# RSA-kryptosystemet

Nørre Gymnasium
SRO 2.e

# 1 Kort om kryptologi

Vi vil begynde med at skabe os en forståelse for, hvad kryptologi det er. Dette vil vi gøre gennem tænkte eksempler, hvor vi løbende introducerer terminologi. Vi vil starte med at introducere et simpelt *kryptosystem*, der beskriver idéen bag kryptologi, men har store sikkerhedsbrister. Dette er det såkaldte *Cæsar-chiffer*, som efter sigende skulle være udtænkt af Julius Cæsar. Et eksempel på et mere berømt kryptosystem er Enigma, der under anden verdenskrig blev brugt af Nazityskland til at sende hemmelige beskeder indbyrdes.

### Terminologi

Begrebet kryptologi dækker overordnet over to begreber: Det at skabe kryptosystemer, som Cæsar-chifferet eller Enigma, som vi kalder for *Kryptografi*, og det at bryde eller knække kryptosystemer, som vi kalder for *Kryptoanalyse*. Et *chiffer* er en algoritme, der kan bruges til *kryptering* og *dekryptering*. Kryptering består i at gøre en tekst eller lignende ulæseligt for uvedkommende og dekryptering er den omvendte proces. Vi kalder den ukrypterede tekst for *klarteksten*, og den krypterede tekst for *chifferteksten*. Klarteksten og chifferteksten består begge af *bogstaver* fra et *alfabet*. Bemærk, at et alfabet ikke nødvendigvis behøver at bestå af A-Å, men kan også består af tal, figurer eller alt muligt andet. Klartekst-alfabetet og chiffertekst-alfabetet behøver heller ikke at være ens. Vi betegner mængden af alle klartekster med $\mathscr{P}$ og mængden af alle chiffertekster med $\mathscr{C}$. Til slut har vi en *krypteringsfunktion* $e : \mathscr{P} \to \mathscr{C}$, der krypterer klartekster til chiffertekster, og en dekrypteringsfunktion $d : \mathscr{C} \to \mathscr{P}$, der dekrypterer chiffertekster til klartekster. Vi ønsker selvfølgelig, at for alle klartekster $m \in \mathscr{P}$, at der gælder, at

$$d(e(m)) = m,$$

altså at kryptering af en besked og en efterfølgende dekryptering giver os den oprindelige besked tilbage. Princippet bag et kryptosystem kan ses af diagrammet i Fig. 1,

$$\mathscr{P} \xrightarrow{e} \mathscr{C} \xrightarrow{d} \mathscr{P}$$

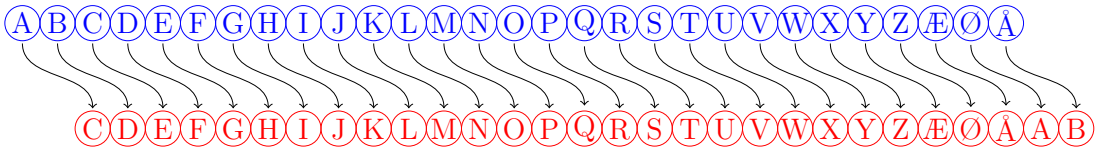
Figur 1: Kryptosystem

## Cæsar-chiffer

Om Cæsar rent faktisk selv opfandt Cæsar-chifferet er svært at afgøre, men det er forståeligt, hvorfor de i Romerriget kan have haft brug for kryptografi af forskellig art. Hvis senatet og kejseren i Rom blev enige om at ændre krigsstrategi i

udkanten af riget, så var en besked lang tid undervejs, og romerne var selvfølgelig ikke interesserede i, at beskeden skulle falde i deres modstanderes hænder. At gøre denne besked ulæselig for uvedkommende vil derfor have været et kærkomment redskab.

Cæsar-chifferet er ret simpelt. Alfabetet er vores sædvanlige alfabet {A,B, . . . , Å}, og både $\mathscr{P}$ og $\mathscr{C}$ består af alle mulige bogstavkombinationer. Krypteringsfunktionen $e_k : \mathscr{P} \to \mathscr{C}$ består i at forskyde alle bogstaver $k$ pladser til højre i alfabetet, og vi kalder $k$ for *krypteringsnøglen*. I tilfældet at $k = 2$, så vil A blive til C, B til D osv. Dekrypteringsfunktionen $d_k$ forskyder tilsvarende alle bogstaver $k$ pladser til venstre i alfabetet. Fig. 2 beskriver kryptering for dette kryptosystem. Dekryptering foregår tilsvarende ved at gå mod pilenes retning i figuren.




Figur 2: kryptering med Cæsar-chiffer hvor $k = 2$

**Eksempel 1.1.** Vi modtager chifferteksten $c$ =CPFGP G GNUMGT OCV, og vi ved, at den er krypteret med krypteringsfunktionen $e_2$. Derfor har vi, at

$$\text{CPFGP G GNUMGT OCV} = e_2(m),$$

hvor $m$ er klarteksten. For at finde $m$ bestemmer vi

$$m = d_2(\text{CPFGP G GNUMGT OCV}).$$

Dette gøres ved at gå baglens i tabellen i Fig 2, og vi får, at klarteksten er

$$m = \text{ANDEN E ELSKER MAT.}$$

Med dette kryptosystem følger nogle ret store problemer. Hvis chifferteksten $c$ opsnappes af en tredjepart og tredjeparten ved, at beskeden er krypteret med et Cæsar-chiffer, så kan tredjeparten blot prøve alle 29 nøgler for at dekryptere. Dette kan højst tage 29 gange så lang tid som at dekryptere en besked, hvis man ikke kender nøglen, som hvis man kender den, hvilket er alt for kort tid.

Alternativt kunne man parre bogstaverne på andre måder. Dette ville øge sikkerheden mod et angreb, der består i at afprøve alle nøgler, da man kan parre bogstaverne på

$$29! = 8841761993739701954543616000000$$

måder. Men så kan man udnytte strukturen i en gennemsnitlig tekst. På dansk er det hyppigst brugte bogstav E, der optræder som omtrent 15% af alle bogstaver.

Derfor vil det hyppigste bogstav i en chiffertekst formentlig tilsvare E i klarteksten. Denne analyse kan så gennemgåes med det næsthyppigste bogstav, tredjehyppigste osv. indtil chifferteksten kan dechifreres til klarteksten.

Et kryptosystem som Cæsar-chifferet kaldes for et symmetrisk kryptosystem, da begge parter skal blive enige om en krypteringsnøgle $k$.

## Public-key-kryptosystemer

RSA-kryptosystemet er et såkaldt public-key-kryptosystem (eller asymmetrisk kryptosystem). Dette betyder, at krypteringsnøglen, der skal bruges til at kryptere klarteksten til chifferteksten er offentligt tilgængelig i den forstand at vi ikke gør noget for at skjule den for tredjeparter. Vi vil gennemgå princippet bag public-key-kryptosystemer uden at komme ind på, hvad RSA gør mere specifikt.

Vi betragter en situation, hvor en person Alice ønsker at sende en krypteret besked til en anden person Bob. Et public-key-system består af følgende skridt:

i) Bob bestemmer to nøgler: en offentlig nøgle $k_o$ og en privat nøgle $k_p$. Den offentlige nøgle $k_o$ deler han med Alice.

ii) Alice bruger nøglen $k_o$ til at kryptere klarteksten $m$ med krypteringsfunktionen $e_k$. Hun bestemmer altså chifferteksten

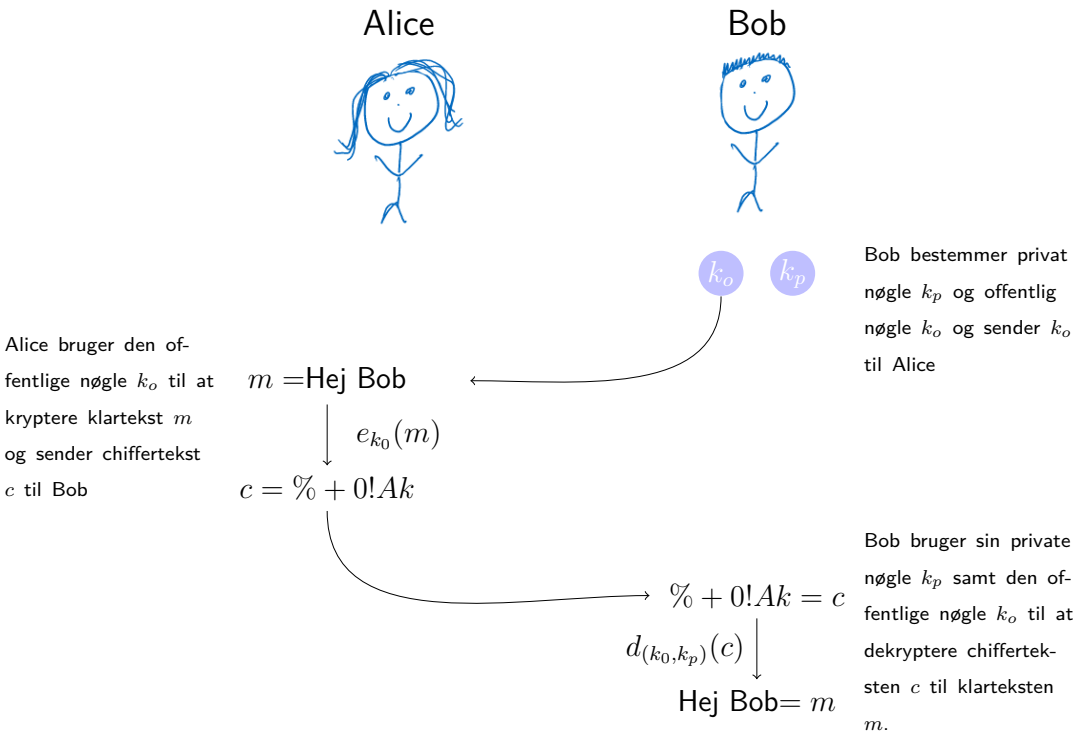
$$c = e_{k_o}(m),$$

og sender $c$ til Bob.

iii) Bob bruger sin private nøgle $k_p$ sammen med den offentlige nøgle $k_o$ til at bestemme dekrypteringsnøglen $k$. Han dekrypterer nu chifferteksten $c$:

$$m = d_k(c),$$

og Bob har modtaget Alices besked.




Figur 3: Princip i public-key-kryptosystem

Et godt public-key-kryptosystem skal opfylde, at det er let at beregne $c = e_{k_o}(m)$, men meget svært at beregne $m$ hvis man kun kender $c$ og $k_o$. Det skal desuden være let at beregne $d_{k_o,k_p}(c) = m$, når man kender både den private og offentlige nøgle. RSA-kryptosystemet udnytter, at det er let at gange store primtal sammen, men svært at bestemme hvilke primtalsfaktorer store tal har.

## Opgaver

i) Brug krypteringsfunktionen $e_2$ til at kryptere klarteksten

$$m = \text{HEJ VERDEN}$$

ii) Brug dekrypteringsfunktionen til at dekryptere chifferteksten

$$c = \text{EÅUCTEJKHHCT}$$

iii) Vi har opsnappet chifferteksten

$$c = \text{TG ØLEPGÅCP G KMPECL IJ LG,}$$

og vi ved, at den er krypteret med et Cæsar-chiffer, men vi kender ikke nøglen. Udnyt, at der kun er få ord med et bogstav til at dekryptere denne besked (Hint: Ét ord med kun et bogstav er meget hyppigt.)

# 2 Matematisk grundlag for RSA

Før vi kan begynde at beskrive Hvis kl. er 9:00 nu, hvad er så klokken om 60 timer? For at løse dette problem bestemmer vi først, hvor mange gange 24 går op i 60. Hvis vi ganger 2 med 24, får vi $24 \cdot 2 = 48$, og ganger vi 3 med 24 får vi $24 \cdot 3 = 72$, hvilket er mere end 60. Vi får derfor, at der om 60 timer er gået to døgn, og resten er $60 - 48 = 12$. Klokken om 60 timer er derfor $12 + 9 = 21$. At regne på denne måde kaldes for moduloregning, og det, vi har bestemt er $9 + 12$ modulo 24, som skrives

$$9 + 60 \equiv 21 \pmod{24},$$

og læses som, at $9 + 60$ er kongruent med 21 modulo 24. Vi definerer kongruens på følgende vis:

**Definition 2.1.** Vi siger, at et heltal $a$ er kongruent med $r$ modulo $b$, hvis det gælder, at

$$a = q \cdot b + r.$$

for et heltal $q$. I så fald skriver vi

$$a \equiv r \pmod{b}.$$

Bemærk, at hvis $a \equiv r \pmod{b}$ så er $r \equiv a \pmod{b}$. Denne definition lyder måske lidt besværlig, men er præcist det, vi gjorde før definitionen. Vi bestemte

$$a = q \cdot b + r \Leftrightarrow 69 = 2 \cdot 24 + 21,$$

og derfor fik vi, at

$$69 \equiv 21 \pmod{24}.$$

Vi vil typisk gerne bestemme det mindste $r$, der opfylder, at $a = qb + r$.

**Definition 2.2** (Heltal $\pmod{n}$)**.** Vi definerer mængden

$$\mathbb{Z}/n\mathbb{Z} = \{0, 1, \ldots, n-1\},$$

og udregninger i denne mængde laves modulo $n$.

**Eksempel 2.3.** I

$$\mathbb{Z}/4\mathbb{Z} = \{0, 1, 2, 3\}$$

gælder der eksempelvis, at $2 \cdot 2 = 0$, da $4 \equiv 0 \pmod{4}$. Alle udregninger kan bestemmes ved såkaldte multiplikations og additionstabeller:

| $+$ | 0 | 1 | 2 | 3 |
|---|---|---|---|---|
| 0 | 0 | 1 | 2 | 3 |
| 1 | 1 | 2 | 3 | 0 |
| 2 | 2 | 3 | 0 | 1 |
| 3 | 3 | 0 | 1 | 1 |

| $\cdot$ | 0 | 1 | 2 | 3 |
|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 |
| 1 | 0 | 1 | 2 | 3 |
| 2 | 0 | 2 | 0 | 2 |
| 3 | 0 | 3 | 2 | 1 |

**Eksempel 2.4.** Vi bemærker, at Cæsar-chifferet kan beskrives ved moduloregning. Til ethvert bogstav A-Å tildeler vi et tal i $\mathbb{Z}/29\mathbb{Z}$. Vi kan gøre det på den oplagte måde, så A tilsvarer 0, B tilsvarer 1 osv. Vi får så, at $\mathscr{P}$ og $\mathscr{C}$ begge består af alle ord med bogstaver fra $\mathbb{Z}/29\mathbb{Z}$. krypteringsfunktionen er så

$$e_k(m) = m + k \pmod{29}.$$

Udtrykket $m + k \pmod{29}$ skal forstås som at $k$ adderes og dette så reduceres (mod 29). Dekryptering foregår så modsat:

$$d_k(c) = c - k \pmod{29}.$$

Ønsker vi eksempelvis at kryptere beskeden HEJ BOB, får vi oversat til tal 7 4 9 1 14 1. Dette krypteres med nøglen $k = 20$, og vi får

$$e_{20}(7\ 4\ 9\ \ 1\ 14\ 1) = 27\ 24\ 0\ \ 21\ 5\ 21 = c$$

For at dekryptere kan vi så enten vælge at trække 20 fra eller lægge 9 til, da $-20 \equiv 9 \pmod{29}$. Vi lægger 9 til for at dekryptere og får

$$d_{20}(27\ 24\ 0\ \ 21\ 5\ 12) = 7\ 4\ 9\ \ 1\ 14\ 1 = m.$$

### Inverse elementer

Hvis vi med normal multiplikation ganger 10 og $\frac{1}{10}$ sammen, så får vi $10 \cdot \frac{1}{10} = 1$. Vi siger, at den multiplikative inverse til 10 er $\frac{1}{10}$. Tilsvarende har vi, at $10+(-10) = 0$, og derfor er 10 og $-10$ hinandens additive inverse.

**Definition 2.5.** Et element $a$ er multiplikativ invers til et element $b$, hvis det gælder, at $ab = 1$. I så fald skriver vi, at $b = a^{-1}$. Et element $a$ er additiv invers til et element $b$, hvis det gælder, at $a + b = 0$.

**Eksempel 2.6.** I $\mathbb{Z}/4\mathbb{Z}$ har vi, at 2 er additiv invers til 2, da $2+2 = 0$. Tilsvarende har vi, at 1 og 3 er hinandens additive inverse, da $3 + 1 = 0$. For multiplikative inverse har vi, at 1 er multiplikativ invers til 1, da $1 \cdot 1 = 1$, og 3 er multiplikativ invers til 3, da $3 \cdot 3 = 1$. Elementet 2 har ikke en multiplikativ invers, da 2 går op i 4.

**Største fælles divisor og Eulers phi-funktion.**

Hvis vi har to heltal $a$ og $b$, så får vi brug for at bestemme det største tal, der går op i begge tal.

**Definition 2.7** (Største fælles divisor)**.** Har vi to heltal $a, b \in \mathbb{Z}$, så betegner vi det største heltal, der går op i $a$ og $b$ som

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

og dette tal kalder vi for den største fælles divisor for $a$ og $b$. I tilfælde af, at at $\gcd(a, b) = 1$, så siger vi, at $a$ og $b$ er indbyrdes primiske.

**Eksempel 2.8.** Hvis vi skal bestemme $\gcd(20, 25)$, så skal vi først bestemme de tal, der går op i både 20 og 25. Disse tal er 1, 2 og 5. Det største af disse tal er 5, så

$$\gcd(20, 25) = 5.$$

**Eksempel 2.9.** Det gælder, at

$$\gcd(6, 3) = 3,$$

da 3 går op i 6. Derfor vil det største tal der går op i begge disse tal klart være 3.

**Definition 2.10** (Eulers $\varphi$-funktion)**.** Vi definerer Eulers $\varphi$-funktion som funktionen $\varphi : \mathbb{Z} \to \mathbb{Z}$, der for $\varphi(n)$ bestemmer hvor mange tal mindre end $n$, der er indbyrdes primiske med $n$.

**Eksempel 2.11.** For 9 gælder der, at

$$\begin{aligned} &\gcd(1, 9) = 1 && \gcd(2, 9) = 1 \\ &\gcd(3, 9) = 3 && \gcd(4, 9) = 1 \\ &\gcd(5, 9) = 1 && \gcd(6, 9) = 3 \\ &\gcd(7, 9) = 1 && \gcd(8, 9) = 1 \\ &\gcd(9, 9) = 9. \end{aligned}$$

Der er altså 6 tal $x$ mindre end 9, der har $\gcd(x, 9) = 1$. Derfor gælder der, at

$$\varphi(9) = 6.$$

**Eksempel 2.12.** Hvis $p$ er et primtal, så er det klart, at $\varphi(p) = p - 1$, da alle tal mindre end $p$ er indbyrdes primiske med $p$.

### Euklids algoritmer

I RSA-algoritmen skal vi i $\mathbb{Z}/n\mathbb{Z}$ for et invertibelt element $a \in \mathbb{Z}/n\mathbb{Z}$ kunne finde det multiplikative inverse element til $a$, altså $a^{-1}$. Da $\mathbb{Z}/n\mathbb{Z}$ var lille, kunne vi bare lave en gangetabel og så aflæse det inverse element, men hvis $n$ er stor, kan dette tage meget lang tid. Derfor skal vi bruge en algoritme, der kan finde inverse elementer relativt hurtigt. Først skal vi dog bruge en sætning, der fortæller os, hvilke elementer, der er invertible:

**Sætning 2.13.** *Et element $a \in \mathbb{Z}/n\mathbb{Z}$ har en multiplikativ invers $a^{-1}$ hvis og kun hvis $gcd(a, b) = 1$.*

Vi vil ikke bevise det, men blot se på nogle eksempler:

**Eksempel 2.14.** I $\mathbb{Z}/9\mathbb{Z}$ har elementerne 1, 2, 4, 5, 7 og 8 multiplikative inverse, da største fælles divisor mellem disse tal og 9 alle er 1. Derimod har elementerne 0, 3 og 6 ikke multiplikative inverse, da de har største fælles divisor større end 1 med 9.

**Eksempel 2.15.** For $p$ et primtal gælder der, at alle elementer $a \in \mathbb{Z}/p\mathbb{Z}$, hvor $a \neq 0$ har multiplikative inverse, da ingen tal mindre end $p$ og større end 1 går op i $p$.

Vi vil nu beskrive Euklids algoritme. Den bestemmer største fælles divisor mellem to tal.

**Definition 2.16** (Euklids algoritme)**.**
Input: To heltal $a$ og $b$.
Output: $\gcd(a, b)$.
Procedure:
$y := a$;
$x := b$;
Så længe $r > 0$ udfør:
$r := y \pmod{x}$;
$y := x$;
$x := r$;
Returnér $x = \gcd(a, b)$.

Det er ikke nødvendigvis klart, hvorfor Euklids algoritme virker. Derfor lad os betragte et eksempel.

**Eksempel 2.17.** Vi skal finde største fælles divisor af 90 og 31. Vi anvender Euklids algoritme. Den består af gentagen division med rest

$$\begin{aligned}
\underbrace{90}_{y} &= 2 \cdot \underbrace{31}_{x} + \underbrace{28}_{r} \\
\underbrace{31}_{y} &= 1 \cdot \underbrace{28}_{x} + \underbrace{3}_{r} \\
\underbrace{28}_{y} &= 9 \cdot \underbrace{3}_{x} + \underbrace{1}_{r} \\
\underbrace{3}_{y} &= 3 \cdot \underbrace{1}_{x} + \underbrace{0}_{r}
\end{aligned}$$

Da $r = 0$, stopper algoritmen og returnerer $\gcd(90, 31) = x = 1$. Derfor har 90 og 31 største fælles divisor 1.

Selvom det nu står klarere, hvordan Euklids algoritme virker, så har vi fortsat ikke vist, at dens output er største fælles divisor af $a$ og $b$. Det er dog ikke svært at overbevise sig selv, om at dette er tilfældet. Vi skal blot vise, at

$$\gcd(a, b) = \text{b,r}$$

for $a = qb + r$. Antag derfor, at $d$ er en divisor for $a$ og $b$. Vi har så

$$r = a - qb = \tilde{a}d - q\tilde{b}d = d(\tilde{a} - q\tilde{b}),$$

for en divisor $\tilde{a}$ af $a$ og en divisor $\tilde{b}$ af $b$, så $d$ går op i $r$. Derfor er alle divisorer af $a$ og $b$ også divisorer af $b$ og $r$ og så må det gælde, at $\gcd(a, b) = \gcd(b, r)$.

### Euklids udvidede algoritme

Lad os vende tilbage til problemstillingen at finde et multiplikativt inverst element til $a \in \mathbb{Z}/n\mathbb{Z}$. For at dette var muligt, så skulle der gælde, at $\gcd(a, n) = 1$. Vi kan bruge Euklids algoritme til at verificere, at dette er tilfældet, men vi skal bruge Euklids udvidede algoritme til at bestemme det inverse element. Idéen kommer af Bézouts identitet.

**Sætning 2.18** (Bézouts identitet)**.** *For positive heltal $a$ og $b$ findes der heltal $s$ og $t$, så*

$$\gcd(a, b) = sa + tb.$$

Vi vil ikke vise denne sætning, men blot anvende den. For indbyrdes primiske tal $a$ og $b$ lyder Bézouts identitet

$$1 = sa + tb,$$

og Euklids udvidede algoritme giver os tallene $s$ og $t$. Reducerer vi nu denne identitet $\pmod b$, så får vi

$$1 \equiv sa \pmod b,$$

og vi har fundet $a^{-1} = s$. Vi vil kun præsentere den udvidede algoritme gennem et eksempel, da alt arbejdet mere eller mindre er gjort i Euklids algoritme.

**Eksempel 2.19.** Fra Eksempel 2.17 fik vi

$$\underbrace{90}_{y} = 2 \cdot \underbrace{31}_{x} + \underbrace{28}_{r} \tag{2.1}$$

$$\underbrace{31}_{y} = 1 \cdot \underbrace{28}_{x} + \underbrace{3}_{r} \tag{2.2}$$

$$\underbrace{28}_{y} = 9 \cdot \underbrace{3}_{x} + \underbrace{1}_{r} \tag{2.3}$$

$$\underbrace{3}_{y} = 3 \cdot \underbrace{1}_{x} + \underbrace{0}_{r} \tag{2.4}$$

Idéen i algoritmen er som følger: Vi ved at $28 = 9 \cdot 3 + 1$ fra (2.1),

$$1 = 28 - 9 \cdot 3. \tag{2.5}$$

Fra (2.2) ved vi, at $31 = 1 \cdot 28 + 3$, så $3 = 31 - 28$. Dette udtryk for 3 indsættes i (2.5), og vi får

$$\begin{aligned} 1 &= 28 - 9 \cdot 3 \\ &= 28 - 9(31 - 28) \\ &= -9 \cdot 31 + 10 \cdot 28 \end{aligned} \tag{2.6}$$

Fra (2.1) ved vi, at $90 = 2 \cdot 31 + 28$, så $28 = 90 - 2 \cdot 31$. Dette indsættes i (2.6), og vi får

$$\begin{aligned} 1 &= -9 \cdot 31 + 10 \cdot 28 \\ &= -9 \cdot 31 + 10(90 - 2 \cdot 31) \\ &= 10 \cdot 90 - 29 \cdot 31. \end{aligned}$$

Vi har nu nøjagtigt det udtryk, vi ønskede. Er vi interesserede i at finde den multiplikative inverse for 31 i $\mathbb{Z}/90\mathbb{Z}$ ved vi nu, at det er 61, da $61 \equiv -29 \pmod{90}$.

### Eulers sætning

Før vi går videre til RSA-algoritmen vil vi vise Eulers sætning, men springe over nogle få detaljer i beviset. Eulers sætning er essentiel for RSA-dekrypteringen. Vi skal dog bruge en hjælpesætning, som vi først vil vise.

**Lemma 2.20.** *Hvis $a$ og $b$ er indbyrdes primiske og $a$ og $c$ er indbyrdes primiske, så er $a$ og $bc$ indbyrdes primiske.*

*Bevis.* Bezouts identitet giver os $s_1$ og $t_1$, så

$$1 = s_1 a + t_1 b$$

og $s_2$ og $t_2$, så

$$1 = s_2 a + t_2 c.$$

Vi ganger nu disse sammen og får

$$\begin{aligned} 1 &= (s_1 a + t_1 b)(s_2 a + t_2 c) \\ &= s_1 s_2 a + s_1 a t_2 c + t_1 b s_2 a + t_1 b t_2 c \\ &= (s_1 s_2 + s_1 t_2 c + t_1 b s_2) a + t_1 t_2 bc. \end{aligned}$$

Deraf følger resultatet ■

**Sætning 2.21** (Eulers sætning)**.** *Lad $a$ og $n$ være indbyrdes primiske, og $n > 0$. Så gælder der, at*

$$a^{\varphi(n)} \equiv 1 \pmod{n}.$$

*Bevis.* Per definition er der $\varphi(n)$ tal mindre end $n$, der er indbyrdes primiske med $n$. Vi opskriver disse tal:

$$S = \{a_1, a_2, \ldots, a_{\varphi(n)}\} \subseteq \mathbb{Z}/n\mathbb{Z}.$$

Vi bestemmer en ny mængde $aS$ ved

$$aS = \{aa_1, aa_2, \ldots, a_{\varphi(n)}\} \subseteq \mathbb{Z}/n\mathbb{Z}.$$

Vi vil gerne kende størrelsen på $aS$. Der er højest $\varphi(n)$ elementer i $aS$, men hvis

$$aa_i \equiv aa_j \pmod{n},$$

for $i \neq j$, så vil der være færre. Derfor betragter vi

$$\begin{aligned} aa_i \equiv aa_j \pmod{n} &\Leftrightarrow aa_i - aa_j \equiv 0 \pmod{n} \\ &\Leftrightarrow a(a_i - a_j) \equiv 0 \pmod{n}. \end{aligned}$$

Dette vil betyde, at $n$ går op i $a$ eller $n$ går op i $(a_i - a_j)$. Da $n$ og $a$ er indbyrdes primiske, kan dette ikke være tilfældet. Derfor må $n$ gå op i $a_i - a_j$, men da $a_i, a_j < n$, så må $a_i - a_j = 0$. Derfor gælder der, at hvis

$$aa_i \equiv aa_j \pmod{n},$$

så er $i = j$. Altså er der $\varphi(n)$ elementer i $aS$. Det gælder desuden af Lemma 2.20, at $\gcd(n, aa_i)$ siden $n$ og $a$ er indbyrdes primiske og $n$ og $a_i$ er indbyrdes primiske. Efter reduktion modulo $n$ vil $n$ og $aa_i$ fortsat være indbyrdes primiske, og derfor har vi, at $aS$ består af $\varphi(n)$ elementer i $\mathbb{Z}/n\mathbb{Z}$ alle indbyrdes primiske med $n$. Men dette var jo $S$. Derfor må der gælde, at $aS = S$.

Vi multiplicerer nu alle elementer i $aS$ og alle elementer i $S$. Dette må give de samme tal, da mængderne er ens og derfor

$$\begin{aligned} aa_1aa_2 \cdots aa_{\varphi(n)} \equiv a_1a_2 \cdots a_{\varphi(n)} \pmod{n} &\Leftrightarrow a^{\varphi(n)}a_{\equiv}a_1a_2 \cdots a_{\varphi(n)} \pmod{n} \\ &\Leftrightarrow a_1a_2 \cdots a_{\varphi(n)}(a^{\varphi(n)} - 1) \equiv 0 \pmod{n}. \end{aligned}$$

Dette betyder, at $n$ går op i $a_1a_2 \cdots a_{\varphi(n)}$ eller at $n$ går op i $a^{\varphi(n)} - 1$. Men Lemma 2.20 giver, at $\gcd(n, a_1a_2 \cdots a_{\varphi(n)}) = 1$, så $n$ må gå op i $a^{\varphi(n)} - 1$. Derfor har vi, at

$$a^{\varphi(n)} \equiv 1 \pmod{n}.$$

■

## Opgaver

i) Afgør, hvilke af følgende kongruenser, der er korrekte og forkerte:

| | |
|---|---|
| 1) $7 \equiv 1 \pmod{2}$ | 2) $48 \equiv 0 \pmod{24}$ |
| 3) $21 \equiv 11 \pmod{20}$ | 4) $60 \equiv 2 \pmod{2}$ |

ii) Bestem det mindste $a$, så følgende kongruenser er sande:

| | |
|---|---|
| 1) $a \equiv 11 \pmod{2}$ | 2) $a \equiv 120 \pmod{25}$ |
| 3) $a \equiv 125 \pmod{60}$ | 4) $a \equiv 55 \pmod{24}$ |

iii) Kryptér klarteksten HEJ VERDEN med et Cæsar-chiffer, hvor $k = 22$ nu med brug af modulo-notation.

iv) Bestem addtions- og multiplikationstabeller for $\mathbb{Z}/2\mathbb{Z}$, $\mathbb{Z}/3\mathbb{Z}$ og $\mathbb{Z}/5\mathbb{Z}$, og bestem hvilke elementer, der er additive og multiplikative inverse til hinanden.

v) Bestem største fælles divisor af følgende tal.

| | |
|---|---|
| 1) 7 og 9 | 2) 100 og 20 |
| 3) 6 og 21 | 4) 21 og 28 |
| 5) 20 og 30 | 6) 31 og 47 |

vi) Bestem følgende værdier for Eulers $\varphi$-funktion:

| | |
|---|---|
| 1) $\varphi(7)$ | 2) $\varphi(4)$ |
| 3) $\varphi(10)$ | 4) $\varphi(1)$ |

**Opgaver i Euklids algoritme**

i) Bestem største fælles divisor for følgende tal

| | |
|---|---|
| 1) 232 og 40 | 2) 91 og 57 |
| 3) 601 og 201 | 4) 42 og 103 |
| 5) 1024 og 302 | 6) 1023 og 423 |

ii) Argumentér for, at Euklids algoritme altid afslutter.

iii) Brug Euklids udvidede algoritme til at bestemme tal $s$ og $t$, så

$$\gcd(a, b) = sa + tb$$

for tallene fra i).

iv) Afgør om følgende elementer i $\mathbb{Z}/531\mathbb{Z}$ har multiplikative inverse. Hvis det er tilfældet, find så disse inverse

| | |
|---|---|
| 1) 47 | 2) 102 |
| 3) 103 | 4) 204 |
| 5) 2 | 6) 401 |

v) Gennemgå og forstå trinene i Eulers sætning i fællesskab.

# 3 RSA-algoritmen

Vi har nu beskrevet den teori, der skal bruges for at få en forståelse for, hvordan RSA-algoritmen virker. Der er fortsat en del hængepartier, men vi vil ikke beskrive dem nærmere. Da RSA-algoritmen er et public-key-system, vil vi bruge terminologien fra Fig. 3 til at beskrive fremgangsmåden.

**Definition 3.1** (RSA-algoritmen)**.**

1. Bob gør følgende:

   i) Bob bestemmer to store primtal $p, q$ og udregner $N = pq$.

   ii) Bob bestemmer en krypteringsnøgle $1 \leq e \leq \varphi(N)$, så $\gcd(e, \varphi(N)) = 1$. Krypteringsnøglen $e$ skal altså være indbyrdes primisk med $\varphi(N)$.

   iii) Bob bestemmer den private dekrypteringsnøgle $d$ som den multiplikative inverse til $e$ i $\mathbb{Z}/\varphi(N)\mathbb{Z}$. Altså opfylder $d$ og $e$ at $de \equiv 1 \pmod{\varphi(N)}$.

   Han deler den offentlige nøgle $(N, e)$ med Alice og beholder selv den private nøgle $(d, p, q)$.

2. Alice tager sin klartekst $m$ og bestemmer $c = m^e \pmod{N}$. Alice sender $c$ til Bob.

3. Bob modtager $c$ og bestemmer $m = c^d \pmod{N}$.

Det første spørgsmål vi kan stille os selv er, hvorfor $m = c^d \pmod{N}$, da dette er grundlaget for korrektheden af denne algoritme. Dette betyder, at

$$m^{ed} \equiv m \pmod{N}.$$

Vi indser først, at

$$\varphi(N) = \varphi(pq) = pq - (p-1) - (q-1) = (p-1)(q-1).$$

Siden det gælder, at $ed \equiv 1 \pmod{\varphi(N)}$, må det gælde, at $ed \equiv 1 \pmod{(p-1)(q-1)}$. Det betyder altså, at

$$ed = 1 + k(p-1)(q-1)$$

for et $k \in \mathbb{Z}$. Vi skal derfor vise, at det gælder, at

$$m^{1+k(p-1)(q-1)} \equiv m \pmod{N}$$

Det må være nok at vise, at

$$m^{1+k(p-1)(q-1)} \equiv m \pmod{p}$$

og

$$m^{1+k(p-1)(q-1)} \equiv m \pmod{q}.$$

Vi viser den første kongruens, da den anden følger på nøjagtigt samme vis. I tilfældet, at $p$ går op i $m$ får vi, at

$$m \equiv 0 \pmod{p},$$

og derfor også, at

$$m^{ed} \equiv 0 \pmod{p},$$

så derfor gælder der, at

$$m^{ed} \equiv m \pmod{p}.$$

I tilfældet, at $p$ ikke går op i $m$ får vi, at $\gcd(m, p) = 1$. Derfor giver Eulers sætning os, at

$$m^{\varphi(p)} = m^{p-1} \equiv 1 \pmod{p}.$$

Dette giver os så

$$\begin{aligned}(m^{p-1})^{k(q-1)} \equiv (1)^{k(q-1)} \pmod{p} &\Leftrightarrow m^{k(p-1)(q-1)} \equiv 1 \pmod{p} \\ &\Leftrightarrow m^{k(p-1)(q-1)+1} \equiv m \pmod{p},\end{aligned}$$

hvilket var hvad vi ønskede.

Vi kan overveje, hvorfor algoritmen her er sikker. Hvis en tredjepart kan bestemme $p$ og $q$ ud fra $N$, så kan han bryde systemet. Derfor må vi antage, at dette er svært, og der er endnu ikke nogen, der har fundet en god måde at faktorisere store sammensatte tal til primtalsfaktorer. Tilsvarende kan systemet brydes, hvis vi kan bestemme $\varphi(N)$ ud fra $N$. Derfor må vi også antage, at dette tilmed er svært.

Det er også vigtigt at bemærke, at hvis $m \geq N$, så vil vi ikke dekryptere den rigtige besked, da den bliver reduceret $\pmod{N}$. Dette må selvfølgelig ikke ske. Dette begrænser beskedlængden temmeligt meget. Derfor bruges dette kryptosystem hovedsagligt til at dele nøgler til et symmetrisk kryptosystem som Cæsar-chifferet.

# Opgaver

i) Vælg to primtal mellem 200 og 500 og brug RSA-algoritmen til at kryptere og dekryptere en klartekst af eget valg.