**HÖGSKOLAN I HALMSTAD**
**Akademin för informationsteknologi**
Mikael Hindgren

**Algebra och diskret matematik**
**Projektuppgift 1.5 hp**
18 september 2025

*Projektuppgiften löses med hjälp av Mathematica och redovisas såväl muntligt som med en skriftlig rapport i form av en Mathematica notebook. Rapporten ska innehålla fullständiga redogörelser för hur deluppgifterna lösts. Projektuppgiften bedöms med betyget godkänd eller underkänd. Godkänd projektuppgift ger 1.5 högskolepoäng.*

# Krypteringssystemet RSA

Inom kryptologi studerar man system för säker kommunikation, sk *kryptosystem*. I ett kryptosystem transformerar (*krypterar*) avsändaren sitt meddelande innan avsändandet på ett sådant sätt att endast behöriga mottagare kan rekonstruera (*avkryptera*) det ursprungliga meddelandet. Innan meddelandet krypteras omvandlas det till ett positivt heltal $M$ som vi antar tillhör mängden $\mathbb{N}_n = \{1, 2, 3, 4, \ldots, n-1\}$ för något heltal $n$. Man kan t.ex. använda A = 1, B = 2, C = 3, osv. eller ASCII-koden för respektive tecken.

Ett krypteringssystem består av två funktioner: $K$ som krypterar och $A$ som avkrypterar så att

$$A(K(M)) = M \text{ för alla } M \in \mathbb{N}_n$$

dvs. om man avkrypterar ett krypterat meddelande får man tillbaka meddelandet i sin ursprungliga form. Funktionerna $K$ och $A$ innehåller i de flesta krypteringssystem parametrar som kallas krypteringsnycklar respektive avkrypteringsnycklar.

Det mest kända och kanske också det bästa krypteringssystemet är RSA-systemet[1, 2, 3] från 1977. Systemet är uppkallat efter sina upphovsmän **R**onald L. Rivest, Adi **S**hamir och Leonard M. **A**dleman och anses vara säkert. RSA är patentskyddat i USA men får användas fritt av privatpersoner. Många av de stora mjukvaruföretagen som t.ex. Microsoft, IBM, Adobe och Apple använder idag systemet. RSA används också för att säkra de elektroniska nycklarna till den amerikanska kärnvapenarsenalen. Krypteringsprogrammet PGP (Pretty Good Privacy), som bl.a. används för säker e-mail, är ett annat exempel som bygger på RSA.

*Symmetriska* krypteringssystem använder samma nyckel för kryptering och avkryptering och därför måste hemliga nycklar utväxlas mellan användarna vilket alltid är en säkerhetsrisk. RSA var det första krypteringssystemet som använder så kallad *asymmetrisk kryptering* vilket innebär att en publik nyckel används för att kryptera meddelandet och en annan hemlig nyckel för att avkryptera det. Denna egenskap gör att RSA även kan användas för att signera ett meddelande så att mottagaren garanterat vet vem som är avsändaren[4]. Asymmetrisk kryptering är säkrare men långsammare än symmetrisk kryptering och därför använder en del system en kombination av båda. Detta gäller t.ex. PGP (Pretty Good Privacy) som är ett program som bl.a. används för att kryptera och dekryptera e-post[7]. Meddelandet (normalt stort) krypteras med ett symmetriskt krypto och nyckeln (normalt liten) som överförs mellan sändare och mottagare krypteras med RSA vilket ger både snabbhet och hög säkerhet.

## Metod för kryptering och avkryptering i RSA-systemet

I RSA-systemet har varje deltagare både offentliga krypteringsnycklar och hemliga avkrypteringsnycklar. Systemet fungerar schematiskt enligt Figur 1:

- Avsändaren transformerar det hemliga meddelandet till ett heltal $M \in \mathbb{N}_n$ enligt något känt system och väljer sedan två (stora) hemliga primtal $p$ och $q$ ($p \neq q$) och sätter $n = pq$.
- Därefter väljs talet $e$ sådant att $1 < e < \phi(n) = (p-1)(q-1)$ där $e$ och $\phi(n)$ dessutom är relativt prima. I praktiken väljs ofta $e$ som ett primtal.
- Det krypterade meddelandet $K(M)$ bestäms enligt[1]

$$K(M) = M^e \bmod n$$

  De positiva heltalen $n$ och $e$ är alltså krypteringsnycklarna (inte hemliga).

[1] $a \bmod b$ = resten då $a$ divideras med $b$. Vi har t.ex. $17 = 3 \cdot 5 + 2$ dvs. $17 \bmod 5 = 2$.

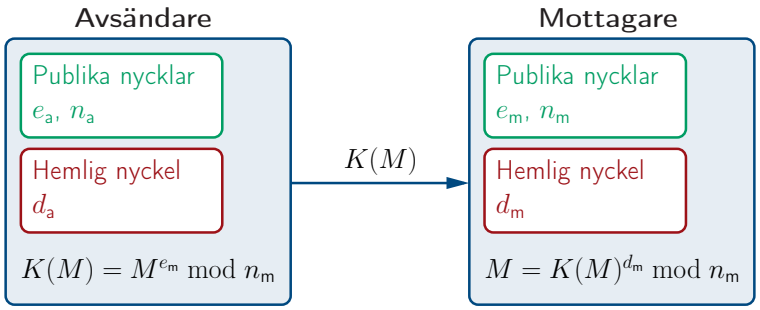
- Mottagaren avkrypterar meddelandet enligt

$$M = A(K(M)) = (K(M))^d \bmod n$$

där $n$ och $d < n$ är avkrypteringsnycklarna ($d$ hemlig). Talet $d$ beräknas enligt följande: Eftersom $\text{sgd}(\phi(n), e) = 1$ finns det heltal $d$ och $b$ sådana att

$$ed + \phi(n)b = 1 \Leftrightarrow ed = 1 - \phi(n)b \Leftrightarrow ed \equiv 1 \pmod{\phi(n)}.$$

Talet $d$ kallas en multiplikativ invers till $e$ modulo $\phi(n)$ och en sådan kan bestämmas med hjälp av Euklides algoritm.




Figur 1: Kryptering och avkryptering av ett meddelande $M$ med RSA.

## Teorin bakom RSA-systemet

För att förklara varför systemet fungerar behöver vi några teoretiska resultat kopplade till följande funktion:

**Definition 1 (Eulers $\phi$-funktion).**
*För positiva heltal $n$ är $\phi(n)$ antalet positiva heltal mindre än $n$ som är relativt prima med $n$.* □

Om t.ex. $n = 15$ är talen $1, 2, 4, 7, 8, 11, 13, 14$ relativt prima med $n$ dvs. $\phi(15) = 8$. Om $p$ är ett primtal är alla positiva heltal mindre än $p$ relativt prima med $p$ dvs. $\phi(p) = p - 1$.

Man kan visa[4, 5] följande egenskaper hos Eulers $\phi$-funktion:

**Sats 1**
Om $\text{sgd}(m, n) = 1$ så är $\phi(mn) = \phi(m)\phi(n)$. ∗

Sats 1 medför att om $p$ och $q$ är primtal och $p \neq q$ så är $\phi(pq) = \phi(p)\phi(q) = (p-1)(q-1)$.

**Sats 2 (Eulers sats)**
Om $a, n \in \mathbb{Z},\ n \geq 2$ och $\text{sgd}(a, n) = 1$ så är $a^{\phi(n)} \equiv 1 \pmod{n}$. ∗

För $n = 10$ är $\phi(10) = 4$ (talen 1, 3, 7, och 9 är relativt prima med 10) så för alla $a$ sådana att $\text{sgd}(a, 10) = 1$ är $a^4 \equiv 1 \pmod{10}$ enligt Sats 2. Eulers sats kan därför användas för att förenkla moduloberäkningar. Vi har t.ex.

$$7^{322} = 7^{320} \cdot 7^2 \equiv 1 \cdot 7^2 = 49 \pmod{1200}$$

eftersom $\phi(1200) = 320$.

Vi valde $e$ och $d$ ovan så att $ed \equiv 1 \pmod{\phi(n)}$. Enligt divisionsalgoritmen finns det då ett heltal $k$ sådant att $ed = k\phi(n) + 1$ och vi får

$$\begin{aligned} M^{ed} = M^{k\phi(n)+1} = M^{k\phi(n)} \cdot M = (M^{\phi(n)})^k \cdot M \equiv 1^k \cdot M = M \pmod{n} \\ \Leftrightarrow M^{ed} \bmod n = M \bmod n = M \end{aligned}$$

där vi utnyttjade Sats 2 samt att $M < n$ vilket innebär att $M \bmod n = M$. Vi får nu

$$A(K(M)) = (M^e \bmod n)^d \bmod n = (M^{ed} \bmod n) \bmod n = M^{ed} \bmod n = M$$

där den andra likheten följer av att det enligt divisionsalgoritmen finns heltal $k$ och $r$ sådana att

$$M^e = kn + r \Rightarrow M^{ed} = (kn + r)^d = (d \text{ st termer som innehåller } n) + r^d = n(\ldots) + r^d$$
$$\Rightarrow r^d = (M^e \bmod n)^d = M^{ed} \bmod n.$$

Kryptering följt av avkryptering ger alltså tillbaka det ursprungliga meddelandet $M$ och metoden fungerar.

**Anmärkning**
För att vi ska kunna utnyttja Eulers sats krävs att $\text{sgd}(M, n) = 1$ men eftersom vi endast antagit att $M < n = pq$ skulle vi i princip kunna ha $M = ap,\ 1 < a < q$ (eller motsvarande för $q$) vilket då innebär att $\text{sgd}(M, n) = p$ och $\text{sgd}(M, q) = 1$. Fortfarande gäller dock att $ed \equiv 1 \pmod{\phi(n)} \Leftrightarrow ed = k\phi(n) + 1$ för något $k$ vilket medför att:

$$(M^{ed} - M) \bmod n = M(M^{ed-1} - 1) \bmod n = ap(M^{k\phi(n)} - 1) \bmod pq = p(a(M^{k\phi(n)} - 1) \bmod q).$$

Eftersom $\phi(q) = q - 1$ får vi:

$$a(M^{k\phi(n)} - 1) \bmod q = a(M^{k(p-1)(q-1)} - 1) \bmod q = a((M^{k(p-1)})^{(q-1)} - 1) \bmod q = 0$$

där vi i sista likheten utnyttjar Eulers sats ($\text{sgd}(M^{k(p-1)}, q) = 1$). Vi får alltså

$$(M^{ed} - M) \bmod n = 0 \Leftrightarrow M^{ed} \bmod n = M \bmod n = M$$

och algoritmen fungerar även i det här fallet. ∗

## Beräkningar i praktiken

Vid kryptering och avkryptering kan man ibland få mycket stora tal. För att sända det lilla meddelandet $M = 9$ till en mottagare med krypteringsnycklarna $n = 55, e = 27$ måste avsändaren beräkna $K(9) = 9^{27} \bmod 55$. Men detta kan göras utan att beräkna $9^{27}$. Vi skriver 27 som en summa av tvåpotenser: 27=16+8+2+1 och får

$$9^2 = 81 = 1 \cdot 55 + 26 \equiv 26 \pmod{55}$$
$$9^4 = (9^2)^2 \equiv 26^2 = 676 = 12 \cdot 55 + 16 \equiv 16 \pmod{55}$$
$$9^8 = (9^4)^2 \equiv 16^2 = 256 = 14 \cdot 55 + 36 \equiv 36 \pmod{55}$$
$$9^{16} = (9^8)^2 \equiv 36^2 = 1296 = 23 \cdot 55 + 31 \equiv 31 \pmod{55}$$

Vi får nu

$$9^{27} = 9^{16} \cdot 9^8 \cdot 9^2 \cdot 9 \equiv 31 \cdot 36 \cdot 26 \cdot 9 = 261144 = 4748 \cdot 55 + 4 \equiv 4 \pmod{55}.$$

## Säkerheten

Säkerheten hos RSA-systemet hänger på att det förnärvarande inte finns någon effektiv algoritm för att primtalsfaktorisera heltal. Väljer vi bara $p$ och $q$ tillräckligt stora (mer än 100 siffror) finns det ingen möjlighet att faktorisera $n = pq$ för att kunna beräkna $\phi(n) = (p - 1)(q - 1)$ och därmed den hemliga nyckeln $d$ inom en rimlig tidsrymd.

1991 publicerade RSA Laboratories en lista[6] på ett antal stora *semiprimtal*, dvs. tal som är produkter av två primtal. De utlovade olika stora belöningar till dem som lyckas faktorisera något av dem. Det största tal i listan som man hittills lyckats faktorisera betecknas RSA-250 och är 250 siffror långt. Faktoriseringen gjordes så sent som 2020 och krävde en motsvarande sammanlagt CPU-tid på ca 2700 år för en modern standard-PC.

# Uppgifter

1. Använd Eulers sats för att beräkna $17^{146}$ mod 105 för hand. Du kan bestämma $\phi(n)$ för lämpligt $n$ med hjälp av Mathematicafunktionen `EulerPhi`. Kontrollera ditt svar med `PowerMod`.
2. Beräkna $636^{37}$ mod 816 med hjälp av "kvadreringsmetoden" ovan.
3. Bestäm primtalen $p$ och $q$ om $n = pq = 57925877$ och $\phi(n) = 57910656$.
4. För heltalen $a$ och $b$ gäller att $b \equiv a \pmod{106}$ och $\text{sgd}(a, 106) = 1$.
   (a) Visa att $\text{sgd}(b, 106) = 1$.
   (b) Bestäm ett positivt tal $k > 1$ sådant att $b^k \equiv a \pmod{106}$.
   (c) Bestäm $a^k$ mod 106 om $b = 251$.
5. Bestäm ett fungerande värde på den publika nyckeln $e$ om $p = 11$ och $q = 31$. Bestäm därefter en hemlig nyckel $d$ och kryptera meddelandet $M = 13$. Kontrollera dina beräkningar genom att avkryptera det krypterade meddelandet. Vilka meddelanden (tal) $M$ kan krypteras med ovanstående nycklar?
6. Skriv ett program i Mathematica som krypterar och avkrypterar ett godtyckligt textmeddelande med RSA enligt ovan. Använd Unicode för tecknen då de transformeras till heltal och kryptera/avkryptera ett tecken i taget. Koden ska säkerställa att samtliga villkor för de ingående parametrarna är uppfyllda.

   Indata till programmet ska vara:

   1) Ett valfritt meddelande i textform.
   2) Antalet siffror i primtalen $q$ och $p$. Dessa ska sedan genereras slumpmässigt.

   Utdata från programmet ska som minimum vara:

   1) Inmatat okrypterat meddelande i textform.
   2) Okrypterat meddelande i talform: $M$.
   3) Primtalen $p$ och $q$.
   4) Nycklarna $n$, $e$ och $d$.
   5) Krypterat meddelande i talform $K(M)$.
   6) Avkrypterat meddelande i tal- och textform $A(K(M))$.

   Tips: Följande Mathematicafunktioner kan vara användbara i ovanstående uppgift: `RandomPrime, EulerPhi, Mod, PowerMod, InputString, FromCharacterCode, ToCharacterCode`.

# Referenser


[1] R. Rivest, A. Shamir and L. Adleman, *A Method for Obtaining Digital Signatures and Public-Key Cryptosystems*, Communications of the ACM, 21 (2), pp. 120-126 (1978)

[2] RSA Laboratories, *PKCS #1 v2.1: RSA Encryption Standard* (2002)
ftp://ftp.rsasecurity.com/pub/pkcs/pkcs-1/pkcs-1v2-1.pdf

[3] T. H. Cormen, C. E. Leiserson och R. L. Rivest, *Introduction to algorithms*, MIT Press (1990)

[4] J. Johansson och S. Lemurell, *Algebra och diskret matematik*, Studentlitteratur (2013)

[5] I. Niven, H. S. Zuckerman och H.L. Montgomery, *An introduction to the theory of numbers, 5th ed.*, Wiley (1991)

[6] Wikipedia, *RSA Numbers*, [Uppdaterad 12 mars 2024; hämtad 22 september 2024].
Hämtad från: http://en.wikipedia.org/wiki/RSA_numbers

[7] Wikipedia, *Pretty Good Privacy*, [Uppdaterad 14 juni 2023; hämtad 22 september 2024].
Hämtad från: https://sv.wikipedia.org/wiki/Pretty_Good_Privacy