

- utilizare simultană, decide cine folosește resursa și cât timp;
  - alocarea dispozitivelor periferice și inițiază operația de intrare/ieșire;
  - dezalocarea dispozitivelor periferice la terminarea execuției operațiilor de intrare/ieșire.
- 4) *gestiunea informației*, care se materializează în:
- evidențierea resursei (informația), localizarea ei, utilizarea, starea, etc.;
  - decide cine utilizează informația, impune protecția cerută și oferă rutine de acces necesare;
  - alocă resursele prin deschiderea fișierului;
  - dezalocă resursele prin închiderea fișierului.

## **SECURITATEA ȘI PROTECȚIA INFORȚIEI ÎN SIST. INF.**

### ***Elaborarea unei arhitecturi de securitate pentru sistemele informatice (Popa, S., Securitatea și protecția informației în sist. inf., Note de curs, pag. 13-17)***

Elaborarea unei arhitecturi de securitate pentru un SI se constituie într-un proces a cărui parcurgere este absolut necesară a fi realizată atunci când se dorește asigurarea securității și protecției informațiilor din sistemul respectiv.

În general, metodologia elaborării unei astfel de arhitecturi de securitate se bazează pe următoarele trei etape, și anume: *identificarea tuturor amenințărilor împotriva cărora este cerută protecția, stabilirea politicii de securitate, selecția serviciilor și mecanismelor de securitate.*

#### **1.5.1 Identificarea amenințărilor**

Această primă etapă este una de analiză și constă din următorii trei pași: *analiza vulnerabilităților, evaluarea amenințărilor și analiza riscurilor.* Pe baza rezultatelor obținute în cadrul acestei etape se poate trece apoi la stabilirea cerințelor de securitate ale sistemului.

##### *Analiza vulnerabilităților*

Sistemele informaționale nu sunt concepute, de regulă, având la bază cerințe de securitate, ci necesități funcționale, de aceea ele conțin o multitudine de vulnerabilități ce pot fi exploatare de către o gamă largă de amenințări. Ca urmare, efectuarea unei analize a vulnerabilităților, este necesară, și ea are ca scop identificarea vulnerabilităților, adică a elementelor potențial slabe din în cadrul sistemului, serviciilor sau aplicațiilor care le fac susceptibile la amenințări și care pot fi critice pentru securitate. Astfel de vulnerabilități sunt:

- echipamentele hardware;
- comunicațiile;
- mediile de stocare a informației;
- software-ul de aplicație, dacă este utilizat înainte de a fi complet testat;
- software-ul de sistem (de bază);
- personalul, atunci când este insuficient instruit în ceea ce privește riscurile asociate cu procesarea informației.

Vulnerabilitățile pot fi rezultatul unor erori de proiectare, de implementare sau de administrare/utilizare. Cele mai dificil de corectat și eliminat sunt cele de proiectare.

Mărimea gradului de vulnerabilitate al unui SI este direct proporțională cu numărul său de utilizatori. Aceasta înseamnă că, cu cât sistemul este mai disponibil prin serviciile oferite, unui număr mai mare de utilizatori, cu atât el devine mai vulnerabil. În particular, sistemele ce se bazează pe rețele deschise, cum este Internet-ul, în virtutea faptului că furnizează acces unui număr foarte mare de utilizatori, sunt deosebit de vulnerabile. Efectuarea unei analize a

vulnerabilităților cu un grad de consistență și obiectivitate ridicat astfel încât toate vulnerabilitățile potențiale ale sistemului să poată fi identificate, se constituie într-o încercare dificilă. De aceea o bună investigație constă, pe lângă studiul documentelor și specificațiilor caracteristice, în observarea sistemului și în purtarea de discuții cu cei ce cunosc bine detaliile sale.

#### *Evaluarea amenințărilor*

Scopul acestei evaluări îl constituie identificarea în cadrul SI, a amenințărilor, adică a atacurilor posibile la care acestea pot fi supuse datorită vulnerabilităților pe care le prezintă. Astfel, plecând de la rezultatele analizei vulnerabilităților și luând în considerare motivațiile și intențiile atacatorilor, precum și condițiile de mediu în care anumite componente ale sistemului își derulează activitatea se pot stabili următoarele categorii de amenințări potențiale:

1. Amenințări asupra sistemului: utilizarea neautorizată incluzând abuzul de drepturile utilizatorilor autorizați, accesul neautorizat la nivelul sistemului, refuzul de servicii;
2. Amenințări împotriva informației din sistem: alterarea sau duplicarea datelor sau software-ului de aplicație și de bază;
3. Amenințări asupra comunicațiilor: alterarea (copierea, distrugerea, modificarea, substituirea) mesajelor precum și schimbarea destinației sau sursei acestora;
4. Amenințări de la utilizatorii distribuiți: utilizatori ce folosesc un serviciu ce este disponibil pe Internet și care nu sunt cunoscuți, utilizatori care nu pot fi de încredere, sisteme terțe (autorități de certificare) de neîncredere, repudierea mesajelor și tranzacțiilor;
5. Amenințări externe și accidentale: distrugerea echipamentelor, incendiul, inundația, penele de tensiune, funcționări defectuoase;
6. Amenințări datorate personalului: neglijența, rea-voință, inițierea de activități neautorizate sau de activități ce depășesc competențele atribuite.

#### *Analiza riscurilor*

Deciziile relative la tratarea amenințărilor trebuie să fie fundamentate pe o evaluare a riscurilor în cauză. Pentru aceasta, amenințările trebuie să fie caracterizate, de fiecare dată când este posibil, printr-o categorie de gravitate și printr-un nivel de probabilitate.

Categoriile de gravitate furnizează o măsură calitativă, plecând de la estimarea celei mai grave consecințe posibile a unui eveniment nedorit (amenințare). Aceste categorii pot fi: catastrofică (distrugerea totală a sistemului), critică (distrugerii grave ale sistemului), marginală (distrugerii reduse), neglijabilă.

Probabilitatea unei amenințări în cursul duratei de viață prevăzute a unui sistem poate fi definită ca fiind numărul de apariții al evenimentului nedorit (amenințării), raportat la o durată de timp. Cuantificarea probabilității este în general imposibilă în faza de concepție a sistemului, dar se poate da o apreciere calitativă bazată pe analize, cercetări, și pe exploatarea experienței acumulate în cadrul altor sisteme similare. O evaluare calitativă este de exemplu, frecvența: probabilă, ocazională, rară, improbabilă.

Plecând de la gravitate și de la probabilitatea riscului asociat fiecărei amenințări, se poate stabili o ordine de prioritate pentru tratarea lor.

Deoarece măsurile de securitate și protecție ce se aplică în cadrul unui SI au întotdeauna un cost, și per ansamblul sistemului acesta poate să fie foarte important. De aceea, pentru a determina gradul în care aceste costuri se justifică, este necesară efectuarea unei proceduri numită "*analiză a riscurilor*". În cadrul ei sunt trecute în revistă diferitele amenințări relevate, estimându-se frecvențele de apariție asociate precum și mărimile pierderilor corespondente ce le implică. Plecând de la aceste date, este previzionată "*pierderea anuală*" asociată fiecăreia dintre amenințări. Ideea ce stă la baza evaluării respective este că, prin compararea pierderii anuale

previzionate și a costului furnizării unei protecții adecvate se vor putea determina care sunt aspectele de securitate ce trebuie tratate, adică se vor putea identifica punctele în care amenințarea este cea mai mare și investiția de securitate este cea mai potrivită în a produce rezultate. O formulă pentru calcularea pierderii anuale previzionate, bazată pe combinarea frecvenței de apariție a unei amenințări (F) cu mărimea pierderii corespondente (P), este următoarea:

$$\text{Pierdere anuală previzionată} = 10^{\frac{(F+P)}{3000}}$$

Pentru fiecare dintre cele două mărimi utilizate în cadrul formulei se folosește câte o scară de reprezentare a magnitudinii.

**F Estimare pentru frecvența de apariție a unei amenințări**

- 0 virtual imposibilă
- 1 o dată la 300 de ani
- 2 o dată la 30 de ani
- 3 o dată la 3 ani (1000 de zile lucrătoare)
- 4 o dată la 3 luni (100 de zile lucrătoare)
- 5 o dată la 10 zile
- 6 o dată în fiecare zi
- 7 o dată la fiecare 2 ore (de 10 ori pe zi)
- 8 o dată la fiecare 15 minute (de 100 ori pe zi)

**P Estimare pentru mărimea pierderii corespondente**

- 0 mai mică de 1USD
- 1 10 USD
- 2 100 USD
- 3 1000 USD
- 4 10000 USD
- 5 100000 USD
- 6 1000000 USD
- 7 10000000 USD
- 8 100000000 USD

Beneficiile unei analize a riscurilor sunt direct proporționale cu gradul de credibilitate al măsurătorilor, și dacă acestea sunt efectuate la timp și pot fi cuantificate. Dar în practică riscul poate, de regulă, să fie cuantificat cu dificultate, iar analizele sunt costisitoare. Cuantificarea necesită statistici despre frecvența amenințărilor și mărimea pierderilor înregistrate în sisteme similare. Asemenea statistici sunt, în general, dificil de obținut, iar frecvența pierderilor poate fi prea mică pentru a putea fi utilă sau nu poate fi aplicabilă unui sistem particular. În plus, incidentele privind pierderile din cadrul unui sistem sunt de obicei neraportate sau nedetectate.

### 1.5.2 Stabilirea politicii de securitate

O politică de securitate aferentă unui SI definește toleranța la risc a acestuia, identificând totodată un număr de principii și linii directoare de urmat în scopul asigurării cerințelor de securitate ale sistemului. Prin aceasta ea garantează că:

1. Sistemul va prezenta o securitate conform cerințelor, iar această securitate va fi obținută într-un timp și cu costuri convenabile;
2. Amenințările asociate sistemului sunt identificate și evaluate iar riscul corespondent este redus la un nivel acceptabil;
3. Experiența acumulată în materie de securitate, asupra altor sisteme, este luată în calcul și utilizată;
4. Cerințele tehnice și operaționale nu antrenează riscuri inacceptabile;
5. Toate incidentele întâlnite sunt exploatate, iar experiența acumulată este înregistrată în

bănci de date, în manuale de utilizare, în norme și reglementări.

De asemenea, politica de securitate trebuie să specifice necesarul de măsuri de securitate, să definească și să asigneze responsabilitățile pentru implementarea și întărirea acestora.

În general, măsurile de securitate se referă la o combinație de măsuri procedurale, logice și fizice ce au ca scop prevenirea și detecția amenințărilor la care este supus un SI, precum și corecția efectelor acestora asupra lui.

*Măsurile procedurale* includ o serie de controale administrative ce sunt specificate prin intermediul unor instrucțiuni, reglementări și norme și care se referă la aspectele de gestiune și de operare a sistemelor și componentelor acestora. Astfel, sarcini cum ar fi gestiunea cheilor criptografice, sunt subiectul unor controale de acces strict și sunt separate geografic și administrativ.

*Măsurile logice* sunt cele destinate asigurării confidențialității, integrității, autentificării și non-repudierii informațiilor din sistem, la care se adaugă asocierea de responsabilități pentru manipularea și procesarea acestora. Măsurile respective includ tehnicile criptografice.

*Măsurile fizice* se referă la protecția împotriva amenințărilor externe ce pot fi exercitate asupra SI. Ele se prezintă sub forma controlului accesului fizic în localurile tehnice, la echipamente, incluzând și protecția împotriva catastrofelor (incendiu, inundație, explozii).

Măsurile de securitate, procedurale, logice și fizice sunt complementare, un nivel înalt de securitate asigurat prin adoptarea de măsuri dintr-o anumită categorie, poate necesita un nivel de securitate mai scăzut asigurat prin măsurile adoptate din altă categorie. De asemenea, pentru a avea succes și a putea stăpâni riscurile ce decurg din amenințările expuse anterior, aplicarea măsurilor respective trebuie făcută într-o manieră coordonată, menținându-se în același timp obiectivele funcționale ale SI.

### **1.5.3 Selecția serviciilor și mecanismelor de securitate**

Odată stabilite obiectivele politicii de securitate, următoarea etapă o constituie selecția serviciilor de securitate. Acestea sunt funcții individuale care sporesc securitatea sistemului.

Serviciile de securitate necesare a fi asigurate în cadrul SI sunt: confidențialitatea, autentificarea, integritatea, certificarea și non-repudierea. Anumite dintre ele pot fi subîmpărțite în mai multe sub-servicii, în funcție de metoda de comunicație utilizată și de particularitățile serviciului cerut.

Fiecare serviciu poate fi implementat prin diverse metode, numite mecanisme de securitate care detaliază maniera în care serviciile sunt furnizate. La rândul lor, mecanismele ce permit furnizarea acestor servicii depind de protocoalele criptografice utilizate.

Când politica de securitate pentru un SI a fost definită, în continuare trebuie luate decizii privind implementarea acesteia. Pentru aceasta se pornește de la o apreciere a riscurilor asociate cu amenințările și vulnerabilitățile identificate în cadrul procesului de formulare a politicii de securitate, în relație cu valoarea informației care trebuie protejată. De asemenea se va ține seama dacă anumite componente ale sistemului respectiv au fost evaluate sau certificate de către organisme recunoscute, precum și de existența unor constrângeri legale, organizaționale, contractuale, sau a unora impuse de anumite reglementări.

### ***Conceptul de semnătură digitală***

***(Popa, S., Securitatea și protecția informației în sist. inf., Note de curs, pag. 29-34)***

### **2.3.1 Rolul și importanța semnăturilor digitale**

În mod tradițional, semnăturile olografe de pe documente servesc la autentificarea acestora, fiind utilizate în dovedirea identității semnatarului și ca mijloc de exprimare a acordului acestuia cu conținutul documentului.

O astfel de semnătură “convențională” angajează, de regulă, responsabilitatea semnatarului și pentru a fi valabilă ea trebuie să asigure următoarele cerințe: să nu poată fi falsificată, să nu fie reutilizabilă, să fie autentică și să nu poată fi renegată.

Necesitatea utilizării *semnăturilor digitale*, a apărut odată cu introducerea în mediile de afaceri a comunicațiilor prin intermediul calculatoarelor, când s-a pus problema semnării unor documente exprimate sub forma mesajelor electronice, și transmiterea semnăturii prin rețelele informatice.

Deși rolul și cerințele pe care trebuie să le îndeplinească o semnătură digitală sunt aceleași ca cele ale unei semnături convenționale, între ele există diferențe fundamentale.

O primă diferență o constituie noțiunea de semnare a unui document. Astfel, în timp ce o semnătură convențională este atașată fizic documentului semnat, acest lucru nu poate fi realizat, de aceeași manieră, pentru o semnătură digitală. Aceasta deoarece, spre deosebire de o semnătură olografă, o semnătură digitală se prezintă fie ca o versiune transformată a mesajului clar complet (documentului în formă electronică), fie ca o valoare suplimentară distinctă ce face pereche cu mesajul respectiv transmis. Prin urmare, procedura de semnare trebuie într-un anumit fel, “să lipească” semnătura la mesajul electronic.

O a doua diferență este cea legată de verificarea semnăturii. O semnătură convențională este autenticată prin compararea sa cu o alta care a fost certificată. Această metodă este puțin fiabilă deoarece semnătura unei persoane se poate imita relativ ușor. Dimpotrivă, o semnătură digitală poate fi verificată de către oricine cunoaște informația publică de verificare, identificându-se astfel emițătorul mesajului la care este atașată.

O altă diferență fundamentală între semnăturile convenționale și cele digitale este că orice “copie” a unui document electronic este identică cu originalul, spre deosebire de copia semnată a unui document de hârtie care poate fi deosebită de original. Ca urmare se impune luarea anumitor precauții pentru ca un document electronic semnat să nu poată fi reutilizat.

De asemenea, în timp ce semnătura olografă a unei persoane este constantă, o semnătură digitală depinde într-o modalitate complexă de fiecare caracter al mesajului astfel încât modificarea acestuia să fie imposibilă de făcut fără ca semnătura să rămână neschimbată. Se asigură astfel integritatea mesajului, prevenindu-se posibilitatea schimbării sau compromiterii nedetectate a conținutului său precum și posibilitatea ca semnătura să fie mutată de la un mesaj la altul. Acest lucru necesită ca mărimea câmpului semnăturii să fie suficientă de mare astfel încât căutarea tuturor mesajelor posibile pentru o semnătură dată, să fie computațional dificilă.

O semnătură digitală poate fi calculată doar de către emițătorul mesajului, căruia i se va atașa, pe baza unei informații secrete cunoscută numai de către acesta. Informația respectivă trebuie să fie în legătură cu o informație publică a cărei cunoaștere de către receptor să-i permită acestuia verificarea semnăturii. În acest fel se asigură nonrepudierea originii, adică prevenirea negării de către emițător ca fiind la originea expedierii mesajului. Totodată, emițătorul va fi încrezător în faptul că receptorul nu va putea să schimbe nici măcar un bit al mesajului fără să altereze semnătura.

Rezumând, se poate afirma că semnăturile digitale permit verificarea integrității mesajelor și furnizează funcții de autentificare și nonrepudiere pentru certificarea emițătorului unui mesaj. Din cele prezentate, rezultă de asemenea, că mecanismul de semnătură digitală constă dintr-un proces de semnare a unei unități de date și un proces de verificare a semnăturii respective.

### **2.3.2 Procedee de obținere**

Obținerea semnăturilor digitale se realizează prin intermediul unui procedeu de semnătură a cărui definiție formală, este următoarea:

Un procedeu de semnătură este un cvintet (M, S, K, F, V) ce verifică:

1. M este o mulțime finită de mesaje;

2. S este o mulțime finită de semnături;
3. K este o mulțime finită de chei;
4. Pentru fiecare  $K \in K$ , există o funcție de semnătură  $\text{sig}_K \in F$  și o funcție de verificare  $\text{ver}_K \in V$  corespondentă.

Funcțiile  $\text{sig}_K : M \rightarrow S$  și  $\text{ver}_K : M \times S \rightarrow \{\text{adevărat, fals}\}$  verifică, pentru fiecare mesaj  $M \in M$  și fiecare semnătură  $S \in S$ ,

$$\text{ver}_K(M,S) = \begin{cases} \text{adevărat} & \text{dacă } S = \text{sig}_K(M) \\ \text{fals} & \text{dacă } S \neq \text{sig}_K(M) \end{cases}$$

Pentru fiecare  $K \in K$ , funcțiile  $\text{sig}_K$  și  $\text{ver}_K$  trebuie să fie calculabile în timp polinomial. Funcția  $\text{ver}_K$  trebuie să fie publică, iar funcția  $\text{sig}_K$  trebuie să fie secretă. Contrafacerea unei semnături a emițătorului pentru un mesaj M trebuie să fie nefezabilă. Altfel spus, pentru un M dat, numai emițătorul trebuie să fie capabil să calculeze o semnătură S astfel încât  $\text{ver}_K(M,S) = \text{adevărat}$ . Un procedeu de semnătură nu poate fi sigur necondiționat, deoarece entitatea adversă poate testa toate semnăturile posibile S ale unui mesaj M cu ajutorul funcției publice de verificare ver, până când găsește semnătura bună. Deci, având timp suficient, entitatea adversă poate contraface întotdeauna o semnătură a emițătorului. În consecință, ca și pentru criptosistemele cu cheie publică, se impune studiul securității computaționale a unui procedeu de semnătură.

Pentru obținerea unei semnături digitale, în practică, se utilizează două procedee de bază.

Un prim procedeu constă în obținerea unei semnături prin transformarea a însăși mesajului în clar ce se dorește a fi transmis. Pentru aceasta se folosește, de regulă, un criptosistem cu cheie publică în cadrul căruia se inversează procesul de aplicare al celor două chei. Ca urmare, pentru crearea unui mesaj semnat S, emițătorul A al acestuia va folosi transformarea de decriptare D cu cheia sa secretă  $d_A$ . La rândul său, receptorul B al mesajului semnat va verifica originea autentică a acestuia și integritatea, prin decodificarea sa utilizând transformarea de criptare E cu cheia publică  $e_A$  a emițătorului. Dacă rezultatul este un mesaj clar posibil (mesaj semnificativ), mecanismul de semnătură a funcționat bine și semnătura este considerată ca și validă, fiind acceptată. Procedeu este ilustrat în figura următoare:

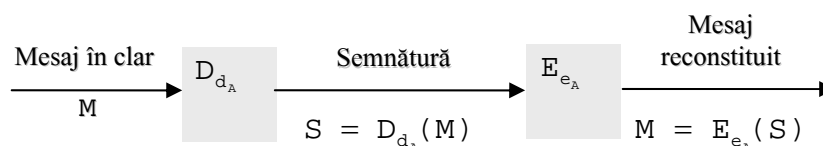


Fig. 2.7 Obținerea unei semnături digitale folosind un criptosistem cu cheie publică

Argumentul doveditor al validității semnăturii este faptul că numai posesia cheii secrete  $d_A$  permite calculul unui mesaj semnat S a cărui transformare, cu ajutorul cheii publice  $e_A$ , produce un mesaj semnificativ. Eficacitatea acestui procedeu se bazează pe un anumit număr de ipoteze.

În primul rând, se presupune că cheia secretă a fost într-adevăr păstrată secretă de către emițătorul mesajului, deoarece, în caz contrar, o entitate adversă va putea să semneze mesaje substituindu-se astfel emițătorului original.

În al doilea rând, identitatea emițătorului și a cheii sale publice se presupune a fi autentice, certificarea acestei autenticități fiind făcută de preferință de către o autoritate de certificare. Dacă o entitate adversă va reuși să facă un destinatar să accepte o cheie publică falsă (a cărei cheie secretă corespondentă o cunoaște), atunci ea va putea să falsifice mesaje cu o semnătură ce va apărea ca fiind validă.

O altă ipoteză este că orice modificare asupra mesajului semnat  $S$  produce la destinatar, după aplicarea cheii publice, o ieșire inacceptabilă pe care acesta va trebui să o distingă de un mesaj semnificativ. Este de asemenea important ca nimeni să nu poată determina prin calcul modificări în mesajul clar lăsând semnătura neschimbată. Acesta presupune, pe de-o parte, că semnătura trebuie să depindă de toți biții unui mesaj clar pentru ca ea să fie inalterabilă, iar pe de altă parte, necesitatea existenței unei redundanțe suficiente în mesajul clar (ca parte integrantă a acestuia) pentru ca probabilitatea de acceptare a unui mesaj aleator să fie foarte redusă. Redundanța este esențială, deoarece, în toată procedura de semnare, verificarea depinde de faptul că o singură porțiune foarte mică din șirul de biți receptați este susceptibilă de a fi acceptată ca și mesaj valid. Securitatea procedurii se bazează pe presupunerea că problema computațională ce constituie suportul criptostemelor cu cheie publică folosite, este computațional dificilă.

În figura 2.7 am presupus că procesul de semnare se aplică unui mesaj scurt văzut ca un singur bloc. Când un mesaj mai lung trebuie semnat, procedeul bazat exclusiv pe utilizarea criptosistemelor cu cheie publică devine inefficient datorită vitezei scăzute a acestora. Dacă se încearcă totuși tratarea mesajului pe porțiuni (bloc după bloc) prin același procedeu, apare riscul ca o entitate adversă să schimbe ordinea blocurilor semnate, semnăturile rămânând în continuare valide. În acest caz este de preferat a se utiliza cel de-al doilea procedeu de obținere, care presupune calculul și utilizarea unei semnături digitale ce ia forma unei valori separate adăugate mesajului în clar. Redundanța necesară mesajului respectiv, pentru ca o falsificare să poată fi detectată cu o mare probabilitate de către destinatar, este reprezentată în acest caz, de însăși semnătura ce se adaugă mesajului.

În figura 2.8 se arată cum se poate construi o semnătură digitală utilizând acest al doilea procedeu:

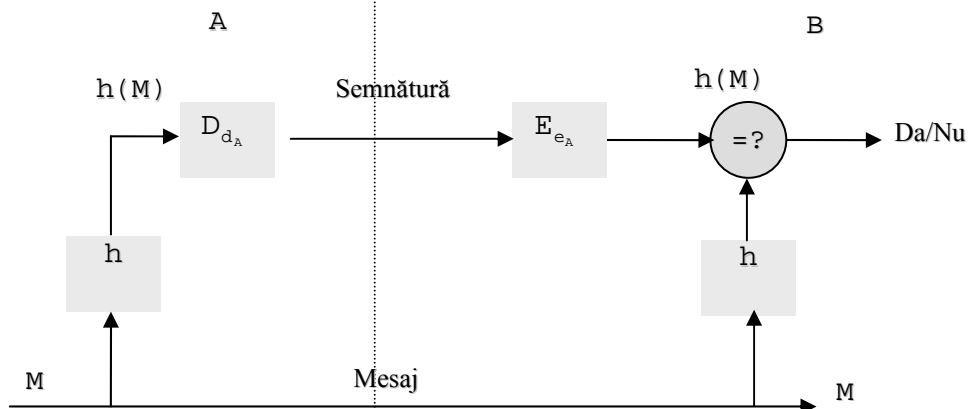


Fig. 2.8 Obținerea unei semnături digitale folosind o funcție de dispersie criptografică

Se aplică asupra mesajului în totalitatea sa, o funcție de dispersie criptografică  $h$ . Rezumatul obținut  $H = h(M)$  servește apoi ca intrare pentru un criptosistem cu cheie publică, care permite obținerea semnăturii mesajului întreg prin aplicarea transformării secrete a emițătorului. Se expediază destinatarului mesajul în clar și semnătura astfel obținută. După recepția acestora destinatarul trece la verificarea semnăturii primite. Pentru aceasta aplică asupra semnăturii cheia publică a expeditorului  $e_A$ , iar asupra mesajului în clar funcția de dispersie criptografică, comparând rezumatele obținute. În caz de coincidență, semnătura și mesajul sunt acceptate, altfel sunt rejectate. Funcția de dispersie criptografică trebuie să fie făcută publică. Tăria procedurii anterior depinde în mod esențial de inabilitatea unei entități adverse de a construi un mesaj care se potrivește cu o valoare a rezumatului dată, în caz contrar putându-se genera o aceeași semnătură pentru două mesaje diferite.

Dacă se dorește obținerea și a confidențialității mesajului în clar, acesta poate fi criptat.

În figura următoare se arată un exemplu de mesaj semnat și criptat printr-un criptosistem cu cheie publică.

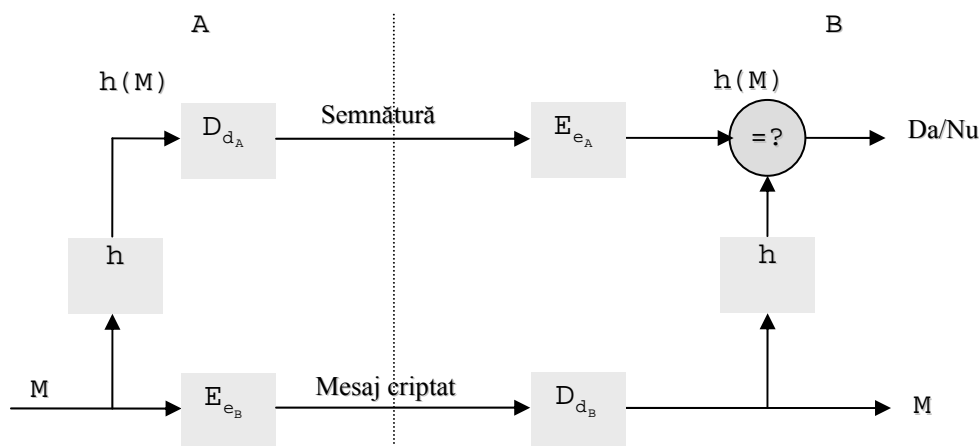


Fig. 2.9 Combinarea semnăturii digitale cu criptarea

Semnătura este trimisă în clar iar mesajul este criptat cu cheia publică  $e_B$  a destinatarului. După decriptare, verificarea semnăturii se face ca și în cazul anterior. Este de notat că pentru asigurarea atât a confidențialității cât și a autenticității mesajelor nu poate fi utilizat orice criptosistem cu cheie publică, ci numai cele care permit inversarea procesului prin care sunt efectuate criptarea și decriptarea.

### ***Tehnici de protecție a programelor***

***(Popa, S., Securitatea și protecția informației în sist. inf., Note de curs, pag. 94-98)***

Ideea ce stă la baza folosirii unei protecții tehnice a aplicațiilor software, este ajungerea la o situație în care operațiile de reverse-engineering să devină neviabile din punct de vedere economic, sau chiar imposibile.

La baza apariției limbajelor de programare ce generează cod independent de platformă, a stat ideea că o singură versiune a aceluși program trebuie creată și aceasta va rula pe „orice” sistem. Costurile de dezvoltare și suport vor fi reduse, deoarece nu este necesară întreținerea unor versiuni diferite ale aceleiași aplicații, pentru fiecare platformă hardware pe care aceasta rulează. Din acest motiv, protecțiile încorporate în programele software trebuie să fie gândite astfel încât să fie independente de orice particularități ale mediului în care ele rulează.

#### **7.2.1 Protecția prin execuția pe server (server-side execution)**

Această modalitate de protecție constă în plasarea aplicației pe un server, iar serviciile ei vor fi oferite printr-o conexiune la distanță (figura 7.1).

Utilizând această arhitectură, accesul fizic la aplicație este împiedicat. Pe de altă parte există două dezavantaje ale executării pe server, față de situația când întreaga aplicație este gazduită pe mașina (calculatorul) client:

1. Lățimea de bandă a conexiunii la rețeaua locală sau la Internet este limitată, făcând ca performanțele aplicației respective să scadă datorită constrângerilor legate de comunicație;
2. Dacă rețeaua în care este serverul, nu funcționează corect, atunci utilizarea aplicației va fi imposibilă.

Se pot stabili priorități între cerințele aplicației dezvoltate, și se poate alege o arhitectură client/server adecvată pentru fiecare caz practic. În același timp, numai anumite părți ale unei aplicații pot fi considerate ca fiind critice din punct de vedere al protecției, nefiind deci necesară protejarea întregului program. Dacă ne aflăm în acest caz și anumite părți nu sunt considerate de interes pentru competitorii firmei producătoare a aplicației, atunci executarea

întregii aplicații pe server va aduce un grad de protecție excesiv față de necesitățile reale. Aplicația va fi împărțită într-o parte privată ce se va executa pe server și o parte publică ce va rula local, pe mașina client. În figura 7.1(b) este prezentată schematic și această configurație cu execuție parțială pe server, a codului. Execuția parțială pe server va avea aceleași beneficii ca și execuția totală pe server. Acestea provin din faptul că partea ce se dorește a fi protejată nu este accesibilă fizic. În plus, se va beneficia și de performanțe mai bune atunci când se vor executa părțile din program ce au rămas local, pe mașina utilizatorului.

În construirea arhitecturii acestui gen de aplicații trebuie avut mereu în vedere volumul de informații ce va fi transferat între client și server. De asemenea trebuie considerată și frecvența comunicațiilor efectuate între cele două părți ale aplicației. De obicei se fac compromisuri între nivelul de protecție și viteza de execuție.

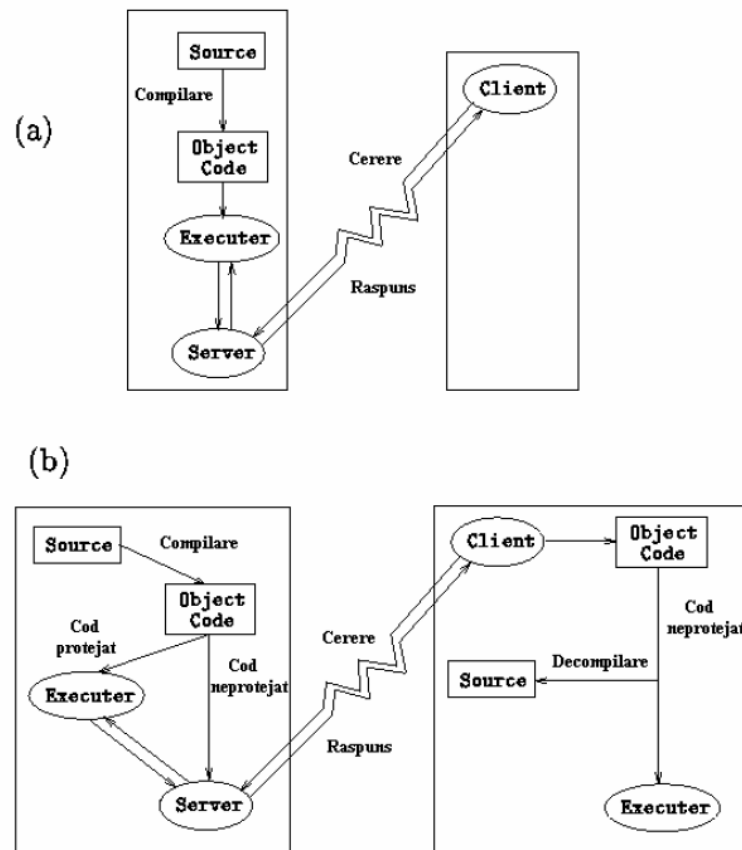


Fig. 7.1 Protejarea aplicațiilor prin execuție pe server(a) și parțial pe server(b)

### 7.2.2 Protecția prin criptare

Pentru a se elimina dezavantajele relative la performanțele aplicațiilor ce au fost descompuse într-o arhitectură client/server, din motive de protecție a codului, trebuie utilizate metode care să protejeze codul direct accesibil unui utilizator, pe mașina pe care acesta rulează.

Criptarea codului distribuit reprezintă una dintre aceste metode (în figura 7.2 este o reprezentare schematică). Interceptarea și decriptarea codului compilat, sunt posibile, cu excepția cazului în care decriptarea se face la nivel hardware. Acest lucru implică existența unui procesor suplimentar (un cip dedicat) care să decripteze instrucțiunile înainte ca acestea să fie executate de către procesorul principal (unitatea centrală) al calculatorului pe care se execută aplicația. Codul decriptat nu va fi niciodată stocat în memoria calculatorului, aceasta fiind accesibilă utilizatorului sau altor programe ce rulează în paralel. Ca urmare, gradul de protecție depinde de schema de criptare utilizată pentru codul programului.

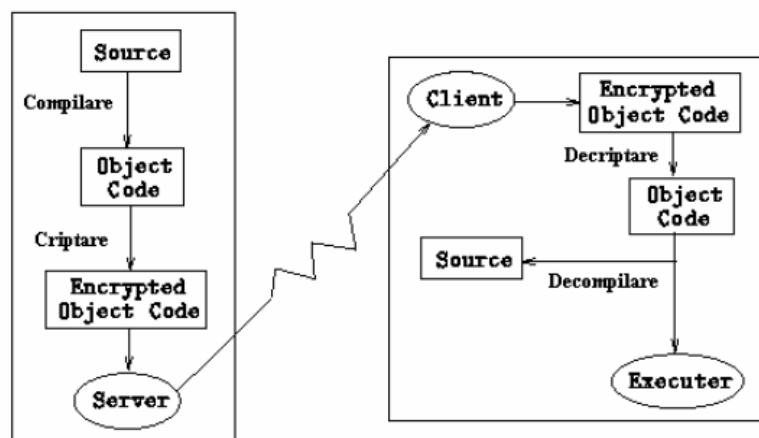


Fig. 7.2 Protecția prin criptare

Aplicațiile protejate prin criptare pot fi programate să nu se execute decât dacă cipul – componenta hardware de decriptare – este conectat la mașina pe care se încearcă execuția. În consecință, pentru a putea rula aplicația pe orice mașină, cel ce va încerca să o copieze și să o folosească va trebui să reușească două operații:

1. să spargă schema de criptare;
2. să modifice codul în așa fel încât verificarea existenței cip-ului de decriptare să fie anulată.

Problemele folosirii unui sistem de criptare care se bazează pe existența unui cip de decriptare hardware provin din varietatea mare de procesoare utilizate în calculatoarele actuale. Procesoarele respective vor avea nevoie de circuite de interfațare diferite pentru comunicația cu cipul de decriptare. Ca urmare, acest mod de protecție este aplicabil numai în cazul aplicațiilor care sunt destinate să ruleze numai pe un set redus de procesoare, sau pe un singur procesor, pentru care există deja interfațare cu cipul de decriptare folosit.

De asemenea, adăugarea unui cip de decriptare obligă utilizatorii să plătească o sumă suplimentară care nu le aduce nici un avantaj direct, beneficii având numai producătorului aplicației respective.

### 7.2.3 Protecția cu ajutorul codului nativ semnat (Signed Native Code)

Aplicațiile Java și .Net sunt executate de către o mașină virtuală ce implementează un interpretor. Aceasta înseamnă că aplicațiile respective se vor executa ceva mai încet decât aplicațiile clasice scrise în limbaje ce se compilează direct în cod mașină. Pe de altă parte, vor avea avantajul portabilității. Pentru a îmbunătăți viteza de execuție în cazul aplicațiilor Java, au fost dezvoltate compilatoare *just-in-time* (JIT)<sup>5</sup>. Există un număr destul de mare de compilatoare JIT, disponibile pentru o varietate de sisteme de operare.

Anumite firme au inclus compilatorul JIT direct în kit-ul lor pentru platforma de dezvoltare, exemple fiind Borland Jbuilder și Microsoft Visual J++.

<sup>5</sup> Tehnologia .NET lansată de firma Microsoft în anul 2002 este o nouă platformă de dezvoltare a aplicațiilor (în special a aplicațiilor distribuite în Internet) pentru sistemele de operare Windows. Platforma .NET suportă utilizarea și integrarea mai multor limbaje de programare: VB.NET, *Managed C++*, J#, C# și limbajul de scripting *JScript*. Compilatoarele aferente acestor limbaje generează, din codul sursă, cod într-un limbaj intermediar unic definit în .NET, numit *MSIL* (*Microsoft Intermediate Language*) (sau, mai simplu *IL*). Acest cod IL, va fi executat ulterior sub controlul unei unități de execuție, denumită *Common Language Runtime* (CLR), în **cod nativ**. Codul executat sub controlul CLR este numit **cod gestionat** (*managed code*), iar codul executat direct de procesor sub controlul sistemului de operare se numește **cod ne-gestionat** (*un-managed code*) sau **nativ**.

Compilatorul JIT convertește codul IL în cod nativ gestionat (*managed native code*), care poate fi executat pe hardware-ul și sistemul de operare al calculatorului gazdă. Avantajul compilatorului JIT este acela că poate genera cod optimizat exact pentru tipul de platformă hardware pe care se va executa.

Pentru a evita consumul de timp aferent compilării de fiecare dată când aceeași secvență de cod este executată pe aceeași mașină, este posibilă stocarea codului mașină deja compilat. Tehnologia care permite acest lucru se numește compilare *way-ahead-of-time* (WAT compilation).

Decompilarea și urmărirea codului nativ (cod mașină), obținut cu ajutorul compilatoarelor JIT și WAT din cod intermediar (IL), este mult mai greu de realizat. Ca urmare, aplicația ce se dorește a fi comercializată este mai întâi compilată cu un compilator standard de Java și încărcată pe un server. Apoi, clienții ce achiziționează aplicația, identifică o combinație de arhitectură și sistem de operare, ce corespunde mediului în care ei doresc a lucra, iar serverul le va furniza o versiune a acelei aplicații direct în cod nativ pentru mediul respectiv.

Faptul de a putea accesa numai codul nativ rezultat va îngreuna foarte mult, dar în nici un caz nu va face imposibilă o operație de reverse-engineering. Nivelul de protecție asigurat de distribuirea aplicației în cod nativ, nu este același ca și cel oferit de utilizarea dispozitivelor hardware de criptare. În plus, este necesară prezența a câte unui compilator JIT sau WAT care să compileze aplicația în cod nativ pentru fiecare combinație de arhitectură/sistem de operare ce poate fi întâlnită la potențialii clienți. Deci portabilitatea aplicației va fi redusă numai la sistemele pentru care compilatoarele JIT sau WAT existente, pot genera cod.

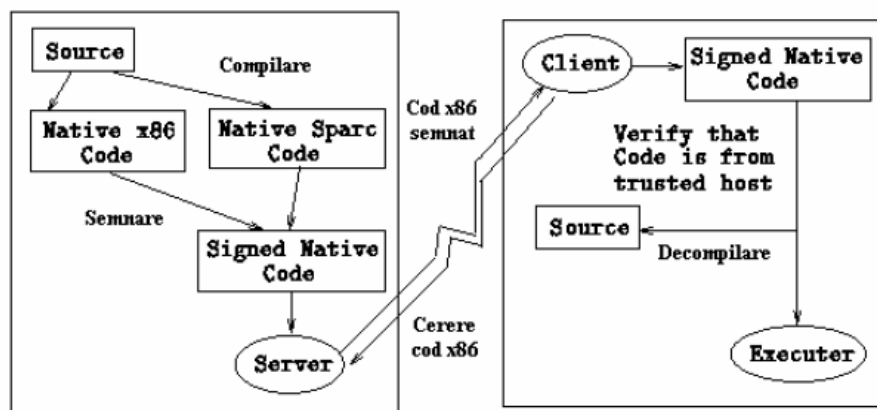


Fig. 7.3 Protecția prin distribuire de cod nativ semnat

În cazul aplicațiilor scrise în limbajul Java mai există o problemă suplimentară legată de distribuirea directă a codului nativ. Spre deosebire de codul intermediar, codul nativ nu mai este supus operației de verificare (*bytecode verification*) care se efectuează înaintea execuției. Deci în cazul codului nativ nu mai putem avea garanția că acesta se va executa fără să efectueze acțiuni ilegale, precum coruperea unor fișiere utilizator. Aceasta nu reprezintă o problemă în cazurile în care firma producătoare a aplicației este una recunoscută, deoarece, evident, nu ne vom aștepta la acțiuni ilegale determinate de utilizarea codului distribuit de aceasta.

Pentru a evita confuziile, companiile dezvoltatoare își semnează digital codul transmis, garantând astfel clienților sau utilizatorilor autorizați că respectiva aplicație este cea originală (figura 7.3).

#### 7.2.4 Protecția prin obfuscarea codului (code obfuscation)

Ideea ce stă la baza acestui tip de protecție este transformarea unei aplicații astfel încât aceasta să fie funcțional identică, relativ la original, dar să fie mult mai greu de înțeles.

Pentru protecția aplicației, firmele dezvoltatoare folosesc utilitare numite *obfuscatoare* (figura 7.4). Trebuie avut în vedere faptul că, spre deosebire de execuția pe server, obfuscarea nu oferă o protecție completă împotriva atacurilor malițioase de tip reverse-engineering. Astfel, alocând suficient timp și efort, este posibilă recuperarea algoritmilor și a structurilor de date importante, din aplicația analizată. De data aceasta se va folosi un utilitar numit de-obfuscator care va încerca să inverseze transformările la care a fost supusă aplicația pentru a fi protejată.

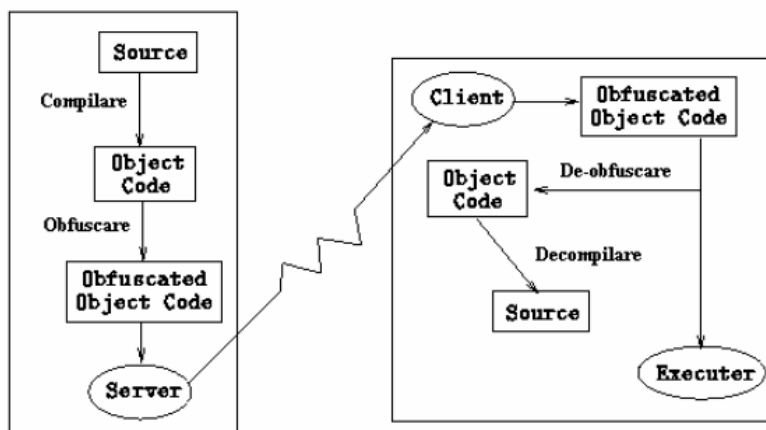


Fig. 7.4 Protejarea prin obfuscarea codului

Nivelul de securitate oferit de un obfuscator împotriva operațiilor de reverse-engineering depinde de:

- complexitatea transformărilor folosite de obfuscator;
- puterea algoritmilor de de-obfuscare;
- cantitatea de resurse (în principal timp) ce sunt disponibile pentru de-obfuscator.

Obfuscarea codului face ca el să fie mult mai greu de înțeles, păstrând în același timp și independența de platformă a acestuia.

Spre deosebire de execuția pe server, o aplicație obfuscată nu va fi afectată suplimentar, de limitarea traficului de rețea și de disponibilitatea acesteia. Obfuscarea nu va necesita hardware special, precum în cazul protecției prin criptare, iar spre deosebire de codul nativ, în cazul obfuscării nu este necesară nici un fel de semnare digitală a aplicațiilor pentru a dovedi că respectivul cod provine dintr-o sursă de încredere.

## PROIECTAREA SISTEMELOR INFORMATICE

### *Etapele de realizare a sistemelor informatice*

*(Bușe, R., Proiectarea sistemelor informatice, Note de curs, cap.2, pag. 1-2)*

Sistemul informatic are un ciclu propriu de viață, care începe cu decizia de realizare, cuprinde faza de elaborare, faza de utilizare, faza de perfecționare și se încheie cu decizia de abandonare în forma existentă și înlocuirea cu un nou sistem.

Acestui ciclu de viață îi corespund etape specifice stărilor succesive prin care trece sistemul informatic, etape caracterizate prin activități distincte. Etapele realizării unui sistem informatic sunt:

- analiza sistemului informațional existent (analiza de sistem);
- proiectarea sistemului informatic;
- elaborarea și testarea programelor;
- implementarea sistemului informatic;
- exploatarea curentă și menținerea în funcțiune a sistemului informatic.

1) Analiza sistemului informațional existent urmărește delimitarea ariei de cuprindere a sistemului și formularea cerințelor și restricțiilor globale de realizare. Pentru a atinge acest scop, în această etapă se face un studiu amănunțit al sistemului existent, se apreciază măsura în care sistemul existent este capabil să răspundă în continuare exigențelor conducerii științifice a agentului economic, se apreciază oportunitatea realizării unui sistem informatic și se formulează principalele restricții și cerințe pentru viitorul sistem informatic.

2) Proiectarea constă în definirea modelului de ansamblu (conceptual) al sistemului informatic, ținând seama de evaluările făcute în etapa anterioară, dar și în transformarea modelului conceptual stabilit anterior într-un model tehnic, operațional.