Co-funded by the Erasmus+ Programme of the European Union

Module de instruire InCyT

Securitatea informațiilor pentru angajați **Unitatea 2 Săptămâna 3** **Numele unității : Introducere în Criptografie**	
Activități de învățare	❑ **Webinar** ❑ **Exerciții** ❑ **Workshop** ❑ **Prezentare** ❑ **Alte**
Responsabil de învățare	Claudio Fornaro
Obiectivele activității de învățare	1. Criptografia 2. Criptoanaliza 3. Algoritm de un singur pad 4. Algoritmi simetrici 5. Algoritmi asimetrici 6. Funcții hash 7. Semnături și infrastructuri cu cheie publică 8. Jetoane criptografice 9. Mecanism provocare-răspuns
Abilități de dobândit	• Cunoștințe de bază despre algoritmi și infrastructuri de criptare • Abilitatea tehnică 2 - TS2
Durata activității de învățare	1 săptămână
Cerințe de învățare	***Ce e necesar?*** • Nu sunt necesare cunoștințe specifice anterioare

Funded by the European Union. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or the European Education and Culture Executive Agency (EACEA). Neither the European Union nor EACEA can be held responsible for them.

Co-funded by the Erasmus+ Programme of the European Union

Informații scurte despre modul

Criptografia se referă la metodologii pentru ascunderea informațiilor de la divulgare, atât în timpul unei comunicări, cât și în scopuri de arhivare. Obiectivele sale sunt atinse prin proiectarea și implementarea protocoalelor care împiedică o parte terță neautorizată să recupereze informațiile. Într-o comunicare securizată, terțul adversar nu poate accesa informațiile transmise, în cazul arhivării terțul nu poate urmări conținutul a ceea ce a fost arhivat.

În Criptografie, terța parte este întotdeauna o entitate rău intenționată care urmărește să recupereze informații la care nu trebuie să aibă acces, fie din motive de confidențialitate, secret comercial sau securitate națională. Confidențialitatea și integritatea datelor, autentificarea părților implicate și nerepudierea a ceea ce a transmis sau stocat sursa de date sunt principiile fundamentale ale criptografiei moderne.

După o parte introductivă, în care sunt descrise conceptele fundamentale, prezentăm o progresie a tehnicilor criptografice de bază, pornind de la Algoritmul One Time Pad și apoi construind pentru a lua în considerare cele două familii criptografice de algoritmi: algoritmi simetrici și asimetrici (sau cheie publică) Algoritmi cu o comparație a acestora. Rolul funcțiilor Hash este, de asemenea, discutat pentru a introduce semnăturile digitale. O descriere a conceptului cheie al infrastructurilor cu cheie publică este prezentată în profunzime cu câteva detalii despre mecanismele și formatele standard. În concluzie, sunt furnizate câteva informații despre Jetoanele Criptografice și Mecanismul Provocare-Răspuns.

Conținutul materialului de învățare

Funded by the European Union. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or the European Education and Culture Executive Agency (EACEA). Neither the European Union nor EACEA can be held responsible for them.

Co-funded by the Erasmus+ Programme of the European Union

Introducere

Criptografia este mecanismul prin care conținutul unui mesaj este ascuns de alte entități. Cu alte cuvinte, criptarea se ocupă cu criptarea mesajelor, adică preluarea unui text simplu, folosind un algoritm și, în general, o cheie, făcând conținutul mesajului neinteligibil.

Criptanaliza studiază cum să decripteze mesajul fără a cunoaște cheia. Într-un atac, criptoanaliza constă în încercarea de a urmări conținutul mesajului fără a avea cheia. Criptografii și criptoanalistii sunt de fapt aceiași oameni: cine scrie algoritmii criptografici îi analizează și pentru a le găsi punctele slabe sau pentru a le valida robustețea. Ca întotdeauna, atunci când aveți de-a face cu securitatea, ideea de a putea face față problemelor de protecție fără a ști cum funcționează tehnicile de atac nu este foarte eficientă.

Informațiile din mesaje sunt în general redundante, într-un sens formal, mesajele folosesc în general un număr mult mai mare de biți decât cel cu adevărat necesar pentru a transporta informațiile. **Redundanța informațiilor facilitează criptoanaliza**: redundanța în informații înseamnă că mesajele pot prezenta o structură care facilitează urmărirea conținutului mesajului, deoarece nu toate mesajele sunt la fel de probabile. Dacă ne gândim la un text reprezentat cu caractere ASCII normale în octet, știm că combinațiile de biți care pot reprezenta mesaje semnificative sunt relativ puține în comparație cu toate combinațiile posibile. Adevăratele mesaje care pot fi reprezentate cu un anumit număr de octeți sunt foarte puține. Acest lucru face posibilă identificarea mesajelor valide prin eliminarea celor care nu poartă informații adevărate. Toate acestea ajută foarte mult activitatea criptoanalistului, astfel încât scopul algoritmului de criptare este și acela de a ascunde această redundanță pentru a ascunde mai bine informațiile..

Exemplu: Codificarea unei secvențe din cele patru culori ALBASTRU, VERDE, ROȘU și GALBEN.

Acestea pot fi codificate:

- cu 2 biți (00, 01, 10, 11)
- ca 4 litere mari diferite
- criptarea numelor de culori

În primul caz, se folosește numărul minim de entități (biți), prin înlocuirea fiecărui nume de culoare cu cei 2 biți corespunzători va ascunde secvența originală, dar nu și ordinea deoarece există o corespondență între CULOARE și cei 2 biți. În al doilea caz, deoarece fiecare literă folosește 8 biți în codul ASCII, 6 biți sunt redundanți pentru fiecare culoare → Redundanță de 75% și obținem același grad de ascundere ca și cei 2 biți. Încă nu putem asocia culorile și valorile.

Luați în considerare următorul cifr foarte simplu, așa-numitul cifru Caesar, care nu face altceva decât să deruleze literele din alfabet cu un anumit număr de poziții. De exemplu, dacă le glisăm cu

Funded by the European Union. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or the European Education and Culture Executive Agency (EACEA). Neither the European Union nor EACEA can be held responsible for them.

Co-funded by the Erasmus+ Programme of the European Union

3, toate „A” devin „D”, toate „B” devin „E” și așa mai departe. Acesta este un cifr foarte simplu; de fapt atribuite lui Iulius Caesar, dar dacă numele culorilor sunt criptate cu această metodă, modelul repetat al literelor lor codificate este ușor de identificat (codul ASCII pentru R urmat de codul ASCII de E urmat de codul ASCII de D sunt ușor de recunosc ca entitate). Deci secvența poate fi identificată, iar dacă cuvintele sunt cunoscute și în acest caz doar numărarea numărului de litere este suficientă pentru a identifica cuvintele originale.

Acest exemplu simplu ne permite să înțelegem că criptoanaliza nu este o analiză care este independentă de conținutul mesajului: un mesaj complet randomizat ar fi mult mai dificil de identificat și analizat decât un mesaj care are o structură de recunoscut.

Unele tipuri de atacuri criptografice și algoritmi se bazează pe cunoașterea unei părți a textului. În unele cazuri, cunoașterea unei părți a textului face posibilă urmărirea cheii și apoi accesarea restului textului. Acest lucru este foarte semnificativ atunci când vine vorba de comunicațiile în rețea, deoarece în comunicațiile în rețea protocoalele de la diferite niveluri au de obicei o anumită cantitate de informații care este în orice caz recunoscută, de ex. într-un pachet IP știm că antetul conține adresa expeditorului și adresa destinatarului.

Un algoritm de criptare ascunde și informații redundante, așa că aplicarea unui algoritm de compresie unui mesaj criptat nu aduce prea multe avantaje, deoarece ar reduce rata de compresie. Dacă vrem să folosim compresia și criptarea, trebuie să folosim mai întâi compresia și apoi criptarea.

Algoritmul One Time Pad

One Time Pad (OTP) este un algoritm, ca să spunem așa, „perfect”. Este un algoritm de criptare foarte simplu, foarte simplu de implementat și are o caracteristică care îl face deosebit de interesant: este dovedit că fără a cunoaște cheia este imposibil să se revină la textul original. Acum ne uităm la el și apoi discutăm de ce, deoarece avem un algoritm atât de bun, trebuie să găsim alții.

SAU EXCLUSIV⊕ (XOR) este o funcție logică în care rezultatul este Adevărat când cei doi operanzi sunt diferiți, operează pe valori logice dar traduse în cifre binare, de obicei Adevărat=1 și Fals=0, tabelul său de adevăr este:

$0 \oplus 0 = 0$

$0 \oplus 1 = 1$

$1 \oplus 0 = 1$

$1 \oplus 1 = 0$

Funded by the European Union. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or the European Education and Culture Executive Agency (EACEA). Neither the European Union nor EACEA can be held responsible for them.

Co-funded by the Erasmus+ Programme of the European Union

Din acest tabel reiese clar că dacă X ⊕ Y → Z, atunci Z ⊕ Y → X și Z ⊕ X → Y. SAU EXCLUSIV este o funcție foarte simplă și, de asemenea, ușor de implementat ca circuit electronic.

Mecanismul OTP este utilizat pentru a transfera un mesaj de la două părți. În literatura științifică este o practică obișnuită de a identifica actorii cu nume personale, aceștia fiind de obicei numiți Alice (A) și Bob (B). Așadar, vom descrie modul în care Alice poate transmite în siguranță un mesaj de n octeți (PlainText) către Bob.

Algoritmul One Time Pad necesită o fază de configurare:

- Este generată o secvență aleatorie de exact n octeți, acesta este One Time Pad (OTP) sau pur și simplu Cheia;
- OTP este partajat în siguranță între cele două părți într-un fel (dat în avans, trimis pe un canal deja securizat etc.).

Când ambele părți dețin cheia, o pot folosi pentru a transmite mesaje. În cazul în care Alice vrea să-i trimită un mesaj lui Bob:

- Alice calculează puțin cu bit EXCLUSIV SAU din PlainText cu cheia, producând un CypherText:
  Criptare: PlainText ⊕ Cheie → CypherText
- CypherText rezultat este transmis de Alice lui Bob pe un canal posibil nesigur
- Bob calculează bit cu bit EXCLUSIVE SAU din CypherText și Key, astfel recuperând PlainText-ul original:
  Decriptare: CypherText ⊕ Cheie → PlainText

Exemplu

Să presupunem că atât Alice, cât și Bob au convenit deja asupra unui OTP (cheie) aleatoriu.

Alice vrea să-i trimită „HELLO” lui Bob (în codul ASCII, aici sunt introduse spații pentru ușurință de citit):

	H	**E**	**L**	**L**	**O**
PlainText:	01001000	01000101	01001100	01001100	01001111
			⊕		
Cheia:	10101001	01110010	10110010	10000110	11000110
			=		
CypherText:	11100001	00110111	11111110	11001010	10001001

Funded by the European Union. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or the European Education and Culture Executive Agency (EACEA). Neither the European Union nor EACEA can be held responsible for them.

Co-funded by the Erasmus+ Programme of the European Union

Bob primește *CypherText* și aplică *Cheia*:

CypherText:	11100001	00110111	11111110	11001010	10001001
			⊕		
Cheia:	10101001	01110010	10110010	10000110	11000110
			=		
PlainText:	01001000	01000101	01001100	01001100	01001111
	H	**E**	**L**	**L**	**O**

Textul *PlainText* a fost recuperat.

Deoarece *Cheia* este o secvență aleatorie, nu există nicio modalitate de a spune dacă un anumit bit al secvenței este „0” sau „1”, deoarece acestea sunt la fel de probabile. Fără *Cheie*, dat fiind un bit *CypherText*, nu avem nicio șansă să încercăm să ghicim care ar putea fi bitul original. Acest lucru este valabil pentru biți individuali, pentru întregul mesaj, pentru orice fragment al mesajului nostru.

În cazul în care o parte a *Cheii* este cunoscută, este posibil să se recupereze partea corespunzătoare din *PlainText*, dar fără restul *Cheii* nu există lipsă de informații despre ceilalți biți din *PlainText*. Prin urmare, criptoanaliza menționată anterior nu este eficientă. În acest caz, pentru a descoperi partea rămasă a mesajului original, este necesară partea rămasă a *Cheii*. Toate informațiile transmise de *Cheie* sunt necesare pentru a recupera toate informațiile textului.

Probleme cu pad o singură dată

Puterea mecanismului OTP se bazează pe lungimea cheii fiind exact egală cu lungimea textului și pe utilizarea acesteia o singură dată. Principalul dezavantaj este că transmiterea OTP-ului are aceleași probleme ca și mesajul.

În unele cazuri, această problemă se rezolvă cu ușurință, de exemplu o servietă diplomatică cu o panglică care conține o cheie foarte lungă ar putea fi furnizată unei ambasade și ulterior mesajele care vor fi schimbate cu acea ambasadă pot fi criptate prin consumarea biților cheii treptat. . Acestea sunt utilizări deosebite, dar destul de eficiente. Este clar că dacă luăm în considerare comunicarea în rețea, utilizările reale ale One Time Pad sunt foarte limitate. Este nevoie de ceva mai flexibil, ceva care să permită schimbul unei chei care poate fi folosită foarte mult timp.

Să luăm în considerare fiecare dintre probleme.

Funded by the European Union. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or the European Education and Culture Executive Agency (EACEA). Neither the European Union nor EACEA can be held responsible for them.

Co-funded by the Erasmus+ Programme of the European Union

Transmitere offline în avans

Aceasta este o problemă generală cu cheile: cheile trebuie să fie convenite într-un fel de către părți. Această nevoie de a conveni asupra cheilor este cea mai importantă problemă a utilizării criptografiei. Problema criptografiei nu este atât de algoritmi și criptoanaliza, cât și de managementul cheilor: adică să fii sigur că toți și numai subiecții potriviți au acces la chei, apoi le schimbă în avans și asigură-te că cheile schimbate sunt cele cele corecte.

Utilizare o singură dată

One Time Pad este folosit o singură dată, așa cum sugerează și numele, acesta este un aspect și o limitare importantă. Dacă am încerca să folosim aceeași cheie pe două mesaje, am slăbi securitatea metodei.

Generarea de numere aleatorii într-un sistem determinist este foarte dificilă

Calculatoarele de uz general folosesc generarea numerelor pseudoaleatoare în care fiecare număr este calculat de la cel anterior cu unele calcule, aceasta produce o secvență de numere care se repetă după o perioadă foarte lungă. Acestea nu sunt numere aleatoare adevărate, deoarece un computer este o mașină deterministă. În criptografie, totuși, această caracteristică este dăunătoare. Dacă este posibil să se obțină următorul număr aleatoriu dintr-unul anterior, protecțiile oferite de algoritmii sau protocoalele criptografice devin mult mai puțin sigure. Scrierea de cod care implementează algoritmi criptografici nu este banal: sunt multe detalii de care trebuie să fii atent, toate defectele identificate de-a lungul anilor de dezvoltatorii de cod pe care doar dezvoltatorii de cod din această zonă anume au învățat să le gestioneze. Aceasta înseamnă că este întotdeauna o idee proastă să dezvoltați și/sau să implementați algoritmi criptografici interni în loc să folosiți biblioteci bine testate, utilizate pe scară largă și validate pe scară largă.

Algoritmi decât OTP

Ceilalți algoritmi folosesc chei cu lungime fixă. Dacă folosim o cheie de n biți, avem 2n chei diferite, dacă valoarea lui n este suficient de mare și doar aplicați un atac de forță brută (adică încercați toate cheile posibile), ar fi nevoie de mulți ani pentru a găsi cea potrivită. una astfel încât „probabil” conținutul mesajului să nu mai prezinte interes. De exemplu, cu unul dintre cele mai rapide supercalculatoare disponibile, verificarea unei chei pe 256 de biți ar necesita 3,67×1055 ani. În mod clar, utilizarea cheilor de lungime suficientă are un cost de calcul suplimentar chiar și pentru cei care criptează și decriptează în mod legitim, dar impactul este limitat.

Există algoritmi simetrici și asimetrici.

Funded by the European Union. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or the European Education and Culture Executive Agency (EACEA). Neither the European Union nor EACEA can be held responsible for them.

Co-funded by the Erasmus+ Programme of the European Union

Algoritmi simetrici

Cea mai tipică și veche familie de algoritmi criptografici, cei la care ne gândim de obicei când ne gândim la criptarea mesajelor, sunt ***algoritmii simetrici***, numiți după simetria operațiunilor de criptare și decriptare. Acești algoritmi folosesc o singură cheie care trebuie cunoscută atât de expeditor, cât și de destinatar; este folosit pentru a cripta un mesaj și apoi pentru a-l decripta.

O schemă simetrică simplă ar putea fi aceeași cu OTP, deoarece OTP este un algoritm simetric, cu marea diferență că cheia are o lungime fixă (de exemplu, 256 de biți) și se repetă într-un fel, eventual după o schimbare complexă de fiecare dată:

Criptare: *encryption_function*(PlainText, Key) → CypherText

Decriptare: *decryption_function*(CypherText, Key) → PlainText

Mai în general, chiar dacă folosim aceeași cheie, funcția folosită pentru decriptare nu este neapărat aceeași folosită pentru criptare. De exemplu, dacă luăm în considerare cifrul lui Cezar, este clar că funcția care descifrează mesajul glisează alfabetul la stânga și nu la dreapta, dar decriptarea folosește aceeași cheie: numărul 3.

Câțiva algoritmi importanți cu cheie simetrică

DES (Data Encryption Standard)

Acest algoritm istoric, acum depășit, folosește o cheie de 56 de biți care la sfârșitul anilor 1970 (când a fost dezvoltat algoritmul) era absolut adecvată pentru multe utilizări, deoarece un atac de forță brută asupra unei chei de 56 de biți pentru a obține mesajul cu computerul. resursele disponibile la momentul și în contextele în care a fost utilizat acel algoritm, a fost impracticabilă. 40 de ani mai târziu, algoritmul DES nu ar dura nici măcar câteva minute dintr-un computer de acasă. Acesta este un aspect foarte important: lungimea cheii este aleasă pe baza capacităților de calcul disponibile reale. O cheie care este considerată suficient de robustă acum ar putea să nu mai fie atât de puternică în 20 de ani. În general, avem tendința de a abunde cu lungimea cheii pentru a ne asigura că există o anumită marjă, atât pentru că capacitățile de calcul cresc, dar și pentru că criptoanaliza ca disciplină face progrese și poate fi capabilă să găsească puncte slabe în algoritm. Acesta este un aspect destul de interesant. Contrar a ceea ce s-ar putea crede, în mod normal, un algoritm criptografic nu eșuează brusc, adică în ziua în care un defect este descoperit într-un algoritm, acesta trece instantaneu de la sigur la complet nesigur și oricine poate găsi textul simplu într-un moment. Ce se întâmplă este că cercetătorii analizează algoritmul și poate au găsit mici slăbiciuni care pot ușura atacarea acestuia în anumite cazuri. Atunci cineva poate găsi o optimizare a algoritmului pentru a rezolva problema. Pe măsură ce trece timpul, algoritmul este slăbit treptat din cauza unor probleme de nerezolvat, până când nu mai este adecvat pentru anumite utilizări sau deloc. Există o erodare progresivă a apărărilor algoritmilor criptografici, iar

Funded by the European Union. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or the European Education and Culture Executive Agency (EACEA). Neither the European Union nor EACEA can be held responsible for them.

această slăbire progresivă este o problemă mai greu de gestionat, pentru că evident nu putem ști ce progrese vor fi în viitor din punctul de vedere al criptoanalizei.

AES (Advanced Encryption Standard, cunoscut și sub numele său original Rijndael)

AES a fost ales nu numai pe baza robusteței, ci și pe eficiența implementării în arhitecturile tipice. Nu s-a căutat niciun algoritm secret deoarece un algoritm secret nu este considerat mai robust decât unul non-secret: dimpotrivă, algoritmul a fost destinat să fie analizat de cât mai mulți criptoanaliști valoroși pentru a se asigura că este intrinsec robust. Un algoritm care este sigur, dar prea lent, prea scump din punct de vedere al resurselor de calcul, nu ar fi utilizabil. Această constrângere a fost stabilită a fi cea mai dificil de depășit de către concurenți, în comparație cu cea a robusteței.

Algoritmi asimetrici (sau chei publice).

Algoritmii asimetrici nu folosesc doar o cheie, ci două: o cheie este folosită pentru a cripta și cealaltă pentru a decripta. Prin urmare, în schimbul de mesaje, expeditorul și destinatarul nu folosesc aceeași cheie. Aceste chei sunt generate și gestionate în perechi: ceea ce criptează o cheie (oricare dintre perechi) poate fi decriptat numai de cealaltă cheie a perechii.

Motivul utilizării algoritmilor asimetrici derivă din problemele pe care le are criptografia simetrică:

1. Comunicațiile confidențiale între 2 entități necesită partajarea unei chei unice numai între ele, dacă entitățile sunt *n*, atunci sunt necesare *n·(n–1)/2* chei diferite pentru a permite comunicarea confidențială între fiecare pereche de părți
2. Atât expeditorul, cât și destinatarul își cunosc cheia partajată, astfel încât *nerepudierea* (repudierea înseamnă negarea de a fi autorul unui anumit mesaj) nu este posibilă

Prima problemă nu ține de numărul de chei pe care trebuie să le gestioneze fiecare subiect, problema este *schimbarea* lor. Fiecare subiect trebuie să convină asupra unei chei cu fiecare dintre ceilalți n-1 subiecți, ceea ce este de fapt o operație complexă; acest lucru face criptografia simetrică nepractică pentru multe utilizări.

A doua problemă, *nerepudierea*, constă în faptul că de multe ori vrem să ne asigurăm că un anumit mesaj a fost trimis efectiv de un anumit subiect. Este clar că dacă un subiect primește un mesaj de la alt subiect și cei doi subiecti sunt singurii care cunosc cheia folosită pentru a-l cripta, fiecare subiect este sigur că celălalt subiect este autorul mesajului. Viceversa, din moment ce ei cunosc doar cheia, fiecare subiect știe că doar celălalt subiect poate citi mesajul. Problema apare atunci când este nevoie să se dovedească acest lucru unei terțe părți, pentru un fel de litigiu, deoarece acest mecanism nu funcționează: fiecare parte este capabilă să creeze un mesaj, să-l cripteze și apoi să spună că cealaltă parte este autor. Problema non-repudierii nu este rezolvabilă

Funded by the European Union. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or the European Education and Culture Executive Agency (EACEA). Neither the European Union nor EACEA can be held responsible for them.

prin utilizarea criptografiei simetrice.

Schema asimetrică este similară cu cea simetrică, dar cheile folosite sunt cele două care sunt generate în pereche:

Criptare: *encryption_function*(*PlainText*, ***Key1***) → *CypherText*

Decriptare: *decryption_function*(*CypherText*, ***Key2***) → *PlainText*

Numele **Key1** și **Key2** au fost folosite în loc de *EncryptionKey* și *DecryptionKey*, deoarece cele două chei sunt interschimbabile matematicSubiectul care generează cheile păstrează în general unul privat/secret (numit *cheie privată*) și îl face pe celălalt public (numit *cheie publică*), de unde și numele clasei de algoritm care se mai numește și *algoritmul cheii publice*. Mai multe despre termenul public vor fi discutate mai târziu.

Funcția folosită pentru criptare nu este neapărat aceeași cu cea folosită pentru decriptare. Important este că împreună reprezintă algoritmul criptografic, o parte se bazează pe o cheie, iar cealaltă parte se bazează pe cealaltă cheie.

Cele două chei au asemenea proprietăți matematice încât descoperirea unei chei din cealaltă este destul de complexă și necesită o perioadă enormă de timp. În special, chiar și cunoscând cheia publică, *PlainText* și *CypherText*, reconstruirea cheii private ar necesita o perioadă nerezonabilă de timp.

<u>Utilizări ale algoritmului asimetric</u>

Să presupunem (detaliile vor fi date mai târziu) *cheia publică* este disponibilă public și proprietarul ei este singurul care cunoaște cheia privată corespunzătoare. Fiind cheia publică disponibilă oricui, oricine o poate folosi pentru a cripta un mesaj, dar numai proprietarul cheii private corespunzătoare va putea decripta mesajul, acest lucru asigură **confidențialitatea** transmiterii mesajului.

Deci am trecut de la o situație în care un subiect avea nevoie de *n* chei diferite pentru a comunica în mod confidențial cu n alți subiecți la o situație în care este nevoie de o singură cheie (cheia publică a receptorului) pentru ca toată lumea să trimită un mesaj confidențial unui subiect.

Pentru a permite *n* subiecți să comunice între ei, fiecare subiect trebuie să aibă o pereche de chei, deci numărul total de chei este *2·n*. Punctul interesant nu este atât de mult că cheile au scăzut ca număr; dar că fiind chei publice acestea pot fi comunicate într-un mod mult mai simplu. Dacă altcineva ajunge să cunoască acele chei publice, nu este deloc o problemă. Prin urmare, aceste chei pot fi într-adevăr publicate, puse la dispoziția tuturor și oricine le poate folosi fără a slăbi protecția mesajelor.

Dacă subiectul este singurul proprietar al unei chei private (secrete), al cărei omolog public este

Funded by the European Union. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or the European Education and Culture Executive Agency (EACEA). Neither the European Union nor EACEA can be held responsible for them.

disponibil public, oricine poate verifica (decripta) mesajele criptate cu cheia privată a subiectului. Apoi, dacă este posibil să se folosească cheia publică a unui subiect pentru a decripta un mesaj, înseamnă că numai subiectul care are cheia privată corespunzătoare ar fi putut cripta mesajul, aceasta prevede **nerepudiarea** mesajului..

O problemă este încă de rezolvat: asigurarea faptului că o cheie publică este într-adevăr asociată subiectului legitim. Aici trebuie furnizată o metodă de certificare *super partes*, aceasta este furnizată de o infrastructură cu *chei* publice (PKI), discutată mai târziu.

Ca o notă secundară, algoritmii *asimetrici* pot fi utilizați și ca *algoritm de provocare-răspuns*. Pentru ca subiectul A *să autentifice* (demonstrează că un subiect este cu adevărat cine pretinde a fi) subiectul B, A trimite (în clar) un număr aleatoriu lui B; apoi B îl criptează cu cheia sa privată și îl trimite înapoi către A. Dacă A este capabil să decripteze mesajul cu cheia publică a lui B, A este sigur că cealaltă parte este B, deoarece este singura care poate cripta ceva cheia publică a lui B poate decripta.

Avantaje și dezavantaje

Numărul total de chei este redus semnificativ, ceea ce scade complexitatea garantării gestionării corecte a tuturor cheilor (și numai proprietarul știe cheia secretă). Deoarece cheile publice sunt puse la dispoziția publicului, nu este nevoie de acorduri specifice între diferiți subiecți. Această utilizare a cheilor publice este, de exemplu, foarte importantă în comerțul electronic, deoarece comerțul electronic nu poate cere să existe un acord a priori al vânzătorului cu clienții.

Pentru a obține această legătură între cele două chei și, în același timp, a garanta că una nu poate fi calculată din cealaltă și că ceea ce este criptat cu una este decriptat cu cealaltă, faceți algoritmii asimetrici mult mai complexi din punct de vedere al problemelor matematice decât algoritmi simetrici. Aceste probleme matematice necesită în esență algoritmi mai lenți decât algoritmii simetrici și utilizarea cheilor mai lungi, cel puțin cu algoritmii utilizați în prezent. Prin urmare, dacă pe de o parte există o flexibilitate mai mare, pe de altă parte, acestea sunt mai oneroase în ceea ce privește resursele de calcul.

Algoritm proeminent: RSA *(de la inițialele lui Ronald **R**ivest, Adi **S**hamir, si Leonard **A**dleman)*
Acest algoritm se bazează pe complexitatea factorizării prime: este ușor să înmulțiți două numere prime și să obțineți rezultatul, dar este complex/insolubil să găsiți cele două numere prime din rezultat. Factorizarea prime este o problemă dificilă. De asemenea, știm că nu este ușor să găsești numere prime foarte mari, deoarece nu există nicio regulă care să permită aflarea numerelor prime în afară de a încerca să le factorizezi și să descoperi dacă sunt prime. Aceasta înseamnă că lucrul pe numere foarte mari, a afla care sunt cele două numere prime din care este derivat un anumit număr este o problemă care necesită resurse de calcul semnificative. Pe de altă parte, operațiunile legitime pot fi efectuate cu resurse de calcul limitate.

Funded by the European Union. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or the European Education and Culture Executive Agency (EACEA). Neither the European Union nor EACEA can be held responsible for them.

Co-funded by the Erasmus+ Programme of the European Union

Algoritm proeminent: Criptografie eliptică *(ECC - Elliptic Curve Cryptography)*

„Problema dificilă” în acest caz este calculul logaritmului discret (într-un câmp finit). Un „câmp finit” înseamnă că valorile întregi sunt într-un set limitat (definiție foarte grosieră).

Problema ECC este mai complexă decât factorizarea prime, așa că obțineți aceeași securitate cu chei mai scurte. Algoritmii ECC sunt mai rapizi decât RSA.

Comparația algoritmilor simetrici și asimetrici

Algoritmii asimetrici exploatează alte probleme matematice decât cele ale criptografiei simetrice. Aceasta înseamnă că lungimile cheilor nu sunt comparabile cu cele ale criptografiei simetrice. Problema cu criptarea simetrică este doar că cheia este suficient de lungă pentru a preveni încercarea tuturor numerelor posibile care ar putea să se potrivească cu o cheie, deci cu o cheie de 128 de biți există 2^{128} (acesta este $3.4 \cdot 10^{38}$) chei posibile. Algoritmii simetrici de astăzi folosesc de obicei 128-, 256- și uneori chei de 512 biți.

Dacă luăm în considerare RSA, de exemplu, care exploatează numere prime, este clar că nu toate cheile sunt posibile, deoarece sunt legate de prezența numerelor prime. Nu toate numerele sunt prime, deci nu toate numerele sunt chei posibile. Foarte simplificator, pentru a avea aceeași robustețe a algoritmului (deci aceeași dificultate din punct de vedere computațional) a unui algoritm simetric de 128 de biți, algoritmul RSA are nevoie de o cheie de ordinul a 1024 de biți. Algoritmii asimetrici de astăzi folosesc de obicei chei de 1024, 2048 și 4096 de biți.

Funcții hash

Funcțiile hash calculează un scurt rezumat cu lungime fixă a unui text de lungime arbitrară; acest rezumat se numește *rezumat*. O funcție hash trebuie să fie simplă și rapidă de calculat.

Proprietăți de securitate

Funcțiile hash sunt „funcții unidirecționale” în sensul că este foarte complex și lent din digest recuperarea unui posibil text care îl generează, dar acest text nu este util. Aceasta este o consecință a faptului că rezumatul textului original este (și ar trebui să fie) mai mare decât rezumatul, astfel încât secvențe practic infinite de octeți pot avea ca rezultat un rezumat specific. Chiar dacă se constată că un text are rezumatul dat, este extrem de improbabil să fie textul original sau chiar să aibă sens. Crearea a două mesaje semnificative cu același rezumat sau construirea unui mesaj semnificativ dintr-un rezumat dat este extrem de dificilă, orice modificare a unui mesaj, chiar și un singur bit, va oferi un rezumat complet diferit (aproximativ 50% dintre biții săi sunt diferiți).

Verificarea integrității

Funded by the European Union. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or the European Education and Culture Executive Agency (EACEA). Neither the European Union nor EACEA can be held responsible for them.

Co-funded by the Erasmus+ Programme of the European Union

Având în vedere un mesaj M, rezumatul său D este calculat prin intermediul unei funcții hash H: D = H(M)

O transmisie sigură a rezumatului D permite să se asigure că mesajul M este corect: receptorul mesajului M calculează rezumatul de pe acesta și îl compară cu cel (D) primit; dacă sunt identice, mesajul nu a fost modificat. Este esențial ca rezumatul să fie transmis într-un mod sigur, altfel nu există deloc protecție.

Acest mecanism este folosit și ca verificare a unui software descărcat în locul altor algoritmi mai puțin puternici (dar mai rapidi), cunoscuți sub numele de CRC (Cyclic Redundancy Checks), obișnuiți în protocoalele de rețea.

Câțiva algoritmi hash importanți

Următoarele sunt doi dintre cei mai cunoscuți algoritmi hash.

MD5 (Message Digest #5, 1991)

MD5 necesită o putere de calcul scăzută (comparativ cu majoritatea celorlalți algoritmi hash) și produce un hash de 128 de biți. De-a lungul timpului a fost slăbit treptat pentru a fi complet nesigur din motive de securitate (în prezent este posibil să găsim două mesaje distincte cu același hash MD5 – ceea ce se numește *coliziune* – în secunde), oricum este încă util în non-criptografie. aplicații, de exemplu pentru a descoperi erori de transmisie în loc de cele mai comune (și slabe, dar mai rapide) CRC-32.

SHA-3 (Secure Hash Algorithm #3, 2015)

SHA sunt de fapt o familie de algoritmi hash securizați, SHA-3 este ultima versiune și rezultatul unei competiții între criptografi. Algoritmul poate fi reglat echilibrând securitatea cu viteza; produce valori hash de 224, 256, 334 și 512 biți. Este de 2-3 ori mai lent decât MD5, dar până acum nu s-a găsit nicio slăbiciune.

Semnături (semnături digitale)

Semnăturile digitale, numite semnătură electronică în legislația unor țări, preiau un mesaj, calculează un hash și îl criptează (doar hash-ul) cu cheia secretă a unui algoritm asimetric. Acest hash semnat cu cheia secretă se numește *semnătură*. O semnătură digitală nu criptează apoi mesajul original, ci doar permite să se garanteze că textul corespunzător nu a fost modificat, pentru că altfel s-ar modifica rezumatul. De asemenea, garantează că originea poate fi verificată (nerepudiere), deoarece a fost de fapt criptată cu cheia privată a unui algoritm asimetric. Aceasta înseamnă că oricine poate verifica dacă textul este intact și a fost de fapt generat de un anumit subiect.

Similar semnăturilor digitale sunt **codurile de autentificare a mesajelor** (MAC). Ele urmăresc, de

Funded by the European Union. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or the European Education and Culture Executive Agency (EACEA). Neither the European Union nor EACEA can be held responsible for them.

Co-funded by the Erasmus+ Programme of the European Union

asemenea, să asigure integritatea și autenticitatea unui mesaj. Rezumatul este calculat prin hashing textul care trebuie protejat și o cheie de *algoritm simetric* (nu există nicio asigurare a nerepudierii, deci este potrivit doar atunci când această caracteristică nu este necesară). Unul dintre cei mai cunoscuti algoritmi este HMAC: Keyed-hash MAC.

Infrastructuri cu cheie publică

Cerințele de confidențialitate, integritate și autenticitate ale operațiunilor de criptare asimetrică și semnătură digitală necesită stabilirea unei relații de încredere între părțile implicate. Deoarece de multe ori părțile nu au avut niciun contact înainte, o terță parte de încredere (de către acestea) trebuie să ofere garanțiile necesare, cu alte cuvinte, această terță parte trebuie să asigure autenticitatea proprietarului unei chei publice.

O *autoritate de certificare* (CA) este o parte a unui model de încredere ierarhic care garantează identitatea utilizatorului. Trebuie recunoscut ca „de încredere” de către toate entitățile implicate în comunicare.

Pentru a asigura autenticitatea unui subiect, o CA semnează un *certificat digital* care leagă subiectul la cheia sa publică (este similar din punct de vedere conceptual cu un document de identitate al unei persoane). Semnătura aplicată de CA este o semnătură digitală, deci este calculată folosind cheia privată a CA. Desigur, certificatul digital trebuie validat folosind cheia publică a CA.

O infrastructură cu *chei publice* (PKI) este o structură ierarhică a CA, în care fiecare CA are cheia publică semnată de CA de nivel anterior (care își folosește cheia privată în acest scop), procesul merge până la Root Certificate Authority care auto- semnează un certificat care deține cheia publică utilizând cheia sa privată.

Un model ierarhic permite construirea de ierarhii de orice adâncime. De exemplu, într-o țară am putea avea autoritățile de certificare municipale, …, regionale, până la o rădăcină CA națională.

Certificatul CA rădăcină trebuie făcut public în mai multe locații și moduri pentru a face imposibilă înlocuirea acestuia cu unul falsificat. De exemplu, sistemele de operare în sine sunt livrate cu un set de CA intermediare rădăcină și de încredere. Figura 1 prezintă un extras din lista de certificate rădăcină dintr-un sistem de operare MS Windows 10.

Funded by the European Union. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or the European Education and Culture Executive Agency (EACEA). Neither the European Union nor EACEA can be held responsible for them.

Co-funded by the Erasmus+ Programme of the European Union

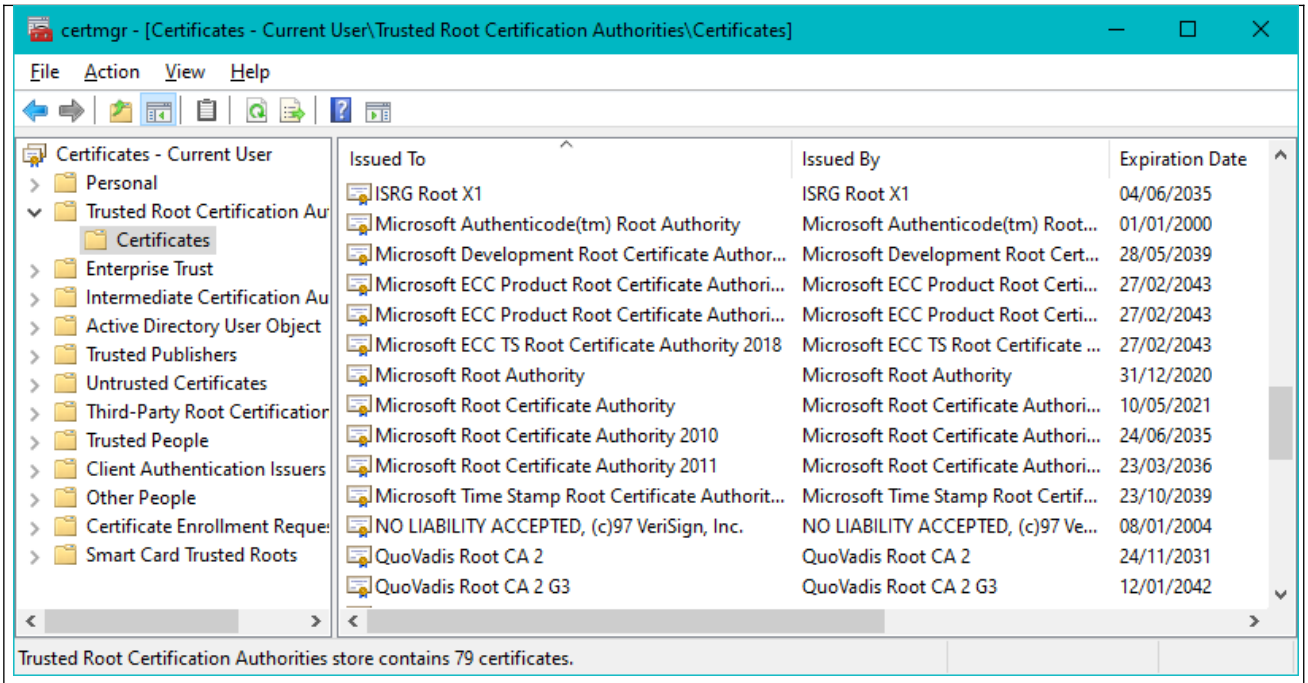

Figura 1. Manager de certificate Windows 10

Componentele unui PKI

Actorul principal este **Autoritatea de Certificare (CA)**, aceasta gestionează cheile *criptografice* și certificatele PKI. Este componenta care deține cheile reale și care apoi efectuează operațiunile criptografice. Evident, în acest sens este și cel mai critic element al autorității de certificare.

Autoritatea de înregistrare (RA) gestionează cererile de certificate și furnizează certificate eliberate. Se ocupă cu distribuția de token-uri criptografice și software (este front-end-ul CA). Este ca un ghișeu, de exemplu, la care un cetățean se poate adresa să solicite un certificat digital și de la care poate fi obținut. Poate fi un contor fizic sau un site web, tipul este asociat cu diferite niveluri de securitate. În plus, un RA poate, în general, să asiste utilizatorul în generarea perechii de chei și gestionarea în siguranță a acestor certificate, de exemplu prin furnizarea de instrumente software sau hardware ad-hoc care sprijină solicitantul în generarea de semnături, criptarea documentelor, verificarea semnăturilor și decriptarea informațiilor.

Un **server de certificate** este un director în care certificatele de utilizator sunt publicate și, într-un fel, pot fi căutate. Directorul urmează standardul X.500 și este interogat utilizând protocolul LDAP.

Certificatul digital

Structura și formatul certificatului digital sunt descrise de standardul ITU-T X.509 (este un standard al Uniunii Internaționale de Telecomunicații numerotat X.509).

Un certificat digital X.509 trebuie să conțină cel puțin următoarele câmpuri:

Funded by the European Union. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or the European Education and Culture Executive Agency (EACEA). Neither the European Union nor EACEA can be held responsible for them.

Co-funded by the Erasmus+ Programme of the European Union

DN (nume distinctiv): Gaius Julius Caesar

CN (nume comun): Iulius Caesar

O (organizație): Guvern

OU (unitatea organizației): Senat

C (țara): Imperiul Roman

Acesta este modelul clasic al unui certificat care îndeplinește standardul X.500, din care X.509 este membru al familiei. Un certificat poate avea alte atribute (de exemplu, e-mail). Poate exista o adresă IP sau o adresă URL care să corespundă unei căi de director X.500 (de exemplu, LDAP – Protocol ușor de acces la director):

IT / Imperiul Roman / Guvern / Iulius Caesar / Gaius Julius Caesar /

Un certificat digital X.509 trebuie să conțină, de asemenea, cel puțin următoarele câmpuri:

Emitent DN: denumirea Autorității în format X.500 (autoritatea de certificare are și un nume distinctiv propriu, care apare în certificat. Evident formatul cu care este reprezentat este același cu numele distinctiv al subiectului la care este se aplică certificat.

Număr de serie: numărul de serie al certificatului deținut de Autoritate (fiecare certificat este identificat în mod unic prin numărul de serie atribuit de către autoritate. Pe de altă parte, perechea (*numărul de serie și numele distinctiv al emitentului*) identifică în mod unic un certificat pe tot parcursul o multitudine de certificate care pot fi eliberate în întreaga lume ca o pereche unică).

NotBefore: data activarii

NotAfter: data expirării

Acestea sunt două informații strict legate între ele. Un certificat are așadar o dată de început de viață și o dată de expirare, după care nu mai este valabil. Motivul derivă din dorința de a proteja cheile de utilizarea excesivă.

Cheie publică: cheia publică a subiectului certificat (certificatul conține date personale și cheia publică)

Algoritmul cheii publice: algoritmul pentru care este potrivită cheia publică

Algoritm de semnătură: algoritmul utilizat de Autoritate pentru a semna Certificatul

Semnătura: semnătura aplicată pe Certificat de către Autoritatea de Certificare (deci elementul

Funded by the European Union. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or the European Education and Culture Executive Agency (EACEA). Neither the European Union nor EACEA can be held responsible for them.

Co-funded by the Erasmus+ Programme of the European Union

care garantează autenticitatea certificatului în sine)

KeyUsage: scopurile pentru care poate fi utilizată cheia

Generarea şi revocarea certificatelor

Standardul RFC 2314 (din PKCS # 10) definește formatul mesajului *certificateRequest* și conține cheia publică și datele solicitantului (ambele date unui RA). Mesajul de solicitare necesită așa-numita *dovadă* a deținerii cheii private: mesajul certificateRequest trebuie semnat cu cheia privată a solicitantului. Autoritatea, prin verificarea acestei semnături, verifică în mod substanțial dacă solicitantul deține de fapt această cheie (altfel solicitantul nu ar fi putut produce această semnătură).

La data corespunzătoare atributului „notAfter", Certificatul este considerat „expirat". Expirarea este un mecanism care servește și pentru a se asigura că aceleași chei asimetrice asociate cu certificatul digital nu pot fi folosite pentru o perioadă excesivă de timp, ceea ce le-ar putea expune la o serie de atacuri criptoanalitice.

Expirarea se poate referi la:

- **certificatul** (cheia subiectului asocierii)
- **cheile asociate**, în raport cu utilizarea pentru care sunt destinate (o cheie poate fi folosită numai pentru anumite scopuri și nu pentru altele)

Un certificat expirat nu poate fi folosit de proprietarul său, deoarece a expirat, utilizarea sa este în practică interzisă. In realitate, nimic nu il impiedica pe detinatorul certificatului sa-l foloseasca in alte scopuri insa toti cei care comunica cu el vor detecta cumva ca acest certificat a expirat si nu il vor accepta. Certificatul este, desigur, disponibil pentru verificarea și decriptarea mesajelor emise înainte de expirare. În unele cazuri, este posibil să „reînnoiți" un certificat, care de obicei nu este automat.

Uneori, un certificat poate fi invalid înainte de expirare prin *revocare* (acest lucru se întâmplă înainte de valoarea atributului „notAfter". Motivul ar putea fi, de exemplu, cheia asociată certificatului a fost compromisă, subiectul nu mai este asociat companiei). sau unitatea schimbată, CA care a emis certificatul (sau una dintre CA-urile sale de nivel superior) a fost compromisă sau din alte motive. Un certificat revocat nu mai poate fi utilizat, decât atunci când este doar menținut „în așteptare". Acesta este un anumit formular de revocare în care certificatul este inserat temporar în CRL din motive specifice (cum ar fi „posibil compromitere a cheii", „cerere de revocare de către o altă persoană decât proprietarul", „neutilizarea temporară a certificatului", de exemplu, pentru concediu)", etc. iar valabilitatea acestuia ar putea fi restabilită ulterior. În cazul „reactivării" de la suspendare, o nouă intrare în CRL este inserată cu un cu un cod motiv special: *removeFromCRL*, deoarece nicio intrare nu poate fi ștearsă vreodată din CRL.

Funded by the European Union. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or the European Education and Culture Executive Agency (EACEA). Neither the European Union nor EACEA can be held responsible for them.

Co-funded by the Erasmus+ Programme of the European Union

Pentru a informa utilizatorii PKI că un certificat și-a pierdut valabilitatea, numărul său de serie este listat într-o *listă de revocare a certificatelor* (CRL). Această listă este emisă periodic de către Autoritatea de Certificare emitentă și propagată (activ sau pasiv) tuturor utilizatorilor, astfel încât toți subiecții care dețin un astfel de certificat nu îl mai pot folosi. CRL respectă standardul ITU-T X.509 - rfc3280 - rfc4325, ceea ce înseamnă că formatul său este standardizat. Este semnat periodic de CA și include un câmp numit „nextUpdate" care conține data și ora la care va fi disponibilă noua Listă de revocare a certificatelor. Dacă această dată aparține trecutului înseamnă că CRL-ul nostru este „expirat", dacă data aparține viitorului știm când va trebui să ne actualizăm lista de certificate revocate. Fiecare certificat revocat include motivul revocării. CRL este furnizat ca serviciu prin protocolul LDAP.

Deoarece CRL-urile pot deveni destul de mari, mai ales dacă autoritatea de certificare are sute de mii sau chiar milioane de utilizatori, acestea pot fi partiționate și CA furnizează doar părți din Lista de revocare a certificatelor, de exemplu, referindu-se la o anumită perioadă de timp. Prin urmare, atunci când este necesară verificarea unui certificat, cetățeanul descarcă singura bucată de CRL de interes. În loc să descărcați un CRL sau o parte a acestuia, un serviciu care implementează *Online Certificate Status Protocol* (OCSP - RFC 2560) poate fi utilizat pentru a afla dacă un anumit număr de certificat este în CRL. Răspunsurile sunt semnate de CA pentru a fi de încredere.

Lanțul de încredere

Pentru a verifica un certificat, un subiect trebuie să verifice tot lanțul CA-urilor intermediare pornind de la unul de încredere, eventual CA-ul rădăcină. Mai întâi, certificatul de verificat este verificat față de CA emitent, apoi certificatul CA este verificat față de CA părinte și așa mai departe până la un CA intermediar de încredere (de către utilizator) sau CA rădăcină în sine..

KeyUsage

De fapt, cheile pot fi folosite în diferite scopuri și un certificat specifică utilizările permise, printre care avem:

- **nonRepudiation**: cheia poate fi folosită pentru producerea unei semnături cu valoare legală care poate fi apoi opusa terților ca semnătură de mână
- **dataEncipherment:** cheia poate fi folosită pentru a cripta datele
- **keyAgreement:** cheia poate fi folosită pentru a schimba în siguranță cheile

TimeStamping

O PKI poate fi, de asemenea, utilizată pentru a ștampila un document în conformitate cu Protocolul de timbru temporal (RFC 3161) pentru a dovedi existența acestuia într-un punct din trecut. Un CA va semna documentul și include o marcă temporală. Semnătura asigură, de asemenea, că documentul nu a fost modificat după aplicarea marcajului de timp. O ștampilă de timp poate prelungi valabilitatea unei semnături electronice dincolo de durata de viață a certificatului digital asociat cheilor utilizate, de exemplu prin solicitarea aplicării unei noi ștampile

Funded by the European Union. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or the European Education and Culture Executive Agency (EACEA). Neither the European Union nor EACEA can be held responsible for them.

Co-funded by the Erasmus+ Programme of the European Union

de timp pe contract în fiecare an care trece. Aplicarea se va face în fiecare an folosind tehnici criptografice cu o rezistență adecvată perioadei în care se aplică semnătura în sine.

Mecanisme și formate standard

Următoarele sunt cele două mecanisme și formate standard utilizate în mod obișnuit.

PKCS #7 - Standard de criptare cu cheie publică #7

PKCS #7 descrie formatul mesajelor/fișierelor semnate, criptate (cu criptare simetrică și asimetrică) și verificabile prin MAC (date digerate)

S/MIME - Secure Multipurpose Mail Extension

Secure Multipurpose Mail Extension (RFC 3851 - SMIMEv3) este o extensie a standardului MIME (RFC 2045, 2046 și 2047 ca extensie a RFC 822) și descrie criptarea mesajelor de e-mail. Entitățile PKCS #7 sunt traduse în mod substanțial în noi entități MIME și, prin urmare, descriu containere noi în care putem găsi porțiuni de mesaje de e-mail semnate sau criptate sau atașamente semnate sau criptate.

Formate de schimb și stocare

PKCS #12 (Personal Information Exchange Syntax Standard #12) definește un format pentru schimbul și stocarea securizată (în fișier) a cheilor criptografice și a certificatelor digitale (lanț de certificate de încredere). Stocarea este sigură deoarece se bazează pe criptarea atât a cheilor, cât și a certificatelor și a întregului lanț de certificate în care un subiect alege să aibă încredere. Este formatul folosit de browsere, clienți de e-mail și sisteme de operare atunci când un utilizator exportă o semnătură și cheie de criptare și un certificat, de exemplu pentru o copie de rezervă sau pentru a o copia pe alt dispozitiv.

Un PKCS #12 criptează datele cu un algoritm simetric a cărui cheie este derivată dintr-o parolă furnizată de utilizator. Acesta garantează integritatea conținutului prin HMAC, sau printr-o semnătură digitală.

Beneficiile utilizării PKCS #12 sunt:

- **nu există costuri de implementare** deoarece este un format disponibil în multe biblioteci pentru multe dispozitive (de la computere la telefoane mobile)
- sunt deja **suportate de majoritatea sistemelor de operare și software-ului**, asta înseamnă că este disponibil imediat
- folosește **algoritmi simetrici „puternici”** (criptarea care se aplică este o criptare puternică, foarte rezistentă, deci sunt chei care rezistă de exemplu atacurilor de forță brută
- este **portabil pe diferite dispozitive**, este posibil să mutați un fișier pe alte dispozitive sau să îl copiați pe toate dispozitivele pe care le folosim pentru a avea cheile mereu disponibile

Funded by the European Union. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or the European Education and Culture Executive Agency (EACEA). Neither the European Union nor EACEA can be held responsible for them.

Co-funded by the Erasmus+ Programme of the European Union

Există și unele dezavantaje:

- **poate fi șters, copiat, transmis**, aceasta este o problemă importantă deoarece poate fi furat, șters, copiat pentru a-l elimina sau pur și simplu nu poate fi disponibil
- **cheia privată trebuie utilizată direct de software-ul de semnare/decriptare,** asta înseamnă că software-ul folosește decriptează acest fișier și accesează cheia direct, așa că există un moment când cheia este în clar pe sistem, neprotejată, deși poate este doar memoria volatilă care este zona cel mai puțin ușor accesibilă; cu toate acestea, este disponibil în text clar pe sistem într-un fel și aceasta este o limitare foarte puternică a acestui format
- **este posibil să se efectueze atacuri de forță brută,** de exemplu, în cazul în care cineva fură fișierul PKCS #12, poate încerca toate parolele posibile într-un sistem computerizat separat

Jetoane criptografice

Tokenurile criptografice sunt dispozitive hardware care implementează nativ algoritmi de criptare. Nu permit accesul direct la materialul criptografic din exterior, sunt singurii abilitati să acceseze și să utilizeze cheia. Mai mult, aceste comenzi pot fi executate în general numai după ce utilizatorul, iar sistemul în numele utilizatorului, se autentifică la token, de exemplu prin introducerea unui PIN. Tokenurile pot avea un sistem de operare mic prin care utilizatorul solicită operațiuni de generare a cheilor, criptare și semnătură a datelor. Distrugerea în siguranță a jetonului înseamnă adesea nu doar anularea, ci și incapacitatea de a accesa zonele de memorie în care a fost scrisă cheia, astfel încât să evităm orice formă de atac pe care nu am putut-o concepe astăzi. Sistemul de operare și hardware-ul token pot fi certificate conform standardelor de securitate specifice.

Mecanismul provocare-răspuns

Un mecanism de provocare/răspuns permite două părți să se autentifice fără a expune informații utile pe canalul de comunicare. Ideea este că o parte A „provocă” cealaltă parte B să opereze pe un număr aleatoriu (numit nonce pentru că este un număr folosit o singură dată) cu propria sa cheie, de exemplu cu o funcție unidirecțională H. Apoi B returnează H(nonce) la A. Dacă B returnează ceea ce Așteaptă, A este sigur de identitatea lui B. În cel mai simplu caz, B criptează nonce cu o cheie secretă cunoscută de A. A, care are aceeași cheie, în turn aplică funcția nonce folosind aceeași cheie și verifică dacă numărul dat de B se potrivește cu cel calculat.

Aceasta este o autentificare bazată pe o parolă care nu a fost trimisă pe canal, deci nu este posibilă urmărirea cheii. Ceea ce face mecanismul eficient este faptul că numărul aleatoriu este diferit pentru fiecare sesiune. Deoarece informațiile citite pe canal sunt diferite pentru fiecare sesiune, nu sunt utile pentru autentificarea ulterioară. Un avantaj este că nu este necesară sincronizarea (spre deosebire de mecanismul One-Time-Password, unde este important să cunoașteți punctul de pornire al Padului de utilizat). Deoarece este dificil să fie folosit de oameni,

Funded by the European Union. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or the European Education and Culture Executive Agency (EACEA). Neither the European Union nor EACEA can be held responsible for them.

Co-funded by the Erasmus+ Programme of the European Union

este de obicei folosit în autentificarea de la mașină la mașină. Un mecanism de provocare-răspuns se găsește în mecanismele de autentificare în rețea ale routerelor ADSL către serverul de autentificare a furnizorului.

Întrebări:

În fișier separat.

Atasamente:

Powerpoint într-un fișier separat.

Referințe:

Fiecare subiect are propria sa pagină pe Wikipedia, uneori deja prea amănunțită pentru scopurile acestui curs; cărțile sunt prea specializate și sunt utile doar după o pregătire foarte profundă pe tema specifică.

Funded by the European Union. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or the European Education and Culture Executive Agency (EACEA). Neither the European Union nor EACEA can be held responsible for them.