algoritmen *EuclidGCD*(a, b) udfører $O(\log \max(a, b))$ aritmetiske operationer

15

Multiplikativ invers (1)



Resterne modulo et positivt heltal n er mængden

$$\mathbb{Z}_n = \{0, 1, 2, \dots, (n-1)\}$$

Lad x og y være to positive elementer i \mathbb{Z}_n , hvor

$$xy \bmod n = 1$$

Vi siger da, at y er **multiplikativ invers** til x i \mathbb{Z}_n , og vi skriver $y = x^{-1}$

Eksempel:

Multiplikative inverse for resterne modulo 11

| | | | | | | | | | | | |
|----------|---|---|---|---|---|---|---|---|---|---|----|
| x | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
| x^{-1} | | 1 | 6 | 4 | 3 | 9 | 2 | 8 | 7 | 5 | 10 |

16

Multiplikativ invers (2)



Sætning:

Et element x i Z_n har en multiplikativ invers, hvis og kun hvis x og n er indbyrdes primiske

Eksempel: Elementerne i Z_{10} , der har en multiplikativ invers, er 1, 3, 5, 7

| x | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
|----------|---|---|---|---|---|---|---|---|---|---|
| x^{-1} | | 1 | | 7 | | | | 3 | | 9 |

Følgesætning:

Hvis p er et primtal, har alle rester i Z_p , undtagen 0, en multiplikativ invers

Sætning:

Euclids GCD-algoritme kan udvides til at beregne den multiplikative inverse til et element x i Z_n , eller afgøre, at en sådan ikke eksisterer

17

Potenser

Lad p være et primtal

Følgen af positive potenser af elementerne i Z_p indeholder gentagende delfølger

Længden af de gentagende delsekvenser og antallet af deres gentagelser er divisorerne til $p - 1$

Eksempel ($p = 7$): $[p - 1 = 1 \cdot 6 = 2 \cdot 3]$

| x | x^2 | x^3 | x^4 | x^5 | x^6 |
|-----|-------|-------|-------|-------|-------|
| 1 | 1 | 1 | 1 | 1 | 1 |
| 2 | 4 | 1 | 2 | 4 | 1 |
| 3 | 2 | 6 | 4 | 5 | 1 |
| 4 | 2 | 1 | 4 | 2 | 1 |
| 5 | 4 | 6 | 2 | 3 | 1 |
| 6 | 1 | 6 | 1 | 6 | 1 |

18

Fermats lille sætning



1601-65

Sætning:

Lad p være et primtal. For enhver rest $x \neq 0$ i Z_p gælder, at $x^{p-1} \bmod p = 1$

Eksempel ($p = 5$):

$$\begin{aligned} 1^4 \bmod 5 &= 1 \\ 2^4 \bmod 5 &= 16 \bmod 5 = 1 \\ 3^4 \bmod 5 &= 81 \bmod 5 = 1 \\ 4^4 \bmod 5 &= 256 \bmod 5 = 1 \end{aligned}$$

Følgesætning:

Lad p være et primtal. For enhver rest $x \neq 0$ i Z_p gælder, at den multiplikative inverse til x er $x^{p-2} \bmod p$

Bevis:

$$x(x^{p-2} \bmod p) \bmod p = xx^{p-2} \bmod p = x^{p-1} \bmod p = 1$$

19

Eulers sætning



1707-83

Den **multiplikative gruppe** for Z_n , betegnet med Z_n^* , er den delmængde af Z_n , der er indbyrdes primiske med n

Euler's **totientfunktion** af n , betegnet med $\phi(n)$, er antallet af elementer i Z_n^*

Eksempel:

$$Z_{10}^* = \{1, 3, 7, 9\} \quad \phi(10) = 4$$

Hvis p er et primtal, har vi

$$Z_p^* = \{1, 2, \dots, (p-1)\} \quad \phi(p) = p-1$$

Sætning:

For ethvert element x i Z_n^* gælder

$$x^{\phi(n)} \bmod n = 1$$

Eksempel ($n = 10$):

$$\begin{aligned} 3^{\phi(10)} \bmod 10 &= 3^4 \bmod 10 = 81 \bmod 10 = 1 \\ 7^{\phi(10)} \bmod 10 &= 7^4 \bmod 10 = 2401 \bmod 10 = 1 \\ 9^{\phi(10)} \bmod 10 &= 9^4 \bmod 10 = 6561 \bmod 10 = 1 \end{aligned}$$

Eulers sætning er en generalisering af Fermats lille sætning

$$x^{-1} \equiv x^{\phi(n)-1} \bmod n$$

20

Kryptosystemet RSA



Kryptosystemet RSA

Eksempel:

Opstilling:

$n = pq$, hvor p og q er primtal
 e er indbyrdes primisk med $\phi(n)$
 $\phi(n) = (p-1)(q-1)$
 d er multiplikativ invers til e i $Z_{\phi(n)}$

Opstilling:

$p=7, q=17$
 $n=7 \cdot 17=119$
 $\phi(n)=6 \cdot 16=96$
 $e=5$
 $d=77 \quad (77 \cdot 5 \bmod 96 = 1)$

Nøgler:

Offentlig nøgle: $K_E=(n, e)$
 Privat nøgle: $K_D=d$

Nøgler:

Offentlig nøgle: **(119, 5)**
 Privat nøgle: **77**

Kryptering:

Klartekst M i Z_n
 Kodetekst $C = M^e \bmod n$

Kryptering:

$M=19$
 $C=19^5 \bmod 119 = 66$

Afkryptering:

$M = C^d \bmod n$

Afkryptering:

$M=66^{77} \bmod 119 = 19$

Komplet RSA eksempel

Opstilling:

$p=5, q=11$
 $n=5 \cdot 11=55$
 $\phi(n)=4 \cdot 10=40$
 $e=3$
 $d=27 \quad (3 \cdot 27=81=2 \cdot 40+1)$

Kryptering:

$C=M^3 \bmod 55$

Afkryptering:

$M=C^{27} \bmod 55$

| | | | | | | | | | | | | | | | | | | |
|-----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|
| M | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 |
| C | 1 | 8 | 27 | 9 | 15 | 51 | 13 | 17 | 14 | 10 | 11 | 23 | 52 | 49 | 20 | 26 | 18 | 2 |
| M | 19 | 20 | 21 | 22 | 23 | 24 | 25 | 26 | 27 | 28 | 29 | 30 | 31 | 32 | 33 | 34 | 35 | 36 |
| C | 39 | 25 | 21 | 33 | 12 | 19 | 5 | 31 | 48 | 7 | 24 | 50 | 36 | 43 | 22 | 34 | 30 | 16 |
| M | 37 | 38 | 39 | 40 | 41 | 42 | 43 | 44 | 45 | 46 | 47 | 48 | 49 | 50 | 51 | 52 | 53 | 54 |
| C | 53 | 37 | 29 | 35 | 6 | 3 | 32 | 44 | 45 | 41 | 38 | 42 | 4 | 40 | 46 | 28 | 47 | 54 |

Sikkerhed



Sikkerheden af RSA-kryptosystemet er baseret på den almindelige antagelse om, at faktorisering af store tal er beregningsmæssigt vanskelig

I 1999 blev et 512-cifret tal faktoriseret på 4 måneder ved brug af følgende computere:

| | |
|-----|------------------------|
| 160 | 175-400 MHz SGI og Sun |
| 8 | 250 MHz SGI Origin |
| 120 | 300-450 MHz Pentium II |
| 4 | 500 MHz Digital/Compaq |

Den bedste kendte algoritme bruger tid, der er eksponentiel i antallet af bits i det tal, der skal faktoriseres

Skønnet ressourceforbrug til faktorisering af et tal inden for 1 år:

The RSA Challenge, der er sponsoreret af RSA Security, tilbyder pengepræmier for faktorisering af givne store tal

| Bit | PC'er | Lager |
|-------|----------------------|-------|
| 430 | 1 | 128MB |
| 760 | 215,000 | 4GB |
| 1,020 | 342×10^6 | 170GB |
| 1,620 | 1.6×10^{15} | 120TB |

I april 2002 varierede præmierne fra 10,000\$ (576 bit) til 200,000\$ (2048 bit)

Korrekthed



Vi viser korrektheden for RSA i det tilfælde, at n ikke går op i klarteksten M

Vi viser, at der i dette tilfælde gælder, at $(M^e)^d \bmod n = M$

Da $ed \bmod \phi(n) = 1$, er der et heltal k , for hvilket det gælder, at $ed = k\phi(n) + 1$

Da n ikke går op i M , får vi ved hjælp af Eulers sætning, at $M^{\phi(n)} \bmod n = 1$

Vi får således

$$\begin{aligned} (M^e)^d \bmod n &= \\ M^{ed} \bmod n &= \\ M^{k\phi(n)+1} \bmod n &= \\ MM^{k\phi(n)} \bmod n &= \\ M (M^{\phi(n)})^k \bmod n &= \\ M (M^{\phi(n)} \bmod n)^k \bmod n &= \\ M (1)^k \bmod n &= \\ M \bmod n &= \\ M \end{aligned}$$

Beviset for korrekthed i det tilfælde, hvor n går op i klarteksten M , kan ses i lærebogen

25

Algoritmiske problemer



Implementering af kryptosystemet RSA kræver forskellige algoritmer

• Generelt

Repræsentation af heltal af vilkårlig længde og aritmetiske operationer på disse

• Kryptering

Modulær potensopløftning

• Afkryptering

Modulær potensopløftning

• Opstilling

Generering af tilfældige tal med et givet antal bit (for at generere kandidater for p og q)

Primtalitetstest (for at påvise, at kandidater for p og q er primtal)

Beregning af GCD (for at påvise, at e og $\phi(n)$ er indbyrdes primiske)

Beregning af den multiplikative inverse (for at beregne d ud fra e)

26

Modulær potensopløftning



Brug af gentagen kvadrering øger beregningshastigheden for en modulær potens $a^p \bmod n$

Skriv eksponenten p på binær form:

$$p = p_{b-1}p_{b-2} \dots p_1p_0$$

Start med

$$Q_1 = a^{p_{b-1}} \bmod n$$

Beregn for $i = 2, 3, \dots, b$

$$Q_i = ((Q_{i-1})^2 \bmod n) a^{p_{b-i}} \bmod n$$

Vi har, at

$$Q_b = a^p \bmod n$$

Denne algoritme udfører $O(\log p)$ aritmetiske operationer

Eksempel:

$$3^{18} \bmod 19 \quad (18 = 2^4 + 2^1 = 0x10010)$$

$$Q_1 = 3^1 \bmod 19 = 3$$

$$Q_2 = (3^2 \bmod 19)3^0 \bmod 19 = 9$$

$$Q_3 = (9^2 \bmod 19)3^0 \bmod 19 = 81 \bmod 19 = 5$$

$$Q_4 = (5^2 \bmod 19)3^1 \bmod 19 = (25 \bmod 19)3 \bmod 19 = 18 \bmod 19 = 18$$

$$Q_5 = (18^2 \bmod 19)3^0 \bmod 19 = (324 \bmod 19) \bmod 19 = (17 \cdot 19 + 1) \bmod 19 = 1$$

| | | | | | |
|---------------|---|---|---|----|---|
| p_{5-i} | 1 | 0 | 0 | 1 | 0 |
| $3^{p_{5-i}}$ | 3 | 1 | 1 | 3 | 1 |
| Q_i | 3 | 9 | 5 | 18 | 1 |

27

Modulær potensopløftning i Java (rekursiv udgave)



Hvis p er lige, er $a^p = (a \cdot a)^{p/2}$

Hvis p er ulige, er $a^p = a \cdot a^{p-1}$

```
long power(long a, long p, long n) {
    if (p == 0)
        return 1;
    long value = power((a * a) % n, p / 2, n);
    if (p % 2 == 1)
        value = (a * value) % n;
    return value;
}
```

28

Modulær potensopløftning i Java (iterativ udgave)



```

long power(long a, long p, long n) {
    long value = 1;
    while (p > 0) {
        if (p % 2 == 1)
            value = (value * a) % n;
        a = (a * a) % n;
        p /= 2;
    }
    return value;
}
    
```

29

Modulær invers



Sætning:

Lad der være give to positive heltal, a og b , og lad d være det mindste positive heltal, således at

$$d = ia + jb$$

for heltal i og j .

Der gælder da, at

$$d = \gcd(a, b)$$

Givet to positive heltal, a og b , beregner den udvidede Euclids algoritme et trippel (d, i, j) , hvor

$$d = \gcd(a, b)$$

$$d = ia + jb$$

For at påvise eksistensen af, samt beregne den multiplikative inverse til $x \in \mathbb{Z}_n$, udføres den udvidede Euclids algoritme på inputparret (x, n)

Lad (d, i, j) være det trippel, der blev returneret

$$d = ix + jn$$

Tilfælde 1: $d=1$

i er den inverse til x i \mathbb{Z}_n

Tilfælde 2: $d > 1$

x har intet inverst element i \mathbb{Z}_n

Eksempel:

$$a=21$$

$$b=15$$

$$d=3$$

$$i=3, j=-4$$

$$3 = 3 \cdot 21 + (-4) \cdot 15 =$$

$$63 - 60 = 3$$

30

Den udvidede Euclids algoritme

Algorithm *ExtendedEuclidGCD*(a, b)

Input Nonnegative integers a and b

Output Triplet of integers (d, i, j) such that

$$d = \gcd(a, b) = ia + jb$$

if $b = 0$

return $(a, 1, 0)$

$(d, i, j) \leftarrow \text{ExtendedEuclidGCD}(b, a \bmod b)$

return $(d, j, i - j \lfloor a/b \rfloor)$

Bevís ved induktion:

Basistilfældet: $b = 0$:

$$a = \gcd(a, 0) = 1a + 0b$$

Induktionshypotesen:

Antag, at

$$(d, i, j) = \text{ExtendedEuclid}(b, a \bmod b)$$

er korrekt.

Vi har da, at

$$d = \gcd(b, a \bmod b) = ib + j(a \bmod b) =$$

$$ib + j(a - \lfloor a/b \rfloor b) =$$

$$ja + (i - j \lfloor a/b \rfloor)b$$

Da desuden $\gcd(b, a \bmod b) = \gcd(a, b)$, må *ExtendedEuclid*(a, b) være korrekt

| | | | | | | | |
|-------|-----|-----|-----|-----|----|----|---|
| a | 412 | 260 | 152 | 108 | 44 | 20 | 4 |
| b | 260 | 152 | 108 | 44 | 20 | 4 | 0 |
| a/b | 1 | 1 | 1 | 2 | 2 | 5 | |
| i | 12 | -7 | 5 | -2 | 1 | 0 | 1 |
| j | -19 | 12 | -7 | 5 | -2 | 1 | 0 |

31

Pseudoprimalitetstest

| |
|---|
| p |
| 2 3 5 7 |
| 11 13 17 |
| 19 23 29 31 |
| 37 41 43 47 53 |
| 59 61 67 71 73 79 |
| 83 89 97 101 103 107 |
| 113 127 131 137 139 149 |
| 151 157 163 167 173 179 181 |
| 191 193 197 199 211 223 227 229 |
| 233 239 241 251 257 263 269 271 |
| 277 281 283 293 307 311 313 317 |
| 331 337 347 349 353 359 367 373 379 383 |
| 389 397 401 409 419 421 431 433 439 443 |
| 449 457 461 463 467 479 487 491 499 503 |
| 509 517 521 523 529 533 541 547 557 563 |
| 569 577 581 587 593 599 601 607 613 617 |
| 619 627 631 637 641 643 647 653 659 661 |
| 667 673 677 683 687 691 697 701 709 713 |
| 719 727 733 739 743 751 757 761 769 773 |
| 779 787 791 797 803 809 811 817 821 827 |
| 829 833 839 847 853 857 863 869 877 881 |
| 883 887 893 899 907 911 913 919 927 931 |
| 937 941 947 953 959 967 971 977 983 989 |
| 991 997 1009 1013 1019 1021 1027 1031 1033 1039 |
| 1043 1047 1051 1057 1061 1063 1067 1069 1073 1077 |
| 1081 1087 1091 1093 1097 1103 1107 1111 1117 1121 |
| 1123 1127 1133 1137 1141 1147 1151 1153 1157 1163 |
| 1169 1171 1177 1181 1183 1187 1193 1197 1201 1207 |
| 1211 1213 1217 1223 1229 1231 1237 1241 1243 1247 |
| 1249 1253 1259 1261 1267 1271 1273 1277 1283 1287 |
| 1291 1297 1301 1303 1307 1313 1317 1321 1327 1331 |
| 1333 1337 1343 1347 1351 1357 1361 1363 1367 1373 |
| 1377 1381 1387 1391 1397 1403 1407 1411 1413 1417 |
| 1421 1423 1427 1433 1437 1441 1443 1447 1453 1457 |
| 1461 1463 1467 1471 1477 1481 1483 1487 1493 1497 |
| 1501 1507 1511 1513 1517 1523 1527 1531 1537 1541 |
| 1543 1547 1553 1557 1561 1563 1567 1573 1577 1581 |
| 1583 1587 1591 1597 1601 1603 1607 1613 1617 1621 |
| 1623 1627 1631 1633 1637 1643 1647 1651 1653 1657 |
| 1661 1663 1667 1671 1673 1677 1681 1683 1687 1693 |
| 1697 1701 1703 1707 1711 1713 1717 1723 1727 1731 |
| 1733 1737 1743 1747 1751 1753 1757 1763 1767 1771 |
| 1773 1777 1781 1783 1787 1793 1797 1801 1803 1807 |
| 1811 1813 1817 1823 1827 1831 1833 1837 1843 1847 |
| 1849 1853 1857 1861 1863 1867 1871 1873 1877 1883 |
| 1887 1891 1893 1897 1901 1903 1907 1913 1917 1921 |
| 1923 1927 1931 1933 1937 1943 1947 1951 1953 1957 |
| 1961 1963 1967 1971 1973 1977 1981 1983 1987 1993 |
| 1997 2003 2009 2011 2013 2017 2021 2023 2027 2029 |
| 2033 2039 2041 2043 2047 2053 2057 2063 2067 2071 |
| 2073 2077 2081 2083 2087 2093 2097 2101 2103 2107 |
| 2111 2113 2117 2123 2127 2131 2133 2137 2143 2147 |
| 2149 2153 2157 2161 2163 2167 2171 2173 2177 2183 |
| 2187 2191 2193 2197 2201 2203 2207 2213 2217 2221 |
| 2223 2227 2231 2233 2237 2243 2247 2251 2253 2257 |
| 2261 2263 2267 2271 2273 2277 2281 2283 2287 2293 |
| 2297 2301 2303 2307 2311 2313 2317 2323 2327 2331 |
| 2333 2337 2343 2347 2351 2353 2357 2363 2367 2371 |
| 2373 2377 2381 2383 2387 2393 2397 2401 2403 2407 |
| 2411 2413 2417 2423 2427 2431 2433 2437 2443 2447 |
| 2449 2453 2457 2461 2463 2467 2471 2473 2477 2483 |
| 2487 2491 2493 2497 2501 2503 2507 2513 2517 2521 |
| 2523 2527 2531 2533 2537 2543 2547 2551 2553 2557 |
| 2561 2563 2567 2571 2573 2577 2581 2583 2587 2593 |
| 2597 2601 2603 2607 2611 2613 2617 2623 2627 2631 |
| 2633 2637 2643 2647 2651 2653 2657 2663 2667 2671 |
| 2673 2677 2681 2683 2687 2693 2697 2701 2703 2707 |
| 2711 2713 2717 2723 2727 2731 2733 2737 2743 2747 |
| 2749 2753 2757 2761 2763 2767 2771 2773 2777 2783 |
| 2787 2791 2793 2797 2801 2803 2807 2813 2817 2821 |
| 2823 2827 2831 2833 2837 2843 2847 2851 2853 2857 |
| 2861 2863 2867 2871 2873 2877 2881 2883 2887 2893 |
| 2897 2901 2903 2907 2911 2913 2917 2923 2927 2931 |
| 2933 2937 2943 2947 2951 2953 2957 2963 2967 2971 |
| 2973 2977 2981 2983 2987 2993 2997 3001 3003 3007 |
| 3011 3013 3017 3023 3027 3031 3033 3037 3043 3047 |
| 3049 3053 3057 3061 3063 3067 3071 3073 3077 3083 |
| 3087 3091 3093 3097 3101 3103 3107 3113 3117 3121 |
| 3123 3127 3131 3133 3137 3143 3147 3151 3153 3157 |
| 3161 3163 3167 3171 3173 3177 3181 3183 3187 3193 |
| 3197 3201 3203 3207 3211 3213 3217 3223 3227 3231 |
| 3233 3237 3243 3247 3251 3253 3257 3263 3267 3271 |
| 3273 3277 3281 3283 3287 3293 3297 3301 3303 3307 |
| 3311 3313 3317 3323 3327 3331 3333 3337 3343 3347 |
| 3349 3353 3357 3361 3363 3367 3371 3373 3377 3383 |
| 3387 3391 3393 3397 3401 3403 3407 3413 3417 3421 |
| 3423 3427 3431 3433 3437 3443 3447 3451 3453 3457 |
| 3461 3463 3467 3471 3473 3477 3481 3483 3487 3493 |
| 3497 3501 3503 3507 3511 3513 3517 3523 3527 3531 |
| 3533 3537 3543 3547 3551 3553 3557 3563 3567 3571 |
| 3573 3577 3581 3583 3587 3593 3597 3601 3603 3607 |
| 3611 3613 3617 3623 3627 3631 3633 3637 3643 3647 |
| 3649 3653 3657 3661 3663 3667 3671 3673 3677 3683 |
| 3687 3691 3693 3697 3701 3703 3707 3713 3717 3721 |
| 3723 3727 3731 3733 3737 3743 3747 3751 3753 3757 |
| 3761 3763 3767 3771 3773 3777 3781 3783 3787 3793 |
| 3797 3801 3803 3807 3811 3813 3817 3823 3827 3831 |
| 3833 3837 3843 3847 3851 3853 3857 3863 3867 3871 |
| 3873 3877 3881 3883 3887 3893 3897 3901 3903 3907 |
| 3911 3913 3917 3923 3927 3931 3933 3937 3943 3947 |
| 3949 3953 3957 3961 3963 3967 3971 3973 3977 3983 |
| 3987 3991 3993 3997 4001 4003 4007 4013 4017 4021 |
| 4023 4027 4031 4033 4037 4043 4047 4051 4053 4057 |
| 4061 4063 4067 4071 4073 4077 4081 4083 4087 4093 |
| 4097 4101 4103 4107 4111 4113 4117 4123 4127 4131 |
| 4133 4137 4143 4147 4151 4153 4157 4163 4167 4171 |
| 4173 4177 4181 4183 4187 4193 4197 4201 4203 4207 |
| 4211 4213 4217 4223 4227 4231 4233 4237 4243 4247 |
| 4249 4253 4257 4261 4263 4267 4271 4273 4277 4283 |
| 4287 4291 4293 4297 4301 4303 4307 4313 4317 4321 |
| 4323 4327 4331 4333 4337 4343 4347 4351 4353 4357 |
| 4361 4363 4367 4371 4373 4377 4381 4383 4387 4393 |
| 4397 4401 4403 4407 4411 4413 4417 4423 4427 4431 |
| 4433 4437 4443 4447 4451 4453 4457 4463 4467 4471 |
| 4473 4477 4481 4483 4487 4493 4497 4501 4503 4507 |
| 4511 4513 4517 4523 4527 4531 4533 4537 4543 4547 |
| 4549 4553 4557 4561 4563 4567 4571 4573 4577 4583 |
| 4587 4591 4593 4597 4601 4603 4607 4613 4617 4621 |
| 4623 4627 4631 4633 4637 4643 4647 4651 4653 4657 |
| 4661 4663 4667 4671 4673 4677 4681 4683 4687 4693 |
| 4697 4701 4703 4707 4711 4713 4717 4723 4727 4731 |
| 4733 4737 4743 4747 4751 4753 4757 4763 4767 4771 |
| 4773 4777 4781 4783 4787 4793 4797 4801 4803 4807 |
| 4811 4813 4817 4823 4827 4831 4833 4837 4843 4847 |
| 4849 4853 4857 4861 4863 4867 4871 4873 4877 4883 |
| 4887 4891 4893 4897 4901 4903 4907 4913 4917 4921 |
| 4923 4927 4931 4933 4937 4943 4947 4951 4953 4957 |
| 4961 4963 4967 4971 4973 4977 4981 4983 4987 4993 |
| 4997 5001 5003 5007 5011 5013 5017 5023 5027 5031 |
| 5033 5037 5043 5047 5051 5053 5057 5063 5067 5071 |
| 5073 5077 5081 5083 5087 5093 5097 5101 5103 5107 |
| 5111 5113 5117 5123 5127 5131 5133 5137 5143 5147 |
| 5149 5153 5157 5161 5163 5167 5171 5173 5177 5183 |
| 5187 5191 5193 5197 5201 5203 5207 5213 5217 5221 |
| 5223 5227 5231 5233 5237 5243 5247 5251 5253 5257 |
| 5261 5263 5267 5271 5273 5277 5281 5283 5287 5293 |
| 5297 5301 5303 5307 5311 5313 5317 5323 5327 5331 |
| 5333 5337 5343 5347 5351 5353 5357 5363 5367 5371 |
| 5373 5377 5381 5383 5387 5393 5397 5401 5403 5407 |
| 5411 5413 5417 5423 5427 5431 5433 5437 5443 5447 |
| 5449 5453 5457 5461 5463 5467 5471 5473 5477 5483 |
| 5487 5491 5493 5497 5501 5503 5507 5513 5517 5521 |
| 5523 5527 5531 5533 5537 5543 5547 5551 5553 5557 |
| 5561 5563 5567 5571 5573 5577 5581 5583 5587 5593 |
| 5597 5601 5603 5607 5611 5613 5617 5623 5627 5631 |
| 5633 5637 5643 5647 5651 5653 5657 5663 5667 5671 |
| 5673 5677 5681 5683 5687 5693 5697 5701 5703 5707 |
| 5711 5713 5717 5723 5727 5731 5733 5737 5743 5747 |
| 5749 5753 5757 5761 5763 5767 5771 5773 5777 5783 |
| 5787 5791 5793 5797 5801 5803 5807 5813 5817 5821 |
| 5823 5827 5831 5833 5837 5843 5847 5851 5853 5857 |
| 5861 5863 5867 5871 5873 5877 5881 5883 5887 5893 |

Randomiseret primalitetstest

Vidnefunktion for sammensathed, $witness(x, n)$, med fejlsandsynlighed q for et heltal x :

- Tilfælde 1: n er et primtal
 $witness(x, n) = false$ (altid)
- Tilfælde 2: n er sammensat
 $witness(x, n) = false$ med sandsynlighed $q < 1$

Algoritmen *RandPrimeTest* afgør, om n er et primtal ved gentagne gange at evaluere $witness(x, n)$

En variant af base- x pseudoprimalitet giver en god vidnefunktion for sammensathed (Rabin-Miller's algoritme)

Algorithm *RandPrimeTest*(n, k)

Input integer n , confidence parameter k and composite witness function $witness(x, n)$ with error probability q

Output an indication of whether n is composite or prime with error probability 2^{-k}

$t \leftarrow k / \log_2(1/q) \quad \{q^t = 2^{-k}\}$

for $i \leftarrow 1$ **to** t

$x \leftarrow random()$

if $witness(x, n) = true$

return " n is composite"

return " n is prime"

33

```
import java.math.BigInteger;
import java.util.Random;
import java.io.*;
```

```
public class RSA {
    public static void main(String[] args) throws IOException {
        int bitLength = 1024;
        Random rnd = new Random();
        BigInteger p = BigInteger.probablePrime(bitLength, rnd);
        BigInteger q = BigInteger.probablePrime(bitLength, rnd);
        BigInteger n = p.multiply(q);
        BigInteger phi = p.subtract(BigInteger.ONE).multiply(
            q.subtract(BigInteger.ONE));
        BigInteger e;
        do
            e = new BigInteger(2 * bitLength, rnd);
        while (e.compareTo(phi) >= 0 || !e.gcd(phi).equals(BigInteger.ONE));
        BigInteger d = e.modInverse(phi);

        BufferedReader in = new BufferedReader(
            new InputStreamReader(System.in));

        while (true) {
            BigInteger ciphertext =
                new BigInteger(in.readLine().getBytes()).modPow(e, n);
            String plaintext =
                new String(ciphertext.modPow(d, n).toByteArray());
            System.out.println(plaintext);
        }
    }
}
```



34

Informationssikkerhed



35

Digital underskrift



En **digital underskrift** er streng S forbundet med en meddelelse M og dens forfatter A , som har følgende egenskaber:

- *Uafviselighed*: S identificerer entydigt forfatteren A af M og beviser, at A faktisk underskrev M
- *Integritet*: S godtgør, at M ikke er blevet ændret

Et skema for digital underskrift giver en algoritme til

- (1) underskrivning af en meddelelse (foretages af forfatteren)
- (2) verifikation af underskriften (foretages af læseren)

Underskrift: Alice (forfatteren) bestemmer $S = decrypt(K_D, M)$ ved brug af sin private nøgle K_D og sender parret (M, S) til Bob

Verifikation: Bob (læseren) bestemmer $M' = encrypt(K_E, S)$ ved brug af Alice's offentlige nøgle K_E og checker, at $M' = M$

36

Digital underskrift med RSA

Opstilling:

$n=pq$, hvor p og q er primtal
 e er indbyrdes primisk med
 $\phi(n)=(p-1)(q-1)$
 d inverse af e i $Z_{\phi(n)}$

Nøgler:

Offentlig nøgle: $K_E=(n, e)$
 Privat nøgle: $K_D=d$

Underskrift:

Klartekst M i Z_n
 Underskrift $S = M^d \bmod n$

Verifikation:

Check, at $M = S^e \bmod n$

Opstilling:

$p=5, q=11$
 $n=5 \cdot 11=55$
 $\phi(n)=4 \cdot 10=40$
 $e=3$
 $d=27(3 \cdot 27=81=2 \cdot 40+1)$

Nøgler:

Offentlig nøgle: $K_E=(55, 3)$
 Privat nøgle: $K_D=27$

Underskrift:

$M=51$
 $S=51^{27} \bmod 55=6$

Verifikation:

$M=6^3 \bmod 55=216 \bmod 55=51$

37

Envejs-hashfunktion



En **envejs-hashfunktion** er en funktion H med følgende egenskaber:

- H afbilder en streng M af vilkårlig længde til et heltal $f=H(M)$ med et fast antal bit, kaldet **fingeraftrykket** af M
- H kan beregnes effektivt
- Givet et heltal f er det beregningsmæssigt vanskeligt at finde en streng M , således at $H(M)=f$
- Givet en streng M er det beregningsmæssigt vanskeligt at finde en anden streng M' , således at $H(M)=H(M')$ (*kollisionsresistent*)
- Det er beregningsmæssigt vanskeligt at finde to strenge, M og M' , således at $H(M)=H(M')$ (*stærkt kollisionsresistent*)

To udbredte anvendte envejs-hashfunktioner:

MD5 (Message Digest 5, 1992), som bruger et 128-bit (16 byte) fingeraftryk
 SHA-1 (Secure Hash Algorithm 1, 1995), som bruger et 160-bit (20 byte) fingeraftryk

38

Fingeraftryk til digital underskrift



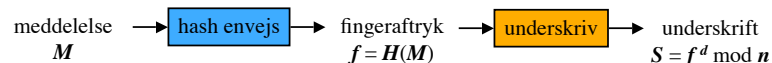
I RSA's skema til digital underskrift med modulo n skal meddelelsen, der skal underskrives, være et heltal i Z_n , d.v.s. meddelelsen kan højst være på log n bit

For at overvinde denne begrænsning kan vi bruge et **fingeraftryk** $f=H(M)$ af meddelelsen i stedet for meddelelsen selv, hvor H er en envejs-hashfunktion

Alice beregner først $f=H(M)$ og dernæst underskriften S af f

Bob beregner først $f=H(M)$ og verificerer derefter S

Hvis H er kollisionsresistent, er det beregningsmæssigt vanskeligt at ændre meddelelsen og samtidig bevare underskriften for fingeraftrykket $f=H(M)$



39

Møntkast over nettet



Alice og Bob vil kaste en mønt tilfældigt i en kommunikation over Internettet

Følgende protokol baseret på en envejs-hashfunktion H sikrer imod snyd:

- Alice vælger et tilfældigt heltal x , beregner fingeraftrykket $f=H(x)$ og sender f til Bob
- Bob sender Alice sit gæt af, om x er lige eller ulige
- Alice bekendtgør resultatet af møntkastet: "ja", hvis Bob har gættet rigtigt; ellers "nej". Hun sender også x til Bob
- Bob verificerer, at Alice ikke har snydt, d.v.s. at $f=H(x)$

Hvis H er stærkt kollisionsresistent, er det beregningsmæssigt vanskeligt for Alice at snyde

40

Certifikater



Offentlig-nøgle kryptografi er baseret på enhver deltagers viden om de offentlige nøgler for de andre deltagere

Det er kompliceret at distribuere de offentlige nøgler til alle deltagere på en sikker måde

Et **certifikat** er en meddelelse af typen (*navn, offentlig nøgle*), der er underskrevet af en tredje part

En person eller virksomhed, som alle stoler på, en såkaldt **certificeringsautoritet** (CA), udsteder til hver deltager et certifikat (*Name, K_E*), der autoritativt binder deltagerne til deres offentlige nøgler

Kun CA's offentlige nøgler behøver at blive distribueret

Inden en krypteret meddelelse sendes til Bob, eller en meddelelse underskrevet af Bob verificeres, bestemmer Alice Bob's offentlige nøgle ved hjælp af Bob's certifikat, (Bob, K_E)

41

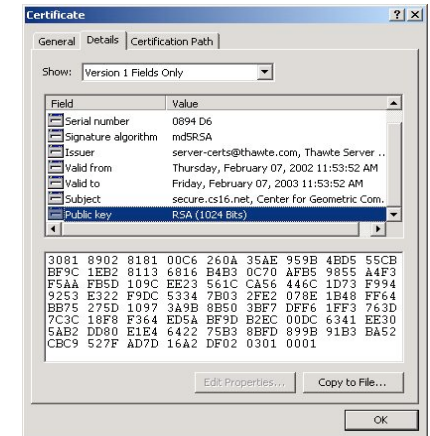
Web server certifikater

Et **web server certifikat** bruges til at autentificere den offentlige nøgle for en web server

Felter i et web server certifikat

- Serienummer
- Hash- og underskriftskema (f.eks. MD5 og RSA)
- Udgiver (certificeringsautoritet)
- Gyldighedsperiode (fra, til)
- Subjekt (URL og organisation)
- Offentlig nøgle

Protokollen SSL (Secure Socket Layer) bruger web server certificering for at tilbyde kryptering og autentificering ved en sikker web forbindelse (http)



42

Ophør af certifikater

Under visse omstændigheder er det nødvendigt at ophæve et certifikat inden dets udløbsdato

- Den private nøgle er blevet afsløret
- Certifikatet blev fejlagtigt udgivet af CA

Certifikatophørsliste

- Tidsstempelt liste over alle ikke-udløbne certifikater, der er blevet ophævet af CA
- Publiceres periodevis og underskrives af CA

Når man præsenteres for et certifikat, skal man

- Verificere CA's underskrift på certifikatet
- Checke at certifikatet ikke er blevet ophævet ved at søge i den senest tilgængelige certifikatophørsliste

Web-browsere checker normalt ikke status for en web servers certifikat, hvilket udgør en sikkerhedsrisiko

43