Grigore Eugen-Andrei  
Grupa 202

**Securitate informațională bazată pe metode asimetrice de criptare**

**Cuprins**

[1. Introducere 1](#_Toc134741034)

[2. Descrierea temei 3](#_Toc134741035)

[3. Metode asimetrice de criptare 5](#_Toc134741036)

[3.1 Descriere 5](#_Toc134741037)

[3.2 Istoria algoritmilor criptografici asimetrici 7](#_Toc134741038)

[3.3 Primul algoritm criptografic asimetric (RSA) 8](#_Toc134741039)

[3.4 Descrierea funcționării RSA 9](#_Toc134741040)

[3.4.1 Generarea cheilor 9](#_Toc134741041)

[3.4.2 Distribuirea cheilor 11](#_Toc134741042)

[3.4.3 Criptarea 11](#_Toc134741043)

[3.4.4 Decriptarea 12](#_Toc134741044)

[3.4.5 Semnarea mesajelor 12](#_Toc134741045)

[3.5 Transport Layer Security (TLS) 13](#_Toc134741046)

[3.5.1 Descriere 13](#_Toc134741047)

[3.6 Performanța algoritmilor asimetrici (RSA) 15](#_Toc134741048)

[4. Concluzii 19](#_Toc134741049)

[5. Bibliografie 20](#_Toc134741050)

# 1. Introducere

Securitatea informațională este o parte importantă a comunicării prin rețelele publice. Există mai multe metode criptografice care asigură securitatea, iar acestea pot fi categorizate în două grupe, simetrice și asimetrice. Metodele simetrice utilizează o singură cheie care poate cripta și decripta mesajul, sau cheia folosită pentru decriptare care este ușor de calculat din cheia de criptare. Metodele simetrice au ca și provocare metoda de împărtășire a cheii intre interlocutori, deoarece este necesar un canal sigur pentru transmisia cheii. Metodele asimetrice rezolvă această problemă prin utilizarea unei perechi de chei, una privată si una publică. Toate mesajele transmise cu cheia publică, pot fi decriptate doar cu cheia privată, iar mesajele criptate cu cheia privată pot fi confirmate ca și provenind de la interlocutorul așteptat. Mărimile cheilor pentru criptare asimetrică sunt de asemenea mai mari decât cele utilizate in criptarea simetrică, ceea ce face compromiterea lor puțin probabilă.

În această lucrare voi discuta despre diferitele metode de criptare asimetrică, utilizarea lor, modul în care au fost folosite în trecut, cum sunt folosite acum, o descriere a sistemelor care utilizează acest tip de criptare, performanța lor și compromisuri pe care acestea le fac. Trebuie sa se țină cont de asemenea că această metodă are și ea neajunsuri. Din acest motiv, lucrarea va mai discuta în scurt și despre alte metode care sunt folosite împreună cu acestea pentru a crea o comunicare cât mai aproape de total sigură.

Într-o lume perfectă, comunicarea poate rămâne sigură cu oricare din metode, cât timp cheile rămân secrete. În acest caz, cheile trebuie știute in prealabil, sau trebuie interschimbate într-un mediu „sanitar”, preferabil fiind ca nimeni să nu știe de existența lor cu excepția interlocutorilor, acest lucru fiind improbabil. Acesta este unul din lucrurile pe care metoda de criptare asimetrică îl rezolvă: asigurarea acestui mediu „sanitar” între doi interlocutori, fie că este utilizat pentru a asigura o altă metodă de criptare, fie că este utilizat ca atare.

# 2. Descrierea temei

Metodele asimetrice de criptare se regăsesc extensiv în sistemele care sunt folosite în zilele noastre. În general, ele asigură o cale securizată de comunicare; după aceea se trece la utilizarea unei metode simetrice pentru viteză. În același timp, criptarea asimetrică este folosită și pe cont propriu, spre exemplu: „End-to-End Encryption” (folosit în sistemele aplicațiilor de mesaje cum ar fi Whatsapp), SSL/TLS (SSL Handshake pe https), PKI (infrastructura pe baza cheilor publice).

Am decis să aleg această temă deoarece de curând am început să lucrez la proiecte care necesită comunicare client-server, care de obicei necesită protejarea credințialelor utilizatorilor. Inspirația principală o reprezintă SSL/TLS și HTTPS care utilizează criptarea asimetrică pentru a negocia o conexiune (handshake). Prin acest handshake sunt asigurate mai multe lucruri, cum ar fi algoritmi de criptare pe care ambele partide îi acceptă și suportă, asigurarea identității serverului, confirmarea validității certificatului și, în final, generarea cheii simetrice care asigură criptarea întregii conexiuni de acum încolo. O altă metodă care mi-a stârnit interesul este End-to-End Encryption, aceasta fiind folosită în general în aplicațiile de mesagerie pentru a asigura confidențialitatea chiar dacă o a treia partidă este implicată (serverul). Schimbul de chei este stabilit între două dispozitive care comunică. Fiecare își împărtășește cheia publică, înainte de a trimite un mesaj către un translocutor criptat cu cheia lui publică, acesta fiind unicul capabil să decripteze mesajul cu cheia privată.

În general, metoda este una destul de lentă, în special comparată cu criptarea simetrică, fiind utilizate chei mult mai lungi între 1024-4096 biți, comparativ cu AES256 care folosește chei de 256 biți.

În acest eseu voi vorbi mai în amănunt despre utilizarea criptării asimetrice, istoricul inventării acesteia, cât și motivul aflat la baza dezvoltării sale, apoi voi enunța și comenta câteva opinii pro și contra. Voi descrie în detaliu și implementarea algoritmilor, modul în care sunt folosiți în domeniu, apoi voi compara performanța algoritmilor de criptare simetrici cu a celor asimetrici.

# 3. Metode asimetrice de criptare

## 3.1 Descriere

Criptografia pe baza cheilor publice sau criptografia asimetrică este partea criptografiei care utilizează perechi de chei corespondente (cheia privată și cea publică). Perechile de chei sunt generate prin algoritmi criptografici bazați pe funcții matematice numite și funcții uni-direcționale (one-way functions). Securitatea acestei criptografii depinde total de păstrarea cheii private secretă, divulgând cheia publică doar partidelor cu care se dorește a comunica. Oricine are o cheie publică poate să cripteze un text, dar acest text în final poate fi decriptat doar de cheia privată corespondentă celei publice utilizate pentru criptare.

Cheile private mai sunt utilizate și în sisteme de semnături digitale, prin care un expeditor folosește cheia privată împreună cu mesajul. Oricine care posedă cheia publică corespondentă poate verifica validitatea mesajului. Un falsificator care nu are acces la cheia privată nu se va putea da drept autoritatea validă, deoarece în momentul validării cu cheia publică se va vedea discrepanța, așadar aflându-se faptul că este vorba de un falsificator.

Înainte de anii 1970 toate sistemele criptografice foloseau metode simetrice, în care o cheie unică este utilizată pentru criptarea și decriptarea datelor. Acest lucru necesită ca această cheie să fie ținută secretă și să fie interschimbată între partidele care comunică printr-o metodă sigură. Cu cât numărul partidelor care sunt angajate în comunicare crește, cu atât crește și dificultatea interschimbării cheii printr-o metodă sigură. De asemenea, schimbarea cheii, care este o metodă de a mări securitatea sistemelor simetrice este un detriment în comunicarea sigură a cheilor. Mărind frecvența, crește și riscul ca cheile să fie dezvăluite partidelor neautorizate. În teorie, am putea utiliza câte o pereche de chei pentru fiecare partidă care se angajează în comunicare, dar și acest lucru ar crea o complexitate exponențială.

Prima metodă prin care aceste sisteme sunt folosite este cea de criptare directă, unde mesajul este criptat folosind cheia publică a destinatarului. Pentru un algoritm bine ales care nu are breșe de securitate cunoscute, mesajul nu poate fi decriptat decât folosind cheia privată care îi corespunde. Acest lucru asigură confidențialitatea mesajului. O a doua metodă prin care acest sistem poate fi folosit este semnătura digitală descrisă anterior. Un mesaj este „semnat” (criptat) folosind cheia primară a expeditorului, validitatea putând fi verificată prin utilizarea cheii publice asociate. Această verificare dovedește că expeditorul este cel care se află în posesia cheii private și cel mai probabil persoana asociată cu cheia publică. De asemenea, ne este asigurat că acesta este mesajul original, deoarece verificarea ar da greș pentru orice alt mesaj creat fără cheia privată. Tot acest sistem se bazează pe încrederea că acea cheie publică este cea autentică și aparține entității revendicate, dar și că nu a fost modificată sau schimbată de nici o partidă care dorește răul.

De aceea, este utilizat sistemul curent de PKI (Public Key Infrastructure), prin care există mai multe entități care nu sunt în nici un fel asociate cu utilizatorii, cunoscute ca și „Certificate Authorities (CA)”, care generează și certifică perechi de chei. În momentul în care se face verificarea mesajului, se verifică dacă această cheie publică este una generată de o entitate de încredere, iar în caz contrar, mesajul este considerat nesigur.

## 3.2 Istoria algoritmilor criptografici asimetrici

Până în 1970, criptografia se baza pe algoritmi simetrici, fiind nevoie de comunicare sigură pentru a distribui cheile. Acestea erau distribuite fie față în față, fie printr-un curier de încredere. Metodele de până acum deveneau complex de menținut, în special când comunicarea avea loc între foarte multe partide. Asta este premiza care ne-a facut sa cautam mai mult.

În 1874 William Stanley Jevons scrie in cartea „The Principles of Science”: „Can the reader say two numbers multiplied together will produce the number 8616460799? I think unlikely that anyone but myself will ever know.” Tradus: „Poate cititorul să găsească 2 numere care înmulțite să dea 8616460799? Cred că e improbabil ca altcineva în afară de mine să știe vreodată”. El a descris relația funcțiilor uni-direcționale, și de asemenea a discutat despre problema factorizării utilizată pentru a creea funcția trapă. (Wikipedia, 2023)

În 1970, James H. Ellis, un criptograf Britanic care lucra la GCHQ (UK Government Communications Headquarters), s-a gândit la posibilitatea unei „criptării non-secretă” (PKI), dar nu a găsit nici o modalitate de a o implementa. În 1973, colegul său Clifford Cocks a implementat ce acum este cunoscut ca și RSA, oferind o metodă practică de „criptare non-secretă”. În 1974 un alt matematician și criptograf, Malcom J. Williamson, a dezvoltat ce acum este cunoscut ca și Diffie-Hellman key exchange (Wikipedia, 2023). Aceste dezvoltări în domeniul criptografic au fost folosite strict în mod militar, fiind recunoscute public abia dupa 27 de ani, când cercetăriile guvernului Britanic au fost declasificate.

## 3.3 Primul algoritm criptografic asimetric (RSA)

RSA este primul algoritm criptografic pe bază de chei publice care este utilizat la scală largă. Acronimul „RSA” provine de la numele celor care l-au descris în mod public în 1977, Ron Rivest, Adi Shamir și Leonard Adleman (RSA (cryptosystem), 2023). Un algoritm echivalent anterior descris a fost inventat în 1973 de Clifford Cocks la GCHQ, care mai târziu, în 1997 a fost declasificat.

Securitatea algoritmului depinde total de dificultatea de factorizare a două numere prime mari, cunoscută drept „problema factorizării” (factoring problem). Încă nu se știe dacă spargerea algoritmului presupune o dificultate la fel de mare ca și problema factorizării (RSA (cryptosystem), 2023). Cât timp o cheie destul de mare este utilizată, nu există o metodă cunoscută pentru a descifra mesajele criptate cu cheia respectivă.

RSA este un algoritm relativ lent, cel mai des fiind folosit pentru a crea un canal de comunicare sigură prin care se transmite o cheie privată. Apoi, cu această cheie privată se continuă comunicarea prin utilizarea unui algoritm simetric, mai rapid. Aceasta este și metoda prin care HTTPS funcționează.

## 3.4 Descrierea funcționării RSA

RSA este descris în pagina „RSA (cryptosystem) Wikipedia” ca și funcționând în patru pași: generarea cheii, distribuirea cheii, criptare și decriptare.

Principiul care stă la baza RSA este observația că este impractic să găsești 3 întregi mari *e*, *d* și *n*, astfel încât exponențierea pentru oricare *m*(0 ≤ m < n):

Știind *e* și *n*, sau *m*, poate fi extrem de dificil în a găsi *d*.

O proprietate utilă este utilizarea comutativității, astfel încât:

RSA implică utilizarea unei chei publice și a uneia private. Cheia publică este reprezentată de întregii *n* și *e*, iar cheia privată de întregul *d* (deși *n* este folosit în procesul de criptare, poate fi și el considerat parte din cheia privată). *m* reprezintă mesajul (pregătit în prealabil).

### 3.4.1 Generarea cheilor

Pașii necesari pentru a genera cheia sunt după cum urmează, conform paginii Wikipedia „RSA (cryptosystem)”:

1. Alegerea a 2 numere prime mari *p* și *q.*
   1. Pentru a face factorizarea mai dificilă *p* și *q* vor fi alese la întâmplare, amândouă vor fi numere mari și vor avea o diferență mare.
   2. *p* și *q* vor fi secrete.
2. Calculează *n = p \* q.*
   1. *n* este folosit ca și modul pentru cheia publică cât și pentru cea privată.
   2. *n* este considerat ca și parte din cheia publică.
3. Calculează *λ(n)*, unde *λ* este funcția toientă a lui Carmichael. Din moment ce *n = p \* q*, *λ(n)* = cmmmc(*λ(p)*, *λ(q)*), și pentru că *p* și *q* sunt prime, *λ(p) = φ(p) = p-1*, și asemănător *λ(q) = q – 1.*

Deci *λ(n)* = cmmmc(*p – 1, q - 1).*

* 1. cmmmc poate fi calculat prin Algoritmul Euclidean
  2. *λ(n)* este păstrat secret.

1. Alege un întreg *e* astfel încât 2 < *e* < *λ(n)* și cmmdc(*e*, *λ(n)*) = 1.
   1. *e* având o lungime scurtă în biți și o greutate Hamming mică rezultă în o criptare mai eficientă. Valoarea cel mai des aleasă este . Cea mai mică valoare posibilă este 3, dar asta produce o criptare mai puțin sigură în anumite condiții.
   2. *e*  este considerat parte din cheia publică.
2. Determină *d* a.î. ; *d* fiind inversa multiplicativă modulară a lui *e* mod *λ(n)*.
   1. *d* este păstrat secret ca și parte din cheia privată.

Cheia publică este formată din modulul *n* și exponentul de criptare *e*. Cheia privată este formată din exponentul de decriptare *d*, care trebuie să ramână secret. *p*, *q* și *λ(n)* trebuie să rămână secrete de asemenea, deoarece pot fi folosite pentru a calcula *d*.

### 3.4.2 Distribuirea cheilor

Următoarele exemple sunt preluate din pagina Wikipedia „RSA (cryptosystem)”.

Presupunem că Andrei dorește să ii trimită un mesaj secret Alexandrei. Dacă este utilizat RSA (ca și orice alt algoritm asimetric), Andrei are nevoie să știe cheia privată a Alexandrei, iar Alexandra trebuie să utilizeze cheia sa privată pentru a decripta mesajul primit.

Pentru ca Andrei să poată cripta mesajele, Alexandra îi transmite cheia ei publică (*n*, *e*) printr-un canal de încredere, dar nu neapărat sigur. Cheia privată a Alexandrei nu este niciodată distribuită.

### 3.4.3 Criptarea

După ce Andrei obține cheia publică de la Alexandra, acesta poate cripta mesajul *M* pe care dorește să îl trimită.

Mai întâi, mesajul (mesaj sub formă de string, fără padding) trebuie să fie transformat într-un număr întreg *m* (mesaj sub formă binară, cu padding), astfel încât 0 ≤ *m* < *n* prin utilizarea unei metode de padding, la care se cade de comun acord. Apoi calculează cifrul *c*, utilizând cheia publica a Alexandrei.

În final Andrei îi va trimite acest cifru Alexandrei prin canalul ales.

### 3.4.4 Decriptarea

Alexandra poate să decripteze mesajul prin utilizarea cheii sale private.

Fiind dat *m* se poate recupera mesajul *M* prin inversarea metodei de padding aleasă.

### 3.4.5 Semnarea mesajelor

Presupunem că Alexandra folosește cheia publică a lui Andrei să îi trimită un mesaj. În mesaj, poate să pretindă că este Alexandra. Andrei nu are nici o metodă de a confirma că el comunică cu Alexandra, deoarece oricine cu cheia sa publică poate cripta mesaje. Pentru a confirma originea unui mesaj, RSA poate fi folosit pentru a semna mesajul.

Presupunem că Alexandra îi trimite un mesaj lui Andrei. Poate să utilizeze cheia sa privată pentru a semna mesajul. Va produce o valoare hash din mesajul original, apoi o va ridica la puterea *d* (ca și la decriptare) și o atașează ca și semnătură mesajului. În momentul în care Andrei primește mesajul, poate folosi funcția de hash împreună cu cheia publică a Alexandrei. Va ridica semnătura la puterea *e* (ca și la criptare), iar apoi va aplica funcția de hash pe mesaj. Comparând astfel rezultatul decriptării cu hash-ul poate confirma originea mesajului.

Acest lucru este posibil datorită comutativității exponenților: ,

Astfel cheile pot fi interschimbate fară a se pierde generalitatea, iar cheia privată poate fi folosită să:

1. Decripteze un mesaj menit doar pentru destinatar, care poate fi criptat de către oricine este în posesia cheii publice.
2. Cripteze un mesaj care poate fi decriptat de către oricine, dar poate fi criptat de către o singură entitate, oferind o semnătura digitală.

## 3.5 Transport Layer Security (TLS)

Transport Layer Security (TLS) este un protocol criptografic care este utilizat la scală largă în aplicații de email, mesagerie instantă, voce prin IP, însă utilizarea sa în HTTPS rămâne cea mai cunoscută.

Protocolul TLS are ca obiective oferirea de securitate, incluzând oferirea de confidențialitate, integritatea datelor și asigurarea autenticității prin utilizarea criptografiei, în principal prin utilizarea certificatelor (criptare asimetrică). Acesta este considerat ca fiind compus din două părți: TLS record și TLS handshake.

### 3.5.1 Descriere

Sistemele client-server se bazează pe TLS pentru comunicare sigură, fiind ferită comunicarea de modificări neautorizate sau de ascultare. (Transport Layer Security, 2023)

Deoarece comunicarea poate folosi TLS sau nu, clientul trebuie să se asigure că serverul cu care comunică folosește TLS. O metodă folosită pentru asigurarea comunicării sigure prin TLS este utilizarea porturilor specifice (Transport Layer Security, 2023). Portul 80 este în mod normal utilizat pentru comunicare fără TLS (HTTP), iar portul 443 este folosit pentru comunicare prin HTTPS.

Îndată ce s-a confirmat utilizarea TLS de către server și client, se negociază conexiunea (handshake). La început se folosește un algoritm asimetric pentru a asigura setările, împreună cu o cheie simetrică specifică sesiunii, care este utilizată pentru comunicarea ce urmează după handshake. (Transport Layer Security, 2023)

Ce se întâmplă într-un handshake:

1. Clientul trimite mesajul inițial prin care se comunică algoritmi pe care clientul vrea să îi utilizeze împreună cu o valoare aleatorie. Valoarea aleatorie trimisă va fi folosită pentru a genera Pre-Master Secret.
2. Serverul răspunde, acesta confirmând că poate folosi versiunea respectivă de TLS. De asemenea, serverul va alege dintre algoritmi pe care clientul îi acceptă. Serverul va genera și el o valoare aleatorie. Împreună cu acestea este trimis pentru autenticitate și Certificatul SSL al serverului.
3. În momentul în care clientul primește certificatul, acesta îi va valida autenticitatea: dacă este valid, se va continua. De asemenea se împărtașește cheia secretă printr-un algoritm asimetric.
4. Clientul criptează numărul aleator și îl criptează folosind cheia publică a serverului; cele 2 partide vor genera o cheie pe care o vor utiliza pentru comunicare secretă prin algoritmi simetrici pentru restul sesiuni.

## 3.6 Performanța algoritmilor asimetrici (RSA)

În criptarea asimetrică, fiind necesare chei mai mari, timpul de criptare/decriptare crește direct proporțional cu mărimea datelor transmise. De aceea TLS folosește algoritmi asimetrici doar pentru handshake. În cazul în care sunt transmise blob-uri de date mai mari timpul de așteptare pentru client va crește.

În această secțiune voi evidenția diferența între performanța RSA și DSA (Data Encryption Standard) un algoritm simetric. Toate testele de performanță și tabelele/graficele sunt din articolul „A performance analysis of DES and RSA cryptography” scris în anul 2013 de S.Singh, S.Maakar și S.Kumar.

|  |  |  |  |
| --- | --- | --- | --- |
| **Input File Size (KB)** | **Encryption Execution Time(Seconds)** | | |
| **DES** | **2DES** | **RSA** |
| 15 | 4.543859 | 9.087718 | 5.637362 |
| 30 | 9.087718 | 18.17544 | 11.27472 |
| 45 | 13.63158 | 27.26315 | 16.91209 |
| 60 | 18.17544 | 36.35087 | 22.54945 |
| 75 | 22.7193 | 45.43859 | 28.18681 |

1. Timp de criptare (Singh, Maakar, & Kumar, 2013)

În tabelul de mai sus se regăsesc timpii de execuție ai algoritmilor DES, 2DES și RSA la criptare. Din tabel reiese clar diferența de viteză între DES și RSA care crește cu mărimea fișierelor.

|  |  |  |  |
| --- | --- | --- | --- |
| **Input File Size (KB)** | **Decryption Execution Time(Seconds)** | | |
| **DES** | **2DES** | **RSA** |
| 15 | 4.543859 | 9.087718 | 5.637362 |
| 30 | 9.087718 | 18.17544 | 11.27472 |
| 45 | 13.63158 | 27.26315 | 16.91209 |
| 60 | 18.17544 | 36.35087 | 22.54945 |
| 75 | 22.7193 | 45.43859 | 28.18681 |

1. Timp de decriptare (Singh, Maakar, & Kumar, 2013)

În cel de-al doilea tabel sunt rezultatele execuției algoritmilor la decriptare. De asemenea și aici poate fi observată o diferentă substanțială, mai ales cu creșterea mărimii datelor.

3 Timp criptare (Singh, Maakar, & Kumar, 2013)

4 Timp decriptare (Singh, Maakar, & Kumar, 2013)

După cum reiese din tabele și grafurile atașate există o diferență considerabilă între criptarea asimetrică (RSA) și cea simetrică (DES). Chiar dacă populația de test este mică, existând similarități între toți algoritmii simetrici, respectiv asimetrici, putem trage o concluzie că cei asimetrici sunt mai lenți. Viteza va depinde de mărimea fișierelor și algoritmii utilizați.

# 4. Concluzii

În concluzie, algoritmii asimetrici sunt o metodă foarte bună de a asigura comunicarea sigură între două partide. Chiar dacă viteza lor este mai mică, au un rol important în criptografie. Cea mai cunoscută și cea mai importantă metodă în care sunt folosiți este asigurarea unui canal sigur de comunicare și semnarea mesajelor, asigurându-ne că acestea sunt transmise de la sursa de la care ne așteptăm.

Cel mai proeminent algoritmii asimetrici sunt folosiți fie în sisteme criptografice strict pentru a asigura distribuirea sigură a unei cheii private, fie în sistemele de mesagerie. Fiind vorba de viteze mai scăzute de criptare/decriptare, sistemul este potrivit fie pentru o singură acțiune de prevenire a „tragerii cu urechea” sau pentru a transmite mesaje de marime mică (aplicații de mesagerie, End-to-End Encryption).

Metodele asimetrice de criptare au fost o descoperire majoră în criptografie. Acestea rezolvă probleme care înainte erau extrem de dificil de rezolvat, necesitând logistică și încredere în toții terți care interacționau cu sistemul.

# 5. Bibliografie

*Deep Dive Into TLS Handshake*. (2021). Retrieved from Medium: https://levelup.gitconnected.com/deep-dive-into-tls-handshake-e029e28e2eb3

*HTTPS*. (2023). Retrieved from Wikipedia: https://en.wikipedia.org/wiki/HTTPS

*RSA (cryptosystem)*. (2023). Retrieved from Wikipedia: https://en.wikipedia.org/wiki/RSA\_(cryptosystem)

Simmons, G. J. (1979). *Symmetric and Asymmetric Encryption.*

Sombir Singh, S. K. (2013). A Performance Analysis of DES and RSA. *International Journal of Emerging Trends & Technology in Computer Science (IJETTCS)*.

*Transport Layer Security*. (2023). Retrieved from Wikipedia: https://en.wikipedia.org/wiki/Transport\_Layer\_Security

Wikipedia. (2023). *Public-key Cryptography*. Retrieved from Wikipedia: https://en.wikipedia.org/wiki/Public-key\_cryptography