7. CRIPTOGRAFIA

Preliminarii

În tema 7, ”Criptografia”, sunt abordate subiecte importante ce țin de criptografie cum ar fi: cerințele organizației și necesitatea implementării criptografiei; terminologia de bază, principiile și conceptele; utilizarea criptografiei în sistemele informatice și aplicațiile de rețea; managementul cheilor și certificatelor digitale.

Scopul:

- *definirea și descrierea tipurilor de algoritmi criptografici: simetrici, asimetrici și hash; pentru analiza utilizării criptografiei în diverse operațiuni și aplicații.*

Obiectivele educaționale:

- *înțelegerea importanței implementării criptografiei pentru asigurarea securității cibernetice;*
- *studierea algoritmilor de hashing, algoritmilor simetrici și asimetrici de criptare și implementare a acestora în scenarii specifice;*
- *cunoașterea provocărilor în procesul de management al cheilor criptografice.*

Finalitățile de referință:

- *utilizarea algoritmilor criptografici diferiți;*
- *familiarizarea cu metodele de calcul al criptogramelor;*
- *cunoașterea modalităților de determinare a hash-urilor pentru divers conținut: informație, aplicație etc.;*
- *dezvoltarea abilităților de implementare a cheilor criptografice puternice.*

Modalitățile de evaluare

Evaluarea masteranzilor se va efectua în baza testelor formative realizate pe parcursul semestrului de studiu, care vor conține întrebări de tip grilă, întrebări deschise și studii de caz. De asemenea, masteranzii vor îndeplini sarcini practice individuale sau de grup și vor prezenta oral o temă relevantă domeniului de securitate cibernetică. La finele semestrului masteranzii vor susține un examen care va acoperi toate temele din acest suport de curs.

7.1. Concepte de bază

Criptografia este conceptul utilizat pentru crearea și utilizarea unui cifru pentru a securiza informația. Etimologia cuvântului provine din greacă și se traduce ca scriere ascunsă. Prima dovadă scrisă cu privire la acest concept datează din 1900 î.H. Cărturarii egipteni au folosit ieroglife nestandardizate în timp ce inscripționau tăblițe de lut; primul algoritm criptografic asemănător cu cele moderne este cifrul lui Caesar care utiliza cheia criptografică +3 pentru a cripta un anumit text. Astfel, A era înlocuit cu D, B cu E și așa mai departe (figura 7.1). Schimbarea textului original într-un mesaj secret folosind criptografia este cunoscut drept concept de criptare, procesul invers este decriptarea [4].

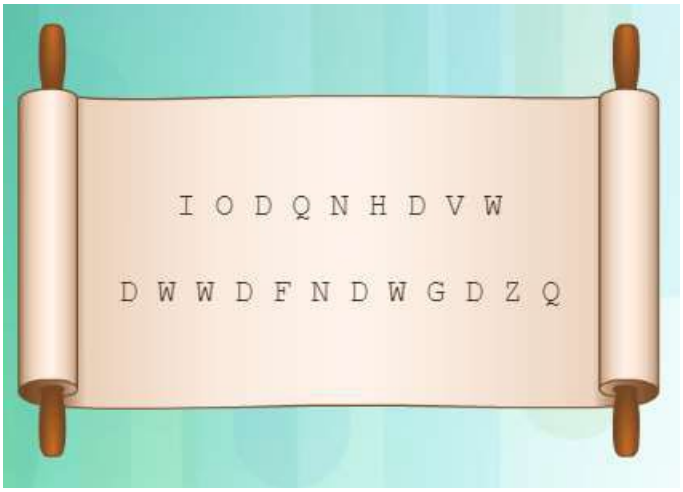

Fig. 7.1. Cifrul lui Caesar

Toate metodele de criptare utilizează o cheie pentru criptarea sau decriptarea unui mesaj (figura 7.2). Cheia este o componentă importantă a algoritmului de criptare. Un algoritm de criptare este pe atât de bun pe cât este cheia folosită. Cu cât cheia este mai complexă, cu atât algoritmul este mai sigur. Managementul cheilor este o piesă importantă în acest proces.

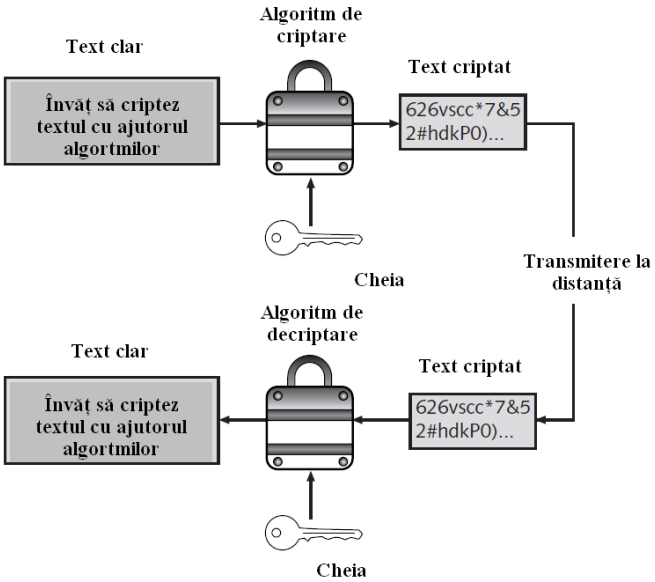

Fig. 7.2. Procesul de criptare a datelor

Criptografia poate oferi protecție de bază de securitate pentru date, deoarece accesul la cheile de criptare poate fi limitat. Criptografia poate asigura cinci protecții de bază:

- *Confidențialitate.* Criptografia poate proteja confidențialitatea informațiilor, asigurându-se că numai părțile autorizate le pot vizualiza. Când informațiile private, cum ar fi lista angajaților care urmează a fi disponibilizați sunt transmise în rețea sau stocate într-un fișier pe server, conținutul acestora poate fi criptat, ceea ce permite doar persoanelor autorizate, care au cheia, de a o vedea.
- *Integritate.* Criptografia poate proteja integritatea informațiilor. Integritatea asigură că informațiile sunt corecte și nici o persoană neautorizată sau software rău intenționat nu au modificat acele date. Deoarece textul cifrat necesită utilizarea unei chei pentru a accesa datele înainte ca acestea să poată fi modificate, criptografia le poate asigura integritatea. De exemplu, lista de angajați care urmează a fi disponibilizați poate fi protejată, astfel încât să nu poată fi adăugate nume sau șterse de personalul neautorizat.

- *Disponibilitate.* Criptografia poate ajuta la asigurarea disponibilității datelor astfel, încât utilizatorii autorizați care dețin cheia să o poată accesa. În loc ca un fișier important să fie stocat pe un hard disk care este blocat într-un seif pentru a preveni accesul neautorizat, un fișier criptat poate fi disponibil imediat de pe un server de fișiere central pentru persoanele autorizate cărora li s-a dat cheia. Lista angajaților care urmează a fi disponibilizați ar putea fi stocată pe un server de rețea și pus la dispoziția angajatului de la Resurse Umane pentru revizuire, deoarece angajatul deține cheia de criptare.
- *Autentificare.* Autentificarea expeditorului poate fi verificată prin criptografie.
- *Non-repudierea.* Criptografia poate impune non-repudierea. Repudierea este definită ca negare; non-repudierea este incapacitatea de a nega. În tehnologia informației, non-repudierea este procesul prin care se dovedește că un utilizator a efectuat o acțiune, cum ar fi trimiterea unui mesaj de e-mail. Non-repudierea împiedică o persoană să fraudeze „renunțarea" la o acțiune. Caracteristicile de non-repudiere ale criptografiei pot împiedica managerul să susțină că nu a trimis niciodată lista de angajați pentru a fi disponibilizați unei terțe părți neautorizate.

Cel mai frecvent utilizate tipuri de criptografie sunt ***cifrurile în blocuri*** și ***cifrurile în flux***. Fiecare metodă diferă în modul în care grupează biți de date pentru a-i cripta. Sistemele criptografice complexe pot combina criptarea pe blocuri cu criptarea în flux în același proces.

Cifrarea în flux

Cifrarea în flux criptează un text clar, caracter cu caracter, octet cu octet sau câte un bit la un moment dat, după cum se arată în figura 7.3. Cifrurile în flux pot fi interpretate ca blocuri cu dimensiunea de un bit. Cu un cifru de flux, transformarea acestor unități de text clar mai mici variază în funcție de momentul în care acestea se întâlnesc în timpul procesului de criptare. Cifrurile de flux pot fi mult mai rapide decât cifrurile bloc și, în general, nu măresc dimensiunea mesajului, deoarece pot cripta un număr arbitrar de biți. A5 este un exemplu de cifru în flux care asigură confidențialitatea vocii și criptează comunicațiile prin telefonul mobil. De asemenea, este posibil să se utilizeze algoritmul DES în modul criptare în flux.

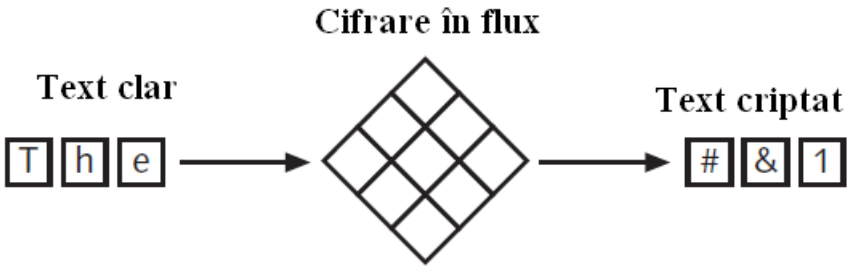

Fig. 7.3. Cifrare în flux

Cel mai simplu tip de cifru în flux este un cifru de substituție. Cifrurile de substituție pur și simplu înlocuiesc o literă sau un caracter cu altul (un cifru de substituție monoalfabetic), așa cum se arată în figura 7.4. Un cifru de flux mai complex, care poate fi mai dificil de spart, este un cifru de substituție homoalfabetic ce mapează un singur caracter din textul clar cu mai multe caractere text criptat. De exemplu, un F se poate mapa la ILS.

Fig. 7.4. Substituția

Cifrarea în bloc

Alți algoritmi folosesc un cifru în bloc. În timp ce un cifru în flux funcționează prin criptarea unui caracter la un moment dat, un cifru în bloc manipulează un întreg bloc de text simplu la un moment dat. Textul clar este împărțit în blocuri separate de la 8 până la 16 octeți, apoi fiecare bloc este criptat independent. Pentru securitate suplimentară, blocurile pot fi randomizate. Cifrarea în bloc generează de obicei date de ieșire care sunt mai mari decât datele de intrare, deoarece textul cifrat trebuie să fie mai mare decât dimensiunea blocului. De exemplu, standardul de criptare a datelor (DES) este un algoritm simetric care criptează blocurile de 64 de biți, utilizând o cheie de 56 de biți. Pentru a realiza acest lucru, algoritmul de blocuri ia datele, un bloc la un moment dat, de exemplu, 8 octeți per bloc, până când întregul bloc este plin. Dacă există mai puține date de intrare decât un bloc complet, algoritmul adaugă date artificiale sau semne false până când blocul are 64 de biți.

Cifrurile în flux și bloc au fiecare avantaje și dezavantaje. Cifrul în flux este rapid când textul simplu este scurt, dar poate consuma mult mai multă putere de procesare dacă textul clar este lung. În plus, cifrurile de flux sunt mai predispuse la atac, deoarece algoritmul care generează fluxul nu variază; singura modificare este textul simplu. Datorită acestei consistențe, un atacator poate examina fluxurile și poate determina cheia. Cifrurile bloc sunt considerate mai sigure, deoarece ieșirea este aleatorie. Când este utilizat un cifru bloc, cifrul este resetat la starea inițială după procesarea fiecărui bloc. Rezultă textul cifrat care este mai greu de spart.

Algoritmii criptografici pot fi divizați în trei mari categorii:

- algoritmi de hashing;
- algoritmi criptografici simetrici;
- algoritmi criptografici asimetrici.

7.2. Algoritmi hash

Cel mai elementar tip de algoritm criptografic este un algoritm hash unidirecțional. ***Un algoritm hash creează o „amprentă digitală" unică a unui set de date și se numește în mod obișnuit hashing***. Această amprentă, numită digest (uneori numit un mesaj digest sau hash), reprezintă conținutul. Deși hashingul este considerat un algoritm criptografic, scopul său nu este de a crea text cifrat care poate fi decriptat ulterior. Hashingul este „unidirecțional", în sensul că conținutul său nu poate fi folosit pentru a dezvălui setul original de date. Hashingul este folosit în primul rând în scopuri de comparație. Un hash securizat care este creat dintr-un set de date nu poate fi inversat. De exemplu, dacă 12 este amplificat cu 34 rezultatul este 408. Dacă unui utilizator i s-a cerut să determine cele două numere utilizate pentru crearea numărului 408, nu ar fi posibil a obține originalul, deoarece pot exista foarte multe combinații: 204+204, 407+1, 999–591, 361+47 etc.

Un algoritm de hashing este considerat sigur dacă are următoarele caracteristici:

- ***Mărime fixă.*** Un rezumat al unui set scurt de date ar trebui să producă aceeași dimensiune ca și un rezumat al unui set lung de date. De exemplu, un rezumat al unei singure litere **a** este 86be7afa339d0fc7cfc785e72f578d33, în timp ce un rezumat de 1 milion de apariții ale literei **a** este 4a7f5723f954eba1216c9d8f6320431f, ambele hashuri având aceeași lungime.
- ***Unic.*** Două seturi diferite de date nu pot produce același rezumat, care este cunoscut ca o coliziune. Schimbarea unei singure litere într-un set de date ar trebui să producă un digest diferit. De exemplu, rezumatul cuvântului "*Duminică*" este *0d716e73a2a7910bd4ae63407056d79b*, în timp ce rezumatul cuvântului "duminică" este *3464eb71bd7a4377967a30da798 a1b54*.
- ***Original.*** Ar trebui să fie imposibil să se producă un set de date care are o valoare hash predefinită.
- ***Sigur.*** Hashul rezultat nu poate fi inversat pentru a determina textul simplu original. Hashingul este folosit în primul rând pentru a determina integritatea unui mesaj sau a

conținutului unui fișier. În acest caz, rezumatul servește drept verificare dacă conținutul original nu s-a schimbat (figura 7.5). Valorile digest sunt adesea postate pe site-uri web pentru a verifica integritatea fișierelor care pot fi descărcate. Un utilizator poate crea un rezumat al fișierului după ce acesta a fost descărcat, apoi să compare acea valoare cu valoarea inițială a rezumatului postată pe site. Potrivirea perfectă indică faptul că integritatea fișierului a fost păstrată.

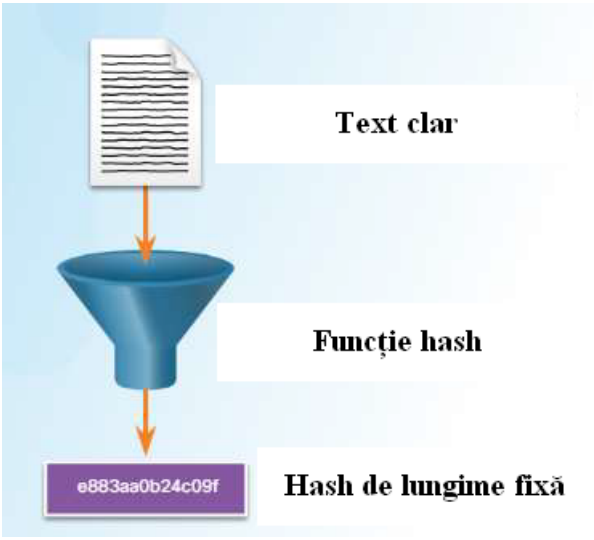

Fig. 7.5. Obținere valoare hash [3]

O variantă care oferă securitate îmbunătățită este codul de autentificare a mesajelor hashed (HMAC). Un cod de autentificare a mesajelor (MAC) combină o „cheie secretă partajată" doar între expeditor și receptor [24]. Când receptorul primește mesajul, știe că a venit de la expeditor pentru că numai el are cheia secretă. Aceasta servește la autentificarea expeditorului mesajului. Cu toate acestea, un MAC nu criptează mesajul în sine.

Un HMAC este un cod de autentificare a mesajelor bazat pe hash în care este aplicată o funcție hash atât la cheie, cât și la mesaj. HMAC este utilizat pe scară largă de protocoalele de securitate Internet pentru a verifica integritatea datelor transmise în timpul comunicațiilor securizate. Hashingul poate fi folosit pentru a verifica doar integritatea datelor.

Funcțiile Hash sunt utile pentru a se asigura că o eroare a utilizatorului sau de comunicare nu a modificat accidental datele. De exemplu, un expeditor ar putea dori să se asigure că nimeni nu modifică un mesaj în drum spre destinatar. Dispozitivul de trimitere introduce mesajul într-un algoritm de hash și calculează digestul sau amprenta cu lungime fixă.

Există mai mulți algoritmi de hashing moderni utilizați astăzi pe scară largă. Doi dintre cei mai populari sunt MD5 și SHA.

MD5

Ron Rivest a dezvoltat algoritmul de hash MD5, iar astăzi îl folosesc mai multe aplicații Internet. MD5 este o funcție one-way prin care se calculează ușor un hash din datele de intrare, dar este foarte dificilă calcularea datelor de intrare dintr-o valoare hash. MD5 produce o valoare hash de 128 biți. Malware-ul Flame a compromis securitatea MD5 în 2012. Autorii malware-ului Flame au folosit o coliziune MD5 pentru a crea un certificat de semnare a codurilor Windows.

SHA

Institutul Național de Standarde și Tehnologie din SUA (NIST) a dezvoltat algoritmul SHA. Algoritmul a fost specificat în Standardul Secure Hash (SHS). NIST a publicat SHA-1 în 1994. Între timp, SHA-2 a înlocuit SHA-1 cu patru funcții de hash suplimentare care alcătuiesc familia SHA și cresc rezilienta algoritmului:

- SHA-224 (224 biți);
- SHA-256 (256 biți);
- SHA-384 (384 biți);
- SHA-512 (512 biți).

SHA-2 este un algoritm mai puternic și înlocuiește MD5. Algoritmii SHA-256, SHA-384 și SHA-512 sunt algoritmi de generație următoare.

Alți algoritmi criptografici apăruți de curând sunt Whirlpool și RIPEMD.

Whirlpool este o funcție hash criptografică relativ recentă care a obținut recunoaștere internațională și adoptarea de către organizațiile de standardizare, inclusiv internaționale – Organizația pentru Standardizare (ISO). Numit după prima galaxie recunoscută ca o structură în spirală, creează un digest de 512 biți. Whirlpool este implementat în câteva aplicații comerciale noi de criptografie.

RACE Integrity Primitives Evaluation Message Digest (RIPEMD) este un alt algoritm hash dezvoltat de Centrul de Cercetare și Dezvoltare în Comunicații Avansate (RACE), organizație afiliată la Uniunea Europeană (UE), proiectat după algoritmul vechi MD4.

Pentru a sparge un hash, un atacator trebuie să ghicească parola. Primele două atacuri utilizate pentru a ghici parolele sunt dicționarul și atacurile cu forță brută. ***Un atac de dicționar utilizează un fișier care conține cuvinte, fraze și parole comune***. Fișierul are sumele de control calculate. Un atac de dicționar compară hash-urile din fișier cu hash-urile parolei. Dacă un hash se potrivește, atacatorul va cunoaște un grup de parole potențial bune. ***Un atac brute-force încearcă orice combinație posibilă de caractere până la o anumită lungime.*** Un atac de forță brută durează mult timp și consumă intens resursele procesorului, dar este doar o chestiune de timp înainte ca această metodă să descopere parola. Parolele trebuie să fie suficient de lungi pentru a face timpul necesar de execuție a atacului de forță brută prea lung pentru a se merita. Utilizarea hashing-ului și păstrarea digestului în locul parolei pe servere îngreunează munca infractorului de a descoperi parolele.

O tehnică utilizată pentru păstrarea hash-ului parolelor pe servere este saltarea. Saltarea face hash-ul parolei mai sigur. Dacă doi utilizatori au aceeași parolă, aceștia vor avea, de asemenea, același hash. Sarea, care este un șir de caractere aleatoare, este o intrare suplimentară a parolei adăugată înainte de hashing. Acest lucru creează un rezultat hash diferit pentru două parole identice. În baza de date se stochează atât hash-ul, cât și sarea.

Utilizarea algoritmilor de hash

Funcțiile hash criptografice sunt utilizate în următoarele situații:

- pentru a furniza o dovadă de autenticitate când este utilizată o cheie de autentificare secretă simetrică, cum ar fi autentificarea protocolului IP Security (IPsec) sau autentificarea protocolului de rutare;
- pentru a furniza autentificare prin generarea de răspunsuri unice la provocările din protocoalele de autentificare;
- pentru a furniza dovada verificării integrității mesajelor, cum ar fi cele utilizate în contractele semnate digital și certificatele de infrastructură de cheie publică (PKI), cum ar fi cele acceptate când accesați un site securizat utilizând un browser web.

Când se selectează un algoritm de hash este recomandat a utiliza cel puțin SHA-256 sau o versiune superioară, deoarece acestea sunt în prezent cele mai sigure. E necesar a evita SHA-1 și MD5 datorită descoperirii deficiențelor de securitate. În rețelele de producție se implementează SHA-256 sau mai mare. În timp ce hashing-ul poate detecta modificări accidentale, nu se poate proteja împotriva modificărilor intenționate. Nu există informații unice de identificare de la expeditor în procedura de hash. Acest lucru înseamnă că oricine poate calcula un hash pentru orice date, atât timp cât acesta cunoaște funcția hash corectă. De exemplu, când un mesaj traversează rețeaua, un potențial atacator ar putea intercepta mesajul, l-ar putea modifica, recalculând hash-ul și adăugând hash-ul nou calculat la mesaj. Dispozitivul

de recepție va valida hash-ul adăugat. Prin urmare, hash-ul este vulnerabil la atacurile "man-in-the-middle" și nu asigură securitatea datelor transmise.

7.3. Algoritmi simetrici de criptare

Algoritmii criptografici originali pentru criptarea și decriptarea datelor sunt **algoritmii criptografici simetrici**. Algoritmii criptografici simetrici folosesc aceeași cheie unică pentru criptarea și decriptarea unui document. Spre deosebire de hashing, în care hash-ul nu este destinat să fie decriptat, algoritmii simetrici sunt proiectați pentru a cripta și decripta textul [8]. Un text clar criptat cu cheia privată a lui Dan va fi decriptat de către Andrei cu aceeași cheie. Prin urmare, este esențial ca cheia să fie păstrată privată (confidențială), deoarece dacă un atacator ar obține cheia ar putea citi toate documentele criptate. Din acest motiv, criptarea simetrică este numită și criptare cu cheie privată. Criptarea simetrică este ilustrată în figura 7.6, unde sunt folosite chei identice pentru a cripta și decripta un document.

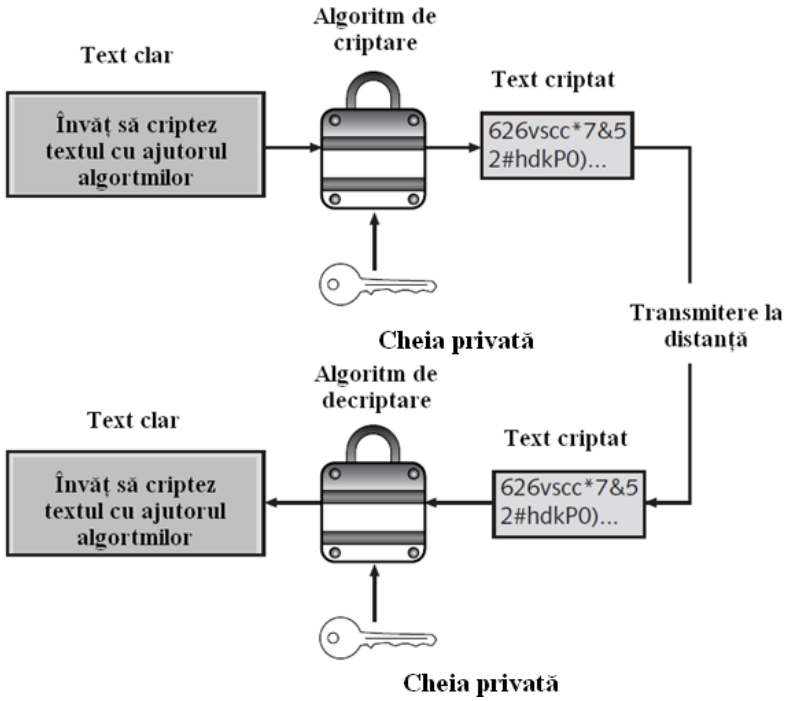

Fig. 7.6. Criptare simetrică

Criptarea simetrică este utilizată de numeroase sisteme de criptare. Standardele de criptare care utilizează criptarea simetrică sunt:

- ***3DES (Triple DES)***: standardul digital de criptare (DES) este un cifru simetric de tip bloc cu dimensiunea blocurilor pe 64 biți care utilizează o cheie pe 56 biți. Este nevoie de un bloc de 64 biți de text clar ca intrare și de ieșire a unui bloc de 64 biți de text criptat. Funcționează întotdeauna pe blocuri de dimensiuni egale și utilizează atât permutări, cât și substituții în algoritm. O permutare este o modalitate de a aranja toate elementele unui set. Triple DES criptează datele de trei ori și folosește o cheie diferită pentru cel puțin una dintre cele trei operații, având o dimensiune a cheii cumulativă de 112-168 biți. 3DES este rezistent la atac, dar este mult mai lent decât DES.
- ***IDEA***: algoritmul internațional de criptare a datelor (IDEA) utilizează blocuri pe 64 biți și chei de 128 biți. IDEA efectuează opt runde de transformări pe fiecare dintre cele 16 blocuri care rezultă din împărțirea fiecărui bloc de 64 biți. IDEA a fost înlocuitorul lui DES, iar acum PGP (Pretty Good Privacy) îl folosește. PGP este un program care oferă confidențialitate și autentificare pentru comunicațiile de date. GNU Privacy Guard (GPG) este o versiune gratuită, licențiată a PGP.

- ***AES***: standardul avansat de criptare (AES) are o dimensiune bloc fixă de 128 biți, cu o dimensiune a cheii de 128, 192 sau 256 biți. Institutul Național de Standarde și Tehnologie (NIST) a aprobat algoritmul AES în decembrie 2001. Guvernul american utilizează AES pentru a proteja informațiile clasificate. AES este un algoritm puternic care utilizează chei lungi. AES este mai rapid decât DES și 3DES, deci, oferă atât o soluție pentru aplicațiile software, cât și utilizarea intensă hardware în firewall-uri și routere.

  Alți algoritmi simetrici sunt *Skipjack* (dezvoltat de NSA), *Blowfish* și *Twofish*.

  Utilizarea algoritmilor criptografici simetrici

Industria de plăți electronice utilizează 3DES. Sistemele de operare utilizează DES pentru a proteja fișierele utilizatorilor și datele de sistem cu parole. Cele mai multe sisteme de fișiere criptate, cum ar fi NTFS, utilizează AES.

7.4. Algoritmi asimetrici de criptare

Criptarea asimetrică, denumită și criptare cu chei publice, utilizează o cheie pentru criptare diferită de cea folosită pentru decriptare. Un infractor nu poate calcula cheia de decriptare bazată pe cunoașterea cheii de criptare, și invers, într-un timp rezonabil.

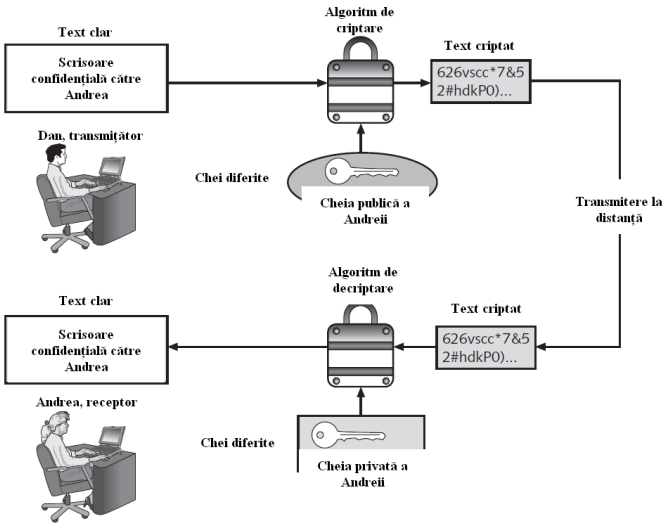

Fig. 7.7. Criptare asimetrică

Dacă Andreea și Dan schimbă un mesaj secret folosind criptarea cu chei publice, folosesc un algoritm asimetric ca în figura 7.7. De data aceasta, Andreea și Dan nu schimbă cheile înainte de a trimite mesaje secrete. În schimb, Andreea și Dan au fiecare un lacăt separat, cu taste separate. Pentru ca Dan să trimită un mesaj secret Andreei, trebuie să o contacteze mai întâi și să-i ceară să-i trimită lacătul deschis (cheia publică). Andreea îi trimite lacătul, dar își păstrează cheia. Când Dan primește lacătul, scrie mesajul secret și îl pune într-o cutie mică, încuie lacătul fără cheie și transmite cutia cu mesajul încuiat către Andreea. Când Dan blochează lacătul, nu îl mai poate deschide, pentru că nu are o cheie. El trimite poșta către Andreea și, în timp ce cutia călătorește prin sistemul de poștă, nimeni nu o poate deschide. Când Andreea primește caseta, ea poate folosi cheia sa privată pentru a debloca lacătul și a prelua mesajul de la Dan.

Modelele asimetrice utilizează formule pe care oricine le poate găsi. Perechea cheilor independente este ceea ce face ca acești algoritmi să fie siguri.

Algoritmii asimetrici cel mai des utilizați sunt:

- ***RSA (Rivest-Shamir-Adleman)*** - utilizează produsul a două numere prime foarte mari cu o lungime egală între 100 și 200 de cifre. Browserele utilizează RSA pentru a stabili o conexiune sigură.
- ***Diffie-Hellman*** - oferă o metodă de schimb electronic pentru partajarea cheii secrete. Protocoalele securizate cum ar fi Secure Sockets Layer (SSL), Secure Transport Layer (TLS), Secure Shell (SSH) și Securitatea Protocolului de Internet (IPsec) utilizează Diffie-Hellman.
- ***ElGamal*** - utilizează standardul guvernamental al SUA pentru semnăturile digitale. Acest algoritm este gratuit pentru utilizare, deoarece nimeni nu deține brevetul.
- ***Criptografie cu folosirea curbelor eliptice (ECC)*** - utilizează curbele eliptice ca parte a algoritmului. În SUA, Agenția Națională de Securitate utilizează ECC pentru generarea semnăturii digitale și schimbul de chei criptografice.

Utilizarea algoritmilor criptografici asimetrici
Patru protocoale folosesc algoritmii cu chei asimetrice [3]:
- ***Internet Key Exchange (IKE)* –** componentă fundamentală a rețelelor virtuale private VPN (VPN).
- ***Secure Socket Layer (SSL)* –** mijloc de implementare a criptografiei într-un browser web.
- ***Secure Shell (SSH)* –** protocol care oferă o conexiune securizată de acces la distanță la dispozitivele de rețea.
- ***Pretty Good Privacy (PGP)* –** program de calculator care oferă confidențialitate criptografică și autentificare pentru a spori securitatea comunicărilor prin e-mail.

7.5. Managementul cheilor de criptare

Gestiunea cheilor include:
- generarea cheii – în cifrurile moderne este făcută automat pentru a asigura un bun proces de alegere a cheii printr-un algoritm aleator;
- verificarea cheii – în orice algoritm de criptare există chei care sunt slabe din punct de vedere al recunoașterii lor, de aceea ele trebuie identificate și omise;
- schimb de chei – managementul cheilor trebuie să asigure un schimb de chei sigur între 2 entități chiar și printr-un mediu nesigur;
- stocarea cheilor în sistemele de operare moderne care sunt multi-user poate fi o problemă, deoarece stocarea are loc pe HDD, iar un program de tip Troian poate obține acces la aceasta;
- durata de viață a cheilor – utilizarea cheilor cu durată mică de viață mărește securitatea cifrului folosit la conexiunile de mare viteză. În IPSec durata de viață a cheii este de 24 ore;
- revocarea și distrugerea cheilor – revocarea are loc în cazul când toate entitățile implicate sunt informate că cheia a fost compromisă și nu mai poate fi folosită, distrugerea are loc când toate cheile vechi sunt definitiv distruse ca să nu poată fi recuperate de către atacatori.

Managementul cheilor este cea mai dificilă parte a proiectării unui sistem criptografic. Multe cripto-sisteme au eșuat din cauza unor greșeli în procedurile lor de gestionare a cheilor. În practică, majoritatea atacurilor asupra sistemelor criptografice vizează managementul cheilor, mai degrabă decât algoritmul criptografic în sine.

Există câteva caracteristici esențiale ale managementului cheilor care trebuie luate în considerare, și anume:
- lungimea cheii, numită și dimensiunea cheii, aceasta este măsurată în biți;

- spațiul cheilor - este numărul de posibilități pe care le poate genera o anumită lungime a cheilor.

În ceea ce privește creșterea lungimii cheii, spațiul pentru chei crește exponențial. Spațiul de chei al unui algoritm este setul tuturor valorilor-cheie posibile. Cheile mai lungi sunt mai sigure; cu toate acestea, ele consumă mai multe resurse. Aproape fiecare algoritm are câteva taste slabe în spațiul său de chei care permit unui infractor să spargă codul criptat printr-o comandă rapidă.

7.6. Criptarea simetrică versus asimetrică

Este important a înțelege diferențele dintre metodele de criptare simetrice și asimetrice. Sistemele de criptare simetrice sunt mai eficiente și pot gestiona mai multe date. Cu toate acestea, gestionarea cheilor cu sisteme de criptare simetrice este mai problematică și mai grea. Criptografia asimetrică este mai eficientă pentru a proteja confidențialitatea unor cantități mici de date, iar mărimea și viteza acesteia o fac mai sigură pentru sarcini precum schimbul de chei electronice, care reprezintă o cantitate mică de date, în loc să cripteze blocuri mari de date.

Păstrarea confidențialității este importantă atât pentru datele în repaus, cât și pentru datele în tranzit. În ambele cazuri, criptarea simetrică este favorizată de viteză și simplitatea algoritmului. Unii algoritmi asimetrici pot mări semnificativ dimensiunea textului criptat. Prin urmare, în cazul datelor în mișcare se utilizează criptografia cu chei publice pentru a schimba cheia secretă între expeditor și destinatar, apoi criptografia simetrică pentru a asigura confidențialitatea datelor trimise.

Întrebări și subiecte pentru aprofundarea cunoștințelor

1. Cum poate fi definit conceptul de HMAC? Cum are loc întreg procesul de utilizare a HMAC?
2. În ce cazuri specifice este utilizată criptarea simetrică? Avantajele și dezavantajele.
3. În ce cazuri specifice este utilizată criptarea asimetrică? Avantajele și dezavantajele.
4. Analizați aplicațiile unde se folosesc standardele de criptare simetrică. Care sunt acestea și care este impactul standardelor de criptare simetrică?
5. Analizați aplicațiile unde se folosesc standardele de criptare asimetrică. Care sunt acestea și care este impactul standardelor de criptare asimetrică?
6. Enumerați și explicați caracteristicile algoritmilor de hash.
7. Cum se generează semnătura digitală? Care este rolul algoritmilor de hash în acest proces?
8. Dacă Dan dorește să trimită un mesaj securizat Andreei folosind criptografia asimetrică, ce cheie folosește pentru a cripta mesajul?
9. Dan vrea să-și trimită o copie a mesajului criptat asimetric pe care i l-a trimis Andreei. De ce chei are nevoie și în ce etapă trebuie să le aplice?
10. Dan vrea să-i trimită Andreei un mesaj cu semnătură digitală Ce cheie va utiliza pentru aceasta? Ce cheie va utiliza Andreea pentru a citi mesajul semnat electronic trimis de Dan?