chapter 1 securitatea cibernetică: concepte, abordare

1.1 ce este securitatea cibernetică?

Securitatea cibernetică constă dintr-un ansamblu de măsuri pentru protejarea integrității rețelelor, a programelor și datelor împotriva atacurilor, distrugerii sau accesului neautorizat la acestea.

Putem de asemenea defini securitatea cibernetică ca un cadru de protecție a activelor informaționale, prin prevenirea amenințărilor la resursele informaționale, stocate și transmise prin intermediul sistemelor informaționale din rețea.

De ce este nevoie de servicii de securitate cibernetică? Enunțăm câteva din rațiunile existenței unei politici și infrastructuri de securitate cibernetică/informatică:

- un nivel tot mai înalt al criminalității cibernetice
- cerințe crescânde de protecție a datelor datorită nivelului tot mai ridicat de informatizare
- existența globală a unor sindicate ale crimei
- structuri și activități de atac cibernetice susținute la nivel guvernamental
- fraudele financiare

Una din componentele esențiale ale securității cibernetice este securitatea informațiilor. Aceasta

1.2 servicii criptografice

Criptografia oferă următoarele servicii:

- autentificare
- integritatea datelor
- non-repudiere
- confidenţialitate

Să detaliem puţin aceste concepte.

Autentificarea permite destinatarului mesajului să valideze identitatea expeditorului. Previne insinuarea unei entităţi neautorizate ca fiind expeditorul legitim al mesajului.

Integritatea datelor garantează că mesajul trimis nu a fost modificat sau alterat pe parcursul comunicării. Acest lucru este realizat, de obicei, prin atașarea la mesaj a unui șir de control (digest, hash) de lungime fixă. Acest șir de control poate fi recreat de către destinatar și permite identificarea unei modificări intenționate sau nu a conținutului mesajului trimis.

Non-repudierea, înseamnă, în securitatea digitală, două lucruri:

- un serviciu care oferă dovada integrității și (mai ales) a originii datelor
- o modalitate de autentificare cu un nivel înalt de încredere

Confidențialitatea, ca serviciu, previne accesul la conținutul real al mesajului din partea entităților neautorizate.

1.3 clasificarea algoritmilor criptografici

Există două tipuri principale de algoritmi de criptare cu cheie:

- cheie secretă – algoritmi cu cheie simetrică

- cheie publică – algoritmi cu cheie asimetrică

Algoritmii cu cheie simetrică se bazează pe caracterul privat (secret) al cheii de criptare/decriptare. Cheie este cunocută de către de către expeditor și de către destinatar.

Acești algoritmi pot fi clasificați în continuare în algorimi de tip **șir** (stream) și în algoritmi de tip **bloc** (block). Cei dintâi au ca unitate de codificare un caracter în timp ce ceilălat tip acționează asupra unui bloc de caractere, care este considerat drept unitate de codificare.

Viteza de execuție a algoritmilor simetrici este mult mai mare ca cea a celor asimetrici. Pe de altă parte, schimbul de chei dintre expeditor și destinatar implicat de utilizarea algoritmilor simetrici ridică noi probleme de securitate. În practică, se obișnuiește utilizarea cripătării asimetrice pentru generarea și schimbul de chei ce urmează a fi utilizate pentru criptare propriu-zisă (cea simetrică) a mesajului.

1.4 algoritmi de criptare cu cheie simetrică

Algoritmii de criptare cu cheie simetrică utiliyeayă o singură cheie (secretă) pentru a efectua atât criptarea cât și decriptarea mesajului. Dn acest motiv, păstrarea caracterului secret a cheii comune este crucială pentru păstrarea caracterului secret al comunicării. Algoritmii cei mai utilizați la ora actuală sunt:

- **DES** – Data Encryption Standard (Standard de criptare a datelor) - a fost dezvoltat la mijlocul anilor 70. Este un standard NIST (US National Institute of Standards and Technology). DES este un algoritm de tip bloc (block) care utilizează blocuri de 64 de biți și o cheie de 56 de biți. Lungimea modestă a cheii îl face vulnerabil la atacuri de forță brută. Specificația inițială a acestui algoritm de criptare este FIPS (Federal Information Processing Standards) 46. Cea mai nouă versiune a DES-ului este Triple-DES (sau 3-DES) și se bazează pe utilizarea de trei ori a DES-ului, cu trei chei diferite, independente. Este, desigur, mai puternic decât DES, dar este și mai lent, dacă îl comparăm cu algoritmii mai noi de criptare. 3-DES este specificat în FIPS 46-3 (Octombrie 1999)
- **AES** – Advanced Encryption Standard (Standard de Criptare Avansată) - obiect al FIPS 197 (noiembrie 2001). AES este un cifru bloc care utilizează blocuri de 128 biți și o cheie de dimensiune 128 biți. Sunt specificate și variante utilizând chei de 192 și 256-biți. Ceea ce este specific pentru acest algoritm este faptul că procesează date la nivel de octet, spre deosebire de procesarea la nivel de biți care a fost folosită anterior. Algoritmul este eficient și considerat sigur.

1.5 distribuirea cheilor secrete

Utilizarea algoritmilor de criptare cu cheie simetrică presupune existența unui sistem robust de distribuție a cheilor. Desigur că aceste chei, fiind ele însele secrete, trebuie încriptate înainte de a fi trimise în mod electronic sau pot fi distribuite prin alte mijloca (prin curieri, de exemplu) pentru a preveni interceptarea cheilor de către o entitate neautorizată.

Există două standarde pentru distribuirea automată a cheilor secrete. Primul standard, numit X9.17 este definit de către ANSI (American National Standards Institute) iar cel de-al doilea este protocolul Diffie-Hellman.

În acest curs ne vom ocupa doar de cea de-a doua metodă, protocolul Diffie-Hellman.

1.6 algoritmi cu cheie asimetrică

Algoritmii cu cheie asimetrică se bazează pe două key distincte pentru implementarea celor două faze: criptarea și, rspectiv, decriptarea:

- o cheie publică, care poate fi distribuită sau facută publică la cerere
- o cheie privată (secretă) care corepunde unei anume chei publice, dar care este cunoscută doar de proprietarul cheii sau de către alte entități autorizate.

Fiecare din aceste două chei definește o funcție de transformare. Cele două transformări definite de către o pereche de chei publice/private sunt invese una celeilalte și pot fi utilzate pentru criptarea/decriptarea unui mesaj. Nu este important care din cele două chei este folosită pentru criptare/decriptare.

Deși algoritmiide criptare cu cheie asimetrică sunt mai lenți decât cei cu cheie simetrică, aceștia au avantajul de a elimina procesul de distribuire a cheilor.

Algotimii principalii de criptare cu cheie asimetrică:

- **RSA** – (Rivest-Shamir-Aldeman) este cel mai utilzat algoritm cu cheie asimetrică. Este utilizat în principal pentrudistribuirea cheilor și pentru semnături digitale. Toate calculele sunt făcute modulo un întreg (de regulă de peste 2048 biți, adică peste 617 cifre zecimale) n = p*q, unde numărul n este public, dar p și q sunt numere prime secrete. Pornind de la reprezentarea numerică a unui mesaj m și de la un exponet public e, este creat un text cifrat c = m^e (mod n). Destinatarul utilizează expunentul invers al lui e modulo (p-1)*(q-1) (i.e.d = e^(-1) (mod (p-1)*(q-1)) pentru decriptare, adică c^d = m^(e*d) = m (mod n). Cheia privată este (n, p, q, e, d) (sau doar p, q, d) în timp ce cheia publică este (n, e). Mărimea lui n trebuie să fie cel puțin de 1024 biți (peste 10^300), dar cerințele curente pretind o valoare de 2048 biți.
- **Rabin** – acest sistem de criptare este echivalent cu procesul de factorizare a unui număr, problemă care este în sine dificilă. Punctul de plecare îl reprezintă (ca și în cazul RSA) un număr mare n = p*q, unde p și q sunt numere prime secrete. Mesajul original m (văzut ca și număr) este ridicat la pătrat (c = m^2(mod n)) și trimis astfel la destinație. Destinatarul calculează rădăcina pătrată a lui c mod p și mod q, rezultând 4 variante distincte ale mesajului original. Una din variante e cea corectă. Deși nu este obiectul unui standard (RSA este standardizat), algoritmul este bine explicat în câteva cărți și prezentări. Cheile de 1024 de biți și peste sunt considerate sigure.

1.7 funcții hash

Funcțiile hash au ca intrare un mesaj (fișier, șir) de lungime arbitrară și generează o sumă/șir de control (numit și hash sau digest) de lungime fixă. Lungimea acestui șir de control depinde de funcția utilizată, dar în general este între 128 și 512 biți.

Fucțiile hash sunt utilizate în special în trei domenii:

- garantarea integrității uni mesaj (sau a unui fișier descărcat) prin atașasrea la mesaj a hash-ului generat. Destinatarul poate recalcula hash-ul mesajului/fișierului primit și acesta poate fi comparat cu hash-ul atașat de către expeditor. Dacă acestea coincid, avem garanția (cu un foarte înalt grad de încredere) că mesajul/fișierul nu a fost alterat.
- Sunt parte a procesului de creare a semnăturii digitale
- stocarea parolelor – parolele nu sunt aproape niciodată (sau nu ar trebui să fie) stocate în forma lor originală. Ceea ce este stocat, în general, este un hash al parolei. Atunci când un utilizator introduce o parolă, este calculat hash-ul acesteia, care este apoi comparat cu hash-ul stocat. Dacă ele coincid, parola este considerată ca fiind cea corectă.

Cele mai utilizate funcții hash sunt cele din familia MD și SHA - și anume MD5 și SHA-1, cele mai noi fiind SHA-2 și SHA-3. SHA înseamnă Secure Hash Algoritm. O altă funcție hash de interes este RipeMD - 160. Funcțiile MD generează un hash de 128 biți și au fost proiectate de compania RSA Security. În timp ce MD5 este încă răspândit, MD4 a fost spart și este considerat nesigur. SHA-1 și RipeMD-160 sunt considerate sigure pentru moment. În timp ce SHA-2 este o extensie a SHA-1, SHA-3 oferă un nou algoritm pentru calculul hash-ului.

Începând cu cele mai noi funcții, iată o listă de funcții hash de interes practic.

- SHA-3 utilizează algoritmul Keccak, o construcție de burete (sponge) în care blocurile de mesaje sunt XOR-ed într-un subset al stării, care este apoi transformat ca un întreg. În versiunea utilizată în SHA-3, starea constă într-o matrice de 5 × 5 cuvinte pe 64 de biți,]n total 1600 de biți. Procesul de standardizare a fost finalizat în august 2015 ca FIPS 202 - https://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.202.pdf .

- SHA-2 include schimbări semnificative față de predecesorul său, SHA-1. Familia SHA-2 constă din șase funcții hash cu digest (valori hash) care sunt de lungime 224, 256, 384 sau 512 biți, cunoscute sub denunirile: SHA-224, SHA-256, SHA-384, SHA-512, SHA-512/224 și SHA-512/256.
- SHA-1 - Algoritm Hash Securizat. Publicat de Guvernul Statelor Unite. Specificația sa este obiectul FIPS 180-1 (aprilie 1995). FIPS reprezintă Federal Information Processing Standard (Standardele Federale de Procesare a Informațiilor). Produce un hash de 160 de biți (5 cuvinte pe 32 de biți).
- RipeMD-160 - conceput ca înlocuitor pentru seria MD. Produce un hash de 160 de biți (sau 20 de octeți, dacă doriți).
- MD5 - Message Digest 5. Dezvoltat de laboratoarele RSA. Produce un hash de 128 biți. Încă în uz, mai ales pentru verificarea integrității mesajelor (sau a fișerelor descărcate).
- MD2, MD4 - Algoritmi de hash mai vechi dezvoltate de către RSA Data Security. Deoarece au defecte cunoscute, ele sunt doar de interes istoric.

1.8 semnătura digitală

Unii algoritmi cu cheie publică pot fi utilizați pentru a genera semnături digitale. O semnătură digitală este o cantitate mică de date care poate fi creată utilizând o cheie privată și există o cheie publică care poate fi utilizată pentru a verifica dacă semnătura a fost generată cu adevărat utilizând cheia privată corespunzătoare. Algoritmul utilizat pentru a genera semnătura trebuie să fie astfel încât, fără a cunoaște cheia privată, nu este posibilă crearea unei semnături care să se confirme ca fiind validă.

Semnăturile digitale sunt folosite pentru a verifica dacă un mesaj vine de la expeditorul revendicat (presupunând că numai expeditorul cunoaște cheia privată corespunzătoare cheii publice). Aceasta se numește autentificare (a originii datelor). Ele pot fi, de asemenea, folosite pentru a marca temporal (timestamp) documente: o entitate de încredere semnează documentul și marca temporală cu cheia privată, certificând astfel că documentul a existat la momentul menționat.

Semnăturile digitale pot fi, de asemenea, folosite pentru a certifica faptul că o cheie publică aparține unei anumite entități. Aceasta se face prin semnarea combinației cheii publice și a informațiilor despre proprietarul acesteia printr-o cheie de încredere. Structura de date rezultată este adesea numită un certificat de cheie publică (sau pur și simplu, un certificat). Certificatele pot fi considerate ca fiind similare cu pașapoartele care garantează identitatea purtătorilor lor.

Entitatea de încredere care eliberează certificate entităților identificate se numește o autoritate de certificare (CA – Certificate Authority). Autoritățile de certificare pot fi considerate ca fiind similare cu guvernele care eliberează pașapoarte pentru cetățenii lor.

O autoritate de certificare poate fi administrată de un furnizor extern de servicii de certificare sau chiar de un guvern sau CA poate aparține aceleiași organizații ca entitățile. De asemenea, CA pot emite certificate altor (sub-) CA. Acest lucru duce la o ierarhie de certificare asemănătoare arborilor. Cel mai înalt CA de încredere din arbore este numit CA rădăcină. Ierarhia de încredere formată de entitățile finale, sub-CA-uri și rădăcină CA este numită infrastructură cu cheie publică (PKI – Public Key Infrastructure).

O infrastructură cu cheie publică nu necesită neapărat o ierarhie sau rădăcini acceptate universal și fiecare parte poate avea puncte de încredere diferite. Acesta este conceptul web de încredere utilizat, de exemplu, în PGP (Pretty Good Privacy).

O semnătură digitală a unui document arbitrar este în mod tipic creată prin calcularea unui hash (digest) al mesajului din document și prin concatenarea acestuia cu informații despre semnatar, un marcaj temporal (de timp) etc. Aceasta se poate face prin aplicarea unei funcții hash criptografice pe date. Șirul rezultat este apoi criptat utilizând cheia privată a semnatarului utilizând un algoritm adecvat. Blocul de biți criptat rezultat este semnătura. Acesta este adesea distribuit împreună cu informații despre cheia publică care a fost utilizată pentru semnarea acestuia.

Pentru a verifica o semnătură, destinatarul determină mai întâi dacă are încredere că cheia aparține persoanei de care ar trebui să aparțină (folosind un certificat sau o cunoaștere a priori) și apoi decriptează semnătura utilizând cheia publică a persoanei. Dacă semnătura se decriptează în mod corespunzător și informațiile se potrivesc cu cea a mesajului (digest (hash) corect al mesajului etc.), semnătura este acceptată

ca valabilă. Pe lângă autentificare, această tehnică asigură și integritatea datelor, ceea ce înseamnă că se detectează modificări neautorizate sau accidentale ale datelor în timpul transmiterii.

Câteva metode pentru realizarea și verificarea semnăturilor digitale sunt disponibile în mod liber. Cel mai cunoscut algoritm este RSA.

1.9 protocoale și standarde criptografice

- **DNSSEC** – Domain Name Server Security. Protocol pentru servicii de numire distribuite
- **SSL** – Secure Socket Layer – principalul protocol pentru conexiuni WWW securizate. Increasing importance due to higher sensitive information traffic. The latest version of the protocol is called **TLS** – Transport Security Layer. Was originally developed by Netscape as an open protocol standard.
- **SHTTP** – un protocol mai nou, mai flexibil decât SSL/TKS. Specificat în RFC 2660.
- **PKCS** – Public Key Encryption Standards – dezvoltat de către RSA Data Security și definește modalități sigure de utilizare a RSA.

1.10 vulnerabilitatea algoritmilor criptografici

Sistemele criptografice bune trebuie întotdeauna proiectate astfel încât să fie cât mai greu de spart. Este posibil să se construiască sisteme care nu pot fi sparte în practică (deși acest lucru nu poate fi, de obicei, dovedit). Acest lucru nu crește în mod semnificativ efortul de implementare a sistemului; cu toate acestea, este necesară atenție și expertiză. Nu există nici o scuză pentru un proiectant de sistem să facă sistemul vulnerabil din start. Orice mecanisme care pot fi folosite pentru a ocoli securitatea trebuie să fie explicate, documentate și aduse în atenția utilizatorilor finali.

În teorie, orice metodă criptografică cu o cheie poate fi spartă prin încercarea succesivă a tuturor cheilor. Dacă utilizarea forței brute pentru a încerca toate cheile este singura opțiune, puterea de calcul necesară crește exponențial cu lungimea cheii. O cheie pe 32 de biți are nevoie de 2^32 (aproximativ 10^9) pași. Acesta este ceva ce poate face oricine pe computerul său de acasă. Un sistem cu chei pe 56 de biți, cum ar fi DES, necesită un efort substanțial, dar utilizarea unor sisteme distribuite masive necesită doar câteva ore de calcul. În 1999, o căutare de forțe brute folosind un supercomputer special conceput și o rețea mondială de aproape 100.000 de PC-uri pe Internet a găsit o cheie DES în 22 de ore și 15 minute. Se crede că cheile cu cel puțin 128 de biți (cum ar fi AES, de exemplu) vor fi suficiente împotriva atacurilor de forță brută în viitorul apropiat.

Cu toate acestea, lungimea cheii nu este singura problemă relevantă. Multe cifruri pot fi sparte fără a încerca toate cheile posibile. În general, este foarte dificil să se proiecteze metode de cifrare care nu ar putea fi sparte mai eficient folosind alte metode.

Modelele nepublicate sau secrete ar trebui, în general, să fie privite cu suspiciune. Destul de des designerul nu este sigur de siguranța algoritmului, sau securitatea acestuia depinde de secretul algoritmului. În general, nici un algoritm care depinde de secretul algoritmului nu este sigur. Pentru profesioniști, este ușor să dezasambleze și să inverseze algoritmul. Experiența a arătat că marea majoritate a algoritmilor secrete care au devenit cunoscute mai târziu au fost în realitate slabe.

Cheile folosite în algoritmii cu chei publice sunt de obicei mult mai lungi decât cele utilizate în algoritmii simetrici. Acest lucru este cauzat de structura suplimentară care este disponibilă pentru cript-analizator. Problema nu este aceea de a ghici cheia corectă, ci de a deduce cheia privată potrivită din cheia publică. În cazul RSA, acest lucru se poate face prin factorizarea unui număr întreg, care are doi factori principali mari. În cazul altor sisteme criptosisteme, acest lucru este echivalent cu calculul logaritmului discret modulo un număr întreg (care se consideră a fi aproximativ comparabil cu factorizarea atunci când moduli este un număr mare mare). Există criptosisteme cu cheie publică bazate pe alte principii.

Pentru a da o idee despre complexitatea criptosistemului RSA, un modul de 256 de biți este ușor de luat în considerare la domiciliu, iar cheile de 512 biți pot fi sparte de către grupurile de cercetare universitare în câteva luni. Cheile cu 768 de biți probabil că nu sunt sigure pe termen lung. Cheile cu 1024 de biți și mai mult ar trebui să fie în siguranță pentru moment, dacă nu se fac progrese criptografice majore împotriva RSA.

RSA Security pretinde că cheile de 1024-biți sunt echivalente cu rezistența la cheile simetrice de 80 de biți și recomandă utilizarea acestora până în 2010. Cheile RSA de 2048-biți sunt susținute a fi echivalente cu cheile simetrice de 112 biți și pot fi utilizate cel puțin până în 2030 .

Trebuie subliniat faptul că puterea unui sistem criptografic este de obicei egală cu cea mai slabă verigă. Nici un aspect al designului sistemului nu ar trebui să fie trecut cu vederea, de la alegerea algoritmilor la politicile de distribuție și utilizare a cheilor.

1.11 analiza criptografică și spargerea sistemelor de criptare

Criptanaliza este arta descifrării comunicațiilor criptate fără cunoașterea cheilor corecte. Există multe tehnici criptanalitice. Unele dintre cele mai importante pentru un implementator de sistem sunt descrise mai jos.

• atac de tip **text cifrat** (ciphertext): aceasta este situația în care atacatorul nu știe nimic despre conținutul mesajului și trebuie să lucreze doar cu textul cifrat. În practică, este destul de des posibil să se facă presupuneri despre textul original, deoarece multe tipuri de mesaje au anteturi de format fix. Chiar și scrisorile și documentele obișnuite încep într-un mod foarte previzibil. De exemplu, multe atacuri clasice folosesc analiza de frecvență a textului cifrat, însă acest lucru nu funcționează bine împotriva algoritmilor moderni.

Sistemele de criptare moderne nu sunt vulnerabile la atacurile numai de tip text cifrat, deși uneori acestea sunt considerate cu presupunerea suplimentară că mesajul conține unele particularități statistice.

• atac de tip **text original parțial** (known plaintext): Atacantul știe sau poate ghici textul original pentru anumite părți ale textului cifrat. Sarcina este de a decripta restul blocurilor de tip text cifrat utilizând aceste informații. Acest lucru se poate face prin determinarea cheii folosite pentru criptarea datelor sau prin intermediul unei scurtături.

Una dintre cele mai cunoscute metode cunoscute de text original parțial este criptanaliza liniară împotriva cifrurilor de tip bloc.

• atac de tip **text original ales** (chosen plaintext): Atacatorul este capabil să aibă orice text pe convenabil criptat cu cheia necunoscută. Sarcina este de a determina cheia utilizată pentru criptare.

Un bun exemplu al acestui atac este criptanaliza diferențială care poate fi aplicată împotriva cifrurilor (algoritmilor) de tip bloc (și, în unele cazuri, și împotriva funcțiilor de hash).

Unele sisteme de criptare, în special RSA, sunt vulnerabile la atacurile de text original ales. Atunci când se utilizează astfel de algoritmi, trebuie să se aibă grijă să se proiecteze aplicația (sau protocolul) astfel încât un atacator să nu aibă niciodată posibilitatea de a cripta un text ales.

• atac de tip **man-in-the-middle** (intermediar): acest atac este relevant pentru comunicarea criptografică și pentru protocoalele de distribuție a cheilor. Ideea este că atunci când două părți, A și B, schimbă cheile pentru o comunicare securizată (de exemplu, folosind protocolul Diffie-Hellman), un adversar se poziționează între A și B pe linia de comunicație. Adversarul interceptează apoi semnalele pe care A și B le trimit reciproc și efectuează un schimb de chei cu A și B separat. A și B vor avea în final chei diferite, fiecare dintre ele fiind cunoscută adversarului. Intermediarul poate apoi decripta orice mesaj de la A cu cheia pe care o împarte cu A și apoi să retransmită mesajul la B prin criptarea din nou cu cheia pe care o împarte cu B. Atât A cât și B vor crede că comunică în siguranță, dar de fapt adversarul aude totul.

Modalitatea obișnuită de a preveni atacul de tip "man-in-the-middle" este utilizarea unui sistem de criptare cu cheie publică capabil să furnizeze semnături digitale. Pentru configurare, părțile trebuie să-și cunoască în avans cheile publice. După ce secretul partajat a fost generat, părțile trimit semnăturile digitale între ele. Intermediarul (man-in-the-middle) nu reușește în atacul său, deoarece nu reușește să creeze aceste semnături fără cunoașterea cheilor private folosite pentru semnare.

Această soluție este suficientă dacă există și o modalitate de a distribui în siguranță cheile publice. Un astfel de mod este o ierarhie de certificare, cum ar fi X.509. Este folosit de exemplu în IPSec.

• **Corelația** dintre cheia secretă și ieșirile sistemului de criptare este sursa principală de informații pentru criptanalizator. În cel mai simplu caz, informațiile despre cheia secretă sunt scurgeri directe ale sistemului de criptare. Cazurile mai complicate necesită studierea corelației (în esență, a oricărei relații care nu ar fi de

așteptat doar pe baza hazardului) între informațiile observate (sau măsurate) despre sistemul de criptare și informațiile esențiale ghicite.

De exemplu, în atacurile liniare (sau diferențiale) împotriva cifrurilor de tip bloc, cript-analizatorul studiază textele cunoscute (respectiv alese) și textul cifrat observat. Ghicind câțiva biți ai cheii sistemului de criptare, analistul determină prin corelarea dintre textul plat și textul cifrat dacă a ghicit corect. Acest lucru poate fi repetat și are multe variații.

Criptanaliza diferențială introdusă de Eli Biham și Adi Shamir la sfârșitul anilor 1980 a fost primul atac care a folosit pe deplin această idee împotriva cifrurilor de tip bloc (mai ales împotriva DES). Mai târziu, Mitsuru Matsui a venit cu criptanaliza liniară, care era și mai eficientă împotriva DES. Mai recent, s-au dezvoltat noi atacuri folosind idei similare.

Poate că cea mai bună introducere la acest material este procesul de EUROCRYPT și CRYPTO pe parcursul anilor 1990. Se poate găsi discuția lui Mitsuru Matsui despre criptanaliza liniară a DES și despre ideile unor diferențe trunchiate de Lars Knudsen (de exemplu, cryptanaliza IDEA). Cartea lui Eli Biham și Adi Shamir despre criptanalizarea diferențiată a DES este lucrarea "clasică" pe acest subiect.

Ideea de corelare este fundamentală pentru criptografie și câțiva cercetători au încercat să construiască criptosisteme care sunt promițător de protejate împotriva acestor atacuri. De exemplu, Knudsen și Nyberg au studiat securitatea dovedită împotriva criptanalizei diferențiate.

• Atacul **împotriva sau utilizarea hardware-ului subiacent**: în ultimii ani, pe măsură ce tot mai multe dispozitive mobile de criptare mobile au intrat în uz larg, o nouă categorie de atacuri a devenit relevantă și urmărește direct implementarea hardware a sistemului de criptare.

Atacurile utilizează datele de la măsurătorile foarte fine ale dispozitivului crypto, care fac, de exemplu, criptarea și calculează informațiile cheie din aceste măsurători. Ideile de bază sunt apoi strâns legate de cele din alte atacuri de corelație. De exemplu, atacatorul ghicește niște biți ai cheii și încearcă să verifice corectitudinea presupunerii studiind corelația cu măsurătorile ei.

S-au propus mai multe atacuri, cum ar fi utilizarea temporizării fine a dispozitivului, măsurători fine ale consumului de energie și modele de radiații. Aceste măsurători pot fi utilizate pentru a obține cheia secretă sau alte tipuri de informații stocate pe dispozitiv.

Acest atac este, în general, independent de algoritmii criptografici utilizați și poate fi aplicat oricărui dispozitiv care nu este protejat în mod explicit împotriva acestuia.

Mai multe informații despre analiza puterii diferențiale sunt disponibile la http://www.cryptography.com.

• **Defectele în criptosisteme** pot duce la criptanaliză și chiar la descoperirea cheii secrete. Interesul pentru dispozitivele criptografice conduce la descoperirea faptului că unii algoritmi s-au comportat foarte prost odată cu introducerea de mici defecte în calculul intern.

De exemplu, implementarea obișnuită a operațiunilor cu cheie privată RSA este foarte susceptibilă la atacurile de eroare. S-a demonstrat că prin producerea unui bit de eroare la un punct adecvat se poate dezvălui factorizarea modulului (adică dezvăluie cheia privată).

Idei similare au fost aplicate la o gamă largă de algoritmi și dispozitive. Este deci necesar ca dispozitivele criptografice să fie proiectate pentru a fi foarte rezistente împotriva defectelor (și împotriva introducerii a defectelor de către criptanalizatori).

• **Calculul cuantic**: lucrarea lui Peter Shor despre factorizare în timp polinomial și algoritmi pentru logaritmul discret , utilizând computerele cuantice a provocat un interes tot mai mare în calculul cuantic. Calculul cuantic este un domeniu recent de cercetare care utilizează mecanica cuantică pentru a construi computere care, în teorie, sunt mai puternice decât computerele seriale moderne. Puterea derivă din paralelismul inerent al mecanicii cuantice. Deci, în loc să efectuăm sarcini una câte una, cum fac mașinile de serie, computerele cuantice le pot efectua pe toate odată (în paralel). Astfel, se speră că, cu ajutorul calculatoarelor cuantice, putem rezolva problemele imposibile pentru computerele seriale.

Rezultatele lui Shor implică faptul că, dacă calculatoarele cuantice ar putea fi implementate eficient, atunci majoritatea criptografiei cu cheie publică vor deveni istorie. Cu toate acestea, ele sunt mult mai puțin eficiente împotriva criptografiei cu cheie secretă.

Starea actuală a computerelor cuantice nu pare alarmantă, deoarece au fost implementate doar mașini foarte mici. Teoria calculului cuantic oferă promisiuni pentru performanțe mai bune decât computerele seriale, cu toate acestea, dacă se vor realiza în practică, este o întrebare deschisă.

Mecanica cuantică este, de asemenea, o sursă pentru noi modalități de secretizare a datelor și o comunicare sigură cu potențialul de a oferi o securitate nevulnerabilă, acesta fiind domeniul criptografiei cuantice. Spre deosebire de calculul cuantic, multe implementări experimentale de succes ale criptografiei cuantice au fost deja realizate. Cu toate acestea, criptografia cuantică este încă departe de a fi realizată în aplicații comerciale.

• criptografia ADN: Leonard Adleman (unul dintre inventatorii RSA) a venit cu ideea de a folosi ADN-ul ca calculatoare. Moleculele de ADN ar putea fi văzute ca un calculator foarte mare capabil de execuție paralelă. Această natură paralelă ar putea da computerelor ADN o creștere a performanțelor exponențială față de computerele seriale moderne.

Există, din păcate, probleme cu calculatoarele ADN, una fiind faptul că accelerarea exponențială necesită și o creștere exponențială a volumului de material necesar. Astfel, în practică, computerele ADN ar avea limite asupra performanței lor. De asemenea, nu este foarte ușor să construiești unul.

chapter 2 probleme de actualitate și tendințe

2.1 probleme de actualitate

Care sunt noile amenințări pe frontul securității cibernetice? O analiză a acestor amenințări precum și a tendințelor înacest domeniu este prezentată în articolul Cybersecurity Threats and Trends for 2020. [AME-TEN].

Prezentăm mai jos elementele principale ale acestui articol.

2.2 amenințări de actualitate

Phishingul devine mai sofisticat - Atacurile de phishing, în care mesajele digitale direcționate cu atenție sunt transmise pentru a păcăli oamenii să facă clic pe un link care poate apoi să instaleze malware sau să expună date sensibile, devin tot mai sofisticate.

Acum, când angajații celor mai multe organizații sunt mai conștienți de pericolele legate de phishing-ul prin e-mail sau de a da clic pe linkuri cu aspect suspect, hackerii avansează - de exemplu, folosind învățarea automată pentru a crea și distribui mai rapid mesaje false convingătoare în speranța că destinatarii vor compromite în mod involuntar rețelele și sistemele organizației lor. Astfel de atacuri permit hackerilor să fure datele de conectare ale utilizatorilor, acreditările cardului de credit și alte tipuri de informații financiare personale, precum și să aibă acces la baze de date private.

Strategiile Ransomware evoluează - Se crede că atacurile Ransomware costă victimele miliarde de dolari în fiecare an, deoarece hackerii implementează tehnologii care le permit să răpească literalmente bazele de date ale unei persoane sau ale unei organizații și să dețină toate informațiile pentru răscumpărare. Creșterea criptomonedelor precum Bitcoin este creditată cu contribuția la alimentarea atacurilor de ransomware, permițând ca cererile de răscumpărare să fie plătite în mod anonim.

Pe măsură ce companiile continuă să se concentreze pe construirea unor sisteme de apărare mai puternice pentru a proteja împotriva încălcărilor ransomware-ului, unii experți consideră că hackerii vor viza din ce în ce mai multe victime ale ransomware-urilor potențial profitabile, cum ar fi persoanele cu valoare netă mare.

Cryptojacking - Mișcarea criptomonedelor afectează, de asemenea, securitatea cibernetică în alte moduri. De exemplu, criptojacking-ul este o tendință care implică infractorii cibernetici care deturnă computerele de la terți sau de la locul de muncă pentru a „mina” pentru criptomonedă. Deoarece exploatarea criptomonedelor (cum ar fi Bitcoin, de exemplu) necesită o cantitate imensă de putere de procesare a computerului, hackerii pot câștiga bani prin ascunderea secretă a sistemelor altcuiva. Pentru companii, sistemele criptocupite pot provoca probleme grave de performanță și timp de funcționare costisitor, deoarece IT funcționează pentru a depista și rezolva problema.

Atacuri cibernetice asupra infrastructurii - Aceeași tehnologie care ne-a permis modernizarea și computerizarea infrastructurii critice aduce, de asemenea, risc. Amenințarea continuă a hacks-urilor care vizează rețelele electrice, sistemele de transport, instalațiile de tratare a apei etc. reprezintă o vulnerabilitate majoră în viitor. Potrivit unui raport recent publicat în The New York Times, chiar și sistemele militare de mai multe miliarde de dolari ale Americii sunt expuse riscului de a juca în mod high-tech.

Atacuri sponsorizate de stat - Dincolo de hackerii care doresc să obțină profit prin furtul de date individuale și corporative, state naționale întregi își folosesc acum abilitățile cibernetice pentru a se infiltra în alte guverne și a efectua atacuri asupra infrastructurii critice. Criminalitatea informatică astăzi reprezintă o

amenințare majoră nu numai pentru sectorul privat și pentru persoane fizice, ci pentru guvern și națiunea în ansamblu. Pe măsură ce trecem în 2020, atacurile sponsorizate de stat sunt de așteptat să crească, atacurile asupra infrastructurii critice fiind de o preocupare deosebită.

Multe astfel de atacuri vizează sisteme și infrastructuri administrate de guvern, dar și organizațiile din sectorul privat sunt expuse riscului. Potrivit unui raport al Thomson Reuters Labs: „Atacurile cibernetice sponsorizate de stat reprezintă un risc semnificativ emergent pentru întreprinderea privată, care va provoca din ce în ce mai mult acele sectoare ale lumii afacerilor care oferă obiective convenabile pentru soluționarea nemulțumirilor geopolitice.”

Atacuri asupra IoT - Internetul obiectelor devine din ce în ce mai omniprezent (conform Statista.com, numărul dispozitivelor conectate la IoT este de așteptat să ajungă la 75 miliarde până în 2025). Include laptopuri și tablete, desigur, dar și routere, camere web, electrocasnice, ceasuri inteligente, dispozitive medicale, echipamente de producție, automobile și chiar sisteme de securitate la domiciliu.

Dispozitivele conectate sunt la îndemână pentru consumatori, iar multe companii le folosesc acum pentru a economisi bani prin colectarea unor cantități imense de date inteligente și eficientizarea proceselor de afaceri. Cu toate acestea, mai multe dispozitive conectate înseamnă un risc mai mare, făcând rețelele IoT mai vulnerabile la invaziile și infecțiile cibernetice. Odată controlate de hackeri, dispozitivele IoT pot fi utilizate pentru a crea ravagii, supraîncărca rețele sau bloca echipamente esențiale pentru câștiguri financiare. [AME-TEN]

2.3 tendințe pentru 2021

În această secțiune trecem în revistă cîteva articole care prezintă principalele tendințe în domeniul securității cibernetice pentru anu 2021.

2.3.1 cinci tendințe cheie în securitatea cibernetică pentru 2021

Acest articol poate fi consultat la link-ul https://www.forbes.com/sites/forbestechcouncil/2021/12/30/five-key-cybersecurity-trends-for-2021/?sh=34ab88295035 .

Prezentăm în continuare principalele idei ale acestui articol.

1. Securizarea muncii online

Trecerea lentă către munca la distanță s-a accelerat enorm în 2020, odată cu închiderea birourilor, extinzând masiv suprafața potențială de atac a majorității companiilor. Inevitabil, utilizarea rețelelor private virtuale (VPN) și a protocoalelor desktop la distanță a crescut, ducând la o creștere îngrijorătoare de 127% a punctelor finale RDP expuse, potrivit Agenției de securitate cibernetică și securitate a infrastructurii a Departamentului pentru Securitate Internă.

În lupta de a menține continuitatea afacerii, securitatea a luat locul secund. Acest lucru trebuie abordat de urgență în anul următor. În 2021, companiile își vor evalua sistemele, vor analiza ceea ce este necesar pentru a le scala în siguranță și va remodela infrastructura pentru a sprijini forța de muncă la distanță. Politicile trebuie revizuite și modernizate. Conducerea trebuie să trimită mesaje clare și să mențină comunicări regulate.

Așteptați o creștere a VPN-ului de întreprindere, virtualizarea desktop-ului, securitatea punctelor finale, detectarea și răspunsul punctelor finale, autentificarea cu mai mulți factori (MFA) și soluțiile de brokeri de securitate cu acces în cloud.

2. Detectare și răspuns prin intermediul serviciilor specializate

Cu cît o breșă de securitate este depistată mai repede și măsurile adecvate sunt luate, cu atât mai ieftin și mai ușor poate fi rezolvată problema.

Întreruperile pot fi reduse la minimum dacă acționați rapid, dar acest lucru necesită un serviciu bun de detectare și răspuns gestionat (Managed Detection and Response)(MDR). Până în 2025, jumătate din toate organizațiile vor folosi serviciile MDR, potrivit Gartner. Pe măsură ce mai multe companii adoptă servicii MDR, este important să vă asigurați că oferă monitorizare cuprinzătoare 24 de ore din 24, alerte în timp real și răspunsuri rapide.

Serviciile MDR ar trebui să ofere expertiza pe care companiile trebuie să o remedieze, oferind sfaturi practice alături de alerte, astfel încât să poată fi abordate cauzele fundamentale ale încălcărilor. De asemenea, acestea ar trebui să ofere asistență 24/7 și să fie la curent cu cele mai recente cerințe de conformitate și reglementări pentru industriile relevante. Furnizorii MDR care oferă o abordare adaptată care se pot adapta și adapta la diferite toleranțe de risc și bugete vor fi la mare căutare în 2021.

3. Modelul de securitate de încredere zero

Odată cu creșterea rapidă a forței de muncă la distanță, mai multe dispozitive decât oricând încearcă să acceseze rețelele. Încrederea este o problemă majoră, deoarece atacatorii vor găsi adesea o cale de a intra și apoi vor intra mai adânc în rețele, mutându-se lateral. Cu un model de încredere zero, întregul proces este simplificat, deoarece alegeți să nu aveți încredere și să verificați întotdeauna conexiunile. Prin asumarea ostilității, este posibil să preveniți mișcarea laterală, oprind un atacator care a ocolit paravanul, de exemplu, să se deplaseze mai departe.

Frumusețea încrederii zero este că simplifică lucrurile pentru echipele dvs. IT, deoarece acestea au mai puține lucruri de administrat. Utilizatorii sunt limitați să acceseze doar aplicațiile și sistemele de care au nevoie pentru a-și îndeplini sarcinile, conectându-se o singură dată pentru a accesa resursele disponibile printr-un director activ. Utilizatorii sunt autentificați înainte de a obține acces la rețeaua dvs., un proces consolidat cu MFA sau software de gestionare a identității și accesului.

4. Educație și instruire pentru conștientizarea securității

Cea mai bună apărare din lume poate fi rapid inutilizată prin erori umane sau sabotaje deliberate. Indiferent dacă sunt incompetenți sau rău intenționați, persoanele din interior pot permite atacatorilor să vă ocolească securitatea, iar acestea o fac frecvent. În ultimul an, 30% din încălcări au implicat actori interni, conform raportului investigațiilor privind încălcarea datelor din 2020 de la Verizon.

Angajații trebuie să fie educați cu privire la modul de recunoaștere a încercărilor complicate de phishing și de a practica o bună igienă de securitate. Instruirea ar trebui să fie continuă, iar organizațiile trebuie să ia măsuri pentru a-i măsura eficacitatea. Asigurați-vă că directorii sunt la bord cu programe de formare, că resursele necesare sunt alocate și că programele sunt conforme cu reglementările relevante. Scopul este de a transforma angajații de la a fi cea mai slabă legătură a voastră la a fi cel mai puternic atu al companiei.

5. Construirea unei infrastructuri cloud sigure

Câștigul de cost și eficiență pe care îl promite cloud computing, împreună cu scalabilitatea sa ușoară, i-au asigurat ascendența în lumea afacerilor. Cu toate acestea, organizațiile nu își pot permite să facă presupuneri cu privire la standardele de securitate ale partenerilor cloud. Serviciile cloud sunt o țintă principală pentru atacatori. Companiile trebuie să facă bilanț și să alcătuiască o imagine clară a modului în care serviciile lor cloud se potrivesc împreună și unde se află datele.

Securizarea infrastructurii cloud a companiei dvs. necesită o arhitectură holistică de securitate cibernetică, care este informată de scenarii de amenințare, întrucât o abordare parțială a securizării datelor și aplicațiilor va eșua cu siguranță. Controalele ar trebui să ia în considerare totul, de la punctele finale până la gestionarea accesului la reglementări. Construiți controale de securitate în infrastructură pe măsură ce se extinde. Testați pentru noi vulnerabilități și evaluați riscurile noi pe măsură ce apar.

MDR este un răspuns perfect. Detectarea și răspunsul gestionat este un proces prin care datele sunt colectate de la fiecare punct final, serviciu cloud și rețea afiliată companiei pentru a construi o viziune holistică a organizației dvs. suficient pentru a permite un mijloc prin care să poată fi efectuată analiza pentru a elimina anomaliile, vulnerabilitățile sau orice amenințare subiacentă.

Deși 2020 a fost o provocare, a accelerat multe tendințe interesante în securitatea cibernetică și există o oportunitate de a consolida și asigura aceste schimbări în anul următor.

chapter 3 puncte de reper istorice

Acest capitol se bazează pe un articol publicat în blogul firmei de securitate Avast. [AVA-HIS]

3.1 Anii ‘40 – timpul de dinaintea atacurilor

Aproape două decenii după crearea primului computer digital din lume în 1943, efectuarea atacurilor cibernetice a fost dificilă. Accesul la mașinile electronice gigantice era limitat la un număr mic de oameni și nu erau conectați la rețea - doar câțiva oameni știau cum să le lucreze, astfel încât amenințarea era aproape inexistentă.

Interesant este faptul că teoria care stă la baza virușilor informatici a fost făcută publică pentru prima dată în 1949, când pionierul computerului, John von Neumann, a speculat că programele de computer ar putea fi reproduse.

3.2 Anii ‘50 – ciudații telefoniei

Rădăcinile tehnologice și subculturale ale hacking-ului sunt legate atât de mult timp de telefoanele timpurii, cât și de computere.

La sfârșitul anilor 1950, a apărut „phreaking-ul telefonic”. Termenul surprinde mai multe metode pe care „le folosește” - persoanele cu un interes deosebit în funcționarea telefoanelor - folosite pentru a deturna protocoalele care permiteau inginerilor din domeniul telecomunicațiilor să lucreze în rețea de la distanță pentru a efectua apeluri gratuite și pentru a evita taxele pe distanțe lungi. Din păcate pentru companiile de telefonie, nu a existat nicio modalitate de a opri fracțiunile, deși practica s-a stins în anii 1980.

Frații deveniseră o comunitate, chiar emitând buletine informative, și includeau pionieri tehnologici precum fondatorii Apple, Steve Wozniak și Steve Jobs. Matrița a fost setată pentru tehnologia digitală.

3.3 Anii ‘60 – liniște și pace pe frontul de Vest

Prima referință vreodată la hacking-ul rău intenționat a fost în ziarul studențesc al Institutului de Tehnologie din Massachusetts.

Chiar și la mijlocul anilor 1960, majoritatea computerelor erau sisteme imense, închise în camere securizate, controlate de temperatură. Aceste mașini erau foarte costisitoare, astfel încât accesul - chiar și la programatori - a rămas limitat.

Cu toate acestea, au existat incursiuni timpurii în hacking de către unii dintre cei cu acces, adesea studenți. În această etapă, atacurile nu au avut beneficii comerciale sau geopolitice. Cei mai mulți hackeri au fost curioși de răufăcători sau cei care au căutat să îmbunătățească sistemele existente făcându-i să funcționeze mai rapid sau mai eficient.

În 1967, IBM a invitat copiii de la școală să-și încerce noul computer. După ce au explorat părțile accesibile ale sistemului, studenții au lucrat pentru a cerceta mai adânc, învățând limba sistemului și obținând acces la alte părți ale sistemului.

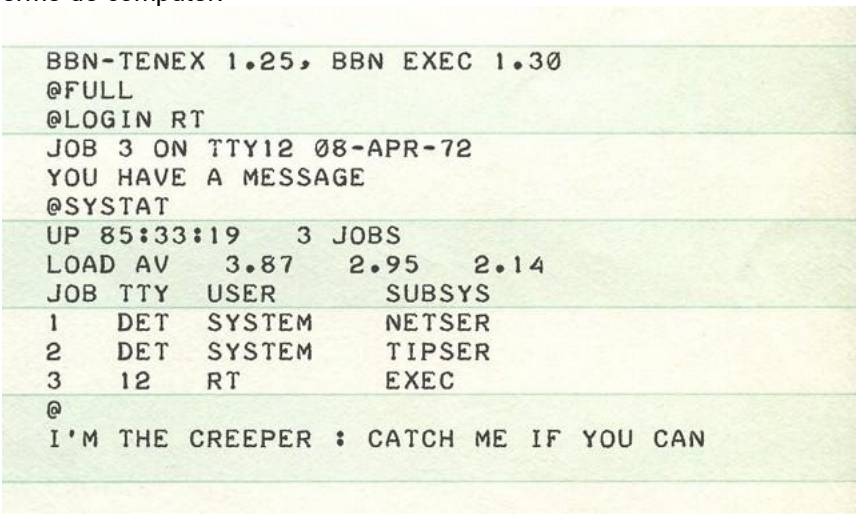
Aceasta a fost o lecție valoroasă pentru companie și și-au recunoscut recunoștința față de „un număr de elevi de liceu pentru constrângerea lor de a bombarda sistemul”, ceea ce a dus la dezvoltarea unor măsuri defensive - și, eventual, la o mentalitate defensivă care s-ar dovedi esențială pentru dezvoltatori. de atunci înainte. Hackingul etic este practicat și astăzi.

Pe măsură ce calculatoarele au început să reducă dimensiunea și costurile, multe companii mari au investit în tehnologii pentru stocarea și gestionarea datelor și sistemelor. Stocarea lor sub cheie și blocare a devenit redundantă, deoarece mai mulți oameni aveau nevoie de acces la ele și au început să fie folosite parolele.

3.4 Anii ‘70 – se naște securitatea cibernetică

Securitatea cibernetică propriu-zisă a început în 1972 cu un proiect de cercetare pe ARPANET (Rețeaua Agenției pentru Proiecte de Cercetare Avansată), un precursor al internetului.

Cercetătorul Bob Thomas a creat un program de computer numit Creeper care se putea deplasa prin rețeaua ARPANET, lăsând o urmă de pesmet oriunde s-a dus. Scria: „Eu sunt târâtorul, prinde-mă dacă poți”. Ray Tomlinson - inventatorul e-mailului - a scris programul Reaper, care a urmărit și șters Creeper. Reaper nu a fost doar primul exemplu de software antivirus, ci a fost și primul program de auto-replicare, făcându-l primul vierme de computer.



Provocarea vulnerabilităților din aceste tehnologii emergente a devenit mai importantă pe măsură ce mai multe organizații începeau să folosească telefonul pentru a crea rețele la distanță. Fiecare bucată de hardware conectat prezenta un nou „punct de intrare” și trebuia protejat.

Pe măsură ce dependența de computere a crescut și creșterea rețelelor, a devenit clar pentru guverne că securitatea era esențială, iar accesul neautorizat la date și sisteme ar putea fi catastrofal. 1972-1974 a fost martorul unei creșteri semnificative a discuțiilor despre securitatea computerelor, în special de către academicieni în ziare.

Crearea timpurie a securității computerului a fost întreprinsă de ESD și ARPA cu Forțele Aeriene ale SUA și alte organizații care au lucrat în cooperare pentru a dezvolta un design pentru un nucleu de securitate pentru sistemul de calcul Honeywell Multics (HIS level 68). UCLA și Stanford Research Institute au lucrat la proiecte similare.

Proiectul ARPA Protection Analysis a explorat securitatea sistemului de operare; identificarea, acolo unde este posibil, a tehnicilor automatizabile pentru detectarea vulnerabilităților în software.

Până la mijlocul anilor 1970, conceptul de securitate cibernetică se maturiza. În 1976, structurile sistemului de operare pentru a sprijini securitatea și software-urile fiabile au declarat:

„Securitatea a devenit un obiectiv important și provocator în proiectarea sistemelor informatice.”

În 1979, Kevin Mitnick, în vârstă de 16 ani, a piratat în The Ark - computerul de la Digital Equipment Corporation folosit pentru dezvoltarea sistemelor de operare - și a făcut copii ale software-ului. El a fost arestat și închis pentru ceea ce ar fi primul din mai multe atacuri cibernetice pe care le-a condus în următoarele câteva decenii. Astăzi conduce Mitnick Security Consulting.

3.5 Anii ‘80 – de la ARPANET la Internet

Anii 1980 au adus o creștere a atacurilor de profil, inclusiv a celor de la National CSS, AT&T și Laboratorul Național Los Alamos. Filmul Jocuri de război, în care un program de computer necinstit preia sistemele de rachete nucleare sub masca unui joc, a fost lansat în 1983. Acesta a fost același an în care au fost folosiți pentru prima oară termenii Cal troian și Virus computer.

În timpul Războiului Rece, amenințarea spionajului cibernetic a evoluat. În 1985, Departamentul Apărării din SUA a publicat Criteriile de evaluare a sistemelor informatice de încredere (cunoscut și sub numele de „Cartea portocalie”), care oferă îndrumări cu privire la:

- Evaluarea gradului de încredere care poate fi plasat în software-ul care prelucrează informații clasificate sau alte informații sensibile
- Ce măsuri de securitate au avut nevoie producătorii pentru a se încadra în produsele lor comerciale.

În ciuda acestui fapt, în 1986, hackerul german Marcus Hess a folosit o poartă de internet în Berkeley, CA, pentru a intra pe ARPANET. El a spart 400 de computere militare, inclusiv mainframe la Pentagon, intenționând să vândă informații KGB-ului.

Securitatea a început să fie luată mai în serios. Utilizatorii experimentați au învățat rapid să monitorizeze dimensiunea fișierului command.com, după ce au observat că o creștere a dimensiunii a fost primul semn de potențială infecție. Măsurile de securitate cibernetică au încorporat această gândire și o reducere bruscă a memoriei libere de operare rămâne un semn de atac până în prezent.

3.6 1987 – nașterea securității cibernetice

1987 a fost anul nașterii antivirusului comercial, deși există pretenții concurente pentru inovatorul primului produs antivirus.

- Andreas Lüning și Kai Figge au lansat primul lor produs antivirus pentru Atari ST - care a văzut și lansarea Ultimate Virus Killer (UVK)
- Trei cehoslovaci au creat prima versiune a antivirusului NOD
- În SUA, John McAfee a fondat McAfee (pe atunci parte a Intel Security) și a lansat VirusScan.

Tot în 1987:

Una dintre cele mai vechi îndepărtări de virusuri „în sălbăticie” documentate a fost efectuată de germanul Bernd Fix când a neutralizat infamul virus Viena - un exemplu timpuriu de malware care răspândea și corupea fișierele.

Virusul Cascade criptat, care a infectat fișierele .COM, a apărut pentru prima dată. Un an mai târziu, Cascade a provocat un incident grav în biroul belgian IBM și a servit drept impuls pentru dezvoltarea produsului antivirus IBM. Înainte de aceasta, orice soluție antivirus dezvoltată la IBM a fost destinată numai pentru uz intern.

Până în 1988, multe companii antivirus au fost înființate în întreaga lume - inclusiv Avast, care a fost fondată de Eduard Kučera și Pavel Baudiš la Praga, Republica Cehă. Astăzi, Avast are o echipă de peste 1.700 la nivel mondial și oprește în jur de 1,5 miliarde de atacuri în fiecare lună.

Programul antivirus timpuriu consta în scanere simple care efectuează căutări de context pentru a detecta secvențe unice de coduri de virus. Multe dintre aceste scanere includeau și „imunizatoare” care modificau programele pentru a face virușii să creadă că computerul este deja infectat și să nu-i atace. Pe măsură ce numărul virusurilor a crescut în sute, imunizatorii au devenit rapid ineficienți.

Companiile antivirus deveneau de asemenea clare că puteau reacționa doar la atacurile existente, iar lipsa unei rețele universale și omniprezente (internetul) făcea ca actualizările să fie greu de implementat.

Pe măsură ce lumea a început încet să ia cunoștință de virușii computerizați, 1988 a asistat și la primul forum electronic dedicat securității antivirusului - Virus-L - în rețeaua Usenet. Deceniul a cunoscut, de asemenea, nașterea presei antivirus: Buletinul Virus, sponsorizat de Sophos, cu sediul în Marea Britanie și Virus Fax International al Dr. Solomon.

Deceniul s-a încheiat cu mai multe adăugiri pe piața securității cibernetice, inclusiv F-Prot, ThunderBYTE și Norman Virus Control. În 1989, IBM și-a comercializat în cele din urmă proiectul antivirus intern, iar IBM Virscan pentru MS-DOS a fost pus în vânzare cu 35 de dolari.

3.7 Anii '90 – lumea trece online

1990 a fost un an destul de liniștit:

- Au fost creați primii viruși polimorfi (cod care muta în timp ce păstrează intact algoritmul original pentru a evita detectarea)
- Revista britanică de calculatoare PC Today a lansat o ediție cu un disc gratuit care conținea 'accidental' virusul DiskKiller, infectând zeci de mii de computere
- A fost înființat EICAR (Institutul European pentru Cercetarea Antivirusului pe Calculatoare)

Produsele antivirusul timpurii au fost bazate doar pe semnături, comparând binare dintr-un sistem cu o bază de date cu „semnături" ale virusului. Aceasta a însemnat că antivirusul timpuriu a produs multe pozitive false și a folosit o mulțime de putere de calcul - ceea ce a frustrat utilizatorii pe măsură ce productivitatea a încetinit.

Pe măsură ce au apărut mai multe scanere antivirus pe piață, infractorii cibernetici au răspuns și în 1992 a apărut primul program antivirus.

Până în 1996, mulți viruși au folosit noi tehnici și metode inovatoare, inclusiv capacitatea stealth, polimorfism și „macro viruși", reprezentând un nou set de provocări pentru furnizorii de antivirus care au trebuit să dezvolte noi capacități de detectare și eliminare.

Noile cifre de virus și malware au explodat în anii 1990, de la zeci de mii la începutul deceniului, crescând la 5 milioane în fiecare an până în 2007. Până la mijlocul anilor '90, era clar că securitatea cibernetică trebuia produsă în masă pentru a proteja publicul. Un cercetător NASA a dezvoltat primul program de firewall, modelându-l pe structurile fizice care împiedică răspândirea incendiilor reale în clădiri.

Sfârșitul anilor 1990 a fost, de asemenea, marcat de conflicte și fricțiuni între dezvoltatorii de antivirus:

- McAfee l-a acuzat pe doctorul Solomon că a înșelat, astfel încât testarea discurilor neinfectate a arătat rezultate bune la viteză, iar testele de scanare a colecțiilor de virusuri au arătat rezultate bune de detectare. Ca răspuns, doctorul Solomon le-a intentat un proces.
- Dezvoltatorul taiwanez Trend Micro a acuzat McAfee și Symantec că și-au încălcat brevetul privind tehnologia de verificare a scanării virușilor prin internet și poștă electronică. Symantec l-a acuzat apoi pe McAfee că folosește codul de la Norton AntiVirus al Symantec.

Detectarea euristică a apărut, de asemenea, ca o nouă metodă de abordare a numărului mare de variante de virus. Scannerele antivirus au început să utilizeze semnături generice - conținând adesea cod necontiguu și folosind caractere wildcard - pentru a detecta viruși chiar dacă amenințarea fusese „ascunsă" în codul fără sens.

E-mail: o binecuvântare și un blestem

Spre sfârșitul anilor 1990, e-mailul proliferează și, deși promitea să revoluționeze comunicarea, a deschis și un nou punct de intrare pentru viruși.

În 1999, virusul Melissa a fost dezlănțuit. Acesta a intrat pe computerul utilizatorului printr-un document Word și apoi a trimis prin e-mail copii ale sale la primele 50 de adrese de e-mail din Microsoft Outlook. Rămâne unul dintre cei mai răspândiți viruși și daunele au costat aproximativ 80 de milioane de dolari.

3.8 Deceniul 2000 – amenințările se multiplică și se diversifică

Cu internetul disponibil în mai multe case și birouri din întreaga lume, infractorii cibernetici au avut mai multe dispozitive și vulnerabilități software de exploatat ca niciodată. Și, pe măsură ce din ce în ce mai multe date erau păstrate digital, erau mai multe de jefuit.

În 2001, a apărut o nouă tehnică de infecție: utilizatorii nu mai aveau nevoie să descarce fișiere - vizitarea unui site web infectat era suficient, deoarece actorii răi înlocuiau paginile curate cu altele infectate sau „ascundeau" malware pe paginile web legitime. Serviciile de mesagerie instant au început, de asemenea, să fie atacate și au sosit și viermi concepuți să se propage prin canalul IRC (Internet Chat Relay).

Dezvoltarea atacurilor de zi zero, care utilizează „găuri" în măsurile de securitate pentru software și aplicații noi, a însemnat că antivirusul a devenit mai puțin eficient - nu puteți verifica codul împotriva semnăturilor de atac existente decât dacă virusul există deja în baza de date . Revista Computer nu a descoperit că ratele de detecție pentru amenințările de zi zero au scăzut de la 40-50% în 2006 la doar 20-30% în 2007.

Pe măsură ce organizațiile criminale au început să finanțeze puternic atacurile cibernetice profesionale, băieții buni au fost fierbinți pe urmele lor:

- 2000: primul motor antivirus open-source OpenAntivirus Project este disponibil
- 2001: Se lansează ClamAV, primul motor antivirus open source care a fost comercializat
- 2001: Avast lansează un software antivirus gratuit, oferind o soluție de securitate completă pentru masă. Inițiativa a crescut baza de utilizatori Avast la peste 20 de milioane în cinci ani.

O provocare cheie a antivirusului este aceea că poate încetini deseori performanța unui computer. O soluție la aceasta a fost mutarea software-ului de pe computer și în cloud. În 2007, Panda Security a combinat tehnologia cloud cu inteligența asupra amenințărilor în produsul lor antivirus - primul în industrie. McAfee Labs a urmat exemplul în 2008, adăugând funcționalitate anti-malware bazată pe cloud la VirusScan. Anul următor a fost creată Organizația pentru standarde de testare anti-malware (AMTSO) și a început să lucreze la scurt timp după o metodă de testare a produselor cloud.

O altă inovație din acest deceniu a fost securitatea sistemului de operare - securitatea cibernetică integrată în sistemul de operare, oferind un strat suplimentar de protecție. Aceasta include adesea efectuarea actualizărilor regulate de patch-uri ale sistemului de operare, instalarea de motoare și software antivirus actualizate, firewall-uri și conturi securizate cu gestionarea utilizatorilor.

Odată cu proliferarea smartphone-urilor, a fost dezvoltat și antivirusul pentru Android și Windows Mobile.

3.9 Deceniul 2010 – noua generație

Anii 2010 au văzut numeroase încălcări și atacuri de profil înalt care au început să aibă impact asupra securității naționale a țărilor și au costat întreprinderile milioane.

- 2012: hackerul saudit 0XOMAR publică online detaliile a peste 400.000 de carduri de credit
- 2013: Fostul angajat al CIA pentru guvernul SUA Edward Snowden a copiat și a scos informații clasificate de la Agenția Națională de Securitate (NSA)
- 2013-2014: hackerii rău intenționați au intrat în Yahoo, compromitând conturile și informațiile personale ale celor 3 miliarde de utilizatori. Yahoo a fost ulterior amendat cu 35 de milioane de dolari pentru că nu a dezvăluit știrea
- 2017: Ransomware-ul WannaCry infectează 230.000 de computere într-o singură zi
- 2019: atacuri multiple DDoS au forțat bursa din Noua Zeelandă să se închidă temporar

Conectarea crescândă și digitalizarea continuă a multor aspecte ale vieții au continuat să ofere criminalilor cibernetici noi oportunități de exploatare. Securitatea cibernetică adaptată în mod specific nevoilor afacerilor a devenit mai proeminentă și, în 2011, Avast a lansat primul său produs de afaceri.

chapter 3

Pe măsură ce securitatea cibernetică s-a dezvoltat pentru a aborda gama extinsă de tipuri de atac, infractorii au răspuns cu propriile inovații: atacuri multi-vectoriale și inginerie socială. Atacatorii deveneau mai inteligenți, iar antivirusul a fost nevoit să treacă de la metodele de detectare bazate pe semnături la inovațiile „generației următoare”.

Securitatea cibernetică de ultimă generație folosește abordări diferite pentru a crește detectarea amenințărilor noi și fără precedent, reducând totodată numărul de falsi pozitivi. De obicei implică:

- Autentificare multi-factor (MFA)
- Network Behavioral Analysis (NBA) - identificarea fișierelor rău intenționate pe baza abaterilor sau anomaliilor comportamentale
- Inteligența amenințării și automatizarea actualizărilor
- Protecție în timp real - denumită și scanare la acces, protecție de fundal, scut de rezidenți și protecție automată
- Sandboxing - crearea unui mediu de testare izolat în care puteți executa un fișier sau o adresă URL suspectă
- Criminalistică - reluarea atacurilor pentru a ajuta echipele de securitate să atenueze mai bine viitoarele încălcări
- Back-up și oglindire
- Firewall-uri pentru aplicații web (WAF) - protecție împotriva falsificării între site-uri, cross-site-scripting (XSS), includerea fișierelor și injecția SQL.

chapter 4 implementări ale politicilor de securitate

4.1 introducere

Securitatea informațiilor a ajuns să joace un rol extrem de vital într-o mișcare rapidă în zilele noastre, dar invariabil tehnic fragil în mediul de afaceri. În consecință, sunt necesare comunicații securizate pentru ca atât companiile, cât și clienții să beneficieze de progresele cu care ne împuternicește Internetul.

Importanța acestui fapt trebuie evidențiată în mod clar, astfel încât să fie implementate măsuri adecvate, îmbunătățind nu numai procedurile și tranzacțiile zilnice ale companiei, ci și pentru a se asigura că măsurile de securitate atât de necesare sunt implementate cu un nivel acceptabil de competență de securitate.

Este neplăcut să vedem că posibilitatea de a avea datele companiei dvs. expuse unui atacator rău intenționat este în continuă creștere în zilele noastre din cauza numărului ridicat de personal „analfabet de securitate” ce are acces și la informații de afaceri sensibile, și uneori chiar secrete. Imaginați-vă doar implicațiile de securitate ale unei persoane care se ocupă de datele sensibile ale companiei, care navighează pe internet în mod nesigur prin rețeaua companiei, primind e-mailuri suspecte care conțin diverse atașamente distructive și să nu uităm de amenințările semnificative pe care le reprezintă utilizarea constantă a oricărei mesaje instant (IM) sau a aplicațiilor de chat.

4.2 de ce să aveți o politică de securitate?

Întrucât construirea unei bune politici de securitate oferă fundamentele pentru implementarea cu succes a proiectelor legate de securitate în viitor, aceasta este fără îndoială prima măsură care trebuie luată pentru a reduce riscul utilizării inacceptabile a oricărei resurse informaționale a companiei.

Primul pas către îmbunătățirea securității unei companii este introducerea unei politici de securitate precise, dar aplicabile,prin informarea personalului cu privire la diferitele aspecte ale responsabilităților lor, utilizarea generală a resurselor companiei și explicarea modului în care trebuie tratate informațiile sensibile. De asemenea, politica va descrie în detaliu semnificația utilizării acceptabile, precum și enumerarea activităților interzise.

Dezvoltarea (și implementarea corectă) a unei politici de securitate este extrem de benefică, deoarece nu numai că va transforma tot personalul în participanți la efortul companiei de a-și asigura comunicațiile, ci va contribui și la reducerea riscului unei potențiale încălcări a securității prin greșeli de factor uman. Acestea sunt de obicei probleme precum dezvăluirea informațiilor către surse necunoscute (sau surse neautorizate), utilizarea nesigură sau necorespunzătoare a internetului și multe alte activități periculoase.

În plus, procesul de construire a unei politici de securitate va contribui, de asemenea, la definirea activelor critice ale unei companii, a modurilor în care acestea trebuie protejate și vor servi și ca document centralizat, în ceea ce privește protejarea activelor de securitate a informațiilor.

4.3 ce este o politică de seuritate?

Politica de securitate este în esență un plan, ce prezintă care sunt activele critice ale companiei și cum trebuie (și pot) să fie protejate. Scopul său principal este de a oferi personalului o scurtă prezentare generală a „utilizării acceptabile” a oricăreia dintre activele informaționale, precum și de a explica ce este considerat permis și ce nu, angajându-i astfel în asigurarea sistemelor critice ale companiei.

Documentul acționează ca o sursă de informații ce „trebuie citită” de toată lumea care folosește în orice mod sisteme și resurse definite ca potențiale ținte. O politică de securitate bună și bine dezvoltată ar trebui să abordeze unele dintre următoarele elemente:

- Cum trebuie tratate informațiile sensibile
- Cum să vă întrețineți corect ID-urile și parola (parolele), precum și orice alte date contabile

- Cum să răspundeți la un potențial incident de securitate, tentativă de intruziune etc.
- Cum se utilizează stațiile de lucru și conectivitatea la Internet într-un mod sigur
- Cum se utilizează corect sistemul de e-mail corporativ

Practic, principalele motive care stau la baza creării unei politici de securitate este de a stabili bazele de securitate a informațiilor unei companii, de a explica personalului modul în care aceștia sunt responsabili pentru protecția resurselor informaționale și de a evidenția importanța asigurării comunicațiilor securizate în timp ce desfășoară afaceri online.

4.4 noțiuni de bază

Scopul acestei secțiuni este de a vă oferi posibile strategii și câteva recomandări pentru procesul de creare a unei politici de securitate și de a vă oferi un plan de abordare de bază în timp ce construiți cadrul.

Procedura de pornire pentru construirea unei politici de securitate necesită o explorare completă a rețelei companiei, precum și a oricărui alt activ critic, astfel încât măsurile adecvate să poată fi puse în aplicare în mod eficient. Totul începe cu identificarea resurselor informaționale critice ale companiei, un subiect care este discutat în profunzime în următoarea secțiune a lucrării.

4.5 analiza riscurilor (identificarea activelor)

Ca în orice altă procedură sensibilă, analiza riscurilor și gestionarea acestora, joacă un rol esențial în funcționalitatea corectă a procesului. Analiza riscurilor este procesul de identificare a activelor informaționale critice ale companiei și a utilizării și funcționalității acestora - un proces important (cheie) care trebuie luat în serios. În esență, este chiar procesul de definire exact a ceea ce încercați să protejați, de la cine încercați să îl protejați si cel mai important, cum îl veți proteja.

Pentru a putea efectua o analiză de risc de succes, trebuie să vă familiarizați cu modul în care operează o companie; dacă este cazul, modalitățile de lucru și anumite proceduri de afaceri, care resurse informaționale sunt mai importante decât altele (prioritizarea) și identificarea dispozitivelor / procedurilor care ar putea duce la o posibilă problemă de securitate.

Enumerați tot ceea ce este esențial pentru funcționalitatea corectă a proceselor de afaceri; precum aplicații și sisteme cheie, servere de aplicații, servere web, servere de baze de date, diverse planuri de afaceri, proiecte în dezvoltare etc.

O abordare de bază poate fi:

- Identificați ceea ce încercați să protejați
- Uită-te la cel pe care încerci să-l protejezi
- Definiți care sunt riscurile potențiale pentru oricare dintre activele dvs. informaționale
- Luați în considerare monitorizarea continuă a procesului pentru a fi la curent cu cele mai recente puncte slabe de securitate
- O posibilă listă de categorii la care să te uiți ar putea fi:
    - Hardware: Toate serverele, stațiile de lucru, computerele personale, laptopurile, suport amovibil (CD, dischete, benzi etc.), linii de comunicare etc.
    - Software: Identificați riscurile unei potențiale probleme de securitate din cauza software-urilor învechite, a patch-urilor rare și a actualizărilor versiunilor noi etc. De asemenea, luați în considerare problemele potențiale cu personalul care instalează diverse aplicații de partajare a fișierelor (Kazaa, Sharereactor, E-Donkey, etc.), software IM (chat), software de divertisment sau freeware provenit din surse necunoscute și de încredere.
    - Personal: Cei care au acces la informații confidențiale, date sensibile, cei care „dețin”, administrează sau modifică în orice mod bazele de date existente

4.6 administrarea riscurilor (identificarea amenințărilor)

Pe baza cercetărilor efectuate asupra activelor informaționale ale companiei, ar trebui să puteți gestiona corect toate amenințările reprezentate de fiecare dintre resursele dvs.

Scopul acestei secțiuni este de a vă ghida prin crearea unei liste care prezintă diferite amenințări potențiale, lucru care ar trebui inclus și în politica formală de securitate. Fiecare dintre următoarele elemente vor fi discutate în detaliu mai târziu în secțiunea programului de conștientizare a securității, oferind astfel membrilor personalului o mai bună înțelegere a fiecăruia dintre subiectele tratate mai jos.

4.6.1 Securitate fizică / desktop

- Acces la sistem: cele mai bune practici pentru crearea parolelor, învechirea parolelor, lungimea minimă a parolei, caracterele care trebuie incluse în alegerea parolelor, întreținerea parolelor, sfaturi pentru protejarea (oricăror) date contabile; pericolele pentru fiecare dintre aceste probleme trebuie explicate în programul de conștientizare a securității;
- Instalarea software-ului: este interzis software-ul fără licență, dacă este permis, în ce condiții, cum este tolerată pirateria software-ului, sunt permise sau complet interzise jocurile de divertisment sau instalarea oricărui alt program provenind din surse necunoscute și de încredere;
- Suporturi amovibile (CD-uri, dischete): trebuie stabilite măsurile „Utilizare acceptabilă” (poate prin intermediul PUA - Politica de utilizare acceptabilă), trebuie explicate pericolele potențialului cod rău intenționat care intră în rețeaua companiei sau în orice alt sistem critic de asemenea;
- Criptare: explicați când, cum și cine trebuie să cripteze oricare dintre datele companiei;
- Backup-uri de sistem: trebuie explicat avantajul de a avea backup-uri; cine este responsabil și cât de des ar trebui să se facă backup pentru date;
- Întreținerea: riscurile unei potențiale încălcări ale securității fizice trebuie explicate pe scurt;
- Manipularea incidentelor: definiți ce este un eveniment suspect, cui trebuie raportat și ce măsuri suplimentare trebuie luate;

4.6.2 Amenințările de pe internet

- Navigare web: definiți ce constituie site-uri web restricționate, interzise și potențial rău intenționate, oferiți membrilor personalului sfaturi scurte și bine rezumate pentru o navigare mai sigură, informați-le în plus că utilizarea lor de Internet este strict monitorizată pentru a proteja sistemele interne ale companiei;
- Utilizarea e-mailului: definiți criteriile de „utilizare acceptabilă” ale sistemului de e-mail, ce este permis și ce nu, politica companiei privind utilizarea sistemului de poștă electronică pentru mesaje personale etc. De asemenea, explicați pe scurt potențialele amenințări reprezentate de (abuzarea) email-ului și a potențialelor probleme în ceea ce privește răspândirea codului rău intenționat;
- Software de mesagerie instant (IM) (ICQ, AIM, MSN etc.): dacă este permis sau complet interzis, oferiți-le exemple scurte despre modul în care un atacator ar putea folosi aceste programe pentru a pătrunde și a fura / corupe / modifica datele companiei;
- Descărcarea / atașamentele: descărcarea este permisă sau nu, sfaturi utile pentru descărcarea mai sigură, explicarea surselor de încredere și care nu sunt de încredere, cele mai bune practici pentru atașamentele de e-mail dacă sunt permise, discutarea potențialelor amenințări și pericole, utilizarea scanerelor de viruși etc.
- Aceste elemente vor fi ulterior acoperite în detaliu într-un program de conștientizare a securității. Personalul trebuie să înțeleagă de ce unele activități sunt interzise, ce poate avea impactul anumitor pericole asupra companiei, acțiuni pe care trebuie să le urmeze dacă și când a fost suspectată sau descoperită o potențială problemă de securitate. Prin implicarea personalului într-un program de conștientizare a securității, personalul nu doar își va extinde cunoștințele în domeniul securității informațiilor, ci asemenea, învăța cum să acționeze într-un mod sigur în timp ce utilizează oricare dintre activele informaționale ale companiei.

4.7 încălcarea politicilor de securitate

Pentru a conștientiza importanța unei politici de securitate, personalul trebuie să fie conștient și să înțeleagă pe deplin consecințele încălcării politicii, expunând astfel sistemele critice unui atacator rău intenționat sau cauzând daune neintenționate altor companii din întreaga lume. Încălcările ar trebui tratate în consecință; cei care într-un fel sau altul încalcă politica de securitate ar trebui să fie conștienți de faptul că ar putea fi supuși unei „perioade de probă", care implică și utilizarea limitată a unora dintre activele informaționale ale companiei până când pot demonstra că sunt capabili să acționează într-un mod sigur în timp ce folosești sistemele corporative. De asemenea, ar trebui să fie conștienți de faptul că în unele cazuri (dure) pot risca, de asemenea, să fie concediați sau chiar urmăriți penal.

Întrucât acest lucru poate părea exagerat pentru unii, trebuie luate măsuri adecvate în fiecare caz de încălcare, în conformitate cu termenii PUA și ai politicii, cu accent pe reiterarea elementelor de securitate și nu a pedepsei. În caz contrar, cel mai probabil va exista o penetrare reușită, fie din cauza unei erori umane, fie din neînțelegerea politicii.

4.8 implementarea politicii

Când politica de securitate este întocmită, revizuită, actualizată și convenită, va urma procesul de implementare. Acest lucru este de obicei mai greu decât crearea politicii în sine, datorită faptului că, în această etapă, trebuie să vă instruiți și să vă educați personalul să se comporte într-un mod „sigur", urmând fiecare dintre elementele esențiale indicate în politica formală de securitate.

Versiunea finală a politicii de securitate trebuie să fie pusă la dispoziția tuturor angajaților care au acces la oricare dintre activele dvs. de informații. Politica trebuie să fie ușor de obținut în orice moment, cu o copie plasată în rețeaua internă și intranet, dacă este cazul.

O implementare adecvată necesită nu numai educarea personalului cu privire la fiecare dintre elementele de bază semnalate drept critice în politica formală de securitate, ci și schimbarea rolului lor în efortul de a proteja datele critice ale companiei.

Următoarea secțiune va avea ca scop să vă ghideze prin procesul de creare a unui program de bază de conștientizare a securității, împreună cu diferite modalități inovatoare și interesante de educare a personalului dvs., utilizând linii de comunicare ușor de utilizat și informale între membrii Oficiului pentru securitatea informațiilor (ISO) și angajații dumneavoastră.

4.9 procesul de dezvoltare

Această secțiune vă va oferi diferitele strategii de construire a unui program solid de conștientizare a securității. Vom discuta diverse metode, avantajele și dezavantajele acestora și vă vom oferi, de asemenea, o mai bună înțelegere a etapelor esențiale pentru construirea programului.

La început trebuie să vă răspundeți la următoarele întrebări:

- Ce ar trebui să realizeze Programul de conștientizare a securității și cum vei atrage atenția asupra acestui lucru?
- Cine este publicul tău, cât de „educați" sunt; va fi necesar să împărțiți programul în două părți, una pentru cei care au mai multe cunoștințe despre computere și una pentru cei care nu sunt deloc interesați de computere?
- Cum veți ajunge și motiva publicul dumneavoastră.? Mai important, cum veți atrage publicul interesat de îmbunătățirea activelor informaționale ale companiei?
- Programul se va baza pe un mod formal sau informal de comunicare între dumneavoastră și membrii personalului? În ce fel o vei conduce și o vei prezenta?

4.9.1 scopul programului

În primul rând, trebuie să explicați personalului ce va încerca să realizeze programul, cum va viza îmbunătățirea operațiunilor companiei și cât de vitală este cu adevărat protecția activelor informaționale. Va trebui să explicați de ce „securitatea este responsabilitatea tuturor” și să vă asigurați că toată lumea o înțelege; explicați că, chiar dacă compania are cele mai recente îmbunătățiri tehnologice, cum ar fi firewall-uri, sisteme de detectare a intruziunilor etc., un membru al personalului needucat ar putea pune cu ușurință în pericol informațiile sensibile și ar putea face orice măsură tehnică de securitate, complet și absolut inutilă.

O altă neînțelegere obișnuită cu care vă veți confrunta cu siguranță în timpul derulării programului este că majoritatea oamenilor tind să creadă că nu este responsabilitatea lor să contribuie la îmbunătățirea securității companiei lor. În general, oamenii sunt de părere (greșită) că numai departamentul IT sau Biroul de securitate a informațiilor (ISO) pot și trebuie să aibă grijă de astfel de probleme și aceasta este ce contează, în general.

4.9.2 adresarea publicului

O problemă majoră cu care sunt sigur că vă veți confrunta este diferența dintre nivelurile de cunoștințe informatice (ale audienței dumneavoastră.), care uneori vă vor obliga să acordați o atenție suplimentară celor care nu sunt atât de interesați de computere. Pe de altă parte, puteți alege, de asemenea, să faceți diferența dintre cei care au nevoie de educație în materie de securitate și cei care nu au nevoie; ideea este de a separa personalul care are acces la oricare dintre activele informaționale ale companiei de cei care nu (și nu pot pune în pericol datele sensibile în niciun fel), deoarece acest lucru vă va economisi cu siguranță mult timp și resurse. Ar fi o abordare bună să organizați întâlniri informale cu personalul pentru a discuta la nivel personal și, de asemenea, să efectuați mai multe sondaje pentru a măsura nivelul lor de calificare; astfel veți ști unde să vă concentrați atenția.

4.9.3 măsurarea nivelului de conștientizare a securității prin sondaje

Sondajele de conștientizare a securității sunt dezvoltate cu ideea de a măsura nivelul actual de conștientizare a personalului dumneavoastră., dar vor indica, de obicei, greșelile obișnuite și neînțelegerile angajaților dvs.; ceea ce vă va ajuta cu siguranță să îmbunătățiți calitatea programului, chiar înainte de a începe. Este cu încredere recomandat să arhivați sondajele pentru a evalua eficacitatea programului pe o perioadă de timp.

S-ar putea să doriți, de asemenea, să indicați membrilor personalului că ancheta este complet anonimă, că nu este necesar să trișați, deoarece ideea principală este doar de a măsura nivelul general de conștientizare a securității în companie și, mai presus de toate, că acesta este doar un sondaj și nu un examen. Ei ar putea răspunde doar la întrebarea principală fără a fi nevoie să răspundă la secțiunea „De ce cred ei așa”, dacă nu știu ce să dea aici ca răspuns.

Unele exemple de întrebări ale sondajului de măsurare a securității ar putea fi:

1. Care dintre următoarele parole este cea mai sigură și de ce credeți asta?
   - Abc123456
   - HerculeS
   - HRE42pazoL
   - $safe456TY

   De ce credeți asta?
2. Care este cea mai periculoasă extensie a atașamentului și de ce credeți asta?
   - *.exe
   - *.com
   - *.bat
   - *.vbs

- all of the above

De ce credeți asta?

3. Politica dvs. de securitate prevede că Biroul de securitate a informațiilor (ISO) nu vă va trimite niciodată o actualizare a unei aplicații, dar tocmai ați primit una, ce ați face în continuare?

- deoarece vine de la security@company.com, care este adresa noastră de e-mail ISO, o voi rula și voi avea cea mai recentă versiune a software-ului.
- așa cum se specifică în Politica de securitate, trebuie să scanez toate atașamentele înainte de a rula, așa că voi scana și rula după aceea.
- Aș apela imediat la biroul ISO pentru a solicita informații suplimentare.

4. Un prieten de-al tău ți-a oferit aseară un CD multimedia, pe care intenționezi să îl verifici de la stația de lucru la locul de muncă; cum o vei face?

- este un prieten de-al meu și nu mi-ar da niciodată fișiere distructive, cum ar fi viruși etc. Am încredere în el / ea, de aceea o voi verifica imediat.
- deși el este un prieten de-al meu, în politica de securitate se precizează că este permisă suportul amovibil, dar utilizarea acestuia trebuie limitată la minimum; mă voi lipi de asta și aș scana conținutul CD-ului și voi vedea ce este în interior înainte de a face acest lucru.
- aș verifica doar conținutul CD-ului de pe computerul meu personal.

5. Un reprezentant al biroului ISO vă solicită (personal) parola dvs., deoarece aceasta a pierdut-o și ar avea nevoie de ea pentru a implementa măsuri de securitate suplimentare pe stația dvs. de lucru; ce ai face?

- nu pot accesa stația de lucru fără parola mea și, deoarece este vorba despre îmbunătățirea securității, le-aș da-le, deoarece acestea sunt cele responsabile de menținerea securității în cadrul organizației.
- am deja stația de lucru securizată corespunzător, așa că nu le voi da.
- n u îmi voi împărtăși parola cuiva, chiar dacă managerul meu încearcă să mă oblige să o spun; aș păstra-o cât mai secretă posibil.

Acestea sunt câteva exemple de întrebări care acoperă majoritatea amenințărilor evidențiate în politica de securitate. Depinde complet de dvs. să decideți câte întrebări ar trebui să fie în sondaj, precum și aspectele pe care ar trebui să le acopere; dar este recomandabil să luați în considerare emiterea de sondaje în mod regulat pentru a monitoriza continuu nivelul și eficacitatea programului.

4.9.4 atragerea atenției

Personalul are deja o mulțime de lucruri la care să se gândească, o mulțime de decizii de luat, de operat și de gestionat prin majoritatea procedurilor de zi cu zi; prin urmare, trebuie să aveți o strategie foarte bună pentru a-i motiva și a fi dornici să învețe cum pot îmbunătăți securitatea companiei.

În prezent, toată lumea este interesată de povești despre securitatea computerului într-un fel sau altul, în special spargerile (cu profil înalt), și folosind acest lucru, scopul principal va fi să înțelegeți „participanții” la programul pe care îl au vor fi de fapt noii „gardieni” ai datelor critice ale companiei (activele informaționale). Veți primi, fără îndoială, întrebări de genul „Da, este minunat să contribuiți la securitatea companiei, dar ce obțin în schimb”, pe care îl definesc ca întrebări normale, la care trebuie să dați răspunsuri adecvate.

Viitorii „studenți” trebuie să fie conștienți și să înțeleagă cât de scump este pentru o companie să organizeze cursuri de conștientizare a securității și să angajeze experți în securitate, pentru a oferi servicii „accesibile” clienților săi. Explicați-le daunele care ar putea fi cauzate companiei, denumirii companiei (mărcii), imaginii acesteia, etc., care vor avea un impact inevitabil asupra lor.

Pe de altă parte, atrageți-le atenția și asupra beneficiilor personale din întregul program și asupra valorii tuturor cunoștințelor care le vor fi furnizate. Un exemplu bun este să menționăm cum toate aceste informații îi vor ajuta în mod semnificativ să crească nivelul de securitate al propriilor computere personale de acasă. Informațiile ce vor fi furnizate nu se aplică doar pentru computerele lor de la locul de muncă, ci se aplică (în întregime) și pentru computerele de acasă.

Un alt punct important de reținut este diferitele moduri în care oamenii învață și memorează lucrurile sau, cu alte cuvinte, tratează informațiile ce tocmai li s-au furnizat. Unii învață citind materialele, în timp ce alții învață mai multe privind diagramele, deși este dovedit că o combinație a acestor metode are un efect maxim în procesul de înțelegere a subiectului. Prin urmare, trebuie să vă asigurați că stilul de prezentare este de o așa manieră încât să atragă o mulțime de oameni cu diferite grade de cunoștințe și înțelegere.

Toată lumea se plictisește să citească materiale lungi, oricât de interesante ar fi ele; dacă nu există nici o imagine, diagramă sau ceva care să aducă un fel de varietate în proces, oamenii pot abandona procesul. Încercați să „vizualizați" fiecare subiect despre care vorbiți adăugând o mulțime de imagini, diagrame, opere de artă relevante și animații.

Animațiile sunt deosebit de bune, deoarece adaugă un element de umor; oamenii își vor aminti cu siguranță o situație amuzantă care reprezintă o procedură mult mai gravă. Acestea sunt cele mai potrivite pentru afișe și sunt cele mai eficiente atunci când sunt plasate în întreaga companie, scopul lor principal fiind un mediu prietenos pentru a răspândi mesajele programului de conștientizare a securității (de exemplu „Blocați computerul când plecați" sau „nu vă împărtășiți" ID și parolă cu ORICINE ", etc).

Umorul joacă un rol esențial în educația prietenoasă a membrilor personalului; ia în considerare utilizarea acestuia după cum consideri potrivit, dar nu transforma întregul program într-o mare comedie în care toată lumea râde și face doar glume despre cuvântul Securitate. Adăugarea unei mici anecdote pline de umor la oricare dintre prelegerile tale de genul „Am un prieten care este atât de paranoic în ceea ce privește securitatea, încât arde fiecare hârtie legată de muncă, dar, nu stinge alarmele de incendiu, ci doar distruge hârtiile cu etichetate confidențial / secret" ar merge bine.

4.9.5 alegerea abordării

Există mai multe abordări pe care le puteți urma atunci când educați personalul, iar această secțiune va indica cea pe care o definesc ca fiind cea mai bună; o combinație atât de modalități formale cât și informale de educație.

Avantajul metodei formale este că va ajuta personalul să-și dea seama de importanța problemei de securitate, deoarece știu că aceste prezentări costă o sumă echitabilă de resurse, efort și bani. Pe de altă parte, va evidenția faptul că societatea ia securitatea foarte serios și, prin urmare, ia măsuri foarte serioase pentru a-și proteja activele informaționale prin educarea personalului său; și tot ce necesită de la ei este puțin timp, devotament și înțelegerea importanței problemei de securitate.

Un alt punct extrem de benefic atunci când desfășurați un program formal de conștientizare este faptul că mesajul, tutorial-ul, prezentarea dvs. vor fi răspândite între majoritatea, dacă nu toți membrii personalului; veți ajunge la o mulțime de oameni în acest fel, ceea ce vă va economisi mult timp în comparație cu metode precum sesiunile individuale etc.

Modul informal de educație constă în reamintiri prin e-mail, discuții, afișe care răspândesc mesaje orientate spre securitate (care sunt discutate în cea mai mare parte la curs), screen savers, mouse pad-uri, căni, stickere, etc pe măsură ce directorii de conștientizare a securității continuă să găsească modalități noi și inovatoare de educare a membrilor personalului. Avantajul acestei metode este că nu împinge (sau, obligă) oamenii în niciun fel, cum ar fi participarea la o întâlnire, ascultarea prelegerilor etc, și este foarte personalizat, ușor de utilizat și extrem de eficient datorită faptului că se apropie foarte mult de viața de zi cu zi și de procedurile de lucru din cadrul companiei (afișe, plăcuțe mouse pad etc.).

Discuțiile informale sunt o altă modalitate extrem de benefică de a educa și măsura abilitățile personalului în care oamenii pun întrebări, la care a răspuns un reprezentant al ISO; atmosfera este de obicei mult mai informală și mai calmă. Acesta este un mod foarte recomandat de a comunica cu angajații, deoarece inițiază o conversație bidirecțională prin care pot fi acoperite multe puncte de interes.

Ca și în multe alte aspecte, trebuie să găsiți echilibrul corect între modalitățile formale și informale, deoarece ambele metode au diferitele lor avantaje și dezavantaje. Monitorizând îndeaproape reacțiile personalului la întâlnirile și prelegerile desfășurate, veți putea revizui în mod semnificativ și îmbunătăți continuu calitatea programului de conștientizare a securității. Oferiți întotdeauna personalului un mod de educație în continuă evoluție, menținându-i astfel interesați, dornici să cunoască și să învețe și să reducă șansele de plictiseală, în timp ce participați la oricare dintre evenimentele programului.

4.10 gestionarea amenințărilor de securitate

Odată ce ați definit cel mai bun mod de educație, aveți planul și strategia pregătite, ați măsurat nivelul de cunoștințe în computer al personalului, ar trebui să începeți prin a discuta în detaliu fiecare dintre elementele subliniate în politica de securitate. Scopul principal al acestei secțiuni este de a explora fiecare dintre aceste elemente în detaliu și de a discuta diverse amenințări, oferindu-vă „cele mai bune practici" gata făcute pe diferite subiecte pe parcurs. Sunteți încurajați să includeți părți din această secțiune în propriile cursuri de conștientizare a securității, oferind astfel personalului o mai bună înțelegere a problemelor acoperite mai jos.

4.10.1 amenințările fizice / desktop explicate

Amenințările care vor fi discutate în această secțiune se referă la modul în care utilizați stația de lucru, la accesarea zonelor restricționate din companie și la modul în care gestionați informațiile sensibile. Voi acoperi toate amenințările posibile, voi discuta în detaliu despre importanța acestora și vă voi oferi diverse modalități eficiente de gestionare a acestora.

4.10.2 accesul la sistem

Personalul trebuie să fie pe deplin conștient de responsabilitatea lor de a-și păstra ID-ul de utilizator și parola cât mai secrete posibil și totul pentru că aceasta este prima linie de apărare în cadrul oricărui sistem: identificarea utilizatorului. Explicați utilizatorului că este complet interzis să partajați ID-ul și parola cu ORICINE, pornind de la reprezentanții Biroului de securitate a informațiilor (ISO), până la membrii familiei lor. Oricât de prostesc ar putea suna asta pentru unii, ei nu trebuie să o facă; chiar dacă managerul lor le cere parola, trebuie să respingă solicitarea. În acest fel, NIMENI nu îi poate forța să-și dezvăluie ID-ul și parola, în niciun caz. Știu cazuri în care managerii au încercat să-și forțeze (sau chiar să păcălească) personalul să-și dea parolele dintr-un motiv sau altul pentru a-și evalua nivelul de conștientizare a securității; pentru a vedea dacă respectă cele menționate în politica de securitate, adică să nu împărtășească ID-ul și parola cu nimeni. Este întotdeauna util să oferiți personalului astfel de exemple „vii" despre modul în care conștientizarea lor ar putea fi evaluată.

Personalul este obligat să nu scrie date contabile sau informații de identificare / parolă pe hârtii libere, sau note (post-it) sau să lase informații sensibile pe tablele albe (de exemplu, după o ședință, tablele albe și / sau rotativele ar trebui curățate) deoarece acest lucru ar putea duce la o potențială spargere, din cauza manipulării necorespunzătoare a datelor sensibile. Indiferent cât de sigur ar putea crede personalul parola, nu ar trebui să li se permită să le stocheze pe niciunul dintre aceste bucăți de hârtie; trebuie să facă tot posibilul și să-l memoreze în schimb. O altă greșeală obișnuită care nu trebuie trecută cu vederea este faptul îngrozitor că majoritatea utilizatorilor tind să ascundă aceste note sub tastatură sau pe un loc „secret", așa cum o numesc, în jurul biroului lor; o altă activitate care ar trebui complet interzisă din motive evidente. Cineva ar putea găsi cu ușurință ascunzătoarea „secretă" și să afle date vitale.

De asemenea, trebuie să vă educați personalul în modul în care sunt create parole puternice. Modalitățile (sigure) de gestionare a datelor contabile sunt prezentate în documentul „Cele mai bune practici pentru parolă", care rezumă pe scurt aceste două aspecte. Am inclus un eșantion „Cele mai bune practici de creare a parolei" și un exemplu de secțiune „Cele mai bune practici de întreținere a parolei" de mai jos, care va oferi o imagine de ansamblu a ceea ce trebuie luat în considerare la scrierea acestor documente.

4.10.3 cele mai bune practici de creare a parolelor

- Parolele trebuie să fie formate dintr-un amestec de litere minuscule (mici), majuscule (mari), cifre și cel puțin un caracter special, cum ar fi (! @ # $% ^ & * () _ + |);
- Lungimea minimă a parolei trebuie să fie de cel puțin 8 caractere;
- Nu utilizați aceeași parolă pe mai multe computere și / sau servicii deoarece, odată dezvăluită, ar compromite securitatea în toate celelalte dintr-o dată.

Exemple bune

- Ona327(sA

- @865DapzI
- 93Sow#-aq

Toate acestea sunt exemple de parole bune, deoarece respectă pe deplin cele mai bune practici de creare a parolelor; conținând astfel un amestec de litere mici, mari, precum și cifre și caractere speciale.

Exemple rele

- aaa123bbb
- abcdefg
- 76543210

Primul exemplu este unul teribil și orice program de cracking configurat corespunzător îl va recupera în câteva minute și nici măcar să nu menționăm de cel de-al doilea și al treilea. Utilizatorul cu ultima parolă (76543210) s-a gândit, evident, că ar fi o parolă ușor de reținut, fiind și una sigură, deoarece este una lungă, dar ceea ce utilizatorul nu știe sau realizează este faptul că, majoritatea programelor de cracking îl vor găsi în câteva secunde (deoarece parola urmează un model numeric specific). Ar putea fi o idee bună să încorporați o mică demonstrație în cursul dvs. de conștientizare, la un moment dat, oferind personalului oportunitatea unică de a vedea cum funcționează un software de cracking pentru parole.

Sfaturi pentru crearea parolelor puternice

- Folosiți primele litere ale unui citat, cântec etc., de exemplu „Ceva ia o parte din mine ...” ar fi „Cipm”
- Alăturați două cuvinte, includeți un număr, precum și un caracter special, de exemplu „run4life #”;
- O strategie plăcută la memorarea parolelor ar putea fi următoarea:
- Să presupunem că parola dvs. este Naige453 $ lZ; mai întâi, trebuie să o pronunți de mai multe ori în minte, apoi întreabă-te care este parola ta, răspunzând la această întrebare în felul următor: „Parola mea este un amestec format din numele Naigel (un prieten străin de-al meu), mai multe numere și semnul dolarului ; parola mea începe și se termină cu litere mari, înainte de ultima literă a numelui (L) există un semn de dolar ($), iar înainte de semnul de dolar, există numere aleatorii.
- Acesta este un truc foarte util pentru oricine încearcă să memoreze sau să-și amintească parola. Repetându-vă (aproape explicându-vă) ce descrie parola dvs. așa cum am sugerat mai sus, sunt sigur că nu veți avea probleme să vă amintiți parole sofisticate, dar puternice.

4.10.4 cele mai bune practici de întreținere a parolelor

Întreținerea corectă a datelor sensibile precum ID-ul de utilizator și parola sunt responsabilitatea fiecărui membru al personalului. Această secțiune va acoperi pe scurt cele mai bune practici de întreținere a parolei.

- NU partajați ID-ul dvs. de utilizator și parola cu nimeni, nici cu un reprezentant ISO, personalul biroului de asistență, membrii familiei și nici cu managerii dvs. Nimeni nu vă poate forța să dezvăluiți ID-urile și parola de utilizator în niciun caz, nu uitați. Este responsabilitatea dumneavoastră să păstrați datele cât mai secrete posibil;
- NU stocați ID-urile și parola de utilizator pe niciun fel de hârtie liberă, notițe lipicioase (post-it), tablă albă, flip chart-uri etc.;
- NU ascundeți ID-urile și parola de utilizator sub tastatură sau oriunde ar putea deveni o ascunzătoare „secretă”. Faceți tot posibilul și memorați-l;
- Schimbați-vă parola (parolele) după perioada de reînnoire a parolei, menționată în politica de securitate;
- Înainte de a introduce ID-ul de utilizator și parola, asigurați-vă că nimeni nu vă urmărește, pentru a evita așa-numita tehnică „surf surfing” (practica de a spiona utilizatorul pentru a obține informații personale de acces);
- Înainte de a utiliza ID-ul de utilizator și parola pe un computer terță parte, asigurați-vă că este bine protejat și că nu are troieni și key loggers.

4.10.5 protecție antivirus

Pe baza lucrărilor publicate, a previziunilor experților, precum și pe baza experiențelor personale, pot afirma cu ușurință că virușii vor continua să fie o amenințare foarte gravă pentru datele critice de afaceri și vor continua să evolueze, devenind mai sofisticați, periculoși și devastatori.

Când începeți să explicați ce este un virus, limitați-l la fapte, de exemplu, cât de distructiv este, ce daune poate provoca, posibilele pierderi financiare legate de un focar de virus etc. Nu deranjați personalul cu informații tehnice specifice precum modul în care funcționează virușii, cum se ascund și multe alte subiecte care nu îi vor fi de interes. În schimb, oferiți celor care sunt cei mai interesați, câteva legături externe (internet) către subiect.

Luați în considerare să explicați ce este un virus / troian / vierme, funcțiile de bază ale fiecăruia dintre acestea, cum să recunoașteți (funcționarea) prezența unuia pe unul dintre sistemele dvs., problemele potențiale pe care le-ar putea provoca și efectele devastatoare asupra întregului companie. Oferiți-le exemple live, discutați pe scurt și răspundeți la cele mai simple și frecvente întrebări care apar, cum ar fi „Se pot recupera datele corupte de viruși" sau „Ce trebuie să faceți după infectarea cu un virus" Cu toate acestea, trebuie să explicați clar că ideea prezentării este de a preveni o infecție în primul rând, deoarece odată infectat cu un virus distructiv, nu puteți face atât de multe, mai ales dacă nu există copii de rezervă ale datelor.

Pe de altă parte, trebuie să explicați cu precizie care ar fi daunele personale după o infecție cu virus; daune și / sau pierderi potențiale de date critice de afaceri, documente, proiecte, planuri de afaceri, prezentări la care au lucrat, împreună cu orice alte date personale stocate pe computer vor fi deteriorate sau, mai mult decât probabil, vor fi distruse. Cunoscând efectele devastatoare pe care le pot avea virușii, personalul va fi mult mai conștient de acest subiect și va înțelege mai mult decât probabil importanța subiectului și riscurile atât pentru compania lor, cât și pentru computerele de acasă.

Parcurgeți numeroasele scenarii ale modului în care virușii pot intra în rețelele companiei, modul în care personalul ar putea fi păcălit să execute un virus, pericolele pe care le reprezintă descărcările de pe Internet, problemele cu semnăturile de viruși depășite etc. Explicați, de asemenea, faptul că scanerele antivirus (AV) nu sunt cea mai bună soluție, mai exact o soluție „nepotrivită" și nici modul în care se bazează pe semnături (fișiere tipare). Discutați cât de utile sunt scanerele de viruși și cum eficacitatea măsurilor preventive care sunt în vigoare depind în mare parte de conștientizarea și vigilența utilizatorilor înșiși.

Personalul trebuie să înțeleagă că scopul nostru principal este să încercăm să prevenim, nu să acționăm după ce suntem infectați; deși vor exista cu siguranță infecții, putem reduce semnificativ riscul de infecție și putem limita potențialele daune, educând personalul și făcându-l conștient de pericolele pe care le prezintă codul și software-ul rău intenționat (virus / troian / vierme).

Avantajele scanării obișnuite a sistemului, precum și problemele potențiale ale nescanării sistemelor dvs. trebuie evidențiate, de asemenea; deși ei știu că scanerele AV nu vor detecta viruși noi, vor ști cel puțin că pot reduce riscul și pot gestiona corect pericolul.

Scanarea sistemelor folosind fișiere de semnături învechite este o altă problemă comună care trebuie atinsă. Personalul ar trebui să-și actualizeze software-ul antivirus / anti-troian cel puțin o dată pe săptămână și, dacă software-ul permite actualizări automate centralizate (majoritatea o fac), actualizările trebuie programate în mod regulat pentru a se asigura că software-ul detectează cele mai recente viruși / troieni / viermi (cunoscuți de laboratorul furnizorului dvs.).

Diferitele modalități de infectare cu coduri rău intenționate trebuie evidențiate, de asemenea; lăsați angajații să vă pună întrebări în mod deschis: vedeți cum reacționează la întrebări precum „Cum mă infectez", apoi oferiți-le o explicație mai bună sau mai completă despre cele mai frecvente și specifice căi de infecție. Mai jos, am inclus un exemplu pentru „Cele mai bune practici privind codul rău intenționat" pentru confortul dvs.; în niciun caz o listă exhaustivă, dar cel puțin veți putea să vă faceți o idee despre ceea ce este considerat activitate periculoasă.

4.10.6 cele mai bune practici pentru cod malițios

- NU rulați niciun fișier fără a-l scana mai întâi, indiferent de extensia fișierului, adică (.exe, .bat, .com, .doc etc.);

- NU descărcați fișiere și / sau programe din surse necunoscute; dacă aveți dubii, contactați biroul ISO cât mai curând posibil;
- NU deschideți atașamente, chiar dacă au fost trimise de un prieten sau de un membru al familiei; verificați mai întâi că într-adevăr, el / ea v-a trimis fișierul, dar totuși scanați înainte de a deschide / rula ceva;
- NU rulați niciun program pe care l-ați găsit pe dischete / CD-uri în jurul biroului dvs. dacă nu sunteți complet sigur că acestea sunt ale voastre; cineva l-ar fi plasat acolo special pentru ca voi „să-l găsiți și să-l verificați";
- Dacă descărcarea este permisă, limitați-o la minim; dacă aveți nevoie de o anumită aplicație sau de altceva, contactați întotdeauna departamentul IT sau biroul ISO pentru informații suplimentare ÎNAINTE să descărcați și să instalați ceva;
- Scanați (scanare completă a sistemului) sistemul cel puțin o dată pe săptămână cu software-ul implicit de scanare AV. Asigurați-vă că actualizați semnăturile virusului înainte de a face acest lucru și, de asemenea, luați în considerare automatizarea procesului prin programarea unei scanări complete a sistemului pentru o scanare regulată convenabilă în viitor;
- Actualizați fișierele de semnături cât mai des posibil, pentru a vă asigura că sunt detectate cele mai recente modele de software rău intenționat;
- Departamentul IT sau Biroul de securitate a informațiilor nu vă vor trimite NICIODATĂ prin mail cele mai recente actualizări ale vreunui software (cu excepția cazului în care acesta este precedat de o campanie mult mediatizată, bine publicizată, la nivelul întregii companii). Dacă detectați activități suspecte, nu ștergeți e-mailul primit și contactați cât mai curând echipa de gestionare a incidentelor sau biroul de asistență;
- dacă aveți îndoieli cu privire la software-ul rău intenționat (viruși / troieni / viermi), contactați imediat ISO, biroul de asistență sau departamentul IT. În acest fel, veți preveni eventualele neplăceri devastatoare, din cauza manipulării necorespunzătoare și eronate a incidentelor periculoase și dăunătoare.
- Tot ceea ce este definit ca interzis trebuie discutat și explicat; de ce este interzis sau restricționat, cum ar putea dăuna companiei sau afacerii etc. Redați mai multe scenarii potențiale, ajutând astfel utilizatorii să înțeleagă subiectul într-un mod ușor de înțeles în timp ce încearcă să atingă baza consecințelor tuturor acestor activități periculoase.

4.10.7 instalarea software-ului

Software-ul gratuit sau orice alt tip de software obținut sau descărcat din surse necunoscute sau care nu sunt de încredere ar putea afecta cu ușurință securitatea companiei, expunând date comerciale critice și / sau corupându-le pe cele sensibile. Mulți utilizatori tind să instaleze astfel de programe (de la protectori de ecran la jocuri și desene animate amuzante în Flash) așa cum au spus-o, pentru diverse nevoi și activități personale; să se distreze, să aibă ceva frumos la care să se uite sau să se relaxeze. În același timp, ei nu realizează potențialele amenințări la care expun sistemele și rețelele companiei, de la software rău intenționat (viruși / troieni / viermi) până la acțiuni legale împotriva companiei pentru instalarea (posibil) de software piratat pe stația de lucru a companiei (s). Astfel, trebuie să familiarizați utilizatorii cu problemele potențiale legate de fiecare dintre aceste probleme și să explicați, de asemenea, politica companiei în ceea ce privește instalarea oricărui software (neautorizat) pe oricare dintre stațiile de lucru ale companiei. Fișierele descărcate de pe Internet, copiate de pe un CD sau o dischetă provenind dintr-o sursă necunoscută sau orice altceva care nu a fost revizuit de către Biroul de securitate a informațiilor sau care nu au fost scanate pentru a detecta potențialul cod rău intenționat (de către sistemele AV corporative) ar putea fi clasificate ca fiind de neîncredere, necunoscute și periculoase. Aplicațiile gratuite, datorită naturii lor de origine, reprezintă o sursă semnificativă de amenințare și ar trebui abordate cu prudență.

Membrii personalului trebuie să fie conștienți de riscurile implicate și să învețe să se gândească de două ori înainte de a acționa asupra problemelor. Acest lucru poate fi stimulat în multe moduri; prin redarea diferitelor scenarii cu privire la modul în care software-ul descărcat fie de pe Internet sau copiat de pe orice suport de eliminare ar putea pune în pericol compania, afacerea acesteia, confidențialitatea cuiva sau utilizarea lățimii de bandă a companiei pentru a comite acțiuni ilegale.

Depinde în totalitate de dvs. să decideți dacă utilizatorilor ar trebui să li se permită să descarce și / sau să instaleze programe terțe pe stațiile lor de lucru; și să pună în aplicare politicile și procedurile adecvate (de securitate) care merg împreună cu această decizie. Nu va trebui doar să declarați în mod clar consecințele pentru cei care încalcă orice restricții, ci și să furnizați procedurile pentru obținerea și instalarea unui software nou.

Este foarte recomandat ca utilizatorii să nu aibă capacitatea de a instala programe noi care ar putea să expună informații sensibile ale companiei, să risipească lățimea de bandă valoroasă sau să corupă date critice. Dacă utilizatorii au nevoie de un nou software instalat pentru utilizare în afaceri, trebuie să contacteze managerul lor, departamentul IT / IS, biroul de asistență sau ISO (în funcție de procedurile stabilite în politică) în loc să întreprindă ei înșiși o astfel de acțiune.

4.10.8 suporturi detașabile (CD's, dischete, benzi, etc.)

Suporturile detașabile, cum ar fi CD-urile (Compact Disks), dischetele și chiar casetele (casetele de rezervă / ADR / DAT / DLT) pot fi definite ca un alt punct de intrare posibil pentru fișierele periculoase și dăunătoare care intră în rețeaua companiei sau pun în pericol securitatea o singură stație de lucru. Pe de altă parte, acestea pot fi folosite și pentru a copia ilegal date sensibile, după care ar fi ușor să ieșiți din incintă cu informațiile furate.

Software-ul rău intenționat (viruși / troieni / viermi) folosește, de asemenea, suporturi detașabile pentru a se răspândi; unii profită de funcția de rulare automată a CD-ului (executând automat fișierul de pornire automată pe CD, care ar putea fi unul distructiv), alții folosesc în continuare metode „clasice”, cum ar fi dischete, pentru a infecta stația de lucru cu un program rău intenționat . Pentru cele mai bune rezultate, dispozitivele media detașabile ar trebui să fie interzise în totalitate (folosind blocaje pentru dischetă sau stații de lucru „fără CD”; CD-urile pot fi utilizate în continuare prin CD-Towers, de exemplu). Dacă trebuie să utilizați suporturi detașabile în organizația dvs., trebuie stabilite cele mai bune practici, trebuie discutate riscurile posibile și scenarii de pericol pentru a reduce programele rău intenționate care intră în rețelele dvs. în toate punctele, protejându-vă astfel compania de un dezastru major.

4.10.9 criptare

Criptarea poate fi definită ca o altă măsură ce are caracterul de „trebuie implementată” și care nu numai că vă va păstra informațiile sensibile și critice protejate împotriva unui potențial atacator, ci vă va proteja și de o mulțime de probleme dacă în cele din urmă apare o încălcare a securității. În politica și procedurile dvs. de securitate trebuie să definiți în mod clar sistemele, fișierele și documentele care ar trebui criptate, de către cine și, cel mai important, ce algoritmi trebuie folosiți. Se recomandă insistent să folosiți algoritmi standard dovediți în industrie, cum ar fi DES, IDEA, Blowfish sau RC5.

4.10.10 backup sistem

Planurile de recuperare în caz de dezastru (DR) sunt esențiale pentru continuitatea afacerii, precum și pentru funcționalitatea corectă a proceselor actuale. Mai devreme sau mai târziu, veți întâmpina inevitabil problema în care un sistem se blochează, indiferent de sistemul de operare utilizat, dar acest lucru poate fi rezolvat cu promptitudine, dacă există proceduri adecvate de backup și planuri de recuperare în caz de dezastru.

Va trebui să definiți activele care trebuie copiate în mod regulat, persoanele responsabile, cele mai bune practici și proceduri, precum și unde ar trebui stocate copiile de rezervă, adică un seif ignifug, seif, în afara amplasamentului etc.

4.10.11 Maintenance

Întreținerea corespunzătoare a computerului / stației de lucru este o altă problemă vitală care nu trebuie trecută cu vederea pe parcursul derulării Programului de conștientizare a securității. Stațiile de lucru ale utilizatorilor reprezintă o sursă semnificativă de amenințare la adresa securității companiei, adesea vizate de așa-numiții „insideri” care caută în speranța de a găsi stații de lucru neprotejate. Prin urmare, trebuie să educați personalul și asupra aspectului securității fizice; din nou, acest lucru poate fi realizat prin parcurgerea scenariilor posibile, oferind în același timp sfaturi pentru o protecție generală mai bună.

4.10.12 tratarea incidentelor

Până acum, personalul dvs. ar trebui să poată defini o potențială problemă de securitate, în timp ce ar trebui să stabiliți regulile pentru cursul acțiunilor care trebuie luate în caz de incident. În politica dvs. trebuie să precizați clar ce trebuie făcut în diverse situații; ideea principală aici ar trebui să fie minimalizarea și limitarea daunelor. Personalul ar trebui să fie informat pe cine este responsabil pentru gestionarea problemelor și pe cine ar trebui să contacteze imediat ce suspectează o potențială problemă de securitate.

4.10.13 amenințările de pe internet explicate

Unul dintre cele mai mari riscuri de securitate din companie este conectivitatea la Internet și utilizarea abuzivă a acestuia prin angajați (inculți). Este un fapt faptul că majoritatea angajaților vor naviga pe site-uri strict interzise și, cel mai probabil, vor ajunge să descarce fișiere rău intenționate și / sau coduri ostile de pe site-urile hackerilor. Oricare dintre aceste activități ar putea afecta productivitatea companiei dvs., mai ales dacă vă gândiți la procesul de recuperare încercând să remediați greșelile făcute de personal.

Prin urmare, este întotdeauna o idee bună să explicăm în detaliu posibilele pericole ale navigării pe Internet; că nu este nevoie să descărcați nimic pentru a infecta computerul cu un virus, troian sau chiar un vierme, dar doar vizitarea site-ului este suficientă pentru a provoca o problemă. Definiți ce constituie un „site interzis” și explicați de ce este interzis, inclusiv problemele care ar putea apărea doar prin vizitarea acestuia.

4.10.14 navigare Web

Navigarea pe web reprezintă o amenințare la adresa securității stației de lucru, precum și a întregii organizații. A fi expus pericolelor navigării pe internet este foarte ușor, deoarece scripturile ostile ar putea fi descărcate și executate automat; tot ce trebuie, de exemplu, este o versiune învechită a browserului web.

Personalul ar trebui să poată face o distincție între site-urile clasificate ca fiind permise, interzise sau potențial periculoase și să încerce să evite vizitarea celor interzise. Java și ActiveX ar trebui să fie dezactivate în mod implicit (nu va da probleme la accesarea paginilor), trebuie să aveți grijă cu filmele Flash etc. și, dacă apare vreodată un indiciu al unei probleme, trebuie contactat imediat biroul ISO.

Există site-uri web în natură, care ar putea încerca să vă scaneze / inunda rețeaua, doar vizitându-le; o altă variantă a acestui scenariu (teoretic, dar foarte posibil) este posibilitatea ca unul dintre angajații dvs. să utilizeze un fel de serviciu de scanare pentru a verifica securitatea stației sale de lucru, risipind astfel lățimea de bandă valoroasă. Ceva de genul acesta va produce invariabil mai multă muncă pentru biroul ISO, precum și sistemele lor vor înregistra probabil utilizarea acestui serviciu ca o posibilă încercare de spargere. Jocurile de noroc online și site-urile pornografice ar trebui să fie pe deplin interzise și utilizarea internetului de către personal monitorizată pentru a se asigura că respectă regulile și reglementările stabilite de Programul de conștientizare a securității.

4.10.15 utilizarea email-ului

În general, sistemele de poștă electronică ale companiei sunt o zonă cu risc ridicat datorită disponibilității lor constante în lumea exterioară, iar riscul este adesea dublu. Utilizarea e-mailului pentru a desfășura diferite tipuri de activități, a contacta clienții și integrarea acestuia în multe alte procese legate de afaceri expune adresele de e-mail ale companiei și sistemele (de poștă) potențialilor atacatori. Pe de altă parte, acesta este, de asemenea, punctul de intrare numărul unu din care majoritatea programelor rău intenționate intră în companie. Prin urmare, un program bine cunoscut și dovedit de protecție a codului rău intenționat este obligatoriu pe toate gateway-urile de poștă electronică, deoarece va detecta, bloca și / sau filtra majoritatea fișierelor periculoase cunoscute și scripturilor ostile care încearcă să intre în rețelele companiei.

La fel ca în toate aspectele securității IT, securitatea la nivelul întregii companii poate fi îmbunătățită doar prin educarea adecvată a personalului. Prin urmare, este foarte recomandat să stabiliți cele mai bune practici pentru utilizarea e-mailului, concentrându-vă asupra punctelor de mai jos.

4.10.16 cele mai bune practici de utilizare a email-ului

- Dacă atașamentele (prin e-mail) sunt permise, atașamentele trebuie scanate înainte de deschidere, precum și confirmarea cu expeditorul (adică prin telefon) că într-adevăr a fost trimis un atașament. Acest lucru va reduce, de asemenea, riscul de a rula un program care a fost trimis automat prin e-mail (necunoscut inițiatorului) printr-un fel de aplicație rău intenționată care a folosit contul (conturile) de e-mail și / sau sistemul de expediere al expeditorului. Dacă atașamentele sunt interzise, urmați politica și nu descărcați / rulați niciun fișier (e) primit (e) ca atașamente;
- Java și ActiveX trebuie dezactivate în timpul citirii e-mail-urilor pentru a gestiona riscul de executare automată a programelor rău intenționate. La fel ca în browserul de internet, anumite opțiuni ale programului pot fi de obicei setate și blocate prin politici de sistem care stabilesc automat aceste condiții pentru toți utilizatorii la conectare;
- Nu utilizați conturile de e-mail ale companiei în scopuri de înregistrare de niciun fel și nu le utilizați în timp ce postați mesaje în forumuri web sau grupuri de știri. Poate doriți să creați un cont special numai în acest scop;
- Nu utilizați sistemul de poștă electronică al companiei pentru a vă conduce propria afacere, corespondență personală excesivă, trimiterea atașamentelor mari, risipind astfel lățimea de bandă valoroasă;
- Nu răspundeți la scrisori în lanț sau la orice alt tip de spam utilizând sistemele de e-mail ale companiei; dacă aveți dubii, contactați biroul ISO;
- Nu redirecționați niciodată date ale companiei către conturi de e-mail externe (adică trimiteți un document de lucru în contul dvs. de e-mail de acasă, deci să lucrați mai departe de acasă în acea seară), fără a verifica mai întâi managerul dvs. și / sau contactați biroul ISO;
- Utilizarea corectă a sistemului de poștă electronică ar trebui monitorizată continuu și utilizatorii ar trebui să fie conștienți că ar putea fi responsabili pentru activități ilegale, cum ar fi spamul, trimiterea și primirea de conținut ilegal etc.

4.10.17 aplicații de mesagerie instant

Mulți utilizatori tind să folosească aceste programe pentru a comunica cu prietenii, pentru a trimite și primi atașamente, mesaje etc. deoarece aceste aplicații încearcă adesea să păcălească conținutul de blocare a gateway-ului la nivel server pentru a permite trecerea conținutului. Cu toate acestea, ei nu realizează pe deplin pericolele acestor programe și potențialele daune pe care le-ar putea provoca.

Un instantaneu din publicația noastră anterioară „The Complete Windows Trojans Paper”, disponibil pe site-ul nostru la http://www.frame4.com/publications/index.php, analizează diferite scenarii de infectare cu un program rău intenționat prin ICQ:

- Nu puteți fi niciodată 100% sigur cine se află de cealaltă parte a computerului în acel moment. Poate fi cineva care a spart ICQ UIN (Numărul unic de identificare) al prietenului tău și dorește să răspândească niște troieni;
- Versiunile vechi ale ICQ conțineau erori în caracteristica WebServer care creează un site web pe computerul dvs. cu informațiile dvs. din baza de date ICQ. Bug-ul a însemnat că atacatorul ar putea avea acces la ORICE fișier de pe mașina dvs. ... și probabil vă dați seama ce s-ar putea întâmpla dacă cineva are acces la win.ini sau la alt fișier de sistem: un troian instalat pe computer în câteva minute;
- Trojan.exe este redenumit în Trojan .... (150 de spații) .txt.exe, pictograma s-a schimbat într-un fișier .txt real; acest lucru cu siguranță te va infecta. Probabil că această eroare va fi remediată în versiunile mai noi.

Indiferent de aplicația de mesagerie instant pe care o utilizați, ați putea fi întotdeauna infectat sau exploatat; printr-o anumită eroare de aplicație despre care nu ați auzit niciodată sau o versiune de bug-uri pe care nu v-ați deranjat niciodată să o actualizați.

Când vine vorba de schimbul de informații și fișiere, indiferent de unde, de la cine sau cum, vă rugăm să fiți conștienți de faptul că există anumite pericole legate de acesta; realizează posibilele pericole ale acțiunilor și naivității tale și acționează în consecință.

4.10.18 descărcare

Descărcarea oricăror date din surse necunoscute și de încredere în timpul utilizării sistemelor și rețelelor companiei ar putea avea un efect devastator asupra proceselor de afaceri; s-ar putea să vă confruntați cu o situație în care datele dvs. să fie pierdute, corupte sau, în anumite cazuri, modificate. Prin urmare, ar trebui să vă propuneți să educați personalul cu privire la procedurile de descărcare a informațiilor într-un mod sigur; aceasta constă în asigurarea descărcării fișierelor numai atunci când este absolut necesar, scanarea fișierelor descărcate cu soluția corporativă Anti-Virus / Anti-Troian înainte de a o deschide etc.

Pentru comoditate, am creat o secțiune rezumată „Cele mai bune practici pentru utilizarea internetului" de mai jos; din nou, departe de a fi o listă exhaustivă, este menită să vă ofere câteva indicații de bază despre utilizarea sigură a internetului.

4.10.19 cele mai bune practici de utilizare a internetului

- Java și ActiveX sunt blocate în mod implicit. Scripturile care conțin Java și ActiveX prezintă un mare pericol datorită naturii lor nesigure, iar problemele rezultate ar putea avea efecte devastatoare asupra computerului dvs., ca să nu mai vorbim de companie. Vă rugăm să nu blocați, opriți sau manipulați orice măsură (de exemplu, politica de grup) care există pentru a le elimina și, dacă aveți probleme la achiziționarea unui articol sau la vizitarea unui site web de încredere, contactați departamentul IT, Biroul de asistență sau ISO birou pentru asistență;
- Nu vizitați site-uri web necorespunzătoare cu conținut dezacordabil; pornografie, jocuri de noroc, warez (software piratat), site-uri de hacker / hacking, precum și cele considerate în general ca fiind interzise de politica dvs. de securitate;
- Dacă este permisă utilizarea aplicațiilor de mesagerie instantanee (IM), nu acceptați atașamente indiferent de tipul fișierului, extensia sau originatorul;
- Descărcarea de software, fișiere sau orice altceva este interzisă. Dacă aveți nevoie de aplicații pentru activitatea dvs. de zi cu zi, contactați fie departamentul IT, biroul de asistență sau biroul ISO. Pentru a finaliza procesul, va trebui să transmiteți un formular de solicitare (software) semnat de managerul dvs. Dacă obțineți autorizația pentru a descărca o bucată de software, nu uitați să nu o executați niciodată înainte de a o scana cu software-ul corporativ Anti-Virus / Anti-Troian;
- Toată activitatea pe internet ar trebui monitorizată continuu și utilizatorii ar trebui să fie conștienți de faptul că ar putea fi considerați responsabili pentru vizitarea site-urilor web interzise, descărcarea fișierelor și conținutului ilegal, precum și pentru sancționarea accesului la internet limitat (până când pot demonstrează că sunt pe deplin conștienți de riscurile create de acțiunile lor).

chapter 5 elemente de criptografie: funcții hash

Funcțiile hash SHA reprezintă un set de funcții hash criptografice proiectate de Agenția Națională de Securitate (NSA) și publicate de NIST ca standard federal de prelucrare a informațiilor din S.U.A. SHA înseamnă Secure Hash Algorithm. Cei trei algoritmi SHA inițiali sunt structurați diferit și se disting ca SHA-0, SHA-1 și SHA-2. Familia SHA-2 utilizează un algoritm identic cu o dimensiune a hashului variabilă, având denumirile distincte ca SHA-224, SHA-256, SHA-384 și SHA-512.

SHA-1 este cel mai utilizat dintre funcțiile hash existente SHA și este folosit în mai multe aplicații și protocoale de securitate utilizate pe scară largă. În 2005, au fost identificate deficiențe de securitate în SHA-1, și anume că ar putea exista o posibilă slăbiciune matematică, indicând faptul că ar fi de dorit o funcție mai puternică de tip hash. Deși nu s-au raportat încă atacuri asupra variantelor SHA-2, ele sunt asemănătoare algoritmilor cu SHA-1 și astfel s-au făcut eforturi pentru a dezvolta alternative îmbunătățite. Un nou standard hash, SHA-3, folosind un nou algoritm numit Keccak, a fost selectat printr-o competiție deschisă care a avut loc între toamna lui 2008 și 2012.

SHA-3 a fost standardizat în august 2015 ca FIPS 202.

5.1 SHA-0 și SHA-1

Specificația originală a algoritmului a fost publicată în 1993 ca standardul Secure Hash, FIPS PUB 180, de către agenția americană de standardizare NIST (Institutul Național de Standarde și Tehnologie). Această versiune este acum adesea denumită SHA-0. A fost retrasă de către NSA la scurt timp după publicare și a fost înlocuită de versiunea revizuită, publicată în 1995 în FIPS PUB 180-1 și denumită în mod obișnuit SHA-1. SHA-1 diferă de SHA-0 numai printr-o singură rotație la nivel de bit în rotația de mesaje a funcției sale de compresie; acest lucru s-a făcut, conform NSA, pentru a corecta un defect al algoritmului original care reducea securitatea sa criptografică. Cu toate acestea, ANS nu a furnizat nicio explicație suplimentară și nici nu a identificat defectul corectat. Puncte slabe au fost ulterior raportate atât la SHA-0, cât și la SHA-1. SHA-1 pare să ofere o mai mare rezistență la atacuri, susținând afirmația NSA că schimbarea a sporit securitatea.

SHA-1 (ca și SHA-0) produce un hash de 160 biți dintr-un mesaj cu o lungime maximă de (2^64 - 1) biți. SHA-1 se bazează pe principii similare celor utilizate de Ronald L. Rivest de la MIT în proiectarea algoritmilor de generare a hash-ului mesajelor MD4 și MD5, dar are un design mai conservator.

5.2 SHA-2 family

NIST a publicat patru funcții hash suplimentare în familia SHA, denumite după lungimile hashului (în biți): SHA-224, SHA-256, SHA-384 și SHA-512. Algoritmii sunt denumiți în mod colectiv SHA-2.

Algoritmii au fost publicați pentru prima oară în 2001 în draftul FIPS PUB 180-2, moment în care revizuirea și comentariul au fost acceptate. FIPS PUB 180-2, care include și SHA-1, a fost lansat ca standard oficial în 2002. În februarie 2004 a fost publicată o notificare de schimbare pentru FIPS PUB 180-2, specificând o variantă suplimentară, SHA-224, lungimea cheii fiind cea a două chei Triple DES. Aceste variante sunt brevetate în brevetul US 6829355. Statele Unite au eliberat brevetul în baza unei licențe gratuite.

SHA-256 și SHA-512 sunt funcții noi de tip hash calculate cu cuvinte de 32 și, respectiv, 64 de biți. Ele folosesc cantități diferite de șifturi și constante aditive, dar structurile lor sunt altfel practic identice, diferențiind numai numărul de runde. SHA-224 și SHA-384 sunt pur și simplu variante trunchiate ale primelor două, calculate cu valori inițiale diferite.

Spre deosebire de SHA-1, funcțiile SHA-2 nu sunt utilizate pe scară largă, în ciuda securității lor mai bune. Motivele ar putea include lipsa suportului pentru SHA-2 pe sistemele care rulează Windows XP SP2 sau mai vechi, Iprecum și ipsa de motivație percepută din moment ce nu au fost găsite coliziuni SHA-1 sau dorința de a aștepta până la standardizarea SHA-3. SHA-256 este utilizat pentru a autentifica pachetele software Debian Linux și în standardul de semnătură digitală a mesajelor DKIM; SHA-512 face parte dintr-un sistem de autentificare a înregistrărilor video de arhivă de la Tribunalul Penal Internațional pentru genocidul

din Ruanda. SHA-256 și SHA-512 sunt propuse pentru utilizarea în directiva DNSSEC NIST ce sugerează agențiilor guvernamentale americane să oprească cele mai multe utilizări ale SHA-1 după 2010 iar finalizarea SHA-3 poate accelera migrarea de la SHA-1.

În prezent, cele mai bune atacuri publice asupra SHA-2 au spart 24 din cele 64 (respectiv 80) de runde.

5.3 SHA-3

O competiție deschisă pentru o nouă funcție SHA-3 a fost anunțată în mod oficial în Registrul Federal al SUA în 2 noiembrie 2007. "NIST inițiază un efort de a dezvolta unul sau mai mulți algoritmi de hash suplimentari printr-o competiție publică, similar cu procesul de dezvoltare pentru Advanced Encryption Standard (AES)." Propunerile au avut loc pe 31 octombrie 2008, iar proclamarea unui câștigător și publicarea noului standard au avut loc în 2012.

NIST a selectat 51 de intrări pentru Runda 1, iar 14 dintre ele au avansat la Runda 2.

5.3.1 acceptate pentru runda doi

Următoarele propuneri de funcții de tip hash au fost acceptate pentru runda a doua.

- BLAKE
- Blue Midnight Wish
- CubeHash
- ECHO (France Telecom)
- Fugue
- Grøstl (Knudsen et al.)
- Hamsi
- JH
- Keccak (Keccak team, Daemen et al.)
- Luffa
- Shabal
- SHAvite-3
- SIMD
- Skein (Schneier et al.)

5.3.2 iar câștigătorul este … Keccak

În octombrie 2012, algoritmul Keccak a fost declarat câștigător.

5.4 utilizare

SHA-1 este cea mai răspândită funcție hash din familia SHA. Acesta face parte din mai multe aplicații și protocoale de securitate utilizate pe scară largă, inclusiv TLS și SSL, PGP, SSH, S / MIME și IPsec. Aceste aplicații pot utiliza, de asemenea, MD5; atât MD5 cât și SHA-1 sunt descendenți din MD4. De asemenea, funcția de hash SHA-1 este utilizată în sistemele distribuite de control al versiunilor, cum ar fi Git, Mercurial și Monotone, pentru a identifica versiunile și pentru a detecta coruperea sau modificarea datelor.

SHA-256, SHA-384 și SHA-512 sunt algoritmii hash de securitate specificați prin lege pentru a fi utilizați în anumite aplicații ale guvernului SUA, inclusiv utilizarea în cadrul altor algoritmi și protocoale criptografice, pentru protecția informațiilor nesecrete sensibile. FIPS PUB 180-1 a încurajat, de asemenea, adoptarea și utilizarea SHA-1 de către organizațiile private și comerciale. SHA-1 urmează a fi retras pentru majoritatea utilizărilor guvernamentale; Institutul National de Standarde si Tehnologie din SUA spune ca "agențiile federale ar trebui sa inceteze sa utilizeze SHA-1 pentru ... aplicatii care necesita rezistenta la coliziune cât mai curand posibil si trebuie sa utilizeze familia SHA-2 de funcții hash pentru aceste aplicatii dupa 2010".

O primă motivație pentru publicarea algoritmului Secure Hash a fost DSS – Digital Signature Standard (Standardul de Semnătură Digitală), în care este încorporat.

Funcțiile hash SHA au fost folosite ca bază pentru algoritmii de cifrare de tip bloc SHACAL.

5.5 analiză criptografică și validare

Pentru o funcție hash în care L este numărul de biți din digestul mesajului, găsirea unui mesaj care corespunde unui hash de mesaj dat poate fi întotdeauna efectuată folosind o căutare de forță brute în 2^L evaluări. Acesta este numit **atac de preimagine** și poate sau nu poate fi practic în funcție de L și de mediul de calcul utilizat. Al doilea criteriu, găsirea a două mesaje diferite care produc același hash de mesaj, cunoscut sub numele de coliziune, necesită, în medie, numai 2^(L/2) evaluări folosind un atac de tip ziua de naștere (**birthday atack**). Din acest din motiv, puterea unei funcții hash este de obicei comparată cu un cifru simetric cu cheie jumătate din lungimea digestului mesajului. Astfel, SHA-1 a fost considerat inițial că are o putere de 80 de biți.

Criptografii au produs perechi de coliziuni pentru SHA-0 și au găsit algoritmi care ar trebui să producă coliziuni SHA-1 în mult mai puține decât cele 2^80 de evaluări inițial preconizate.

În ceea ce privește securitatea practică, o preocupare majoră cu privire la aceste noi atacuri este că acestea ar putea deschide calea către unele mai eficiente. Dacă se va întâmpla acest lucru, rămâne de văzut, dar se consideră că migrarea la hashuri mai puternice este prudentă. Unele aplicații care utilizează hash-urile criptografice, cum ar fi stocarea parolelor, sunt afectate minimal de un atac de coliziune. Construirea unei parole care să funcționeze pentru un anumit cont necesită un atac de preimage, precum și accesul la hash-ul parolei originale (stocate, în mod obișnuit, în fișiere umbră) ceea ce pare să nu să fie chiar așa de simplu. Inversarea criptării parolei (de exemplu, pentru a obține o parolă pentru a accesa contul unui utilizator în altă parte) nu este posibilă prin aceste atacuri. (Cu toate acestea, chiar și un hash de parole securizat nu poate împiedica atacurile de forță brute asupra parolelor slabe.)

În cazul semnării documentelor, un atacator nu poate falsifica pur și simplu o semnătură dintr-un document existent - atacatorul ar fi trebuit să producă o pereche de documente, unul inofensiv și unul dăunător și să determine titularul cheii private să semneze documentul inofensiv. Există circumstanțe practice în care acest lucru este posibil; până la sfârșitul anului 2008 a fost posibil să se creeze certificate SSL false folosind o coliziune MD5.

5.5.1 SHA-0

La CRYPTO 98, doi cercetători francezi, Florent Chabaud și Antoine Joux, au prezentat un atac asupra SHA-0: coliziunile pot fi găsite cu complexitate 2^61, mai puțin decât cele 2^80 pentru o funcție hash ideală de aceeași dimensiune .

În 2004, Biham și Chen au descoperit semi-coliziuni pentru SHA-0:două mesaje care au aproape același hash; în acest caz, 142 din cei 160 de biți sunt egali. De asemenea, au descoperit coliziunile complete ale SHA-0 redus la 62 din cele 80 de runde.

Ulterior, la 12 august 2004, o coliziune pentru algoritmul SHA-0 complet a fost anunțată de Joux, Carribault, Lemuet și Jalby. Acest lucru a fost făcut folosind o generalizare a atacului Chabaud și Joux. Găsirea coliziunii a avut complexitatea 2^51 și a durat aproximativ 80.000 de ore CPU pe un supercomputer cu 256 procesoare Itanium 2. (Echivalent cu 13 zile de utilizare integrală a calculatorului.)

La 17 august 2004, în cadrul sesiunii Rump de la CRYPTO 2004, rezultate preliminare au fost anunțate de Wang, Feng, Lai și Yu, despre un atac asupra MD5, SHA-0 și a altor funcții hash. Complexitatea atacului lor asupra SHA-0 este 2^40, semnificativ mai bună decât atacul lui Joux et al.

În februarie 2005, a fost anunțat un atac făcut de Xiaoyun Wang, Yiqun Lisa Yin și Hongbo Yu care au putut identifica coliziuni în SHA-0 în 2^39 de operațiuni.

5.5.2 SHA-1

În lumina rezultatelor pentru SHA-0, unii experți au sugerat că planurile de utilizare a SHA-1 în noile criptosisteme ar trebui reconsiderate. După ce au fost publicate rezultatele CRYPTO 2004, NIST a anunțat că intenționează să elimine treptat utilizarea SHA-1 până în 2010 în favoarea variantelor SHA-2.

La începutul anului 2005, Rijmen și Oswald au publicat un atac asupra unei versiuni reduse a SHA-1: 53 din 80 de runde - care găsește coliziuni cu un efort computațional de mai puțin de 2^80 operațiuni.

În februarie 2005 a fost anunțat un atac făcut de Xiaoyun Wang, Yiqun Lisa Yin, Bayarjargal și Hongbo Yu. Atacurile pot găsi coliziuni în versiunea completă a SHA-1, care necesită mai puțin de 2^69 operațiuni. (Căutarea cu forță brute ar necesita 2^80 operațiuni.)

Autorii scriu: "În particular, analiza noastră se bazează pe atacul diferențial original asupra SHA-0, atacul de semi-coliziune asupra SHA-0, tehnicile de coliziune multibloc, precum și pe tehnicile de modificare a mesajului folosite în atacul de căutare a coliziunilor pentru MD5. Spargere SHA-1 nu ar fi posibilă fără aceste tehnici analitice puternice. " Autorii au prezentat o coliziune pentru SHA-1 cu 58 de runde, găsită cu 2^33 operații de tip hash. Lucrarea cu descrierea completă a atacului a fost publicată în august 2005 la conferința CRYPTO.

Într-un interviu, Yin afirmă: "În principiu, exploatăm următoarele două puncte slabe: Unul este că pasul de preprocesare a fișierelor nu este suficient de complicat, altul este că anumite operații matematice în primele 20 de runde au probleme de securitate neașteptate".

La 17 august 2005, o îmbunătățire a atacului SHA-1 a fost anunțată în numele Xiaoyun Wang, Andrew Yao și Frances Yao la sesiunea CRYPTO 2005, reducând complexitatea necesară pentru a găsi o coliziune în SHA-1 la 2^63. La 18 Decembrie 2007 detaliile acestui rezultat au fost explicate și verificate de Martin Cochran.

Christophe De Cannière și Christian Rechberger au îmbunătățit și mai mult atacul asupra SHA-1 în "Finding SHA-1 Characteristics: General Results and Applications", primind premiul pentru cea mai bună lucrare la ASIACRYPT 2006. O coliziune a două blocuri pentru SHA-1 a fost prezentată, găsită folosind metode neoptimizate cu 2^35 de evaluări ale funcțiilor de comprimare. Deoarece acest atac necesită echivalentul a aproximativ 2^35 de evaluări, este considerat a fi un salt teoretic semnificativ. Pentru a găsi o coliziune reală în cele 80 de runde ale funcției hash, cu toate acestea, sunt necesare cantități masive de timp pentru calculator. În acest scop, o căutare de coliziune pentru SHA-1 utilizând platforma distribuită de calcul BOINC a început la 8 august 2007, organizată de Universitatea Tehnică din Graz. Efortul a fost abandonat pe 12 mai 2009 din cauza lipsei de progres.

La sesiunea CRYPTO 2006, Christian Rechberger și Christophe De Cannière au susținut că au descoperit un atac de coliziune asupra SHA-1, care ar permite atacatorului să selecteze cel puțin părți ale mesajului.

Cameron McDonald, Philip Hawkes și Josef Pieprzyk au prezentat un atac de coliziune a hash-ului cu complexitate revendicată de 2^52 la sesiunea Rump a Eurocrypt 2009. Cu toate acestea, lucrarea însoțitoare, "Differential Pathfor SHA-1 with complexity O(2^{52})" a fost retrasă din cauza descoperirii de către autori că estimarea lor a fost incorectă.

În 2017, CWI Amsterdam și Google au anunțat găsirea unei coliziuni (două fișiere pdf distincte care produc același hash) pentru SHA-1. [SHA1C]. Numărul de evaluari a fost de aproximativ 2^63.1, adică de 100000 de ori ma rapid decât prin utilizarea forței brute printr-un atac al zilei de nștere (birthday attack).

5.5.3 SHA-2

Există două atacuri de preimagine împotriva SHA-2 cu un număr redus de runde. Primul atacă 41 de runde SHA-256 din cele 64 de runde cu complexitate de timp de 2^253,5 și complexitate spațială de 2^16 și 46 de runde SHA-512 din cele 80 de runde cu timpul 2^511,5 și spațiu 2^3 . Cel de-al doilea atacă 42 de runde SHA-256 cu complexitate de timp de 2^251,7 și complexitate spațială de 2^12 și 42 de runde SHA-512 cu timpul 2^502 și spațiu 2^22.

5.6 generalități despre SHA-1

SHA-1 este obiectul specificației FIPS 180-1, un document NIST din 1993. Este o funcție hash de tip bloc, mărimea blocului fiind de 512 biți (64 octeți). Produce un hash de 160 de biți, de obicei sub forma a 5 cuvinte de 32 de biți, reprezentate în hexazecimal. Este considerată o funcție hash relativ sigură. Totuși, se preconizează înlocuirea sa treptată (începând cu 2010) cu variante mai sigure, precum SHA-2 și SHA-3.

5.7 preliminarii operaționale

5.7.1 șiruri de biți și întregi

Se va folosi următoarea terminologie referitoare la șiruri de biți și numere întregi:

a. O cifră hexazecimală este un element al mulțimii {0, 1, ..., 9, A, ..., F}. O cifră hexazecimală este reprezentarea unui șir de 4 biți. Exemple: 7 = 0111, A = 1010.

b. Un cuvânt este egal cu un șir de 32 de biți care poate fi reprezentat ca o succesiune de 8 cifre hexazecimale. Pentru a converti un cuvânt la 8 cifre hexazecimale fiecare șir de biți pe 4 biți este convertit în echivalentul său hexa, așa cum este descris la punctul (a) de mai sus. Exemplu:

1010 0001 0000 0011 1111 1110 0010 0011 = A103FE23.

c. Un întreg între 0 și 2^32 - 1 inclusiv poate fi reprezentat ca un cuvânt (de 32 de biți). Cei mai puțin semnificativi patru biți ai întregului sunt reprezentați ca cea mai din dreapta cifră hexa a reprezentării cuvântului. Exemplu: întregul 291 = 2^8 + 2^5 + 2^1 + 20 = 256 + 32 + 2 + 1 este reprezentat de cuvântul hexa: 00000123.

Dacă z este un număr întreg, 0 <= z <2^64, atunci z = 2^32x + y unde 0 <= x <2^32 și 0 <= y <2^32. Deoarece x și y pot fi reprezentate ca și cuvinte X și Y, z poate fi reprezentat ca o pereche de cuvinte (X, Y).

d. blocul = șir de 512 biți. Un bloc (de exemplu, B) poate fi reprezentat ca o secvență de 16 cuvinte.

5.7.2 operații pe cuvinte

Următorii operatori logici vor fi aplicați cuvintelor:

a. Operațiuni la nivel de bit cu cuvinte logice

X ^ Y = "și" (la nivel de bit) al lui X și Y.

X V Y = "sau inclusiv" de X și Y.

X XOR Y = "sau exclusiv" al lui X și Y

~X = "complementul" logic la nivel de bit al lui X.

Exemplu:

```
          01101100101110011101001001111011
XOR       01100101110000010110100110110111
          ----------------------------------------------------
=         00001001011110001011101111001100
```

b. Operația X + Y este definită după cum urmează: cuvintele X și Y reprezintă numerele întregi x și y, unde 0 <= x <2^32 și 0 <= y <2^32. Pentru numerele întregi pozitive n și m, notăm cu (n mod m) restul împărțind lui n cu m. Calculăm:

z = (x + y) mod 2^32.

Atunci 0 <= z <2^32. Convertim z la un cuvânt, Z și definim Z = X + Y.

c. Operația de șiftare circulară stânga S^n (X), unde X este un cuvânt și n este un număr întreg cu 0 <= n^32, este definită prin:

S^n (X) = (X << n) OR (X >> 32-n).

În cele de mai sus, X << n este obținut după cum urmează: desconsiderați cei mai din stânga n biți ai lui X și apoi introduceți n de zero în dreapta (rezultatul va fi în continuare de 32 de biți). X >> n este obținut prin

eliminarea celor mai mulți n biți ai lui X și apoi completarea rezultatului cu n de zero în stânga. Astfel, S^n (X) este echivalent cu o deplasare circulară a lui X cu n poziții spre stânga.

5.8 descrierea funcției SHA-1

5.8.1 completarea mesajului (message padding)

SHA-1 este folosit pentru a calcula hash-ul unui mesaj sau al unui fișier de date care este furnizat ca intrare. Mesajul sau fișierul de date ar trebui considerat a fi un șir de biți. Lungimea mesajului este numărul de biți din mesaj (mesajul gol are lungimea 0). Dacă numărul de biți dintr-un mesaj este un multiplu de 8, pentru compactitate putem reprezenta mesajul în hexa. Scopul completării mesajului este de a face lungimea totală a unui mesaj să fie un multiplu de 512. SHA-1 procesează secvențial blocuri de 512 biți atunci când se compune digestul mesajului. Următoarele specifică modul în care se realizează această completare. Pe scurt, un "1" urmat de m de "0", urmat de un întreg pe 64 de biți sunt atașați la sfârșitul mesajului pentru a produce un mesaj completat (padded) cu lungimea de 512 * n. Numărul întreg pe 64 de biți este l, lungimea mesajului original. Mesajul completat (căptușit) este apoi procesat de SHA-1 ca n blocuri de 512 biți.

Să presupunem că un mesaj are lungimea l <2^64. Înainte de a intra în SHA-1, mesajul este completat (căptușit) în dreapta după cum urmează:

a. un bit cu valoarea "1" este atașat. Exemplu: dacă mesajul original este "01010000", acesta este căptușit la "010100001".

b. Se adaugă biți zero "0". Numărul de zerouri "0" va depinde de lungimea inițială a mesajului. Ultimii 64 de biți ai ultimului bloc de 512 biți sunt rezervați pentru lungimea l a mesajului original.

Exemplu: Să presupunem că mesajul original este șirul de biți

01100001 01100010 01100011 01100100 01100101.

După pasul (a) obținem:

01100001 01100010 01100011 01100100 01100101 1.

Deoarece l = 40, numărul de biți de mai sus este 41 și 407 de "0" sunt atașați, făcând totalul acum 448. Aceasta dă (în hexa):

61626364 65800000 00000000 00000000

00000000 00000000 00000000 00000000

00000000 00000000 00000000 00000000

00000000 00000000.

c. Obțineți reprezentarea de 2 cuvinte a lui l, numărul de biți din mesajul original. Dacă l <2^32 atunci primul cuvânt este zero. Adăugați aceste două cuvinte la mesajul căptușit.

Exemplu: Să presupunem că mesajul inițial este ca în (b). Apoi l = 40 (rețineți că l este calculat înainte de orice umplutură). Reprezentarea cu două cuvinte a lui 40 este hex 00000000 00000028. Prin urmare, mesajul căptușit final este în hexa:

61626364 65800000 00000000 00000000

00000000 00000000 00000000 00000000

00000000 00000000 00000000 00000000

00000000 00000000 00000000 00000028.

Mesajul căptușit va conține 16*n cuvinte pentru un număr n> 0. Mesajul căptușit este considerat ca o secvență de n blocuri M_1, M_2, ..., M_n, unde fiecare M_i conține 16 cuvinte și M1 conține primele caractere (sau biți) ai mesajului.

5.8.2 funcții utilizate

SHA-1 utilizează o secvență de funcții logice f_0, f_1, ..., f_79. Fiecare f_t, 0 <= t <= 79, operează pe trei cuvinte pe 32 de biți B, C, D și produce un cuvânt de 32 de biți ca ieșire. f_t (B, C, D) este definită după cum urmează: pentru cuvintele B, C, D,

f_t(B,C,D) = (B AND C) OR ((NOT B) AND D) (0 <= t <= 19)

f_t(B,C,D) = B XOR C XOR D (20 <= t <= 39)

f_t(B,C,D) = (B AND C) OR (B AND D) OR (C AND D) (40 <= t <= 59)

f_t(B,C,D) = B XOR C XOR D (60 <= t <= 79).

5.8.3 constante utilizate

SHA-1 utilizează o secvență de cuvinte constante K (0), K (1), ..., K (79). În hexa, acestea sunt date de:

K = 5A827999 (0 <= t <= 19)

K_t = 6ED9EBA1 (20 <= t <= 39)

K_t = 8F1BBCDC (40 <= t <= 59)

K_t = CA62C1D6 (60 <= t <= 79)

5.9 pseudo-cod pentru SHA-1

O funcție hash criptografică este o transformare care ia o intrare și returnează un șir de dimensiuni fixe, care se numește valoarea hash.

```
    // inițializare variabile
    h0 = 0x67452301
    h1 = 0xEFCDAB89
    h2 = 0x98BADCFE
    h3 = 0x10325476
    h4 = 0xC3D2E1F0

    // pre-procesare:
    adăugați bitul "1" la mesaj
    adăugați 0 ≤ k < 512 biți cu caloarea"0", astfel încât lungimea mesajului
rezultat (în biți) este congruent cu 448 ≡ -64 (mod 512)
    adăugați lungimea mesajului (înainte de pre-procesare), în biți, ca întreg
întreg pe 64 de biți

    // procesați mesajul în blocuri succesive de 512 de biți:
    // spargeți mesajul în bucăți de 512 biți

    for each bloc
        spargeți blocul în șaisprezece cuvinte pe 32 de biți w [i], 0 ≤ i ≤ 15

        // Extindeți cele 16 cuvinte pe 32 de biți în 80 de cuvinte pe 32 biți:
        for i from 16 to 79
            w[i] = (w[i-3] xor w[i-8] xor w[i-14] xor w[i-16]) leftrotate 1
```

```
        // Inițializați valoarea hash pentru acest bloc:
        a = h0
        b = h1
        c = h2
        d = h3
        e = h4

        // bucla principală:
        for i from 0 to 79
            if 0 ≤ i ≤ 19 then
                f = (b and c) or ((not b) and d)
                k = 0x5A827999
            else if 20 ≤ i ≤ 39
                f = b xor c xor d
                k = 0x6ED9EBA1
            else if 40 ≤ i ≤ 59
                f = (b and c) or (b and d) or (c and d)
                k = 0x8F1BBCDC
            else if 60 ≤ i ≤ 79
                f = b xor c xor d
                k = 0xCA62C1D6

            temp = (a leftrotate 5) + f + e + k + w[i]
            e = d
            d = c
            c = b leftrotate 30
            b = a
            a = temp

        // adaugă hash-ul acestui bloc la resultatul curent:
        h0 = h0 + a
        h1 = h1 + b
        h2 = h2 + c
        h3 = h3 + d
        h4 = h4 + e

    // obțineți valoarea de hash finală (big-endian):
    digest = hash = h0 append h1 append h2 append h3 append h4
```

5.10 un mesaj simplu și valoarea sa de hash

Mesajul este codul binar ASCII al șirului "abc", i.e.,

01100001 01100010 01100011.

Acest mesaj are lungimea l = 24. În pasul (a) din secțiunea 4, adăugăm "1". În pasul (b) adăugăm 423 "0" s. În pasul (c) se anexează hex 00000000 00000018, reprezentarea pe două cuvinte a lui 24. Astfel, mesajul final completat (căptușit)constă dintr-un bloc, astfel încât n = 1 în notația secțiunii 4.

Valorile inițiale ale regiștrilor $\{H_i\}$ în hexa sunt:

H_0 = 67452301
H_1 = EFCDAB89
H_2 = 98BADCFE
H_3 = 10325476

H_4 = C3D2E1F0.

Începem procesarea blocului 1. Cuvintele blocului sunt:

```
W[0] = 61626380
W[1] = 00000000
W[2] = 00000000
W[3] = 00000000
W[4] = 00000000
W[5] = 00000000
W[6] = 00000000
W[7] = 00000000
W[8] = 00000000
W[9] = 00000000
W[10] = 00000000
W[11] = 00000000
W[12] = 00000000
W[13] = 00000000
W[14] = 00000000
W[15] = 00000018.
```

Valorile hexa ale lui A, B, C, D, E după pasul t al buclei "pentru t = 0 la 79" (etapa (d) din secțiunea 7 sau etapa (c) din secțiunea 8) sunt:

```
            A          B          C          D          E

t =  0: 0116FC33   67452301   7BF36AE2   98BADCFE   10325476
t =  1: 8990536D   0116FC33   59D148C0   7BF36AE2   98BADCFE
t =  2: A1390F08   8990536D   C045BF0C   59D148C0   7BF36AE2
t =  3: CDD8E11B   A1390F08   626414DB   C045BF0C   59D148C0
t =  4: CFD499DE   CDD8E11B   284E43C2   626414DB   C045BF0C
t =  5: 3FC7CA40   CFD499DE   F3763846   284E43C2   626414DB
t =  6: 993E30C1   3FC7CA40   B3F52677   F3763846   284E43C2
t =  7: 9E8C07D4   993E30C1   0FF1F290   B3F52677   F3763846
t =  8: 4B6AE328   9E8C07D4   664F8C30   0FF1F290   B3F52677
t =  9: 8351F929   4B6AE328   27A301F5   664F8C30   0FF1F290
t = 10: FBDA9E89   8351F929   12DAB8CA   27A301F5   664F8C30
t = 11: 63188FE4   FBDA9E89   60D47E4A   12DAB8CA   27A301F5
t = 12: 4607B664   63188FE4   7EF6A7A2   60D47E4A   12DAB8CA
t = 13: 9128F695   4607B664   18C623F9   7EF6A7A2   60D47E4A
t = 14: 196BEE77   9128F695   1181ED99   18C623F9   7EF6A7A2
t = 15: 20BDD62F   196BEE77   644A3DA5   1181ED99   18C623F9
t = 16: 4E925823   20BDD62F   C65AFB9D   644A3DA5   1181ED99
t = 17: 82AA6728   4E925823   C82F758B   C65AFB9D   644A3DA5
t = 18: DC64901D   82AA6728   D3A49608   C82F758B   C65AFB9D
t = 19: FD9E1D7D   DC64901D   20AA99CA   D3A49608   C82F758B
t = 20: 1A37B0CA   FD9E1D7D   77192407   20AA99CA   D3A49608
t = 21: 33A23BFC   1A37B0CA   7F67875F   77192407   20AA99CA
t = 22: 21283486   33A23BFC   868DEC32   7F67875F   77192407
t = 23: D541F12D   21283486   0CE88EFF   868DEC32   7F67875F
t = 24: C7567DC6   D541F12D   884A0D21   0CE88EFF   868DEC32
t = 25: 48413BA4   C7567DC6   75507C4B   884A0D21   0CE88EFF
t = 26: BE35FBD5   48413BA4   B1D59F71   75507C4B   884A0D21
t = 27: 4AA84D97   BE35FBD5   12104EE9   B1D59F71   75507C4B
t = 28: 8370B52E   4AA84D97   6F8D7EF5   12104EE9   B1D59F71
t = 29: C5FBAF5D   8370B52E   D2AA1365   6F8D7EF5   12104EE9
t = 30: 1267B407   C5FBAF5D   A0DC2D4B   D2AA1365   6F8D7EF5
t = 31: 3B845D33   1267B407   717EEBD7   A0DC2D4B   D2AA1365
t = 32: 046FAA0A   3B845D33   C499ED01   717EEBD7   A0DC2D4B
```

```
t = 33: 2C0EBC11    046FAA0A    CEE1174C    C499ED01    717EEBD7
t = 34: 21796AD4    2C0EBC11    811BEA82    CEE1174C    C499ED01
t = 35: DCBBB0CB    21796AD4    4B03AF04    811BEA82    CEE1174C
t = 36: 0F511FD8    DCBBB0CB    085E5AB5    4B03AF04    811BEA82
t = 37: DC63973F    0F511FD8    F72EEC32    085E5AB5    4B03AF04
t = 38: 4C986405    DC63973F    03D447F6    F72EEC32    085E5AB5
t = 39: 32DE1CBA    4C986405    F718E5CF    03D447F6    F72EEC32
t = 40: FC87DEDF    32DE1CBA    53261901    F718E5CF    03D447F6
t = 41: 970A0D5C    FC87DEDF    8CB7872E    53261901    F718E5CF
t = 42: 7F193DC5    970A0D5C    FF21F7B7    8CB7872E    53261901
t = 43: EE1B1AAF    7F193DC5    25C28357    FF21F7B7    8CB7872E
t = 44: 40F28E09    EE1B1AAF    5FC64F71    25C28357    FF21F7B7
t = 45: 1C51E1F2    40F28E09    FB86C6AB    5FC64F71    25C28357
t = 46: A01B846C    1C51E1F2    503CA382    FB86C6AB    5FC64F71
t = 47: BEAD02CA    A01B846C    8714787C    503CA382    FB86C6AB
t = 48: BAF39337    BEAD02CA    2806E11B    8714787C    503CA382
t = 49: 120731C5    BAF39337    AFAB40B2    2806E11B    8714787C
t = 50: 641DB2CE    120731C5    EEBCE4CD    AFAB40B2    2806E11B
t = 51: 3847AD66    641DB2CE    4481CC71    EEBCE4CD    AFAB40B2
t = 52: E490436D    3847AD66    99076CB3    4481CC71    EEBCE4CD
t = 53: 27E9F1D8    E490436D    8E11EB59    99076CB3    4481CC71
t = 54: 7B71F76D    27E9F1D8    792410DB    8E11EB59    99076CB3
t = 55: 5E6456AF    7B71F76D    09FA7C76    792410DB    8E11EB59
t = 56: C846093F    5E6456AF    5EDC7DDB    09FA7C76    792410DB
t = 57: D262FF50    C846093F    D79915AB    5EDC7DDB    09FA7C76
t = 58: 09D785FD    D262FF50    F211824F    D79915AB    5EDC7DDB
t = 59: 3F52DE5A    09D785FD    3498BFD4    F211824F    D79915AB
t = 60: D756C147    3F52DE5A    4275E17F    3498BFD4    F211824F
t = 61: 548C9CB2    D756C147    8FD4B796    4275E17F    3498BFD4
t = 62: B66C020B    548C9CB2    F5D5B051    8FD4B796    4275E17F
t = 63: 6B61C9E1    B66C020B    9523272C    F5D5B051    8FD4B796
t = 64: 19DFA7AC    6B61C9E1    ED9B0082    9523272C    F5D5B051
t = 65: 101655F9    19DFA7AC    5AD87278    ED9B0082    9523272C
t = 66: 0C3DF2B4    101655F9    0677E9EB    5AD87278    ED9B0082
t = 67: 78DD4D2B    0C3DF2B4    4405957E    0677E9EB    5AD87278
t = 68: 497093C0    78DD4D2B    030F7CAD    4405957E    0677E9EB
t = 69: 3F2588C2    497093C0    DE37534A    030F7CAD    4405957E
t = 70: C199F8C7    3F2588C2    125C24F0    DE37534A    030F7CAD
t = 71: 39859DE7    C199F8C7    8FC96230    125C24F0    DE37534A
t = 72: EDB42DE4    39859DE7    F0667E31    8FC96230    125C24F0
t = 73: 11793F6F    EDB42DE4    CE616779    F0667E31    8FC96230
t = 74: 5EE76897    11793F6F    3B6D0B79    CE616779    F0667E31
t = 75: 63F7DAB7    5EE76897    C45E4FDB    3B6D0B79    CE616779
t = 76: A079B7D9    63F7DAB7    D7B9DA25    C45E4FDB    3B6D0B79
t = 77: 860D21CC    A079B7D9    D8FDF6AD    D7B9DA25    C45E4FDB
t = 78: 5738D5E1    860D21CC    681E6DF6    D8FDF6AD    D7B9DA25
t = 79: 42541B35    5738D5E1    21834873    681E6DF6    D8FDF6AD.
```

Blocul 1 a fost procesat. Valorile $\{H_i\}$ sunt:

H_0 = 67452301 + 42541B35 = A9993E36
H_1 = EFCDAB89 + 5738D5E1 = 4706816A
H_2 = 98BADCFE + 21834873 = BA3E2571
H_3 = 10325476 + 681E6DF6 = 7850C26C
H_4 = C3D2E1F0 + D8FDF6AD = 9CD0D89D.

Hashul mesajului = A9993E36 4706816A BA3E2571 7850C26C 9CD0D89D

5.11 familia de funcții burete Keccak pentru SHA-3

Prezentarea acestei funțcii este conformă cu [KSF], scris de Gudo Bertoni, Joan Daemen, Michael Peeters și Gilles Van Assche.

Keccak (pronunțat [kɛtʃak], ca și "ketchak") este o familie de funcții hash care a fost adoptată ca algoritmul hash al NIST pentru SHA-3. Textul de mai jos este o descriere rapidă a lui Keccak folosind pseudo-cod. Nimic nu ar trebui ca acest text introductiv să fie considerat o descriere oficială și de referință a lui Keccak. În schimb, obiectivul este de a prezenta Keccak cu accent pe lizibilitate și claritate.

5.11.1 structura algoritmului Keccak

Keccak este o familie de funcții hash care se bazează pe construcția de tip burete și, prin urmare, este o familie de funcții de tip burete. În Keccak, funcția de bază este o permutare aleasă într-un set de șapte permutări Keccak-f, denumită Keccak-f[b], unde b ∈ {25, 50, 100, 200, 400, 800, 1600} este mărimea permutării. Mărimea permutării este, de asemenea, mărimea stării în construcția buretelui.

Starea este organizat ca o serie de 5 x 5 benzi, fiecare având lungimea w ∈ {1, 2, 4, 8, 16, 32, 64} (b = 25w). Când este implementată pe un procesor pe 64 de biți, o bandă de Keccak-f[1600] poate fi reprezentată ca un cuvânt CPU pe 64 de biți.

Obținem funcția de burete Keccak[r, c], cu parametrii de capacitate c și rata de biți r, dacă aplicăm construcția de burete la Keccak-f[r + c] și aplicăm o completare (umplere) specifică mesajului de intrare.

5.11.2 descrierea pseudo-codului

În primul rând începem cu descrierea lui Keccak-f în pseudo-codul de mai jos. Numărul de runde nr depinde de lățimea de permutare și este dat de nr = 12 + 2l, unde 2l = w. Acest lucru dă 24 de runde pentru Keccak-f [1600].

```
Keccak-f[b](A) {
  forall i in 0…nr-1
    A = Round[b](A, RC[i])
  return A
}


Round[b](A,RC) {
  // θ step
  C[x] = A[x,0] xor A[x,1] xor A[x,2] xor A[x,3] xor A[x,4], forall x in 0…4
  D[x] = C[x-1] xor rot(C[x+1],1), forall x in 0…4
  A[x,y] = A[x,y] xor D[x], forall (x,y) in (0…4,0…4)

  // ρ and π steps
  B[y,2*x+3*y] = rot(A[x,y], r[x,y]), forall (x,y) in (0…4,0…4)

  // χ step
  A[x,y] = B[x,y] xor ((not B[x+1,y]) and B[x+2,y]), forall (x,y) in (0…4,0…
4)

  // ι step
  A[0,0] = A[0,0] xor RC

  return A
}
```

În pseudocodul de mai sus, se folosesc următoarele convenții. Toate operațiunile pe indici se fac modulo 5. A denotă matricea completă de stare a permutării, iar A [x, y] denotă o banda specială în acea stare. B [x,

y], C [x], D [x] sunt variabile intermediare. Constantele r [x, y] sunt indecșii de rotație (vezi tabelul 2), în timp ce RC [i] sunt constantele rundei (vezi tabelul 1). rot (W, r) este operația obișnuită de șift circular la biților, deplasând un bit din poziția i în poziția i + r (modulo dimensiunea benzii).

Apoi, prezentăm pseudocodul pentru funcția burete Keccak[r, c], cu parametrii de capacitate c și rata biților r. Descrierea de mai jos este limitată la cazul mesajelor care acoperă un număr întreg de octeți. Pentru mesajele cu un număr de biți care nu pot fi divizate cu 8, consultați specificațiile pentru mai multe detalii. De asemenea, presupunem pentru simplitate că r este un multiplu al mărimii benzii; acesta este cazul pentru parametrii candidați SHA-3.

```
Keccak[r,c](M) {
  // Inițializare și umplere
  S[x,y] = 0, forall (x,y) in (0…4,0…4)
  P = M || 0x01 || 0x00 || … || 0x00
  P = P xor (0x00 || … || 0x00 || 0x80)

  // faza de absorbție
  forall block Pi in P
    S[x,y] = S[x,y] xor Pi[x+5*y], forall (x,y) such that x+5*y < r/w
    S = Keccak-f[r+c](S)

  // faza de presarew
  Z = empty string
  while output is requested
    Z = Z || S[x,y], forall (x,y) such that x+5*y < r/w
    S = Keccak-f[r+c](S)

  return Z
}
```

În pseudo-codul de mai sus, S reprezintă starea ca o serie de benzi. Mesajul căptușit (completat) P este organizat ca o serie de blocuri P_i, ele însele fiind organizate ca o serie de benzi. Operatorul || denotă concatenarea obișnuită a șirurilor de octeți.

chapter 6 sisteme de criptare cu cheie simetrică

6.1 generalități

Originile DES datează de la începutul anilor 1970. În 1972, după încheierea unui studiu privind necesitățile de securitate informatică ale guvernului SUA, organismul american de standardizare NBS (denumit acum NIST) a identificat necesitatea unui standard guvernamental de criptare a informațiilor neclasigeneralități

6.2 digital encryption standard (DES)

Originile DES datează de la începutul anilor 1970. În 1972, după încheierea unui studiu privind necesitățile de securitate informatică ale guvernului SUA, organismul american de standardizare NBS (denumit acum NIST) a identificat necesitatea unui standard guvernamental de criptare a informațiilor neclasificate, sensibile. În consecință, la 15 mai 1973, după consultarea cu NSA, NBS a solicitat propuneri pentru un cifru care să îndeplinească criterii riguroase de proiectare. Niciuna dintre propunerile trimise nu a fost consederată potrivită. O a doua solicitare a fost emisă la 27 august 1974. De data aceasta, IBM a prezentat un candidat care a fost considerat acceptabil - un cifru dezvoltat în perioada 1973-1974 bazat pe un algoritm anterior, cifrul Lucifer al lui Horst Feistel. Echipa de la IBM implicată în proiectarea și analiza cifrului i-a inclus pe Feistel, Walter Tuchman, Don Coppersmith, Alan Konheim, Carl Meyer, Mike Matyas, Roy Adler, Edna Grossman, Bill Notz, Lynn Smith și Bryant Tuckerman.

6.2.1 implicarea NSA în design

La 17 martie 1975, DES propus a fost publicat în Registrul federal. Au fost solicitate observații publice, iar în anul următor au avut loc două ateliere deschise pentru a discuta standardul propus. Au existat unele critici din partea unor participanți, inclusiv de la pionierii criptografiei publice Martin Hellman si Whitfield Diffie, specificând lungimea scurtă a cheiei și misterioasele "S-boxes" ca dovadă a interferențelor incorecte din partea NSA. Suspiciunea era că algoritmul a fost slăbit în secret de către agenția de informații, astfel încât ei - dar nimeni altcineva - să citească cu ușurință mesajele criptate. Alan Konheim (unul dintre designerii DES) a comentat: "Am trimis boxele S la Washington, s-au întors și au fost cu totul altfel". Comitetul selectiv al Senatului Statelor Unite ale Americii privind serviciile de informații a revizuit acțiunile NSA pentru a determina dacă a existat o implicare necorespunzătoare. În rezumatul neclasificat al concluziilor lor, publicat în 1978, Comitetul a scris:

> *"În dezvoltarea DES, NSA a convins IBM că o dimensiune redusă a cheiei este suficientă, a asistat indirect la dezvoltarea structurilor S-box și a certificat că algoritmul final DES era, după cunoștințele sale, liber de orice slabiciune statistică sau matematica. "*

Cu toate acestea, a constatat, de asemenea, că:

> *"NSA nu a manipulat în nici un fel proiectarea algoritmului. IBM a inventat și a proiectat algoritmul, a luat toate deciziile pertinente cu privire la acesta și a fost de acord că dimensiunea-cheie convenită este mai mult decât adecvată pentru toate aplicațiile comerciale pentru care DES este intenționat.“*

Un alt membru al echipei DES, Walter Tuchman, este citat spunând: "Am dezvoltat algoritmul DES în întregime în IBM folosind IBMers. NSA nu a dictat un singur fir!" În schimb, o carte desecretizată a NSA privind istoria criptologică afirmă:

> *"În 1973, NBS a solicitat industriei private un standard de criptare a datelor (DES), iar primele oferte au fost dezamăgitoare, așa că NSA a început să lucreze la algoritmul său propriu", a declarat Howard Rosenblum, director adjunct pentru cercetare și inginerie, că Walter Tuchman de la IBM pe o modificare a lui Lucifer pentru uz general. NSA ia dat lui Tuchman o autorizație și la adus să lucreze împreună cu Agenția pentru modificarea lui Lucifer.*

Unele suspiciuni cu privire la slăbiciunile ascunse în cutiile S au fost eliminate în 1990, cu descoperirea independentă și publicarea deschisă de Eli Biham și Adi Shamir a criptanalizei diferențiale, o metodă generală de spargere a cifrurilor de tip bloc. Caseta S a DES a fost mult mai rezistente la atac decât dacă ar fi fost alese la întâmplare, sugerând puternic că IBM a știut despre tehnică din 1970. Acesta a fost într-adevăr cazul - în 1994, Don Coppersmith a publicat unele dintre criteriile originale de proiectare pentru casetele S. Potrivit lui Steven Levy, cercetătorii IBM Watson au descoperit atacuri criptanalitice diferențiate în 1974 și au fost rugați de către NSA să păstreze secretul tehnicii. Coppersmith explică decizia secretă a IBM prin a spune: "Aceasta a fost pentru că [criptanaliza diferențială] poate fi un instrument foarte puternic, folosit împotriva multor scheme, și există îngrijorare că astfel de informații în domeniul public ar putea afecta negativ securitatea națională". Levy citează pe Walter Tuchman: "Ne-au cerut să declarăm toate documentele confidențiale ... Am pus de fapt un număr pe fiecare și le-am încuiat în seifuri, pentru că erau considerate secrete ale guvernului american.“

6.2.2 algoritmul ca standard

În ciuda criticilor, DES a fost aprobat ca standard federal în noiembrie 1976 și publicat la 15 ianuarie 1977 ca FIPS PUB 46, fiind autorizat pentru utilizare cu toate datele neclasificate. A fost ulterior reafirmat ca standard în 1983, 1988 (revizuit ca FIPS-46-1), 1993 (FIPS-46-2) și din nou în 1999 (FIPS-46-3), acesta din urmă descriind "Triple DES"). La 26 mai 2002, DES a fost în cele din urmă înlocuită de Standardul de criptare avansată (AES), în urma unei competiții publice. La 19 mai 2005, FIPS 46-3 a fost retras oficial, însă NIST a aprobat Triple DES până în anul 2030 pentru informații guvernamentale sensibile.

Algoritmul este de asemenea specificat în ANSI X3.92, NIST SP 800-67 și ISO / IEC 18033-3 (ca o componentă a TDEA).

Un alt atac teoretic, criptanaliza liniară, a fost publicat în 1994, dar a fost un atac de forță brută în 1998 care a demonstrat că DES ar putea fi atacat foarte practic și a subliniat necesitatea unui algoritm de înlocuire.

6.3 descriere

DES este cifrul blocului arhetipal - un algoritm care ia un șir de lungime fixă de biți plaintext și îl transformă printr-o serie de operații complicate într-un alt bitstring de tip ciphertext de aceeași lungime. În cazul DES, dimensiunea blocului este de 64 de biți. DES utilizează, de asemenea, o cheie pentru personalizarea transformării, astfel încât decriptarea se presupune că este realizată numai de cei care cunosc cheia specială folosită pentru a cripta. Cheia aparent constă în 64 de biți; totuși, doar 56 dintre acestea sunt de fapt folosite de algoritm. Opt biți sunt utilizați exclusiv pentru verificarea parității și apoi sunt eliminați. Prin urmare, lungimea efectivă a cheii este de 56 de biți și este de obicei citată ca atare.

Ca și alte cipuri de blocuri, DES nu este un mijloc securizat de criptare, ci trebuie folosit într-un mod de operare. FIPS-81 specifică mai multe moduri de utilizare cu DES. Comentarii suplimentare privind utilizarea DES sunt cuprinse în FIPS-74.

Structura generală a algoritmului este prezentată în Figura 1: există 16 etape identice de prelucrare, numite runde. Există, de asemenea, o permutare inițială și finală, denumită IP și FP, care sunt inverse (IP "anulează" acțiunea FP și invers). IP și FP nu au aproape nici o semnificație criptografică, dar au fost aparent incluse pentru a facilita blocurile de încărcare în și din hardware-ul din anii 1970, precum și pentru a face DES să funcționeze mai lent în software-ul.

Înainte de rundele principale, blocul este împărțit în două jumătăți de 32 de biți și procesat alternativ; această intersecție este cunoscută sub numele de schema Feistel. Structura Feistel asigură că decriptarea și criptarea sunt procese foarte asemănătoare - singura diferență este că subcheiile sunt aplicate în ordine inversă atunci când se decriptează. Restul algoritmului este identic. Acest lucru simplifică foarte mult implementarea, în special în domeniul hardware, deoarece nu este nevoie de algoritmi separați de criptare și decriptare.

Simbolul ⊕ desemnează operațiunea exclusiv-OR (XOR). Funcția F combină jumătate de bloc împreună cu o parte din cheie. Ieșirea din funcția F este apoi combinată cu cealaltă jumătate a blocului, iar jumătățile sunt schimbate înainte de următoarea rundă. După runda finală, jumătățile nu sunt schimbate; aceasta este o caracteristică a structurii Feistel care face similare procesele de criptare și decriptare.

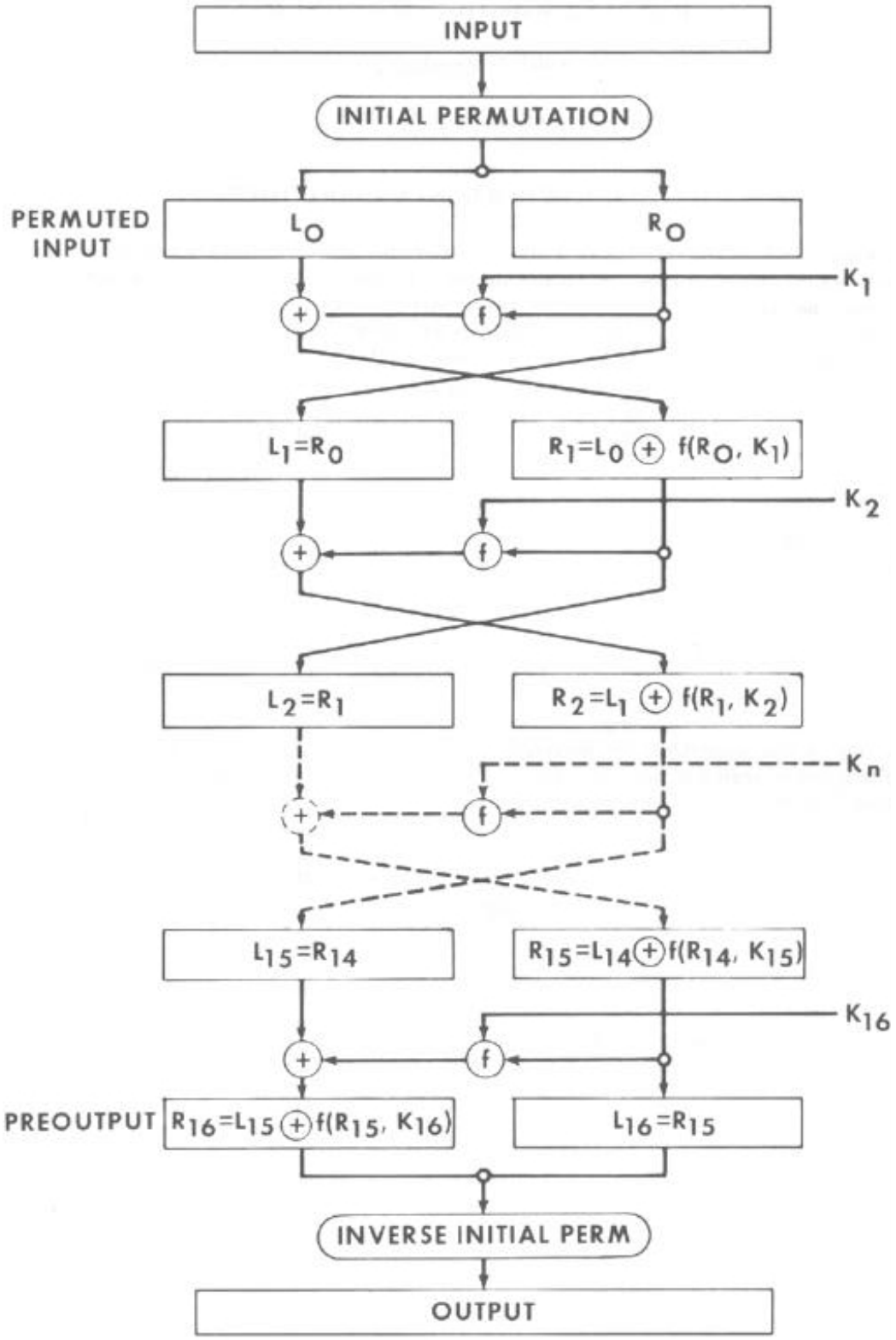


Figura 1 – Structura generală a algoritmului DES

6.4 funcția Feistel (F)

Funcția F, prezentată în figura 2, funcționează pe o jumătate de bloc (32 biți) la un moment dat și constă din patru etape:

1. Extindere - jumătatea de bloc pe 32 de biți este extinsă la 48 de biți folosind permutarea de expansiune, denumită E în diagramă, duplicând jumătate din biți. Ieșirea este compusă din 8 bucăți de 6 biți, fiecare conținând o copie de 4 biți de intrare corespunzători, plus o copie a bitului imediat adiacent din fiecare piesă de intrare în fiecare parte.

2. Mixarea cheii - rezultatul este combinat cu o subcheie folosind o operație XOR. Șasezeci de sub-chei de 48 de biți - câte una pentru fiecare rundă - sunt derivate din cheia principală folosind programul cheilor (descris mai jos).

3. Înlocuire - după amestecare în subcheie, blocul este împărțit în opt bucăți de 6 biți înainte de prelucrare de către cutiile S sau casetele de substituție. Fiecare dintre cele opt casete S înlocuiește cei șase biți de intrare cu patru biți de ieșire, în conformitate cu o transformare neliniară, furnizată sub forma unei tabele de căutare. Casetele S furnizează nucleul securității DES - fără ele, cifrul ar fi liniar și trivial de spart.

4. Permutarea - în cele din urmă, cele 32 de ieșiri din casetele S sunt rearanjate în funcție de o permutare fixă, P. Aceasta este proiectată astfel încât, după expansiune, fiecare bit de ieșire al lui S-box să fie împărțit în 6 casete S diferite în runda următoare.

Alternarea substituției din cutiile S și permutarea biților din cutia P și E-expansiune oferă așa-numita "confuzie și difuzie" respectiv, un concept identificat de Claude Shannon în anii 1940 drept o condiție necesară pentru un cifru securizat și practic.

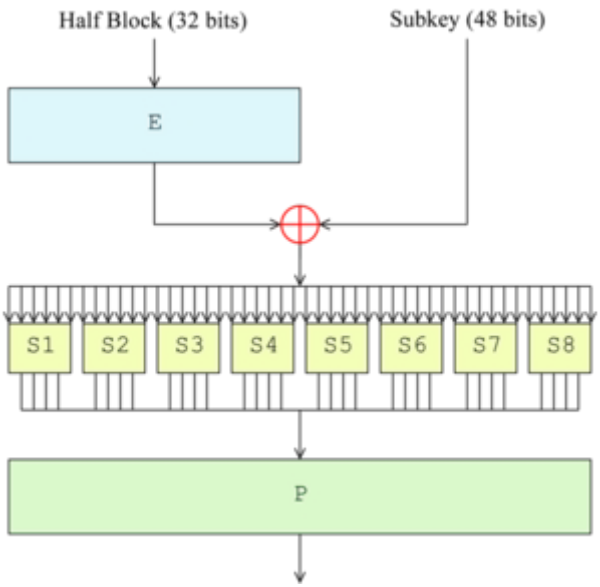


Figura 2 – funcția Feistel

6.5 programul cheilor

Figura 3 ilustrează programul cheilor pentru criptare - algoritmul care generează subcheile. Inițial, 56 de biți ai cheii sunt selectați din cei 64 inițiali cu Permuted Choice 1 (PC-1) - restul de 8 biți sunt fie eliminați, fie utilizați ca biți de verificare a parității. Cei 56 de biți sunt apoi împărțiți în două jumătăți de 28 de biți; fiecare jumătate este apoi tratată separat. În runde succesive, ambele jumătăți sunt rotite la stânga cu unul sau doi

biți (specificați pentru fiecare rundă) și apoi 48 biți de subcheie sunt selectați de Permuted Choice 2 (PC-2) - 24 de biți din jumătatea stângă și 24 de la dreapta . Rotațiile (indicate prin "<<<" în diagramă) înseamnă că un set diferit de biți este utilizat în fiecare subcheie; fiecare bit este utilizat în aproximativ 14 din cele 16 subchei.

Programul cheilor pentru decriptare este similar - subcheile sunt în ordine inversă față de criptare. În afară de această schimbare, procesul este același ca și pentru criptare.

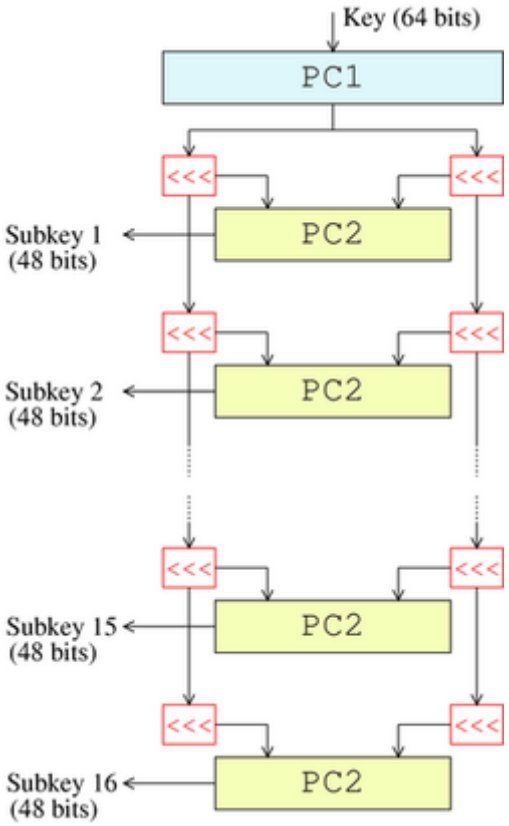


Figura 3

6.6 pașii algoritmului DES

Pozițiile celor 64 de biți ai blocului de intrare ce urmează a fi criptat sunt permutate, conform permutării inițiale ***IP***:

IP

58 50 42 34 26 18 10 2

60 52 44 36 28 20 12 4

62 54 46 38 30 22 14 6

64 56 48 40 32 24 16 8

57 49 41 33 25 17 9 1

59 51 43 35 27 19 11 3

61 53 45 37 29 21 13 5

63 55 47 39 31 23 15 7

Aceasta înseamnă bitul 58 din blocul de intrare va fi mutat în poziția 1, bitul din poziția 50 va fi mutat în poziția 2 iar bitul din poziția 7 va ajunge pe ultima poziție. The permuted input block is then the input to a complex key-dependent computation described below. The output of that computation, called the preoutput, is then subjected to the following permutation which is the inverse of the initial permutation:

IP^{-1}

40 8 48 16 56 24 64 32

39 7 47 15 55 23 63 31

38 6 46 14 54 22 62 30

37 5 45 13 53 21 61 29

36 4 44 12 52 20 60 28

35 3 43 11 51 19 59 27

34 2 42 10 50 18 58 26

33 1 41 9 49 17 57 25

Cu alte cuvinte, în urma aplicării algoritmului, bitul 40 al bocului de intrare devine primul bit în blocul de ieșire, bitul 8 devine al doilea bit din blocul de ieșire, etc., până ce bitul 25 din blocul de intrare devine ultimul bit din blocul de ieșire.

Calculul care folosește blocul de intrare permutat ca intrare pentru a produce blocul de pre-ieșire constă, cu excepția unui schimb final de blocuri, din 16 iterații ale calculului descris mai jos prin funcția de cifrare ***f*** care acționează pe două blocuri, unul de 32 de biți și unul de 48 de biți și produce un bloc de 32 de biți.

Să considerăm cei 64 de biți ai blocului de intrare la o iterație să constea dintr-un bloc L de 32 biți urmat de un bloc de 32 biți R. Folosind notația definită în introducere, blocul de intrare este atunci ***LR***. Fie ***K*** un bloc de 48 de biți ales din cheia de 64 de biți. Atunci ieșirea ***L'R'*** a unei iterații cu intrarea ***LR*** este definită prin:

(1) ***L'*** = ***R***

R' = ***L ^ f(R,K)***

unde ^ denotă adunarea bit-cu-bit modulo 2.

După cum am specificat mai înainte, intrarea pentru prima iterație a calculului este blocul de intrare permutat. Dacă ***L'R'*** este ieșirea celei de-a 16 iterații, atunci ***R'L'*** este blocul de pre-ieșire. La fiecare iterație un bloc diferit ***K*** de biți ai cheii este ales din cheia de 64 de biți desemnată de ***KEY***.

Utilizând notații noi putem descrie iterațiile calculului cu mai multe detalii. Fie ***KS*** funcția care are ca intrare un întreg *n* din domeniul 1 la 16 precum și un bloc ***KEY*** de 64 de biți iar ca ieșire are un bloc ***Kn*** de 48 de biți care este o selecție permutată a biților din ***KEY***. Cu alte cuvinte

(2) ***Kn*** = ***KS(n,KEY)***

unde ***Kn*** este determinat de 48 de biți aflați în poziții distincte din ***KEY***. ***KS*** este numit programul de chei (key schedule) deoarece blocul ***K*** utilizat în iterația *n* din (1) este blocul ***Kn*** determinat în (2).

Ca și mai înainte, notăm cu ***LR*** blocul de intrare permutat. Fie ***L()*** și ***R()*** pe post de ***L*** and ***R*** iar ***Ln*** și ***Rn*** pe post de ***L'*** și ***R'*** în (1) atunci când ***L*** și ***R*** sunt respectiv ***Ln-1*** și ***Rn-1*** iar ***K*** este ***Kn***; adică atunci când *n* este în domeniul 1 - 16,

(3) ***Ln*** = ***Rn-1***

Rn = ***Ln-1 ^ f(Rn-1,Kn)***

Blocul de pre-ieșire este atunci $R_{16}L_{16}$.

Programul cheilor produce cei 16 ***Kn*** care sunt necesari algoritmului.

6.7 un exemplu

6.8 securitate și criptanaliză

Deși au fost publicate mai multe informații despre criptanaliza DES decât despre alt cifru de tip bloc, atacul cel mai practic până în prezent continuă să fie forța brută. Sunt cunoscute diverse proprietăți criptanalitice minore și sunt posibile trei atacuri teoretice care, deși au o complexitate teoretică mai mică decât un atac de forță brută, necesită o cantitate nerealistă de text sub formă de text cunoscut sau ales pentru a fi efectuat și nu reprezintă o problemă în practică.

6.8.1 atacurile de forță brută

Pentru orice cifru, cea mai de bază metodă de atac este forța brută - încercând pe rând toate cheile posibile. Lungimea cheii determină numărul de chei posibile și, prin urmare, fezabilitatea acestei abordări. În ceea ce privește DES, s-au ridicat întrebări cu privire la adecvarea dimensiunii cheii sale, încă înainte de a fi adoptată ca standard și a fost dimensiunea mică a cheii, mai degrabă decât criptanaliza teoretică, care a dictat necesitatea unui algoritm de înlocuire. Ca urmare a discuțiilor care au implicat consultanți externi, inclusiv NSA, dimensiunea cheii a fost redusă de la 128 biți la 56 biți pentru a se potrivi pe un singur cip.

Mașina DES Cracker de 250.000 USD a EFF conținea 1.856 de jetoane personalizate și putea forța o forță cheie DES în câteva zile.

În mediul academic, au fost avansate diverse propuneri pentru o mașină de spargere a DES. În 1977, Diffie și Hellman au propus o mașină care costă aproximativ 20 milioane USD, care ar putea găsi o cheie DES într-o singură zi. Până în 1993, Wiener a propus o mașină de căutare cu cheie cu un milion de dolari SUA, care va găsi o cheie în 7 ore. Cu toate acestea, niciuna dintre aceste propuneri timpurii nu a fost vreodată pusă în aplicare - sau, cel puțin, nicio implementare nu a fost recunoscută public. Vulnerabilitatea DES a fost practic demonstrată la sfârșitul anilor '90. În 1997, RSA Security a sponsorizat o serie de concursuri, oferind un premiu de 10.000 USD primei echipe care a spart un mesaj criptat cu DES pentru concurs. Acest concurs a fost câștigat de proiectul DESCHALL, condus de Rocke Verser, Matt Curtin și Justin Dolske, folosind ciclurile inactive ale mii de computere de pe Internet. Fezabilitatea spargerii DES a fost rapid demonstrată în 1998, când a fost construit un cracker DES personalizat de către Electronic Frontier Foundation (EFF), un grup de drepturi civile din spațiul ciberspațial, cu un cost de aproximativ 250.000 USD (a se vedea crackerul EFF DES). Motivația lor a fost aceea de a arăta că DES poate fi spart atât în practică, cât și în teorie: "Există multe persoane care nu vor crede un adevăr până nu îl vor vedea cu propriii ochi. Arătându-le o mașină fizică care poate sparge DES în câteva zilele este singura modalitate de a convinge unii oameni că nu pot avea încredere în securitatea lor pentru DES. " Mașina a găsit o cheie prin forță brută într-un timp puțin mai mare de 2 zile.

Mașina COPACOBANA, construită în 2006 pentru 10.000 USD de către universitățile din Bochum și Kiel, Germania, conține 120 de FPGA low-cost și ar putea efectua o căutare cheie exhaustivă pe DES în 9 zile în medie.

Singurul alt cracker DES confirmat a fost aparatul COPACOBANA construit în 2006 de echipele universităților din Bochum și Kiel, ambele din Germania. Spre deosebire de aparatul EFF, COPACOBANA constă din circuite integrate, reconfigurabile disponibile comercial. 120 din aceste matrice de canale programabile Field (FPGA) de tip XILINX Spartan3-1000 rulează în paralel. Sunt grupate în 20 de module DIMM, fiecare conținând 6 FPGA. Utilizarea hardware-ului reconfigurabil face ca aparatul să se aplice și altor sarcini de rupere a codului. Unul dintre aspectele mai interesante ale COPACOBANA este factorul său de cost. O mașină poate fi construită pentru aproximativ 10.000 USD. Reducerea costurilor cu aproximativ un factor de 25 pentru mașina EFF este un exemplu impresionant pentru îmbunătățirea continuă a hardware-ului digital. Ajustarea inflației pe 8 ani produce o îmbunătățire și mai mare de aproximativ 30x. Începând cu 2007, SciEngines GmbH, o companie spin-off a celor doi parteneri de proiect ai COPACOBANA, a consolidat și dezvoltat succesorii COPACOBANA. În 2008, COPACOBANA RIVYERA a redus timpul pentru a sparge DES la mai puțin de o zi, folosind 128 Spartan-3 5000. [DESW]

6.8.2 atacuri mai rapide decât forța brută

Se cunosc trei atacuri care pot sparge cele șaisprezece runde ale DES cu mai puțină complexitate decât o căutare cu forța brută: criptanaliza diferențială (DC), criptanaliza liniară (LC) și atacul lui Davies. Cu toate acestea, atacurile sunt teoretice și sunt imposibil de montat în practică; aceste tipuri de atac sunt uneori denumite puncte slabe certificaționale.

• Analiza criptografică diferențială a fost redescoperită la sfârșitul anilor 1980 de către Eli Biham și Adi Shamir; aceasta a fost cunoscută mai devreme atât de IBM, cât și de NSA și păstrată secretă. Pentru a sparge cele 16 runde complete, criptanaliza diferențială necesită 247 de texte alese. DES a fost proiectat să fie rezistentă la DC.

• Criptanaliza liniară a fost descoperită de Mitsuru Matsui și are nevoie de 243 de texte cunoscute; metoda a fost implementată și a fost prima criptanaliză experimentală a DES care a fost raportată. Nu există dovezi că DES a fost adaptat pentru a fi rezistent la acest tip de atac. În 1994 a fost sugerată o generalizare a LC - criptanaliză liniară multiplă și a fost rafinată în continuare de Biryukov și colab. În 2004; Analiza lor sugerează că mai multe aproximări liniare ar putea fi utilizate pentru a reduce cerințele de date ale atacului cu cel puțin un factor de 4 (adică 2^41 în loc de 2^43). O reducere similară a complexității datelor poate fi obținută într-o variantă aleasă în textul criptanalizei liniare (Knudsen și Mathiassen, 2000). Junod (2001) a efectuat mai multe experimente pentru a determina complexitatea timpului real al criptanalizei liniare și a raportat că a fost ceva mai rapid decât a fost prevăzut, necesitând un timp echivalent cu 2^39-2^41 evaluări DES.

• Atacul îmbunătățit al lui Davies: în timp ce criptanaliza liniară și diferențială sunt tehnici generale și pot fi aplicate la o serie de scheme, atacul lui Davies este o tehnică specializată pentru DES, sugerată prima dată de Donald Davies în anii optzeci și îmbunătățită de Biham și Biryukov în 1997. Cea mai puternică formă de atac necesită 250 de planuri cunoscute, are o complexitate de calcul de 250 și are o rată de succes de 51%.

Au fost, de asemenea, atacuri propuse împotriva versiunilor reduse ale cifrului, adică versiuni ale DES cu mai puțin de șaisprezece runde. O astfel de analiză oferă o perspectivă asupra cât de multe runde sunt necesare pentru siguranță și cât de multă „marjă de securitate” păstrează versiunea completă. Langanal și Hellman au propus criptanaliza liniară diferențială și combină criptanaliza liniară diferențială și liniară într-un singur atac. O versiune îmbunătățită a atacului poate sparge 9-round DES cu 215.8 planuri de text cunoscute și are o complexitate de timp 229.2 (Biham și colab., 2002). [DESW]

6.9 triplu DES

6.9.1 algoritmul

Triple DES utilizează un set de 3 chei, K1, K2 și K3, fiecare de 56 biți. Algoritmul de criptare este:

ciphertext = EK3(DK2(EK1(plaintext)))

I.e., DES criptează cu K1, DES *decriptează cu* K2, apoi DES criptează with K3.

Decriptarea are loc în sens invers:

plaintext = DK1(EK2(DK3(ciphertext)))

I.e., DES decriptează cu K3, DES *criptează cu* K2, apoi DES decriptează with K1.

Fiecare tripletă criptează un bloc de 64 de biți de date.

În fiecare caz, operația de la mijloc este inversul primei și ultimei. Acest lucru îmbunătățește puterea algoritmului atunci când utilizați opțiunea de cheie de tip 2 compatibilitatea cu DES cu opțiunea de cheie 3.

6.9.2 opțiuni de cheie

Standardele definesc trei opțiuni de cheie:

- Opțiunea de cheie 1: Toate cele trei taste sunt independente.

- Opțiunea de cheie 2: K1 și K2 sunt independente, iar K3 = K1.
- Opțiunea de cheie 3: Toate cele trei taste sunt identice, adică K1 = K2 = K3.

Opțiunea de cheie 1 este cea mai puternică, cu 3 × 56 = 168 biți de cheie independenți.

Opțiunea de cheie 2 oferă mai puțină securitate, cu 2 × 56 = 112 biți de cheie. Această opțiune este mai puternică decât criptarea DES de două ori, de ex. cu K1 și K2, deoarece protejează împotriva atacurilor de întâlnire la mijloc.

Opțiunea de cheie 3 este echivalentă cu DES, cu doar 56 de biți cheie. Această opțiune oferă o compatibilitate cu DES, deoarece prima și a doua operație DES se anulează. Nu mai este recomandată de Institutul Național de Standarde și Tehnologie (NIST) și nu este susținută de ISO / IEC 18033-3.

6.10 advanced encryption standard (AES)

Standardul avansat de criptare (AES) este un standard de criptare adoptat de guvernul S.U.A. Standardul cuprinde trei algoritmi de cifrare de tip bloc, AES-128, AES-192 și AES-256, adoptați dintr-o colecție mai mare publicată inițial sub numele Rijndael. Fiecare cifru AES are un bloc de 128 biți, cu dimensiuni cheie de 128, 192 și, respectiv, 256 de biți. Algoritmii de cifrare AES au fost analizați pe scară largă și sunt acum utilizați la nivel mondial, cum a fost cazul și cu predecesorul său, Standardul de criptare a datelor (DES).

6.11 istoric AES

Standardul avansat de criptare (AES) este un standard de criptare adoptat de guvernul S.U.A. Standardul cuprinde trei algoritmi de cifrare de tip bloc, AES-128, AES-192 și AES-256, adoptați dintr-o colecție mai mare publicată inițial sub numele Rijndael. Fiecare cifru AES are un bloc de 128 biți, cu dimensiuni cheie de 128, 192 și, respectiv, 256 de biți. Algoritmii de cifrare AES au fost analizați pe scară largă și sunt acum utilizați la nivel mondial, cum a fost cazul și cu predecesorul său, Standardul de criptare a datelor (DES).

AES a fost anunțat de Institutul Național de Standarde și Tehnologie (NIST) ca US FIPS PUB 197 (FIPS 197) pe 26 noiembrie 2001, după un proces de standardizare de 5 ani în care au fost prezentate și evaluate cincisprezece proiecte concurente înainte de a fi selectat Rijndael ca cel mai potrivit (pentru detalii, consultați procesul de adoptare a Advanced Encryption Standard). A devenit efectiv standard guvernamental federal la 26 mai 2002, după aprobarea de către Secretarului de Comerț. Acesta este disponibil în multe pachete de criptare. AES este primul cifru public accesibil și deschis, aprobat de ANS pentru informații de cel mai înalt nivel de secretizare (a se vedea secțiunea "Securitatea AES", mai jos).

Cifrul Rijndael a fost elaborat de doi criptografi belgieni, Joan Daemen și Vincent Rijmen, și a fost prezentat de aceștia la procesul de selecție AES. Rijndael (pronunțat [rɛinda:l]) este o combibație a numelor celor doi inventatori.

6.12 descriere generală

AES se bazează pe un principiu de proiectare cunoscut sub numele de rețea de permutare de substituție. Este rapid atât în software, cât și în hardware. Spre deosebire de predecesorul său, DES, AES nu utilizează o rețea Feistel.

AES are o dimensiune a blocului fixă de 128 biți și o cheie de lungime 128, 192 sau 256 de biți, deși Rijndael poate fi specificat cu blocuri și dimensiuni de cheie orice multiplu de 32 de biți, cu un minim de 128 biți și un maxim de 256 biți.

AES funcționează pe o matrice de octeți 4 × 4, denumită stare (versiunile lui Rijndael cu o dimensiune mai mare a blocului au coloane suplimentare în stare). Majoritatea calculelor AES se fac într-un câmp finit special.

Cifrul AES este specificat ca un număr de repetări ale rundelor de transformare care convertesc textul de intrare în rezultatul final al textului cifrat. Fiecare rundă constă din mai mulți pași de procesare, inclusiv unul care depinde de cheia de criptare. Se aplică un set de runde inverse pentru a transforma textul criptat înapoi în textul original, folosind aceeași cheie de criptare.

6.13 fundamente algebrice

6.13.1 reprezentarea octeților

Unitatea de procesare din AES este octetul. Acest lucru este destul de diferit de DES, unde operațiile de bază au fost efectuate la nivel de biți. Un octet este o secvență de opt biți, numiți b_0, b_1, ..., b_7 - b_0 fiind cel mai puțin semnificativ. Fiecărui octet (sau secvență de opt biți) îi asociem un polinom de grad cel mult 7, cu coeficienți în Z_2, inelul de întregi modulo 2, după cum urmează. Dacă valoarea octetului nostru este b_7 * 2^7 + b_6 * 2^6 + b_5 * 2^5 + b_4 * 2^4 + b_3 * 2^3 + b_2 * 2^2 + b_1 * 2^1 + b_0 * 2^0 cu toți coeficienții în Z_2, atunci polinomul asociat este b_7 * x^7 + b_6 * x^6 + ... + b_2 * x^2 + b_1 * x + b_0.

De exemplu, {01100011} identifică elementul de câmp finit x^6 + x^5 + x + 1. În afară de reprezentările lor binare sau polinomiale, octeții vor fi de asemenea reprezentați în notație hexazecimală fie ca {63} (octetul de mai sus), fie ca 0x63 .

Toți octeții din algoritmul AES sunt interpretați ca elemente de câmp finit. Elementele de câmp finite pot fi adăugate și multiplicate, dar aceste operații sunt diferite de cele utilizate pentru numere. Următoarele subsecțiuni introduc conceptele matematice de bază necesare pentru descrierea algoritmului.

6.13.2 adunarea

Adăugarea a două elemente într-un câmp finit se obține prin "adunarea" coeficienților pentru puterile corespunzătoare în polinoamele pentru cele două elemente. Adunarea se face cu operația XOR (notată cu ^) - modulo 2 - astfel încât 1 ^ 1 = 0, 1 ^ 0 = 1 și 0 ^ 0 = 0.

În consecință, scăderea polinomilor este identică cu adunarea polinomilor.

Alternativ, adăugarea de elemente de câmp finit poate fi descrisă ca adăugarea modulo 2 a biților corespunzători în byte. Pentru doi octeți $\{a_7a_6a_5a_4a_3a_2a_1a_0\}$ and $\{b_7b_6b_5b_4b_3b_2b_1b_0\}$, suma este

$\{c_7c_6c_5c_4c_3c_2c_1c_0\}$, unde pentru fiecare i, $c_i = a_i$ ^ b_i (i.e., $c_7 = a_7$ ^ b_7, $c_6 = a_6$ ^ b_6, ...$c_0 = a_0$ ^ b_0).

De exemplu, următoarele expresii sunt echivalente una cu cealaltă:

$(x^6 + x^4 + x^2 + x + 1) + (x^7 + x + 1) = x^7 + x^6 + x^4 + x^2$ (notație polinomială);

{01010111} ^ {10000011} = {11010100} (notație binară);

{57} ^ {83} = {d4} (notație hexazecimală).

6.13.3 înmulțirea (produsul)

În reprezentarea polinomială, înmulțirea în GF (2^8) (notată cu ·) corespunde cu înmulțirea polinoamelor modulo un polinom ireductibil de gradul 8. Un polinom este ireductibil dacă singuii săi divizori sunt 1 și el însuși. Pentru algoritmul AES, acest polinom ireductibil este

$$m(x) = x^8 + x^4 + x^3 + x + 1$$

sau {01}{1b} în hexazecimal.

De exemplu, {57} · {83} = {c1}, deoarece

$(x^6 + x^4 + x^2 + x + 1)(x^7 + x + 1) = x^{13} + x^{11} + x^9 + x^8 + x^7 +$

$x^7 + x^5 + x^3 + x^2 + x +$

$x^6 + x^4 + x^2 + x + 1$

$= x^{13} + x^{11} + x^9 + x^8 + x^6 + x^5 + x^4 + x^3 + 1$

și

$x^{13} + x^{11} + x^9 + x^8 + x^6 + x^5 + x^4 + x^3 + 1$ modulo ($x^8 + x^4 + x^3 + x + 1$)

$= x^7 + x^6 + 1$.

Reducerea modulo $m(x)$ ne asigură că rezultatul înmulțirii va fi întotdeauna un polinom de grad mai mic decât 8 și de aceea, acesta poate fi reprezentat de către un octet. Spre deosebire de adunare, nu există nicio operație simplă la nivelul octeților care corespunde acestei multiplicări.

Înmulțirea definită mai sus este asociativă, iar elementul {01} este identitatea la operația de înmulțire. Pentru orice polinom binar non-zero b(x) de grad mai mic de 8, inversul multiplicativ al b (x), notat b^{-1} (x), poate fi găsit după cum urmează: algoritmul euclidian extins este folosit pentru a calcula polinoamele a (x) și c (x) astfel încât

$b(x)a(x) + m(x)c(x) = 1$

Deci, $a(x) \cdot b(x) \bmod m(x) = 1$, ceea ce înseamnă

b^{8}-1 $(x) = a(x) \bmod m(x)$

Mai mult decât atât, pentru orice $a(x)$, $b(x)$ și $c(x)$ din câmp, avem

$a(x) \cdot (b(x) + c(x)) = a(x) \cdot b(x) + a(x) \cdot c(x)$.

Rezultă că setul de 256 valori posibile posibile, cu XOR folosit ca adunare și multiplicarea definită ca mai sus, are structura câmpului finit GF(2^8).

6.13.4 înmulțirea cu *x*

Multiplicarea unui polinom binar cu polinomul x rezultă în x · b (x) și se obține prin reducerea rezultatului de mai sus modulo m(x). Dacă b7 = 0, rezultatul este deja în formă redusă. Dacă b_7 = 1, reducerea este realizată prin scădere (adică, prin XOR) a polinomului m(x). Rezultă că multiplicarea cu x (adică, {00000010} sau {02}) poate fi implementată la nivel de octet ca o deplasare la stânga și un XOR cu {1b}. Această operație pe octeți este notată cu xtime().

Înmulțirea prin puteri mai mari a lui x poate fi implementată prin aplicarea repetată a operației xtime().

Prin adăugarea unor rezultate intermediare, poate fi implementată înmulțirea cu orice constantă.

De examplu, {57} · {13} = {fe} deoarece

{57} · {02} = xtime({57}) = {ae}

{57} · {04} = xtime({ae}) = {47}

{57} · {08} = xtime({47}) = {8e}

{57} · {10} = xtime({8e}) = {07},

astfel,

{57} · {13} = {57} · ({01} ^ {02} ^ {10})

= {57} ^ {ae} ^ {07}

= {fe}.

6.14 programul cheilor

Cheia privată utilizată în procesul de criptare (care poate fi de 4, 6 sau 8 cuvinte de 32 de biți) este utilizată pentru generarea unui set de cuvinte care va fi utilizat în fiecare din rundele algoritmului pentru a fi adăugat (utilizând operația XOR) la cuvintele din matricea de stare. Primele Nk cuvinte din acest set sunt formate din chiar cheia privată. Restul sunt generate de ea, confrom procedurii de mai jos.

```
KeyExpansion(byte key[4*Nk], word w[Nb*(Nr+1)], Nk)
begin
   word temp
   i = 0

   while (i < Nk)
         w[i] = word(key[4*i], key[4*i+1], key[4*i+2], key[4*i+3])
         i = i+1
   end while

   i = Nk

   while (i < Nb * (Nr+1)]
         temp = w[i-1]

         if (i mod Nk = 0)
               temp = SubWord(RotWord(temp)) xor Rcon[i/Nk]
         else if (Nk > 6 and i mod Nk = 4)
               temp = SubWord(temp)
         end if

         w[i] = w[i-Nk] xor temp
         i = i + 1
   end while
end
```

Programul cheilor pentru decriptare este similar - subcheiile sunt în ordine inversă față de criptare. În afară de această schimbare, procesul este același ca și pentru criptare.

6.15 funcții utilizate

6.15.1 funcția SubBytes()

În pasul SubBytes(), fiecare octet din stare este înlocuit cu intrarea acestuia dintr-un tabel de căutare fix pe 8 biți, S; $b_{ij} = S(a_{ij})$.

În pasul `SubBytes()`, fiecare octet din matrice este actualizat utilizând o cutie de substituție pe 8 biți, caseta S-Rijndael. Această operație asigură non-liniaritatea în cifru. S-boxul folosit este derivat din inversul multiplicativ peste GF(2^8), cunoscut ca având proprietăți bune de neliniaritate. Pentru a evita atacurile bazate pe proprietăți simple algebrice, S-box-ul este construit prin combinarea funcției inverse cu o transformare afină inversabilă. S-box-ul este, de asemenea, ales pentru a evita orice puncte fixe (și astfel poate fi considerat un deranjament), precum și orice puncte fixe opuse.

6.15.2 funcția ShiftRows()

În pasul `ShiftRows`, octeții din fiecare linie al stării sunt deplasați ciclic spre stânga. Numărul de locuri pe care fiecare octet este mutat diferă pentru fiecare rând.

Pasul `ShiftRows` funcționează pe liniile stării; deplasează (șiftează) ciclic octeții din fiecare rând (linie) cu un anumit număr de poziții. Pentru AES, primul rând este lăsat neschimbat. Fiecare octet al celui de-al doilea rând este deplasat cu o poziție spre stânga. În mod similar, rândurile al treilea și al patrulea sunt deplasate cu două și, respectiv, trei poziții. Pentru blocul de mărimi de 128 biți și 192 de biți, modelul de schimbare este același. În acest fel, fiecare coloană a stării de ieșire a pasului `ShiftRows` este compusă din octeți din fiecare coloană a stării de intrare. (Variantele Rijndael cu o dimensiune mai mare a blocului au offseturi ușor diferite). În cazul blocului de 256 de biți, primul rând este neschimbat, iar schimbarea pentru al doilea, al treilea și al patrulea rând este de 1 octet, 3 octeți și respectiv 4 octeți - această modificare se aplică

numai cifrului Rijndael când este utilizat cu un bloc de 256 de biți, deoarece AES nu utilizează blocuri de 256 biți.

6.15.3 funcția MixColumns()

În pasul `MixColumns`, fiecare coloană a stării se înmulțește cu un polinom fix c(x).

În pasul `MixColumns`, cei patru octeți din fiecare coloană a stării sunt combinați folosind o transformare liniară inversabilă. Funcția `MixColumns` are patru octeți ca intrări și ieșiri de patru octeți, în care fiecare octet de intrare afectează toți cei patru octeți de ieșire. Împreună cu `ShiftRows`, `MixColumns` oferă difuzie în cifru. Fiecare coloană este tratată ca un polinom peste GF(2^8) și apoi este multiplicată modulo x^4 + 1 cu un polinom fix c (x) = 0x03x^3 + x^2 + x + 0x02. (Coeficienții sunt afișați în echivalentul lor hexazecimal al reprezentării binare a polinomilor de biți din GF(2) [x].) Pasul `MixColumns` poate fi de asemenea vizualizat ca o multiplicare de către o matrice specială MDS în câmpul finit.

6.15.4 funcția AddRoundKey()

În pasul `AddRoundKey`, fiecare octet al stării este combinat cu un octet al subcheiei de rundă utilizând operația XOR (⊕).

În pasul `AddRoundKey`, subcheia este combinată cu matrice de stare. Pentru fiecare rundă, o subcheie este derivată din cheia principală utilizând programul de chei al algoritmului Rijndael; fiecare subcheie are aceeași dimensiune ca și matricea de stare. Subcheia este adăugată prin combinarea fiecărui octet al stării cu octetul corespunzător al subcheiei folosind operația XOR la nivel de bit.

6.16 pseudo-cod

```
Cipher(byte in[4*Nb], byte out[4*Nb], word w[Nb*(Nr+1)])
begin
   byte state[4,Nb]
   state = in
   AddRoundKey(state, w[0, Nb-1])

   for round = 1 step 1 to Nr-1
         SubBytes(state)
         ShiftRows(state)
         MixColumns(state)
         AddRoundKey(state, w[round*Nb, (round+1)*Nb-1])
   end for

   SubBytes(state)
   ShiftRows(state)
   AddRoundKey(state, w[Nr*Nb, (Nr+1)*Nb-1])
   out = state
end
```

6.17 funcțiile inverse

Funcțiile prezentate în paragrafele precedente pot fi inversate și aplicate în ordine inversă, permițând astfel decriptarea mesajului cifrat, având ca rezultat mesajul inițial.

Continuăm cu o scurtă descriere a celor patru funcții inverse. Pentru mai multe detalii, consultatți FIPS 197 - https://nvlpubs.nist.gov/nistpubs/fips/nist.fips.197.pdf .

- `InvShiftRows` – liniile (rândurile) din matricea de stare sunt șiftate (deplasate) spre dreapta cu același număr de octeți, spre deosebire de funcția inițială, unde deplasarea se făcea la stânga.

- `InvSubBytes` – funcția folosește caseta de substituție care este inversa casetei de substituție a funcției `SubBytes`
- `InvMixColumns` – polinomul utilizat pentru multiplicare este 0x0bx^3 + 0x0dx^2 + 0x09x + 0x0e
- `AddRoundKey` – este chiar funcția inițială, deoarece este propria sa inversă

6.18 pseudo-cod pentru decriptare

```
InvCipher(byte in[4*Nb], byte out[4*Nb], word w[Nb*(Nr+1)])
begin
   byte state[4,Nb]
   state = in
   AddRoundKey(state, w[Nr*Nb, (Nr+1)*Nb-1])

   for round = Nr-1 step -1 downto 1
         InvShiftRows(state)
         InvSubBytes(state)
         AddRoundKey(state, w[round*Nb, (round+1)*Nb-1])
         InvMixColumns(state)
   end for

   InvShiftRows(state)
   InvSubBytes(state)
   AddRoundKey(state, w[0, Nb-1])
   out = state
end
```

6.19 pseudo-cod echivalent pentru decriptare

Algoritmul AES are două proprietăți care permit elaborarea unei versiuni echivalente pentru pseudo-codul procedurii de decriptare. Aceste proprietăți sunt:

- funcțiile `SubBytes` și `ShiftRows` comută, la felși inversele lor, `InvSubBytes` și `InvShiftRows`
- Operațiile de mixare a coloanelor – `MixColumns` și `InvMixColumns` - sunt liniare în raport cu intrările de la coloane

Aceste proprietăți conduc la o versiune mai eficientă a procedurii de decriptare. Listăm mai jos pseudo-codul pentru procedura echivalentă de decriptare:

```
EqInvCipher(byte in[4*Nb], byte out[4*Nb], word w[Nb*(Nr+1)])
begin
   byte state[4,Nb]
   state = in
   AddRoundKey(state, w[Nr*Nb, (Nr+1)*Nb-1])

   for round = Nr-1 step -1 downto 1
         InvSubBytes(state)
         InvShiftRows(state)
         InvMixColumns(state)
         AddRoundKey(state, dw[round*Nb, (round+1)*Nb-1])

   end for

   InvSubBytes(state)
   InvShiftRows(state)
```

```
   AddRoundKey(state, dw[0, Nb-1])
   out = state
end
```

Matricea dw[] care conține programul de chei se obține adăugînd pseudo-codul următor la sfârșitul procedurii de program al cheilor:

```
for i = 0 step 1 to (Nr+1)*Nb-1
   dw[i] = w[i]
end for

for round = 1 step 1 to Nr-1
   InvMixColumns(dw[round*Nb, (round+1)*Nb-1])
end for
```

6.20 un exemplu

```
PLAINTEXT:     00112233445566778899aabbccddeeff
KEY:           000102030405060708090a0b0c0d0e0f

CIPHER (ENCRYPT):

round[ 0].input 00112233445566778899aabbccddeeff
round[ 0].k_sch 000102030405060708090a0b0c0d0e0f
round[ 1].start 00102030405060708090a0b0c0d0e0f0
round[ 1].s_box 63cab7040953d051cd60e0e7ba70e18c
round[ 1].s_row 6353e08c0960e104cd70b751bacad0e7
round[ 1].m_col 5f72641557f5bc92f7be3b291db9f91a
round[ 1].k_sch d6aa74fdd2af72fadaa678f1d6ab76fe
round[ 2].start 89d810e8855ace682d1843d8cb128fe4
round[ 2].s_box a761ca9b97be8b45d8ad1a611fc97369
round[ 2].s_row a7be1a6997ad739bd8c9ca451f618b61
round[ 2].m_col ff87968431d86a51645151fa773ad009
round[ 2].k_sch b692cf0b643dbdf1be9bc5006830b3fe
round[ 3].start 4915598f55e5d7a0daca94fa1f0a63f7
round[ 3].s_box 3b59cb73fcd90ee05774222dc067fb68
round[ 3].s_row 3bd92268fc74fb735767cbe0c0590e2d
round[ 3].m_col 4c9c1e66f771f0762c3f868e534df256
round[ 3].k_sch b6ff744ed2c2c9bf6c590cbf0469bf41
round[ 4].start fa636a2825b339c940668a3157244d17
round[ 4].s_box 2dfb02343f6d12dd09337ec75b36e3f0
round[ 4].s_row 2d6d7ef03f33e334093602dd5bfb12c7
round[ 4].m_col 6385b79ffc538df997be478e7547d691
round[ 4].k_sch 47f7f7bc95353e03f96c32bcfd058dfd
round[ 5].start 247240236966b3fa6ed2753288425b6c
round[ 5].s_box 36400926f9336d2d9fb59d23c42c3950
round[ 5].s_row 36339d50f9b539269f2c092dc4406d23
round[ 5].m_col f4bcd45432e554d075f1d6c51dd03b3c
round[ 5].k_sch 3caaa3e8a99f9deb50f3af57adf622aa
round[ 6].start c81677bc9b7ac93b25027992b0261996
round[ 6].s_box e847f56514dadde23f77b64fe7f7d490
round[ 6].s_row e8dab6901477d4653ff7f5e2e747dd4f
round[ 6].m_col 9816ee7400f87f556b2c049c8e5ad036
round[ 6].k_sch 5e390f7df7a69296a7553dc10aa31f6b
round[ 7].start c62fe109f75eedc3cc79395d84f9cf5d
round[ 7].s_box b415f8016858552e4bb6124c5f998a4c
round[ 7].s_row b458124c68b68a014b99f82e5f15554c
```

```
round[ 7].m_col c57e1c159a9bd286f05f4be098c63439
round[ 7].k_sch 14f9701ae35fe28c440adf4d4ea9c026
round[ 8].start d1876c0f79c4300ab45594add66ff41f
round[ 8].s_box 3e175076b61c04678dfc2295f6a8bfc0
round[ 8].s_row 3e1c22c0b6fcbf768da85067f6170495
round[ 8].m_col baa03de7a1f9b56ed5512cba5f414d23
round[ 8].k_sch 47438735a41c65b9e016baf4aebf7ad2
round[ 9].start fde3bad205e5d0d73547964ef1fe37f1
round[ 9].s_box 5411f4b56bd9700e96a0902fa1bb9aa1
round[ 9].s_row 54d990a16ba09ab596bbf40ea111702f
round[ 9].m_col e9f74eec023020f61bf2ccf2353c21c7
round[ 9].k_sch 549932d1f08557681093ed9cbe2c974e
round[10].start bd6e7c3df2b5779e0b61216e8b10b689
round[10].s_box 7a9f102789d5f50b2beffd9f3dca4ea7
round[10].s_row 7ad5fda789ef4e272bca100b3d9ff59f
round[10].k_sch 13111d7fe3944a17f307a78b4d2b30c5
round[10].output 69c4e0d86a7b0430d8cdb78070b4c55a

INVERSE CIPHER (DECRYPT):

round[ 0].iinput 69c4e0d86a7b0430d8cdb78070b4c55a
round[ 0].ik_sch 13111d7fe3944a17f307a78b4d2b30c5
round[ 1].istart 7ad5fda789ef4e272bca100b3d9ff59f
round[ 1].is_row 7a9f102789d5f50b2beffd9f3dca4ea7
round[ 1].is_box bd6e7c3df2b5779e0b61216e8b10b689
round[ 1].ik_sch 549932d1f08557681093ed9cbe2c974e
round[ 1].ik_add e9f74eec023020f61bf2ccf2353c21c7
round[ 2].istart 54d990a16ba09ab596bbf40ea111702f
round[ 2].is_row 5411f4b56bd9700e96a0902fa1bb9aa1
round[ 2].is_box fde3bad205e5d0d73547964ef1fe37f1
round[ 2].ik_sch 47438735a41c65b9e016baf4aebf7ad2
round[ 2].ik_add baa03de7a1f9b56ed5512cba5f414d23
round[ 3].istart 3e1c22c0b6fcbf768da85067f6170495
round[ 3].is_row 3e175076b61c04678dfc2295f6a8bfc0
round[ 3].is_box d1876c0f79c4300ab45594add66ff41f
round[ 3].ik_sch 14f9701ae35fe28c440adf4d4ea9c026
round[ 3].ik_add c57e1c159a9bd286f05f4be098c63439
round[ 4].istart b458124c68b68a014b99f82e5f15554c
round[ 4].is_row b415f8016858552e4bb6124c5f998a4c
round[ 4].is_box c62fe109f75eedc3cc79395d84f9cf5d
round[ 4].ik_sch 5e390f7df7a69296a7553dc10aa31f6b
round[ 4].ik_add 9816ee7400f87f556b2c049c8e5ad036
round[ 5].istart e8dab6901477d4653ff7f5e2e747dd4f
round[ 5].is_row e847f56514dadde23f77b64fe7f7d490
round[ 5].is_box c81677bc9b7ac93b25027992b0261996
round[ 5].ik_sch 3caaa3e8a99f9deb50f3af57adf622aa
round[ 5].ik_add f4bcd45432e554d075f1d6c51dd03b3c
round[ 6].istart 36339d50f9b539269f2c092dc4406d23
round[ 6].is_row 36400926f9336d2d9fb59d23c42c3950
round[ 6].is_box 247240236966b3fa6ed2753288425b6c
round[ 6].ik_sch 47f7f7bc95353e03f96c32bcfd058dfd
round[ 6].ik_add 6385b79ffc538df997be478e7547d691
round[ 7].istart 2d6d7ef03f33e334093602dd5bfb12c7
round[ 7].is_row 2dfb02343f6d12dd09337ec75b36e3f0
round[ 7].is_box fa636a2825b339c940668a3157244d17
round[ 7].ik_sch b6ff744ed2c2c9bf6c590cbf0469bf41
round[ 7].ik_add 4c9c1e66f771f0762c3f868e534df256
round[ 8].istart 3bd92268fc74fb735767cbe0c0590e2d
round[ 8].is_row 3b59cb73fcd90ee05774222dc067fb68
```

```
round[ 8].is_box 4915598f55e5d7a0daca94fa1f0a63f7
round[ 8].ik_sch b692cf0b643dbdf1be9bc5006830b3fe
round[ 8].ik_add ff87968431d86a51645151fa773ad009
round[ 9].istart a7be1a6997ad739bd8c9ca451f618b61
round[ 9].is_row a761ca9b97be8b45d8ad1a611fc97369
round[ 9].is_box 89d810e8855ace682d1843d8cb128fe4
round[ 9].ik_sch d6aa74fdd2af72fadaa678f1d6ab76fe
round[ 9].ik_add 5f72641557f5bc92f7be3b291db9f91a
round[10].istart 6353e08c0960e104cd70b751bacad0e7
round[10].is_row 63cab7040953d051cd60e0e7ba70e18c
round[10].is_box 000102030405060708090a0b0c0d0e0f0
round[10].ik_sch 00010203040506070809a0b0c0d0e0f
round[10].i_outp 00112233445566778899aabbccddeeff
```

6.21 atacuri de tip canal lateral (side-channel)

Până în mai 2009, singurele atacuri publicate cu succes împotrivalgoritmului complet AES au fost atacurile canalului lateral (side-channel)asupra unor implementări specifice. Agenția Națională de Securitate (NSA) a revizuit toți finaliștii AES, inclusiv Rijndael, și a declarat că toți au fost suficient de siguri pentru datele ne-clasificate ale guvernului Statelor Unite. În iunie 2003, guvernul american a anunțat că AES poate fi utilizat pentru a proteja informațiile clasificate:

"Designul și rezistența tuturor lungimilor cheie ale algoritmului AES (adică 128, 192 și 256) sunt suficiente pentru a proteja informațiile clasificate până la nivelul SECRET. Informațiile TOP SECRET vor necesita utilizarea cheilor de lungime de 192 sau 256 de biți. Implementarea AES în produsele destinate să protejeze sistemele și / sau informațiile naționale de securitate trebuie revizuită și certificată de NSA înainte de achiziționarea și utilizarea acestora."

AES are 10 runde pentru chei de 128 biți, 12 runde pentru chei de 192 de biți și 14 runde pentru chei de 256 de biți. Până în 2006, cele mai cunoscute atacuri au fost cele 7 runde pentru chei de 128 biți, 8 runde pentru chei de 192 biți și 9 runde pentru chei de 256 de biți.

Pentru criptografi, o "spargere" criptografică este ceva mai rapid decât o căutare exhaustivă. Astfel, un atac XSL împotriva unui AES pe 128 de biți care necesită 2^100 de operații (în comparație cu 2^128 de chei posibile) ar fi considerat o spargere. Cel mai mare atac de forță brute, cunoscut în public, a fost împotriva unei chei RC5 de 64 de biți de către distribuite.net.

Spre deosebire de majoritatea celorlalte blocuri de cifru, AES are o descriere algebrică foarte curată. În 2002, un atac teoretic, denumit "atac XSL", a fost anunțat de Nicolas Courtois și Josef Pieprzyk, pretinzând că prezintă o slăbiciune a algoritmului AES datorită descrierii sale simple. De atunci, alte documente au arătat că atacul prezentat inițial nu este posibil; a se vedea atacul XSL asupra cifrurilor de tip bloc.

În timpul procesului de adoptare a AES, dezvoltatorii de algoritmi concurenți au scris despre Rijndael: "... suntem preocupați de utilizarea sa în aplicații critice de securitate". Cu toate acestea, la finalul procesului AES, Bruce Schneier, un dezvoltator al algoritmului competitiv Twofish, a scris că, deși el credea că atacurile academice reușite asupra lui Rijndael ar putea fi dezvoltate într-o zi, "Nu cred că cineva va descoperi vreodată un atac care permite cuiva să citească traficul de la Rijndael. "

Pe 1 iulie 2009, Bruce Schneier a dezvăluit un atac cu privire la versiunile de 192 biți și 256 de biți ale AES descoperite de Alex Biryukov și Dmitry Khovratovich; atacul cheie asociat cu versiunea de 256 de biți a AES exploatează simplitatea programei de chei a programului AES și are o complexitate de 2^119. Acesta este o continuare a unui atac descoperit anterior în 2009 de Alex Biryukov, Dmitry Khovratovich și Ivica Nikolic, cu o complexitate de 2^96 pentru una din fiecare 2^35 chei.

Un alt atac a fost anunțat pe blog de Bruce Schneier la 30 iulie 2009 și publicat pe 3 august 2009. Acest nou atac de către Alex Biryukov, Orr Dunkelman, Nathan Keller, Dmitry Khovratovich și Adi Shamir este împotriva lui AES-256 care folosește doar două chei asociate și timp de 2^39 pentru a recupera cheia completă de 256 biți a unei versiuni cu 9 runde sau timp de 2^45 pentru o versiune cu 10 runde, cu un tip mai puternic de atac de subcheie asociat, sau cu timp de 2^70 pentru o versiune cu 11 runde. AES cu chei de 256-biți folosește 14 runde, deci aceste atacuri nu sunt eficiente împotriva AES complet.

În noiembrie 2009 a fost publicat primul atac împotriva versiunii cu 8 runde a lui AES-128. Acest atac distinctiv cu cheie cunoscută este o îmbunătățire a reculului sau a atacurilor de la început de la mijloc pentru permutările asemănătoare AES, care văd două runde consecutive de permutare ca fiind aplicarea așa-numitei Super-Box. Funcționează pe versiunea cu 8 runde a modelului AES-128, cu o complexitate de calcul de 2^48 și o complexitate de memorie de 2^32

chapter 7 algoritmul diffie-hellman de schimb chei

7.1 istoria protocolului

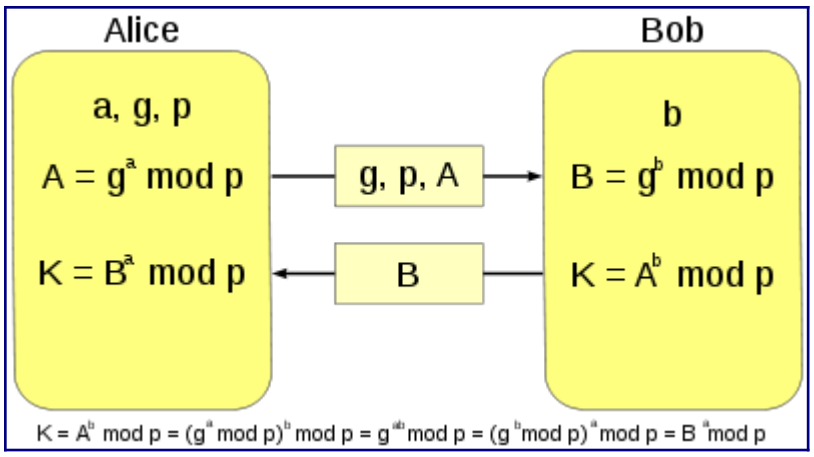
Algoritmul de schimb al cheilor Diffie-Hellman-Merkle a fost inventat în anul 1976 în timpul unei colaborări între Whitfield Diffie și Martin Hellman, aceasta fiind prima metodă practică pentru stabilirea unui secret comun peste un canal de comunicații neprotejat. Munca lui Ralph Merkle în distribuirea cheilor publice a fost o influență. John Gill a sugerat aplicarea problemei logaritmului discret. Prima dată a fost inventat de Malcolm Williamson de la GCHQ în Marea Britanie, cu câțiva ani înainte, dar GCHQ a ales să nu îl facă public până în anul 1997, dar atunci nu a avut nici o influență în domeniul academic de cercetare.

Metoda care a urmat în scurt timp a fost RSA, o altă implementare a criptografiei cu cheie publică ce folosește algoritmi asimetrici. Patentul U.S. 4,200,770, acum expirat, descire algoritmul și îi acreditează pe Hellman, Diffie și Merkle ca inventatori

7.2 descriere

Diffie-Hellman stabilește un secret comun care poate fii folosit în comunicare secretă prin schimbul de date peste o rețea publică.

Exemplu care include elementele matematice folosite în această criptare:



Schimbul de chei Diffie-Hellman

Cea mai simplă și originală implementare a protocolului care folosește grupul multiplicativ al numerelor întregi modulo p, unde p este prim și g este o rădăcină primitivă mod p. Exemplu de schimb, cu valorile publice în verde și valorile secrete în **rosu**:

ALICE				BOB		
Secret	Public	Calcul		Calcul	Public	Secret
	p, g				p, g	
a						b
		$g^a \pmod p$				

				$g^b \pmod p$		
	$(g^a)^b \pmod p$		$\rightarrow$ $\leftarrow$		$(g^b)^a \pmod p$	

1. Alice și Bob cad de acord să folosească numărul prim p=23 si baza g=5.
2. Alice alege un număr întreg secret ***a*=6**, iar apo îl trimite lui Bob A = $g^a \bmod p$
   - A = $5^6 \bmod 23 = 8$.
3. Bob alege un număr întreg secret ***b*=15**, iar apoi îl trimite lui Alice B = $g^b \bmod p$
   - B = $5^{15} \bmod 23 = 19$.
4. Alice calculează **s** = $B^a \bmod p$
   - $19^6 \bmod 23 =$ **2**.
5. Bob calculează **s** = $A^b \bmod p$
   - $8^{15} \bmod 23 =$ **2**.

Atît Alice cît și Bob au ajuns la aceeași valore, pentru că g^{ab} and g^{ba} sunt egale mod p. De remarcat că doar a, b și $g^{ab} = g^{ba} \bmod p$ sunt secrete. Toate celelalte valori – p, g, $g^a \bmod p$, și $g^b \bmod p$ – nu sunt secrete. Atunci când Alice și Bob calculează cheia secretă comună, ei o pot folosi ca cheie de criptare, cunoscută doar de ei, pentru transmiterea de mesage peste un canal de comunicații deschis. Desigur, trebuie folosite valori mult mai mari pentru a, b și p pentru ca acest exemplu să fie securizat, pentru că este ușor să se încerce toate valorile posibile $g^{ab} \bmod 23$ (sunt doar 22 de valori, chiar dacă a si b sunt mari). Dacă p ar fi un număr prim de cel puțin 300 de cifre, iar a și b de cel puțin 100 cifre, atunci nici cei mai buni algoritmi cunoscuți în ziua de azi nu ar putea găsi a dacă se cunoaște doar g, p, și $g^a \bmod p$, chiar dacă se folosește toată puterea de calcul a întregii omeniri.

Problema este cunoscută ca problema logaritmului discret. Observați că nu este nevoie ca g să fie mare și în practică este 2 sau 5.

O descriere generală a protocolului:

1. Alice și Bob aleg în comun un grup ciclic finit G și un elemnet generator g in G. (Acest lucru se face de obicei cu mult înaintea restului de pași ai protocolului; g se presupune a fi cunoscut de toți atacatorii). Noi vom scrie grupul G multiplicativ.
2. Alice alege un număr natural aleatoriu a și trimite g^a lui Bob.
3. Bob alege un număr natural aleatoriu b și trimite g^b lui Alice.
4. Alice calculează $(g^b)^a$.
5. Bob calculează $(g^a)^b$.

Ambii Alice si Bob sunt acum în posesia unui element al grupului, g^{ab}, care pot servi ca și cheie secretă comună. Valorile lui $(g^b)^a$ și $(g^a)^b$ sunt egale deoarece ridicarea la putere este asociativă.

7.3 chart

Exemplu de diagramă care ajută la prezentarea a cine știe ce informație. (Eve este o persoană care „trage cu urechea” – ea urmărește ce este trimis între Alice și Bob, dar Eve nu poate altera conținutul informațiilor aflate pe parcursul acestei comunicări).

- Fie s = cheia secretă comună. s = 2

- Fie a = cheia privată a lui Alice. a = 6
- Fie A = cheia publică a lui Alice. A = g^a mod p = 8
- Fie b = cheia privată a lui Bob. b = 15
- Fie B = cheia publică a lui Bob. B = g^b mod p = 19
- Fie g = baza publică. g=5
- Fie p = număr prim public. p = 23

Alice		***Eve***		***Bob***	
cunoaște	nu cunoaște	cunoaște	nu cunoaște	cunoaște	nu cunoaște
p = 23	b = 15	p = 23	a = 6	p = 23	a = 6
base g = 5		base g = 5	b = 15	base g = 5	
a = 6			s = 2	b = 15	
A = 5^6 mod 23 = 8		A = 5^a mod 23 = 8		B = 5^{15} mod 23 = 19	
B = 5^b mod 23 = 19		B = 5^b mod 23 = 19		A = 5^a mod 23 = 8	
s = 19^6 mod 23 = 2		s = 19^a mod 23		s = 8^{15} mod 23 = 2	
s = 8^b mod 23 = 2		s = 8^b mod 23		s = 19^a mod 23 = 2	
s = 19^6 mod 23 = 8^b mod 23		s = 19^a mod 23 = 8^b mod 23		s = 8^{15} mod 23 = 19^a mod 23	
s = 2				s = 2	

Notă: Ar trebui să fie dificil pentru Alie să rezolve cheia privată a lui Bob sau pentru Bob să rezolve cheia privată a lui Alice. Dacă nu ar fii dificile cazurile acestea, Eve ar putea foarte ușor să substituie cu perechea de cheie să publică / privată, ea poate pune cheia publică a lui Bob în cheia sa privată, produce o cheie secretă comună falsă, și să rezolve cheia privată a lui Bob (și să o folosească pentru a rezolva cheia secretă comună. Eve ar putea încerca să aleagă o pereche de cheie publică / privată care îi va ușura calcularea cheii private a lui Bob).

7.4 securitate

Protocolul este considerat securizat / sigur împotriva celor care ar putea „asculta” dacă G și g sunt aleși cum se cuvine. Acești „ascultători” (Eve) ar trebui să rezolve preblema Diffie-Hellman de a obține g^{ab}. Acest lucru este deocamdată considerată dificil. Un algoritm eficient de a rezolva problema logaritmului discret ar ușura calcularea a sau b și ar rezolva problema Diffie-Hellman, făcând acest și multe alte criptosisteme cu chei publice nesigure.

Ordinul lui G ar trebui să fie prim sau să aibă un facor prim larg pentru a preveni folosirea algoritmului Pohlig-Hellman pentru obținerea lui a sau b. Pentru acest motiv, un prim q de tip Sophie-Germain este câteodată folosit la calcularea lui p=2q+1, considerat a fi un prim sigur, deoarece ordinul lui G este atunci doar divizibil cu 2 și q. g este câteodată ales pentru a genera subgrupul de ordin q al lui G, mai degrabă decât G, astfel încît simbolul Legendre a lui g^a niciodată nu dezvăluie bit-ul de ordin mic a lui a.

Dacă Alice și Bob folosesc generatoare de numere aleatorii al căror output nu este complet aleatoriu și poate fi prezis într-o oarecare măsură, atunci munca lui Eve este mult mai ușoară.

Numerele întregi secrete a și b sunt înlăturate la sfârșitul sesiunii. Așadar schimbul de chei Diffie-Hellman realizează în mod trivial „secretul direcționat” perfect deoarece nici un material cu cheie privată de lungă durata nu există pentru a putea fi dezvăluit.

7.5 autentificare

În descrierea originală, schimbul Diffie-Hellman-Merkel propriu-zis nu oferă autentificarea pentru persoanele care comunică, ceea ce îl face vulnerabil la atacuri man-in-the-middle. O persoană în mijloc poate iniția două schimbări de chei Diffie-Hellman diferite, una cu Alice și alta cu Bob, mascându-se eficient în Alice sau Bob, și vice versa, permițând atacatorului să decripteze (și să citească sau să stocheze) și după să cripteze din nou mesajele trimise între ei. O metodă pentru autentificarea comunicării dintre persoane este, în general, necesară pentru a preveni acest tip de atac

O varietate de soluții de autentificare criptografice includ un schimb Diffie-Hellman. Când Alice și Bob au o infrastructură pentru cheia publică, ei pot să semneze digital cheia la care au căzut de acord, sau g^a și g^b, precum în componentele MQV, STS și IKE din protocolul IPsec folosite pentru securizarea comunicațiilor din Protocolul de Internet (IP).

Când Alice și Bob împart o parolă, ei pot folosi un acord de cheie autentificată prin parolă din Diffie-Hellman, cum este cel descris in recomandarea ITU-T X.1035, care este folosită de către standardul de rețea de domiciliu G.hn.

chapter 8 criptare asimetrică - RSA

În criptografie, RSA este un algoritm pentru criptografia cu chei publice. A fost primul algoritm cunoscut a fi potrivit pentru semnare și criptare și unul dintre primele mari progrese în criptografia cu chei publice. RSA este utilizat pe scară largă în protocoalele de comerț electronic și se crede că este sigur, dacă se utilizează chei suficient de lungi și utilizarea implementărilor actualizate.

Criptografia cu cheie publică, cunoscută și sub denumirea de criptografie asimetrică, este o formă de criptografie în care un utilizator are o pereche de chei criptografice - o cheie publică și o cheie privată. Cheia privată este păstrată secretă, în timp ce cheia publică poate fi distribuită pe scară largă. Cheile sunt legate matematic, dar cheia privată nu poate fi practic derivată din cheia publică. Un mesaj criptat cu cheia publică poate fi decriptat doar cu cheia privată corespunzătoare.

În schimb, criptografia cu chei secrete, cunoscută și sub denumirea de criptografie simetrică, folosește o singură cheie secretă atât pentru criptare cât și pentru decriptare.

Conținutul acestui capitol vine în mare parte din RSA Wikipedia: [RSA-W].

8.1 istoric

Algoritmul a fost descris public în 1977 de Ron Rivest, Adi Shamir și Leonard Adleman la MIT; literele RSA fiind inițialele numelor lor de familie.

Clifford Cocks, un matematician britanic care lucra pentru agenția de informații britanică GCHQ, a descris un un sistem echivalent într-un document intern în 1973, dar având în vedere computerele relativ scumpe necesare implementarii la acea vreme, a fost considerată în cea mai mare parte o curiozitate și, din câte se știe, nu a fost niciodată implementat. Cu toate acestea, descoperirea sa a fost dezvăluită abia în 1997 datorită clasificării sale top-secret și Rivest, Shamir și Adleman au conceput RSA independent de munca lui Cocks.

MIT a primit brevetul SUA 4405829 pentru un „sistem și metodă de comunicații criptografice” care a folosit algoritmul în 1983. Brevetul a expirat la 21 septembrie 2000. Deoarece o lucrare care descrie algoritmul fusese publicat în august 1977, înainte de data depunerii cererii de brevet din decembrie 1977, reglementările din mare parte din restul lumii excludeau brevetele în altă parte și numai brevetul SUA a fost acordat. Dacă munca lui Cocks ar fi fost cunoscută public, un brevet în SUA nu ar fi fost posibil.

8.2 operații

RSA implică o cheie publică și o cheie privată. Cheia publică poate fi cunoscută de toată lumea și este folosită pentru criptarea mesajelor. Mesajele criptate cu cheia publică pot fi decriptate numai folosind cheia privată. Cheile pentru algoritmul RSA sunt generate în felul următor:

- Alegeți două numere prime mari aleatorii distincte p și q
- Calculați $n = pq$
- n este utilizat ca modul atât pentru cheile publice, cât și pentru cele private
- Calculați indicatorul: $\varphi(n) = (p-1)(q-1)$. (Indicatorul unui număr întreg pozitiv n este definit ca fiind numărul de numere întregi pozitive mai mici sau egale cu n care sunt ii sunt coprime lui n. De exemplu, $\varphi(9) = 6$ deoarece cele șase numere 1, 2, 4, 5, 7 și 8 sunt coprime cu 9).
- Alegeți un număr întreg e astfel încât $1 < e < \varphi(n)$ iar e și $\varphi(n)$ să nu aibă alți factori în afară de 1 (adică e și $\varphi(n)$ sunt coprime)
- e este lansat ca exponent al cheii publice
- Calculați d pentru a satisface relația de congruență $de = 1 \pmod{\varphi(n)}$; adică $de = 1 + k\varphi(n)$ pentru un întreg k.

- *d* este păstrat ca exponent al cheii private

Note cu privire la pașii de mai sus:

- Pasul 1: numerele pot fi testate probabilistic pentru primalitate.
- Pasul 3: modificat în PKCS # 1 v2.0 Standard de criptografie cu cheie publică, *l(n) = cmmmc(p-1, q-1)* unde *cmmmc* este cel mai mic multiplu comun, în loc de *φ(n) = (p-1)(q-1).*
- Pasul 4: O alegere populară pentru exponenții publici este e = 2^16 + 1 = 65537. Unele aplicații aleg valori mai mici, cum ar fi e = 3, 5, 17 sau 257. Acest lucru se face pentru a realiza verificarea criptării și semnăturii mai rapidă pe dispozitivele mici, cum ar fi cardurile inteligente, dar exponenții publici mici pot duce la creșterea de riscuri de securitate.
- Pașii 4 și 5 pot fi realizați cu algoritmul Euclidian extins; vezi aritmetica modulară.

Algoritmul euclidian extins este o extensie a algoritmului euclidian pentru găsirea celui mai mare divizor comun (CMMDC) al numerelor întregi *a* și *b*: găsește și numerele întregi *x* și *y* în identitatea lui Bézout

ax + by = cmmdc(a, b)

(De obicei, x sau y este negativ).

Algoritmul euclidian extins este deosebit de util atunci când *a* și *b* sunt coprime, deoarece x este inversul multiplicativ modular al unui modul *b*.

Cheia publică constă din modulul *n* și exponentul public (sau de criptare) *e*.

Cheia privată constă din perechea *(p, q)* și exponentul privat (sau de decriptare) *d*, care trebuie păstrate secrete.

Pentru eficiență, o altă formă a cheii private poate fi stocată:

- *p* și *q*: numerele prime din generarea cheii.
- *d mod (p-1)* și *d mod (q-1)* , denumite deseori dmp1 și dmq1.
- *q^{-1} mod(p)*, denumit și iqmp.

Toate părțile cheii private trebuie păstrate secrete sub această formă. p și q sunt sensibili, deoarece sunt factorii lui n și permit calcularea lui d dacă se cunoaște e. Dacă p și q nu sunt stocate în acest formă a cheii private, atunci acestea sunt șterse în siguranță împreună cu alte valori intermediare din generarea cheii.

Deși această formă permite decriptarea și semnarea mai rapidă utilizând Teorema Chinezească a Restului, acesta este considerabil mai puțin sigur, deoarece permite atacuri pe canale laterale. Aceasta este o problemă specială dacă implementarea este pe pe carduri inteligente, care beneficiază cel mai mult de eficiența îmbunătățită. (Începeți cu *y = x^e (mod n)* și lăsați cardul să decripteze asta. Deci calculează *y^d* (mod *p*) sau *y^d* (mod *q*) ale căror rezultate dau o anumită valoare *z*. Acum, induceți o eroare într-unul dintre calcule. Atunci gcd(*z* − *x*,*n*) va dezvălui *p* sau *q*.)

În criptografie, un atac de canal lateral este orice atac bazat pe informații obținute din implementarea fizica unui cripto-sistem, mai degrabă decât pe slăbiciuni teoretice în algoritmi (comparați criptanaliza). De exemplu, informații de sincronizare, consum de energie, scurgeri electromagnetice sau chiar sunet, pot oferi o sursă suplimentară de informații care poate fi exploatată pentru a sparge sistemul. Atacurile pe multiple canale laterale necesită cunoștințe tehnice considerabile despre funcționarea internă a sistemului pe care se află implementata criptografia.

Încercările de a sparge un cripto-sistem, înșelând sau constrângând persoane cu acces legitim nu sunt denumite de obicei atacuri pe canale laterale: vezi ingineria socială și criptanaliza furtunurilor de cauciuc (*rubber-hose*). Pentru atacuri asupra sistemele informatice în sine (care sunt adesea utilizate pentru a efectua criptografie și, astfel, conțin chei criptografice sau texte simple), consultați securitatea computerului.

8.3 criptarea mesajelor

Alice transmite cheia sa publică (n, e) lui Bob și păsrează secretă cheia sa privată. Bob dorește apoi să transmită un mesaj M lui Alice.

Pentru început, el transformș mesajul M într-un număr *m*. El calculează (criptează) apoi textul cifrat c conform formulei:

c = m^e (mod n)

Acest lucru se poate face rapid folosind metoda de exponențiere prin ridicare la pătrat. Bob îi transmite apoi valoarea lui *c* lui Alice.

8.4 decriptarea mesajelor

Alice poate recupera pe m din c prin utilizarea cheii sale private, conform calculului următor:

m = c^d (mod n)

Dat fiind *m*, se poate recupera mesajul original M.

Procedura descrisă mai sus funcționează deoarece, pentru început:

c^d = (m^e)^d = m^{ed} (mod n).

Acum, *ed = 1 (mod (p-1)(q-1))*, deci

ed = 1 (mod p-1) și *ed = 1 (mod q-1)*

care poate fi scris ca

ed = k(p-1) + 1 și *ed = h(q-1) + 1*

pentru valori convenabile ale lui k și h.

Dacă *m* nu este un multiplu de *p,* atunci *m* și *p* sunt coprime, deoarece *p* este prim; deci, conform teoremei lui Fermat

m^(p-1) = mod (p)

deci, utilizând prima expresie pentru *ed*,

m^{ed} = m^{k(p-1)+1} = (m^{p-1}^k m = 1^k m = m (mod p).

Dacă *m* este un multiplu de *p*, atunci

m^{ed} = 0^{ed} = 0 = m (mod p).

Folosind a soua expresie pentru *ed*, putem conclude, în mod similar

m^{ed} = m (mod q).

Deoarece *p* și *q* sunt numere prime distincte, prin aplicarea teoremei chineze a resturilor pentru aceste două congruențe, obținem

m^{ed} = m (mod pq).

Deci,

c^d = m (mod n).

8.5 un exemplu concret

Iată un exempu concret de criptare și decripatre cu RSA. Parametrii utilizați aici sunt mult mai mici decât normal, dar puteți genera parametri mai apropiați de valorile din viața reală, utilizând OpenSSL pentru a genera și utiliza o pereche de chei reală.

- Alegem două numere prime:

p = 61 și q = 53

- Calculăm $n = pq$

n = 61 * 53 = 3233

- Calculăm indicatorul lui Euler $\varphi(n) = (p-1)(q-1)$

$\varphi(n)$ = (61-1)(53-1) = 3120

- Alegem e > 1 coprim cu 3120

$e = 17$

- Calculăm pe d, inversul lui e modulo $\varphi(n)$, adică $de = 1 \pmod{\varphi(n)}$

d = 2753

*17 * 2753 = 46801 = 1 + 15 * 3120*.

Cheia publică este (n = 3233, e = 17). Pentru un mesaj completat (padded) *m,* funcția de criptare este:

c = m^e mod (n) = m^17 mod (3233)

Cheia privată este (*(61, 53)*, d = 2753). Funcția de decriptare este:

m = c^d mod (n) = c^2753 mod (3233)

De exemplu, pentru a cripta pe m = 123, calculăm

c = 123^17 (mod 3233) = 855.

Pentru a decripta pe c = 855, calculăm

m = 855^2753 (mod 3233) = 123.

Ambele calcule pot fi efectuate eficient folosind algoritmul de ridicare la pătrat și multiplicat pentru exponențierea modulară.

8.6 considerente practice

8.6.1 generarea cheilor

Finding the large primes p and q is usually done by testing random numbers of the right size with probabilistic primality tests which quickly eliminate virtually all non-primes.

p and q should not be 'too close', lest the Fermat factorization for n be successful, if p-q, for instance is less than $2n^{1/4}$ (which for even small 1024-bit values of n is $3x10^{77}$) solving for p and q is trivial. Furthermore, if either p-1 or q-1 has only small prime factors, n can be factored quickly by Pollard's p − 1 algorithm, and these values of p or q should therefore be discarded as well.

It is important that the secret key d be large enough. Michael J. Wiener showed in 1990 that if p is between q and $2q$ (which is quite typical) and $d < n^{1/4}/3$, then d can be computed efficiently from n and e. There is no known attack against small public exponents such as e=3, provided that proper padding is used. However, when no padding is used or when the padding is improperly implemented then small public exponents have a greater risk of leading to an attack, such as for example the unpadded plaintext vulnerability listed above. 65537 is a commonly used value for e. This value can be regarded as a compromise between avoiding potential small exponent attacks and still allowing efficient encryptions (or signature verification). The NIST Special Publication on Computer Security (SP 800-78 Rev 1 of August 2007) does not allow public exponents e smaller than 65537, but does not state a reason for this restriction.

8.6.2 viteza de execuție

RSA is much slower than DES and other symmetric cryptosystems. In practice, Bob typically encrypts a secret message with a symmetric algorithm, encrypts the (comparatively short) symmetric key with RSA, and transmits both the RSA-encrypted symmetric key and the symmetrically-encrypted message to Alice.

Symmetric-key algorithms are a class of algorithms for cryptography that use trivially related, often identical, cryptographic keys for both decryption and encryption.

The encryption key is trivially related to the decryption key, in that they may be identical or there is a simple transform to go between the two keys. The keys, in practice, represent a shared secret between two or more parties that can be used to maintain a private information link.

Other terms for symmetric-key encryption are secret-key, single-key, one-key and eventually private-key encryption. Use of the latter term does conflict with the term *private key* in public key cryptography.

This procedure raises additional security issues. For instance, it is of utmost importance to use a strong random number generator for the symmetric key, because otherwise Eve (an eavesdropper wanting to see what was sent) could bypass RSA by guessing the symmetric key.

8.6.3 distribuirea cheilor

As with all ciphers, how RSA public keys are distributed is important to security. Key distribution must be secured against a man-in-the-middle attack. Suppose Eve has some way to give Bob arbitrary keys and make him believe they belong to Alice. Suppose further that Eve can *intercept* transmissions between Alice and Bob. Eve sends Bob her own public key, which Bob believes to be Alice's. Eve can then intercept any ciphertext sent by Bob, decrypt it with her own secret key, keep a copy of the message, encrypt the message with Alice's public key, and send the new ciphertext to Alice. In principle, neither Alice nor Bob would be able to detect Eve's presence. Defenses against such attacks are often based on digital certificates or other components of a public key infrastructure.

In cryptography, a public key infrastructure (PKI) is an arrangement that binds public keys with respective user identities by means of a certificate authority (CA). The user identity must be unique for each CA. The binding is established through the registration and issuance process, which, depending on the level of assurance the binding has, may be carried out by software at a CA, or under human supervision. The PKI role that assures this binding is called the Registration Authority (RA) . For each user, the user identity, the public key, their binding, validity conditions and other attributes are made unforgeable in public key certificates issued by the CA.

The term trusted third party (TTP) may also be used for certificate authority (CA). The term PKI is sometimes erroneously used to denote public key algorithms which, however, do not require the use of a CA.

8.7 securitate

The security of the RSA cryptosystem is based on two mathematical problems: the problem of factoring large numbers and the RSA problem. Full decryption of an RSA ciphertext is thought to be infeasible on the assumption that both of these problems are hard, i.e., no efficient algorithm exists for solving them. Providing security against *partial* decryption may require the addition of a secure padding scheme.

The RSA problem is defined as the task of taking eth roots modulo a composite n: recovering a value m such that $c=m^e \bmod n$, where (n, e) is an RSA public key and c is an RSA ciphertext. Currently the most promising approach to solving the RSA problem is to factor the modulus n. With the ability to recover prime factors, an attacker can compute the secret exponent d from a public key (n, e), then decrypt c using the standard procedure. To accomplish this, an attacker factors n into p and q, and computes $(p-1)(q-1)$ which allows the determination of d from e. No polynomial-time method for factoring large integers on a classical computer has yet been found, but it has not been proven that none exists. See integer factorization for a discussion of this problem.

As of 2005, the largest number factored by a general-purpose factoring algorithm was 663 bits long (see RSA-200), using a state-of-the-art distributed implementation. RSA keys are typically 1024–2048 bits long.

Some experts believe that 1024-bit keys may become breakable in the near term (though this is disputed); few see any way that 4096-bit keys could be broken in the foreseeable future. Therefore, it is generally presumed that RSA is secure if *n* is sufficiently large. If *n* is 256 bits or shorter, it can be factored in a few hours on a personal computer, using software already freely available. Keys of 512 bits (or less) have been shown to be practically breakable in 1999 when RSA-155 was factored by using several hundred computers. A theoretical hardware device named TWIRL and described by Shamir and Tromer in 2003 called into question the security of 1024 bit keys. It is currently recommended that *n* be at least 2048 bits long.

In 1994, Peter Shor published Shor's algorithm, showing that a quantum computer could in principle perform the factorization in polynomial time. However, quantum computation is still in the early stages of development and may never prove to be practical.

8.7.1 adaptive chosen-ciphertext attack

An adaptive chosen-ciphertext attack (abbreviated as CCA2) is an interactive form of chosen-ciphertext attack in which an attacker sends a number of ciphertexts to be decrypted, then uses the results of these decryptions to select subsequent ciphertexts. It is to be distinguished from an indifferent chosen-ciphertext attack (CCA1).

The goal of this attack is to gradually reveal information about an encrypted message, or about the decryption key itself. For public-key systems, adaptive-chosen-ciphertexts are generally applicable only when they have the property of ciphertext malleability — that is, a ciphertext can be modified in specific ways that will have a predictable effect on the decryption of that message.

8.7.2 preventing the adaptive chosen-ciphertext attack

In order to prevent adaptive-chosen-ciphertext attacks, it is necessary to use an encryption or encoding scheme that limits ciphertext malleability. A number of encoding schemes have been proposed; the most common standard for RSA encryption is Optimal Asymmetric Encryption Padding (OAEP). Unlike ad-hoc schemes such as the padding used in PKCS #1 v1, OAEP has been proven secure under the random oracle model.

8.7.3 signing messages

Suppose Alice uses Bob's public key to send him an encrypted message. In the message, she can claim to be Alice but Bob has no way of verifying that the message was actually from Alice since anyone can use Bob's public key to send him encrypted messages. So, in order to verify the origin of a message, RSA can also be used to sign a message.

Suppose Alice wishes to send a signed message to Bob. She can use her own private key to do so. She produces a hash value of the message, raises it to the power of *d* mod *n* (as she does when decrypting a message), and attaches it as a "signature" to the message. When Bob receives the signed message, he uses the same hash algorithm in conjunction with Alice's public key. He raises the signature to the power of *e* mod *n* (as he does when encrypting a message), and compares the resulting hash value with the message's actual hash value. If the two agree, he knows that the author of the message was in possession of Alice's secret key, and that the message has not been tampered with since.

Note that secure padding schemes such as RSA-PSS are as essential for the security of message signing as they are for message encryption, and that the same key should never be used for both encryption and signing purposes

8.8 atacuri practice

Adaptive-chosen-ciphertext attacks were largely considered to be a theoretical concern until 1998, when Daniel Bleichenbacher of Bell Laboratories demonstrated a practical attack against systems using RSA encryption in concert with the PKCS #1 v1 encoding function, including a version of the Secure Socket Layer (SSL) protocol used by thousands of web servers at the time.

The Bleichenbacher attacks took advantage of flaws within the PKCS #1 function to gradually reveal the content of an RSA encrypted message. Doing this requires sending several million test ciphertexts to the decryption device (eg, SSL-equipped web server.) In practical terms, this means that an SSL session key can be exposed in a reasonable amount of time, perhaps a day or less.

8.8.1 timing attacks

Kocher described a new attack on RSA in 1995: if the attacker *Eve* knows *Alice's* hardware in sufficient detail and is able to measure the decryption times for several known ciphertexts, she can deduce the decryption key *d* quickly. This attack can also be applied against the RSA signature scheme. In 2003, Boneh and Brumley demonstrated a more practical attack capable of recovering RSA factorizations over a network connection (e.g., from a Secure Socket Layer (SSL)-enabled webserver). This attack takes advantage of information leaked by the Chinese remainder theorem optimization used by many RSA implementations.

One way to thwart these attacks is to ensure that the decryption operation takes a constant amount of time for every ciphertext. However, this approach can significantly reduce performance. Instead, most RSA implementations use an alternate technique known as cryptographic blinding. RSA blinding makes use of the multiplicative property of RSA. Instead of computing $c^d \bmod n$, Alice first chooses a secret random value r and computes $(r^e c)^d \bmod n$. The result of this computation is $r\,m \bmod n$ and so the effect of r can be removed by multiplying by its inverse. A new value of r is chosen for each ciphertext. With blinding applied, the decryption time is no longer correlated to the value of the input ciphertext and so the timing attack fails.

8.8.2 adaptive chosen ciphertext attacks

In 1998, Daniel Bleichenbacher described the first practical adaptive chosen ciphertext attack, against RSA-encrypted messages using the PKCS #1 v1 padding scheme (a padding scheme randomizes and adds structure to an RSA-encrypted message, so it is possible to determine whether a decrypted message is valid.) Due to flaws with the PKCS #1 scheme, Bleichenbacher was able to mount a practical attack against RSA implementations of the Secure Socket Layer protocol, and to recover session keys. As a result of this work, cryptographers now recommend the use of provably secure padding schemes such as Optimal Asymmetric Encryption Padding, and RSA Laboratories has released new versions of PKCS #1 that are not vulnerable to these attacks.

8.8.3 branch prediction analysis (BPA) attacks

Many processors use a branch predictor to determine whether a conditional branch in the instruction flow of a program is likely to be taken or not. Usually these processors also implement simultaneous multithreading (SMT). Branch prediction analysis attacks use a spy process to discover (statistically) the private key when processed with these processors.

Simple Branch Prediction Analysis (SBPA) claims to improve BPA in a non-statistical way. In their paper, "On the Power of Simple Branch Prediction Analysis", the authors of SBPA (Onur Aciicmez and Cetin Kaya Koc) claim to have discovered 508 out of 512 bits of an RSA key in 10 iterations.

chapter 9 infrastructura cheilor publice

9.1 ce este?

Infrastructura Cheilor Publice (PKI – Public Key infrastructure) este un sistem pentru crearea, distribuirea, identificarea și revocarea cheilor publice (cunoscute sub numele de certificate digitale). [PKIS]

Sistemul constă dintr-un sistem de încredere care definește:

- roluri
- politici
- proceduri
- elemente hardware
- elemente software

9.2 componentele infrastructurii

O implermentare PKI constă, în condiții normale, din următoarele 4 elemente:

- o autoritate de certificare (CA – Certificate Authority), care acționează ca rădăcină a arborelui de încredere. Aceste autorități eliberează certificate digitale care specifică proprietarul unei chei publice
- o autoritate de înregistrare (RA – Registration Authority) care validează înregistrarea unui certificat digital cu o cheie publică. Orice verificare a unui certificat digital ajunge la RA, care decide dacă să autentifice datele conținute în acesta
- o bază de date de certificate, care eliberează și revocă certificate digitale
- un depozit (store) în care sunt stocate certificatele cu cheile private asociate

9.3 chei publice

O cheie publică este un element care permite identificarea unei entități în spațiul electronic public.

- Protocol dezvoltat de Netscape pentru transmiterea de documente private prin internet.
- Folosește un sistem criptografic care utilizează două chei pentru criptarea datelor: o cheie publică cunoscută de toată lumea și o cheie privată (secretă) cunoscută doar de către proprietarul cheii

9.4 certificat digital

Structura unui certificat digital este specificată în standardul X.509. În versiunea 3 a acestui standard, structura este următoarea:

- versiune – versiunea standardului X.509 utilizată pentru crearea certificatului
- serie – identificator generat de către autoritatea care a emis certificatul
- identificator pentru algoritmul certificatului – combinația de fucție de hash și a algoritmului utilizat de emitent pentru semnarea certificatului
- numele emitentului – numele autorității de certificare (CA) care a emis certificatul
- perioada de validitate – data de start și de expirare a certificatului

- numele proprietarului certificatului
- informații despre cheia publică a proprietarului certificatului
- identificatorul unic al emitentului
- extensii (opționale)
- semnătura digitală creată de emitent (CA), utilizând algoritmul specificat mai sus

Iată mai jos un exemplu – certificatul digital al uvt.ro

- versiune – `V3`
- serie – `03 2e f6 9c a0 65 75 89 d9 59 5b 9e da 22 07 c1 c8 d6`
- identificator pentru algoritmul certificatului – `sha256RSA`
- numele emitentului – `CN = Let's Encrypt Authority X3, O = Let's Encrypt, C = US`
- perioada de validitate – start: `Saturday, September 28, 2019 6:53:27 AM`, end: `Friday, December 27, 2019 6:53:27 AM`
- numele proprietarului certificatului – `uvt.ro`
- informații despre cheia publică a proprietarului certificatului – `30 82 01 0a 02 82 01 01 00 db 7a 0d 3e 33 b0 64 0d aa 5c 93 27 bb c9 09 65 7b de 53 f2 c8 4f 53 be b2 c4 c1 ee 2a 85 94 aa 78 c4 03 fe ad 3c 71 1d 14 78 5e c7 62 7b d5 c4 1b d8 93 53 b1 86 c4 3c 3c 5d 1f 34 af 38 bd e0 af d6 dc 60 bf 31 35 27 f9 49 09 4f 89 bc 3f bd 3a a4 85 79 62 0d 3d 5e 98 76 28 19 6a 03 1b 36 0e 8b c8 2f 39 37 52 20 89 95 99 56 22 8e 6c dd 6a 4a df 1b d2 4a 00 53 ee 79 0a 87 f3 ce db a6 e7 a3 8f 6e 44 65 46 cf 67 db 48 3b 69 12 b2 fb d9 01 97 d6 d0 f6 74 af ba 67 78 cb 85 a1 28 cc f5 7c 62 5d fa 56 93 d0 4f 53 80 0d dd 2c 68 32 11 e0 cc 7b 24 b6 cd 2f 42 5a 79 ea 58 cc 30 36 68 8b 61 db 34 fa 18 51 b1 b6 45 a2 70 a1 2b 86 8e df 5b fe 83 48 33 97 ff 97 f6 df c1 04 25 f5 1a 7f 58 d6 5a 5f c7 c9 53 28 6b ed 7d 11 f1 46 41 d5 42 12 ab 39 5f 10 49 e4 8e 0a 0c 08 87 c3 02 03 01 00 01`
- identificatorul unic al emitentului – `64 3c 2a 7f 0a 5d 97 c8 7e 5e 4d 94 eb af 5f f6 e0 82 d5 8a`
- extensii (opționale)
- semnătura digitală creată de emitent (CA), utilizând algoritmul specificat mai sus – `cd cc de 94 da 53 f4 02 bc 33 3a a4 f0 10 b1 7f b7 82 ac cc`

9.5 autorități de certificare

9.5.1 ce este o autoritate de certificare?

O autoritate de certificare (CA) este o entitate (un furnizor de servicii de încredere, sau un furnizor de servicii de certificare) care emite certificate cu cheie publică, mai cunoscute sub numele de certificate electronice sau digitale. CA este, de asemenea, o parte crucială a infrastructurii cheilor publice (PKI), deoarece folosește un PKI pentru a emite sau revoca certificate de cheie publică și pentru a furniza declarații verificabile cu privire la starea lor.

Un certificat digital este un document digital care poate conține informații legate de identitatea proprietarului său, cea a emitentului și certificatul în sine. A avea atribute de identitate reale într-un certificat digital este opțional; în mod alternativ, poate cuprinde doar un pseudonim.

Certificatul permite utilizatorului să autentifice și să acceseze diverse servicii online sensibile (cum ar fi accesarea aplicațiilor de web banking, platformelor de servicii publice etc.). Certificatele digitale servesc, de

asemenea, companiilor și furnizorilor de servicii care le garantează că persoana care accesează serviciile lor este cea potrivită. [CA]

9.5.2 ce face o autoritate de certificare?

O autoritate de certificare este responsabilă pentru întreg ciclul de viață al unui certificat digital, și anume:

- Activarea unui certificat. Un certificat poate fi activat la eliberare. În general, înainte de a emite un certificat calificat, trebuie verificată identitatea utilizatorului pentru a ne asigura că persoana este cea care pretinde că este.
- Reactivarea unui certificat (în cazul în care este suspendat)
- Re-tastare. CA se poate ocupa și de re-introducerea certificatului. Certificatele au o perioadă de valabilitate limitată, în funcție de tipurile lor și de politicile PKI. De exemplu, certificatele LuxTrust sunt valabile pentru o perioadă de 3 ani.
- Revocarea unui certificat: CA este, de asemenea, autorizată să revoce un certificat care îl face ireversibil inutilizabil. Acest lucru se poate întâmpla dacă există suspiciuni că certificatul a fost compromis sau la cererea autorităților. Un certificat poate fi, de asemenea, revocat în cazul în care CA descoperă că o informație din certificat devine învechită sau inexactă.

9.5.3 cum se poate deveni autoritate de certificare?

Dacă o companie dorește să devină o autoritate de certificare, atunci trebuie să procure un PKI certificat, să definească procesele corespunzătoare și să creeze o politică de certificat (CP) și o declarație de practică (CPS) corespunzătoare. Apoi, acesta trebuie să fie recunoscut și certificat cu succes de către autoritățile competente. Atunci când sunteți o autoritate de certificare, respectarea reglementărilor este esențială. De fapt, autoritățile competente sunt auditate și supravegheate în mod regulat de către autorități pentru a se asigura că respectă legislația actuală. În caz de infracțiune, autoritățile competente sunt supuse unor consecințe grave care pot ajunge la suspendarea serviciilor lor.

9.6 tipuri de certificate

Nu toate autoritățile de certifcare oferă același tip de certificat. Diferența provine din setul de măsuri de verificare efectuat. Astfel, avem mai multe tipuri de certificate:

- Validare domeniu (Domain validation – DV) – doar domeniul este verificat pentru legitimare
- Wildcard – domeniul de bază și toate subdomeniile sale sunt incluse într-un singur certificat
- Validare extinsă (Extended validation – EV)
- Comunicații unificate (Unified communications – UC) – pentru criptarea email-urilor sau altor tipuri de comunicații. Poate include domenii multiple.
- Numele alternativ al subiectului (proprietarului) (Subject Alternative Name – SAN) – domeniul rădăcină și domenii asociate acestuia pot fi incluse într-un singur certificat
- Validare organizație (Organization validation – OV)

9.7 autorități de înregistare

O autoritate de înregistrare poate fi definită ca entitate secundară de încredere. Aceasta poate fi bancă, o instituție educațională, o companie, etc. [RA]

Autoritățile de înregistrare au rolul de a identifica clar viitorul proprietar al certificatului și de a administra datele ce implică această identificare.

Ce face o Autoritate de Înregistrare? Enumerăm principalele atribute:

- identifică solicitantul de certificat printr-o procedură care implică verificarea unui document oficial de identitate
- se asigură de concordanța datelor personale cu cele ce vor apărea în certificat
- administrează ciclul de viață al certificatului în cazul unei cereri de revocare sau reînnoire

9.8 infrastructura cheilor publice în Romania

În România, Ministerul Comunicațiilor și Societății Informaționale (MCSI) exercită atribuțiile de organism de supraveghere pentru prestatorii de servicii de încredere calificați, stabiliți pe teritoriul României, precum și luarea măsurilor, după caz, în legatură cu prestatorii de servicii de încredere necalificați stabiliți pe teritoriul României, în conformitate cu Regulamentul (UE) nr. 910/2014 al Parlamentului European și al Consiliului din 23 iulie 2014 privind identificarea electronică și serviciile de încredere pentru tranzacțiile electronice pe piața internă și de abrogare a Directivei 1999/93/CE.

Statutul de prestator de servicii de încredere calificat se acordă de către Organismul de Supraveghere, conform Regulamentului (UE) nr. 910/2014 și a OMCSI nr. 449/2017.

Semnăturile electronice emise de entitățile înscrise în Trusted List (lista de încredere) publicată la nivel UE sunt valabile în toate statele membre UE.

În Romania, autoritățile de certificare (acreditate de guvern) sunt:

- Trans Sped SRL
- Digisign SA
- Certsign SA
- Alfatrust Certification SA
- Centrul de Calcul SA
- SRI – Institutul pentru tehnologii avansate (?)

Pentru o listă la zi la nivelul Uniunii Europene, utilizați link-ul:- https://webgate.ec.europa.eu/tl-browser/

9.9 exemple din restul lumii

La nivelul **Uniunii Europene**, lista autorităților de cerificare de încredere este Trusted List, link-ul fiind cel din paragraful anterior. La nivel de țară, numărul autorităților de certificare de încredere variază între 1 (Cipru și Danemarca, de exemplu) și 34 (Spania). Spania e mai degrabă excepția, următoarele țări cu un număr mare de autorități de certificare de încredere fiind Franța și Italia (cîte 22), dar media este de 5-6.

În **Statele Unite** rețeaua de autorități de certificare constă din autorități rădăcină (root), intermediare (intermediate) și de emitere (issuing). [USPKI]

Activitatea acestora este guvernată de un set de reguli, cunoscut sub numele de Federal Common Policy Certification Authority (FCPCA), care servește ca root (rădăcină) și trust anchor (ancoră de încredere) pentru autoritățile intermediare și de înscriere operate de Agențiile ramurii executive ale guvernului federal.

Totodată, autoritățile de certificare sunt împărțite (la nivel orizontal) în mai multe categorii, și anume:

- PKI Shared Service Provider (SSP) – o entitate subordonată FCPCA
- Autorități de certificare din domeniul privat – acestea pot oferi servicii și guvernului federal
- Alte autorități de certificare guvernamentale – administrate de administrații la nivel de stat, local, tribal, teritorial sau internațional
- Autorități de certificare de legătură (Bridge CAs)

- Agenții Federale remanente (Legacy) – agenții care au investit și desfășurat propriile PKIs și Cas înainte de 2004

Asigurarea coerenței politicilor în domeniul autorităților de certificare se realizează prin intermediul unei autorități numite Federal BridgeCertification Aothority (FBCA).

In China, agenția responsabilă pentru domeniile .cn (CNNIC – China Internet Network Information Center) oferă servicii în calitate de Autoritate de certificare, deși cota de piață a acestei autorități este sub 0.1%. (conform [SSLMS])

9.10 principalele autorități de certificare la nivel mondial

Top five (conform [TOP5])

- Let's Encrypt – dacă e gratis cu plăcere. Oferă criptare RSA 2048-bit și are suportul unor companii precum Automatic, Mozilla, Sucuri, Facebook, Chrome. Suport pentru ECDSA în curs de implementare. Certificate gratis, instant, reînnoire gratis, nivel ridicat de compatibilitate. Certificate de tip DV, SAN, UC.
- Comodo
- Symantec
- Digicert
- Geotrust

În materie de cotă de piață pentru certificate SSL utilizate de site-uri web, lucrurile arată un pic diferit, conform [SSL-MS]:

- IdenTrust – 52.2%
- DigiCert Group – 19.3%
- Sectigo – 17.3%
- GoDaddy Group – 6.8%
- GlobalSign – 2.9%

chapter 10 standardul de semnătură digitală – DSS

10.1 ce este standardul de semnătură digitală?

DSS este un acronim pentru Digital Signature Standard, un standard al guvernului federal SUA, dezvoltat de către NSA (National security Agency), standard care precizează modalitatea de generare a semnăturilor digitale pentru autentificarea documentelor electronice.

Specificația algoritmului de semnătură digitală (DSA) este conținută în FIPS (Federal Information Processing Standard) 186. Ultima versiune (pentru nov. 2019) a acestei specificații este FIPS 186-4 și a apărut în iulie 2013 - https://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.186-4.pdf .

10.2 DSA – algoritmul de semnătură digitală

O semnătură digitală DSA se calculează utilizând un set de parametri de domeniu, o cheie privată x, un număr secret k specific fiecărui mesaj, datele care urmează a fi semnate precum și o funcție de hash.

Semnătura digitală este verificată utilizând aceiași parametri de domeniu, cheia publică y asociată cheii private x, datele ce urmează a fi verificate precum și funcția hash utilizată la generare.

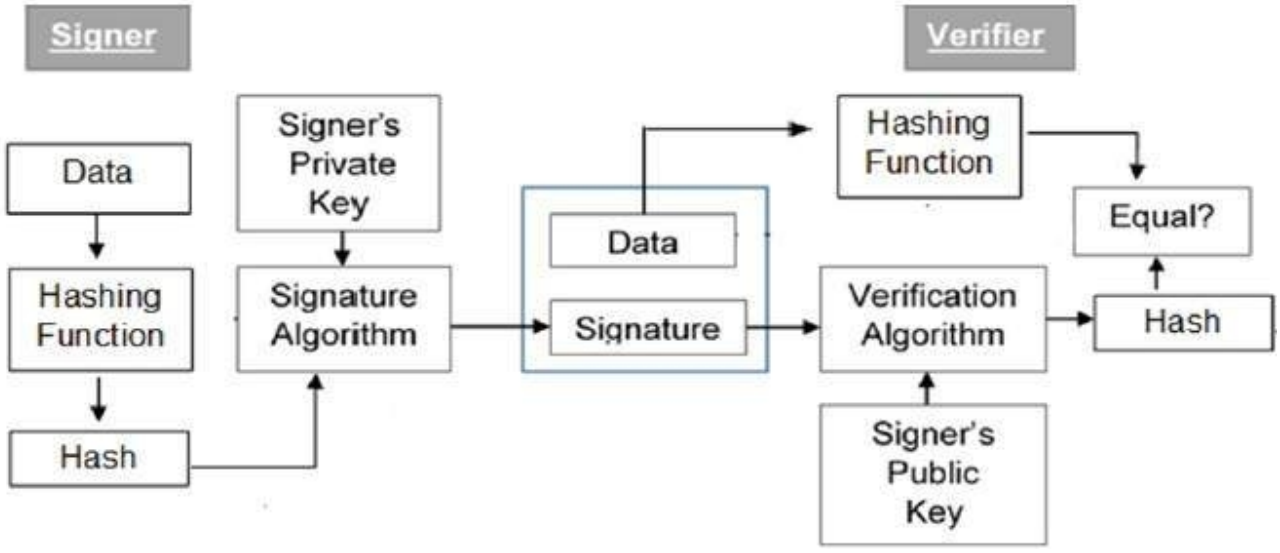


Fig. 10.1 Schema creării și verificării demnăturii digitale (poză de la tutorialspoint.com)

10.3 parametrii DSA

DSA utilizează următorii parametri:

1. p = un număr prim, cu 2^(L-1) < p < 2^L, unde 1024<= L <= 3072, L e multiplu de 64
2. q = un divizor prim al lui p-1, cu 2^(N-1) < q < 2^N, valorile lui N sunt 160, 224 sau 256
3. g = h(p-1)/q mod p, unde h este orice întreg cu 1 < h < p - 1 astfel încât h(p-1)/q mod p > 1

(g are ordinul q mod p)

4. x = un întreg random (aleatoriu) sau pseudorandom (seudo-aleatoriu) generat, cu 0 < x < q

5. y = gx mod p

6. k = un întreg random (aleatoriu) sau pseudorandom (seudo-aleatoriu) generat, cu 0 < k < q

Numerele p, q, și pot fi publice și pot fi comune pentru un grup de utilizatori. Cheile private și publice ale unui utilizator sunt x și y, respectiv. În mod normal, sunt fixate pentru o perioadă de timp. Parametrii x și k sunt folosiți numai pentru generarea de semnături și trebuie păstrați în secret. Parametrul k trebuie regenerat pentru fiecare semnătură.

Parametrii p și q sunt generați după cum se specifică în anexa 2 sau folosind alte metode de securitate aprobate de FIPS. Parametrii x și k sunt generați în conformitate cu specificațiile din anexa 3 sau folosind alte metode de securitate aprobate de FIPS.

10.4 generarea semnăturii DSA

Semnătura unui mesaj M este perechea de numere r și s calculate după ecuațiile de mai jos:

r = (gk mod p) mod q

s = (k^-1(SHA-1(M) + xr)) mod q.

În cele de mai sus, k^-1 este inversul multiplicativ al lui k, mod q; adică, (k^-1 k) mod q = 1 și 0 <k^-1 <q. Valoarea SHA-1 (M) este o ieșire de 160 biți din algoritmul Secure Hash specificat în FIPS 180-1.

Pentru utilizare în calcularea lui s, acest șir trebuie convertit într-un număr întreg. Regula de conversie este prezentată în anexa 2.2.

Ca opțiune, se poate verifica dacă r = 0 sau s = 0. Dacă r = 0 sau s = 0, ar trebui să se genereze o nouă valoare a lui k, iar semnătura să fie recalculată (este extrem de puțin probabil ca r = 0 sau s = 0 dacă semnăturile sunt generate corect).

Semnătura este transmisă împreună cu mesajul către verificator.

10.5 verificarea semnăturii DSA

Înainte de a verifica semnătura într-un mesaj semnat, p, q și g precum și cheia publică și identitatea expeditorului sunt puse la dispoziția verificatorului într-o manieră autentificată.

Fie M ', r' și s ' versiunile primite ale lui M, r și s, respectiv, și fie y cheia publică a semnatarului. Pentru a verifica semnătura, verificatorul verifică mai întâi faptul că 0 <r '<q și 0 <s' <q; în cazul în care oricare dintre condiții este încălcată, semnătura va fi respinsă. Dacă aceste două condiții sunt îndeplinite, verificatorul calculează

w = (s')-1 mod q

u1 = ((SHA-1(M'))w) mod q

u2 = ((r')w) mod q

v = (((g)u1 (y)u2) mod p) mod q.

Dacă v = r ', atunci semnătura este verificată și verificatorul poate avea încredere mare că mesajul primit a fost trimis de partea celui care deține cheia secretă x corespunzătoare lui y. Pentru o dovadă că v = r 'când M' = M, r '= r și s' = s, a se vedea anexa 1.

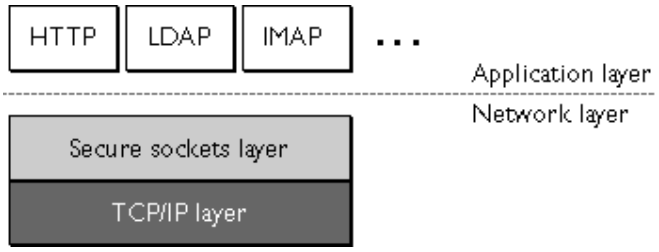
Dacă v nu este egal cu r ', atunci este posibil ca mesajul să fi fost modificat, mesajul să fi fost semnat incorect de către semnatar sau este posibil să fi fost semnat de un impostor. Mesajul trebuie considerat nevalid.

chapter 11 secure socket layer – SSL, TLS

11.1 SSL (Secure Sockets Layer)

- Protocol dezvoltat de Netscape pentru transmiterea documentelor private prin internet
- Utilizează un sistem criptografic care folosește 2 chei pentru criptarea datelor: o cheie publică cunoscută de toată lumea și o cheie privată sau secretă cunoscută numai destinatarului mesajului

11.2 protocolul SSL



Protocolul SSL rulează peste TCP / IP și sub protocoale de nivel superior, cum ar fi HTTP sau IMAP. Utilizează TCP / IP în numele protocoalelor de nivel superior și, în acest proces, permite unui server capabil de SSL să se autentifice la un client compatibil SSL, permite clientului să se autentifice pe server și permite ambelor mașini să stabilească o conexiune criptată.

- **Autentificarea serverului SSL** permite utilizatorului să confirme identitatea serverului
- **Autentificarea clientului SSL** permite unui server să confirme identitatea utilizatorului
- O **conexiune SSL criptată** necesită ca toate informațiile trimise între un client și un server să fie criptate de software-ul care trimite și decriptate de software-ul primitor, oferind astfel un grad ridicat de confidențialitate

11.3 strângerea de mână (handshake) SSL

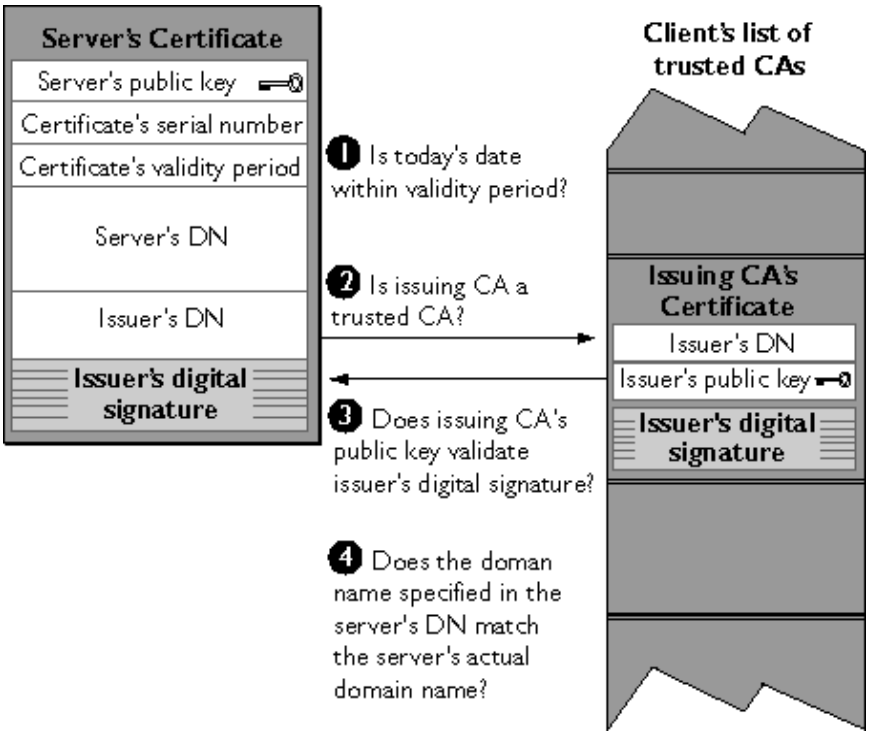
1. Clientul trimite serverului numărul de versiune SSL al clientului, setările de cifrare, datele aleatorii generate și alte informații de care serverul are nevoie ca să comunice cu clientul folosind SSL.
2. Serverul trimite clientului numărul de versiune SSL al serverului, setările de cifrare, datele aleatorii generate și alte informații de care clientul are nevoie pentru a comunica cu serverul prin SSL. Serverul trimite, de asemenea, propriul său certificat și, dacă clientul solicită o resursă server care necesită autentificarea clientului, solicită certificatul clientului.
3. Clientul folosește o parte din informațiile trimise de server pentru a autentifica serverul (consultați autentificarea serverului pentru detalii). Dacă serverul nu poate fi autentificat, utilizatorul este avertizat despre problemă și este informat că nu poate fi stabilită o conexiune criptată și autentificată. Dacă serverul poate fi autentificat cu succes, clientul trece la Pasul 4.
4. Folosind toate datele generate în strângerea de mână până acum, clientul (cu cooperarea serverului, în funcție de cifrul folosit) creează secretul premaster pentru sesiune, îl criptează cu cheia publică a serverului (obținut din certificatul serverului , trimis la pasul 2) și trimite secretul premaster criptat către server.

5. Dacă serverul a solicitat autentificarea clientului (un pas opțional în strângerea de mână), clientul semnează, de asemenea, un pachet de date care este unic pentru această strângere de mână și cunoscută atât de client cât și de server. În acest caz, clientul trimite atât datele semnate, cât și propriul certificat al clientului către server împreună cu secretul premaster criptat.
6. Dacă serverul a solicitat autentificarea clientului, serverul încearcă să autentifice clientul (consultați Autentificarea client pentru detalii). Dacă clientul nu poate fi autentificat, sesiunea este încheiată. Dacă clientul poate fi autentificat cu succes, serverul folosește cheia sa privată pentru a decripta secretul premaster, apoi efectuează o serie de pași (pe care clientul îi realizează pornind de la același secret premaster) pentru a genera secretul master.
7. Atât clientul cât și serverul folosesc secretul master pentru a genera cheile de sesiune, care sunt chei simetrice utilizate pentru criptarea și decriptarea informațiilor schimbate în timpul sesiunii SSL și pentru a verifica integritatea acesteia - adică pentru a detecta orice modificare a datelor între momentul în care a fost trimis și momentul în care este primit prin conexiunea SSL.
8. Clientul trimite un mesaj către server prin care îl informează că mesajele viitoare de la client vor fi criptate cu cheia de sesiune. Apoi trimite un mesaj separat (criptat) care indică faptul că porțiunea client a strângerii de mână a fost terminată.
9. Serverul trimite un mesaj către client informându-l că mesajele viitoare de la server vor fi criptate cu cheia de sesiune. Apoi trimite un mesaj separat (criptat) care indică faptul că porțiunea de server a strângerii de mână este terminată.
10. Strângerea de mână SSL este acum completă și sesiunea SSL a început. Clientul și serverul folosesc cheile de sesiune pentru a cripta și decripta datele pe care le transmit reciproc și pentru a-i valida integritatea.

11.4 autentificarea serverului

În cazul autentificării serverului, clientul criptează secretul premaster cu cheia publică a serverului. Doar cheia privată corespunzătoare poate decripta corect secretul, astfel încât clientul are o anumită certitudine că identitatea asociată cu cheia publică este de fapt serverul cu care este conectat clientul. În caz contrar, serverul nu poate decripta secretul premaster și nu poate genera cheile simetrice necesare sesiunii, iar sesiunea va fi încheiată.

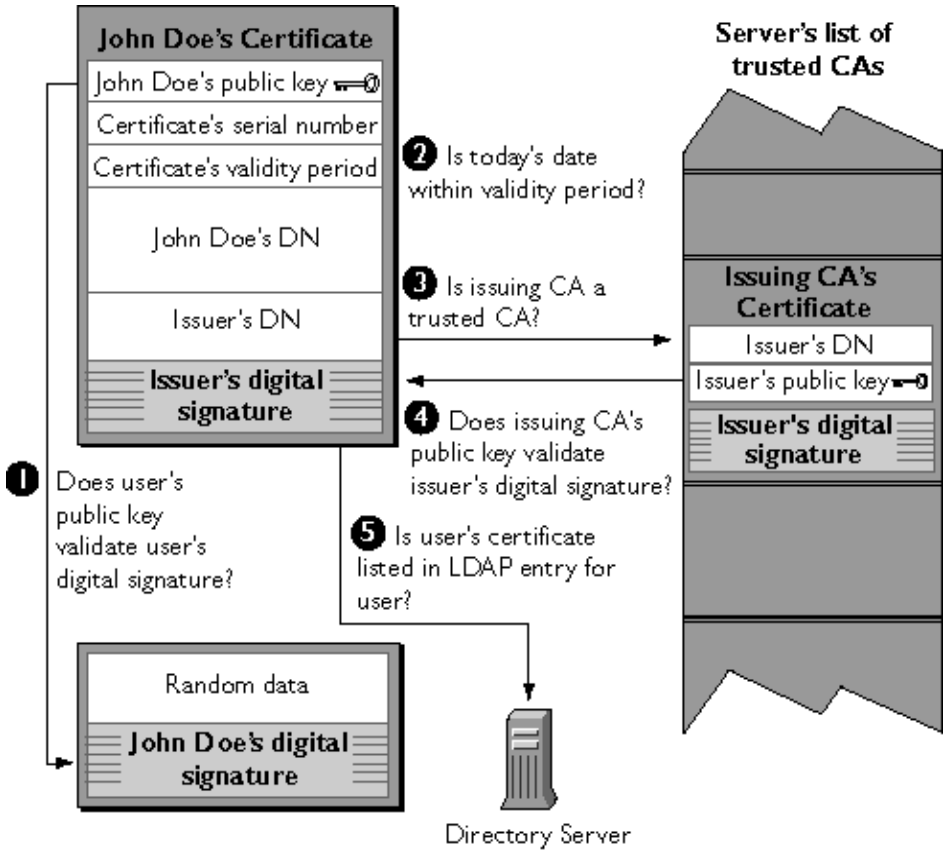
Exemplu: Cum autentifică un client Netscape un certificat de server



11.5 autentificarea clientului

În cazul autentificării clientului, clientul criptează unele date aleatorii cu cheia privată a clientului - adică creează o semnătură digitală. Cheia publică din certificatul clientului poate valida corect semnătura digitală numai dacă a fost utilizată cheia privată corespunzătoare. În caz contrar, serverul nu poate valida semnătura digitală și sesiunea este încheiată.

Exemplu: Cum autentifică un client Netscape un certificat de server



11.6 algoritmi de criptare (cifruri) utilizate cu SSL

- **DES**. Data Encryption Standard, un algoritm de criptare utilizat de guvernul S.U.A.
- **AES**. Advanced Encryption Standard – algoritmul folosit curent pentru încripatre simetrică
- **DSA**. Algoritmul de semnătură digitală, parte a standardului de autentificare digitală utilizat de guvernul S.U.A.
- **KEA**. Algoritmul de schimb de chei, un algoritm utilizat pentru schimbul de chei de către guvernul Statelor Unite.
- **MD5**. Algoritmul Message Digest dezvoltat de Rivest.
- **RC2** și **RC4**. Algoritmul lui Rivest de criptare dezvoltat pentru RSA Data Security.
- **RSA**. Un algoritm cu cheie publică atât pentru criptare cât și pentru autentificare. Dezvoltat de Rivest, Shamir și Adleman.
- Schimb de chei **RSA**. Un algoritm de schimb de chei pentru SSL bazat pe algoritmul RSA.
- **SHA-1**. Algoritmul Secure Hash 1, o funcție de hash folosită de guvernul SUA.
- **SKIPJACK**. Un algoritm secretizat cu cheie simetrică implementat într-un hardware compatibil FORTEZZA utilizat de guvernul SUA.
- **TRIPLE DES**. DES aplicat de trei ori

chapter 12 secure shell

12.1 istoric și dezvoltare

12.1.1 versiunea 1.0

În 1995, Tatu Ylönen, un cercetător la Universitatea de Tehnologie din Helsinki, Finlanda, a proiectat prima versiune a protocolului (acum numită SSH-1), determinată de un atac de furt de parole din rețeaua sa universitară. Scopul SSH a fost înlocuirea protocoalelor anterioare rlogin, TELNET și rsh, care nu ofereau o autentificare puternică sau nu garantau confidențialitatea. Ylönen a lansat implementarea ca freeware în iulie 1995, iar instrumentul a câștigat rapid în popularitate. Spre sfârșitul anului 1995, baza de utilizatori SSH a crescut la 20.000 de utilizatori în cincizeci de țări.

În decembrie 1995, Ylönen a fondat SSH Communications Security pentru a comercializa și dezvolta SSH. Versiunea originală a software-ului SSH a folosit diverse părți de software gratuite, cum ar fi GNU libgmp, dar versiunile ulterioare lansate de SSH Secure Communications au evoluat spre software tot mai proprietar.

12.1.2 versiunea 2.0

„Secsh” a fost numele oficial al grupului de lucru IETF (Internet Engineering Task Force) responsabil pentru versiunea 2 a protocolului SSH. În 1996, o versiune revizuită a protocolului, SSH-2, a fost adoptată ca standard. Această versiune este incompatibilă cu SSH-1. SSH-2 prezintă atât îmbunătățiri de securitate, cât și ale funcționalității față de SSH-1. O mai bună securitate, de exemplu, vine prin schimbul de chei Diffie-Hellman și verificarea puternică a integrității prin intermediul codurilor de autentificare a mesajelor. Noile caracteristici ale SSH-2 includ posibilitatea de a rula orice număr de sesiuni shell pe o singură conexiune SSH.

12.2 OpenSSH

În 1999, dezvoltatorii care doreau să aibă disponibilă o versiune de software gratuită au revenit la versiunea anterioară 1.2.12 a programului SSH inițial, care a fost ultima lansată sub licență open source. OSSH-ul lui Björn Grönvall a fost ulterior dezvoltat din această bază de cod. La scurt timp după aceea, dezvoltatorii OpenBSD au creat o nouă ramură a codului lui Grönvall și au lucrat pe larg la acesta, creând OpenSSH, care a fost livrat cu versiunea 2.6 a OpenBSD. Din această versiune, s-a format o ramură de „portabilitate” pentru a porta OpenSSH către alte sisteme de operare.

Se estimează că, începând cu anul 2000, existau 2.000.000 de utilizatori de SSH.

Începând cu 2005, OpenSSH este cea mai populară implementare SSH, venită implicit într-un număr mare de sisteme de operare. Între timp, OSSH a devenit învechită.

12.3 standardul de internet SSH-2

12.3.1 publicare originală

În 2006, protocolul SSH-2 menționat mai sus a devenit un standard de internet propus, cu publicarea de către grupul de lucru RFC de la IETF „secsh”. A fost publicat pentru prima dată în ianuarie 2006.

- RFC 4250, The Secure Shell (SSH) Protocol Assigned Numbers
- RFC 4251, The Secure Shell (SSH) Protocol Architecture
- RFC 4252, The Secure Shell (SSH) Authentication Protocol

- RFC 4253, The Secure Shell (SSH) Transport Layer Protocol
- RFC 4254, The Secure Shell (SSH) Connection Protocol
- RFC 4255, Using DNS to Securely Publish Secure Shell (SSH) Key Fingerprints
- RFC 4256, Generic Message Exchange Authentication for the Secure Shell Protocol (SSH)
- RFC 4335, The Secure Shell (SSH) Session Channel Break Extension
- RFC 4344, The Secure Shell (SSH) Transport Layer Encryption Modes
- RFC 4345, Improved Arcfour Modes for the Secure Shell (SSH) Transport Layer Protocol

12.3.2 modificări ulterioare

Ulterior a fost modificat și extins de următoarele publicații.

- RFC 4419, Diffie-Hellman Group Exchange for the Secure Shell (SSH) Transport Layer Protocol (March 2006)
- RFC 4432, RSA Key Exchange for the Secure Shell (SSH) Transport Layer Protocol (March 2006)
- RFC 4716, The Secure Shell (SSH) Public Key File Format (Nov 2006)

12.4 utilizări

Exemplu de tunelare a unei aplicații X11 peste SSH: utilizatorul „josh” utilizează SSH de la mașina locală „foofighter” la mașina de la distanță „tengwar” pentru a rula xeyes.

Conectarea la OpenWrt prin SSH folosind PuTTY care rulează pe Windows.

SSH este un protocol care poate fi utilizat pentru multe aplicații pe mai multe platforme, inclusiv UNIX, Microsoft Windows, Apple Mac și Linux. Unele dintre aplicațiile de mai jos pot necesita funcții care sunt disponibile sau compatibile doar cu clienți sau servere SSH specifice. De exemplu, utilizarea protocolului SSH pentru a implementa un VPN este posibilă, dar în prezent doar cu serverul OpenSSH și implementarea clientului.

- Pentru conectare la un shell pe o gazdă de la distanță (înlocuirea Telnet și rlogin)
- Pentru executarea unei singure comenzi pe o gazdă la distanță (înlocuirea rsh)
- Pentru copierea fișierelor de pe un server local pe o gazdă la distanță. Consultați SCP, ca o alternativă pentru rcp
- În combinație cu SFTP, ca o alternativă sigură la transferul de fișiere FTP
- În combinație cu rsync to backup, copiați fișierele în mod eficient și sigur
- Pentru redirecționarea portului sau tunelarea unui port (nu trebuie confundat cu o VPN care rutează pachetele între diferite rețele sau poduri două domenii de difuzare într-unul singur).
- Pentru utilizarea ca VPN criptat cu drepturi depline. Rețineți că numai serverul și clientul OpenSSH acceptă această caracteristică.
- Pentru redirecționarea X11 prin mai multe gazde
- Pentru navigarea pe web printr-o conexiune proxy criptată cu clienții SSH care acceptă protocolul SOCKS.
- Pentru montarea sigură a unui director pe un server la distanță ca sistem de fișiere pe un computer local folosind SSHFS.
- Pentru monitorizarea și gestionarea automată de la distanță a serverelor prin unul sau mai multe dintre mecanismele descrise mai sus.

- Pentru colaborarea sigură a mai multor utilizatori de canale SSH, unde este posibilă transferul, schimbul, schimbul și recuperarea sesiunilor deconectate.

12.5 arhitectură

Diagrama pachetului binar SSH-2.

Protocolul SSH-2 are o arhitectură internă (definită în RFC 4251) cu straturi bine separate. Acestea sunt:

- Stratul de transport (RFC 4253). Acest strat gestionează schimbul inițial de chei și autentificarea serverului și stabilește criptarea, compresia și verificarea integrității. Acesta expune stratului superior o interfață pentru trimiterea și primirea pachetelor sub formă de text de până la 32.768 de octeți fiecare (mai multe pot fi permise de implementare). Stratul de transport prevede, de asemenea, schimbul de chei, de obicei după ce 1 GB de date au fost transferate sau după ce a trecut 1 oră, oricare dintre acestea este mai devreme.
- Stratul de autentificare a utilizatorului (RFC 4252). Acest strat gestionează autentificarea clientului și oferă o serie de metode de autentificare. Autentificarea este bazată pe client: atunci când se solicită o parolă, poate fi solicitat clientul SSH, nu server. Serverul nu răspunde decât la solicitările de autentificare ale clientului. Metodele de autentificare utilizate pe scară largă includ următoarele:
    - parolă: o metodă de autentificare simplă a parolei, inclusiv o facilitate care permite schimbarea unei parole. Această metodă nu este implementată de toate programele.
    - publickey: o metodă pentru autentificarea bazată pe cheie publică, care acceptă, de regulă, cel puțin pachetele de chei DSA sau RSA, alte implementări acceptând și certificatele X.509.
    - Tastatură interactivă (RFC 4256): o metodă versatilă în care serverul trimite una sau mai multe solicitări de introducere a informațiilor, iar clientul le afișează și trimite răspunsuri introduse de utilizator. Folosit pentru a furniza o singură dată autentificare cu parolă, cum ar fi S / Key sau SecurID. Folosit de unele configurații OpenSSH atunci când PAM este furnizorul de autentificare gazdă de bază pentru a furniza în mod eficient autentificarea cu parolă, uneori ducând la imposibilitatea de a vă autentifica cu un client care acceptă doar metoda de autentificare simplă a parolei.
    - Metodele de autentificare GSSAPI care oferă o schemă extensibilă pentru a efectua autentificarea SSH folosind mecanisme externe precum Kerberos 5 sau NTLM, oferind un singur semn pe capacitatea sesiunilor SSH. Aceste metode sunt de obicei implementate prin implementări comerciale SSH pentru utilizare în organizații, deși OpenSSH are o implementare GSSAPI funcțională.
- Stratul de conexiune (RFC 4254). Acest nivel definește conceptul de canale, solicitări de canale și solicitări globale folosind serviciile SSH care sunt furnizate. O singură conexiune SSH poate găzdui mai multe canale simultan, fiecare transferând date în ambele direcții. Solicitările de canal sunt utilizate pentru a transmite date specifice canalului în afara bandei, cum ar fi dimensiunea modificată a unei ferestre a terminalului sau codul de ieșire al unui proces din partea serverului. Clientul SSH solicită să fie redirecționat un port din partea serverului folosind o solicitare globală. Tipurile de canale standard includ:
    - shell pentru shell-uri terminale, solicitări SFTP și exec (inclusiv transferuri SCP)
    - direct-tcpip pentru conexiunile transmise client-server
    - redirecționat-tcpip pentru conexiunile transmise de la server la client
- Înregistrarea SSHFP DNS (RFC 4255) oferă amprentele cheie ale gazdelor publice pentru a ajuta la verificarea autenticității gazdei.

Această arhitectură deschisă oferă o flexibilitate considerabilă, permițând SSH-ului să fie utilizat într-o varietate de scopuri dincolo de shell-ul securizat. Funcționalitatea singurului strat de transport este comparabilă cu TLS; stratul de autentificare a utilizatorului este extrem de extensibil cu metode de autentificare personalizate; iar stratul de conexiune oferă posibilitatea de a multiplexa multe sesiuni secundare într-o singură conexiune SSH, o caracteristică comparabilă cu BEEP și care nu este disponibilă în TLS.

12.6 probleme de securitate

Deoarece SSH-1 are defecte de proiectare inerente care îl fac vulnerabil (de exemplu, atacuri om-în-mijloc), acum este în general învechit și ar trebui evitat prin dezactivarea explicită a recesiunii la SSH-1. În timp ce majoritatea serverelor și clienților moderni acceptă SSH-2, unele organizații utilizează în continuare software fără suport pentru SSH-2 și astfel SSH-1 nu poate fi întotdeauna evitat.

În toate versiunile SSH, este important să verificați cheile publice necunoscute înainte de a le accepta ca valide. Acceptarea cheii publice a unui atacator ca cheie publică validă are ca efect dezvăluirea parolei transmise și permite atacuri de tipul omului în mijloc.

chapter 13 securitatea datelor

Securitatea datelor este modalitatea de a asigura protejarea datelor împotriva modific[rilor nedorite și că accesul la acestea este controlat în mod adecvat. Astfel, securitatea datelor ajută la asigurarea confidențialității. De asemenea, ajută la protejarea datelor cu caracter personal.

În Marea Britanie, Legea privind protecția datelor este utilizată pentru a se asigura că datele personale sunt accesibile celor pe care-l privește și oferă despăgubiri persoanelor fizice în cazul în care există inexactități. Acest lucru este deosebit de important pentru a se asigura că persoanele sunt tratate în mod echitabil, de exemplu în scopul verificării creditului. Legea privind protecția datelor prevede că numai persoanele fizice și companiile cu motive legitime și legale pot prelucra informații personale și că acestea nu pot fi partajate.

Standardul internațional ISO / IEC 17799 acoperă securitatea datelor sub subiectul securității informațiilor, iar unul dintre principiile sale cardinale este că toate informațiile stocate, adică datele, ar trebui să fie deținute astfel încât să fie clar a cui este responsabilitatea de a proteja și controla accesul la aceste date. [DATS]

13.1 coruperea datelor

Coruperea (modificările nedorite) datelor se referă la erorile din datele computerului care apar în timpul transmiterii sau recuperării, introducând modificări neintenționate la datele originale. Sistemele de stocare și transmisie a computerelor utilizează o serie de măsuri pentru a asigura integritatea datelor, lipsa erorilor.

Coruperea datelor în timpul transmisiei are o varietate de cauze. Întreruperea transmiterii datelor provoacă pierderi de informații. Condițiile de mediu pot interfera cu transmisia de date, în special atunci când avem de-a face cu metode de transmisie wireless. Norii puternici pot bloca transmisiile prin satelit. Rețelele wireless sunt susceptibile la interferențe de la dispozitive precum cuptoare cu microunde.

Pierderea datelor în timpul stocării are două mari cauze: eșecul hardware și software. Defecțiunea hard-disk-ului se încadrează în prima categorie, în timp ce eșecul software apare de obicei din cauza erorilor din cod.

Când coruperea datelor se comportă ca un proces Poisson, în care fiecare bit de date are o probabilitate independentă de modificare, coruperea datelor poate fi în general detectată prin utilizarea sumelor de verificare și poate fi adesea corectată prin utilizarea codurilor de corectare a erorilor.

Dacă este detectată o corupere necorectată a datelor, se pot aplica proceduri precum retransmisie automată sau restaurare din copii de rezervă. Matricele de discuri RAID, stochează și evaluează biți de paritate pentru datele dintr-un set de hard disk-uri și pot reconstrui datele corupte în urma eșecului unui singur disc.

Dacă sunt folosite mecanisme adecvate pentru detectarea și remedierea corupției datelor, integritatea datelor poate fi menținută. Acest lucru este deosebit de important în sectorul bancar, în cazul în care o eroare nedetectată poate afecta drastic un sold al contului, cât și în utilizarea datelor criptate sau comprimate, în cazul în care o mică eroare poate face un set de date extinse inutilizabil.

13.2 confidențialitatea datelor

Confidențialitatea datelor este relația dintre colectarea și diseminarea datelor, tehnologiile implicate, așteptările publicului în ce privește ocrotirea vieții private și problemele legale din domeniu.

Problemele de confidențialitate există oriunde sunt colectate și stocate informații de identificare personală - în formă digitală sau în alt mod. Controlul necorespunzător sau inexistent al divulgării poate fi cauza principală a problemelor de confidențialitate. Probleme de confidențialitate a datelor pot apărea ca răspuns la informații dintr-o gamă largă de surse, cum ar fi:

- registre medicale

- investigații și proceduri de justiție penală
- instituții și tranzacții financiare
- trăsături biologice, cum ar fi materialul genetic
- înregistrări privind rezidență și locația
- etnie

Provocarea confidențialității datelor constă în partajarea datelor, protejând în același timp informațiile de identificare personală. Domeniul securității datelor și al securității informațiilor are rolul de a proiecta și utiliza software, hardware și resurse umane pentru a rezolva această problemă.

13.3 remanența datelor

Remanența (persistența) datelor este reprezentarea reziduală a datelor care au fost, într-un fel, șterse sau eliminate în mod nominal. Acest reziduu se poate datora datelor rămase intacte printr-o operație de ștergere nominală sau prin proprietățile fizice ale mediului de stocare. Remanența datelor poate face posibilă dezvăluirea accidentală a informațiilor sensibile, în cazul în care suportul de stocare este eliberat într-un mediu necontrolat (de exemplu, aruncat în coșul de gunoi sau dat unui terț).

De-a lungul timpului, s-au dezvoltat diverse tehnici pentru combaterea remanenței datelor. În funcție de eficacitatea și intenția, acestea sunt adesea clasificate ca fiind fie de compensare sau de purjare / sanitizare. Metodele specifice includ suprascrierea, demagnetizarea, criptarea și distrugerea fizică.

13.4 scurgerea de date

Scurgerile de date pot include incidente, cum ar fi furtul sau pierderea de mass-media digitale, cum ar fi hard disk-uri sau laptop-uri care conțin astfel de suporturi pe care aceste informații sunt stocate necriptat, postarea de astfel de informații pe web la nivel mondial sau de pe un calculator accesibil la Internet fără precauții adecvate de securitate a informațiilor, transferul acestor informații către un sistem care nu este complet deschis, dar nu este acreditat în mod corespunzător sau formal pentru securitate la nivelul aprobat, cum ar fi e-mailul necriptat sau transferul acestor informații către sistemele de informații ale unei agenții eventual ostilă, cum ar fi o corporație concurentă sau o națiune străină, unde poate fi expusă la tehnici de decriptare mai intense.

13.5 furtul datelor

Furtul de date este o problemă în creștere comisă în primul rând de angajații din birouri cu acces la tehnologie, cum ar fi computere desktop și dispozitive portabile capabile să stocheze informații digitale, precum unități flash, iPod-uri și chiar camere digitale. Întrucât angajații petrec adesea o cantitate considerabilă de timp dezvoltând contacte și informații confidențiale și protejate de drepturi de autor pentru compania pentru care lucrează, ei simt adesea că au un anumit drept la informații și sunt înclinați să copieze și / sau să șteargă o parte din aceasta atunci când părăsesc compania, sau să le folosească greșit în timp ce sunt încă la muncă.

În timp ce majoritatea organizațiilor au implementat firewall-uri și sisteme de detectare a intruziunilor, foarte puțini iau în considerare amenințarea venită din partea angajatului mediu care copiază datele proprii pentru câștig sau utilizare personală de către o altă companie. Un scenariu obișnuit este acela în care o persoană de vânzări face o copie a bazei de date de contact pentru a fi utilizată în următoarea lor slujbă. De obicei, aceasta este o încălcare clară a condițiilor lor de angajare.

Daunele cauzate de furtul de date pot fi considerabile, având capacitatea de a transmite fișiere foarte mari prin e-mail, pagini web, dispozitive USB, stocare DVD și alte dispozitive portabile. Dispozitivele media amovibile sunt din ce în ce mai mici, cu o capacitate crescută de stocare, iar activități precum podslurping

sunt din ce în ce mai frecvente. Acum este posibil să stocați 1 TB de date pe un dispozitiv care să încapă în buzunarul unui angajat, date care ar putea contribui la căderea unei afaceri.

13.6 separarea protecției și securității

În informatică, separarea protecției și securității este o alegere de proiectare. Se poate defini **protecția** ca mecanism și **securitatea** ca politică, făcând astfel distincția protecție-securitate ca un caz particular al principiului distincției mecanism-politică.

Adoptarea acestei distincții într-o arhitectură de calculator, de obicei, înseamnă că protecția este asigurată ca un mecanism de toleranță la erori de hardware / firmware și kernel, susținerea sistemului de operare și aplicațiile care rulează pe partea de sus în punerea în aplicare a politicilor lor de securitate. În acest proiect, politicile de securitate se bazează, prin urmare, pe mecanismele de protecție și pe tehnici suplimentare de criptografie.

Cele două abordări hardware majore pentru securitate și / sau protecție sunt domeniile de protecție ierarhice (arhitecturi cu inel cu „modul supraveghetor” și „mod utilizator”) și adresare bazată pe capacitate.

Prima abordare adoptă o politică deja la nivelurile de arhitectură inferioare (hw / firmware / kernel), restricționând restul sistemului să se bazeze pe ea; prin urmare, alegerea de a distinge între protecție și securitate în proiectarea arhitecturii generale duce la respingerea demersului ierarhic în favoarea abordării bazate pe capacități.

Modelul Bell-LaPadula este un exemplu de model în care protecția și securitatea nu sunt separate. În Landwehr 1981 există un tabel care arată care modele pentru securitatea computerului separă mecanismul de protecție și politica de securitate. Cele cu separare sunt: matricea de acces, UCLA Data Secure Unix, preluarea și filtrarea; iar dintre cele fără separare menționăm: high-water mark, Bell și LaPadula (original și revizuit), fluxul de informații, dependență puternică și constrângeri.

chapter 14 controlul accesului

14.1 access control lists

14.2 file system ACLs

For file systems, the access control lists are operating system specific data structures which specify individual or group rights to certain objects, like files, directories or processes.

14.2.1 the NTFS example

NTFS – New Technology File System is the standard file system for MS based operating systems, starting with Windows NT. Besides support for ACLs, NTFS offers file system journaling, as well.

14.3 network ACLs

14.4 passwords

14.5 password efficiency

14.6 password storing

14.7 passwords over the network

14.8 password breaking

chapter 15 securitatea rețelelor

chapter 16 amenințări specifice internetului

16.1 generalități

Virușii, atacurile hackerilor și alte amenințări cibernetice sunt acum o parte din viața de zi cu zi. Malware este raspândit pe tot internetul, hackerii fura date confidențiale și casuțe de email invadate de spam sunt prețul pe care oamenii îl plătesc pentru comoditatea de calcul. Orice calculator sau rețea neprotejată sunt vulnerabile.

Utilizatorii de acasă pot pierde date personale valoroase cu un singur click pe un site greșit. Jocurile pentru copii schimbă virușii fără ca cineva să știe. Primești un mail care solicita o actualizare a detaliilor de plată și un hacker obține accesul la contul vostru bancar. Pe calculator se instaleaza o pagină ascunsă și computerul dumneavoastră devine un zombi.

Internetul a devenit o resursă critică pe care oamenii se bazează pe el pentru a-și face munca sau pentru diverstiment. Ei folosesc internetul pentru a efectua cercetări și pentru a aduna informații. Oamenii folosesc email-ul sau rețelele de socializare populare care să-i ajute să păstreze legătura cu colegii și clienții. Încărcarea, descărcarea, partajarea fișierelor și alte produse de lucru sunt acum activități de zi cu zi.

Din păcate, atunci când oamenii îndeplinesc aceste saricii zilnic, expun companiile pentru care lucrează la riscuri grave de securitate. Angajații trebuie să fie preocupați mai mult decât sa prevină pur și simplu oamenii care fac lucruri la serviciu pe care aceștia nu ar trebui sa le facă – navigarea pe site-uri restricționate sau necorespunzătoare, de exemplu. Acum oamenii sunt expuși la amenințări nocive, distructive în timp ce își fac pur și simplu meseria. Companiile ar trebui sa examineze măsurile IT de securitate a acestora și sa determine dacă sunt suficiente pentru protecția împotriva acestor amenințări transmise prin web.

Firewall-urile Gateway și antivirusul nu pot proteja împotriva complexului și multitudinea de programe daunatoare care amenința infrastructurile IT. Firewall-urile pot detecta traficul web, dar majoritatea nu au nici un mijloc de monitorizare a informațiilor specifice care sunt transferate. Soluțile antivirusului sunt reactive, nu preventive; sunt eficiente numai împotriva amenințărilor foarte specifice și oferă o protecția limitată numai după ce un atac a avut deja loc. Organizațiile trebuie să își completeze sistemele de securitate existente cu o soluție care să completeze aceste măsuri de protecție la nivel de conținut,

16.2 expunerea la amenințări

Amenințările la care oameni sunt expuși zilnic, variază în funcție de ceea ce fac aceștia ce internet.

În paragrafele următoare prezentăm unele dintre aceste amenințări pe baza activității principale în care sunt angajați oamenii.

16.2.1 accesul la internet

În timp ce navigheaza pe web, oamenii pot vizita în mod inconștient site-uri web dăunătoare – site-uri web care au fost deja atacate sau proiectate special pentru a distribui un software dăunător. Când un utilizator vizitează unul dintre aceste site-uri, hackerii pot exercita control asupra calculatorului, fișierelor descărcate, instalarea keylogger-urilor sau a altor coduri dăunătoare.

16.2.2 partajarea fișierelor

Atunci când oamenii partajează fisierele folosind rețelele peer-to-peer, acestia descarcă adesea spyware și coduri de mobil dăunătoare împreuna cu produsul de lucru prevăzut. Spyware aduna informații despre utilizator – adesea înregistrarea tastelor apasate, obiceiurile de navigare web, parole, adrese de email și transferă informațiile respective înapoi în site-ul sursa prin portul 80 de comunicații back-channel.

Codul dăunător poate fi transmis prin web – virusuri, cai troieni, viermi sau cod de internet malițios.

Codurile de mobil dăunatoare achiziționate se distribuie folosind pagini web sau cod HTML, incluzând ActiveX sau cod Javascript și este incorporat în paginile web.

16.2.3 mesagerie instant

Folosind aplicațiile de mesagerie, oamenii pot „vorbi" și partaja fișiere fără nici un efort. Aplicațiile de mesagerie pot ajuta la promovarea comunicării între membrii echipei și de a reduce numărul de întâlniri față în față necesare. Acesta poate fi, de asemenea, un instrument de e-commerce de neprețuit, cu reprezentanți de servicii pentru clienți care sprijină clienții noi prin răspunderea la întrebările despre produs, ajuntând la finalizarea tranzacților etc.

Din păcate, poate fi și utilizat pentru a transmite informații proprietarilor comaniei în format nectriptat și pentru a transfera fișierele atașate care ocolesc complet infrastructura de securitate existentă.

În plus, multe descărcări instant sunt infestate de viruși, cai troieni și viermi. De fapt, mai mulți viermi au vizat clienți IM specifici, trimitând utilizatorilor email-uri de înșelătorie și folosind liste de prieteni IM pentru răspândire.

16.2.4 e-mail

Chiar și trimiterea și răspunderea la email-uri poate fi o activitate riscantă. Email-ul oferă hackerilor o modalitate ușoară de distribuire a conținutului dăunător. Mesajele prin email pot include atașamente de fisiere infectate cu viruși, viermi, cai troieni sau alte programe care daunează. Hackerii trimit fișierele infectate și speră ca destinatarul sa le deschidă. Se folosesc alte email-uri cu intenții rele care acționează asupra vulnerabilității browserului pentru a se răspândi. Un exemplu este viermele Nimda, care a rulat automat pe calculatoare cu o versiune vulnerabilă

16.3 phishing

Activitatea care constă în obtinerea datelor confidențiale(phishing) este o altă amenințare care valorifică popularitatea email-ului ca instrument de comunicare. În multe comploturi, hoții trimit email-uri cu aspect oficial, dar fals pentru a-i păcăli pe destinatari sa dezvăluie un cont confidențial sau informații despre utilizator. Destinatarii sunt încurajați să facă click pe linkurile din email-uri, ducându-i la ceea ce par a fi pagini de servicii pentru clienți, completate cu link-uri, logo-uri, limba și toate aspectele familiare a site-ului web autentic. De fapt, unele site-uri web frauduloase sunt atât de convingătoare încât bara de adrese a utilizatorului arata ca sunt conectați la un site bancar sau site e-commerce.

Hoții sunt considerați hackeri, deorarece folosesc ingineria socială pentru a-și păcăli și înșela țintele. De exemplu, un hoț poate trimite un email folosind fatadele unui bănci majore, a unui card de credit sau a unui serviciu E-money ca PayPal. Email-ul nu va arata doar oficial, ci va avea și un nume de domeniu de rețea cu aspect oficial și adresa de retur. Conținutul va avea un mesaj inofensiv, cum ar fi: „Informațiile contului tău necesită actualizare".

Presupunerea hoților este ca oamenii vor deschide emailul, îl vor citi și vor crede conținutul. Ei speră ca cititorul va face click pe linkul furnizat, deoarece pare oficial și va fi directionat către un site care arată exact ca cel real (PayPal etc.). În realitate, utilizatorul a fost direcționat către un site fals și urmează sa scrie informații confidențiale ale contului care vor fi înregistrate și trimise înapoi către atacator. Impactul înșelătoriei este asupra întreprinderilor cât și asupra cosumatorilor. Sunt bine cunoscute băncile de încredere și alți furnizori de servicii online care sunt îngrijorate de faptul că temerile de a identifica furtul și spargerea conturilor vor opri consumatorii să facă cumpărături și prelucrare online a altor tranzacții finanaciare. Visa Internațional a intrat în prima agregare la nivel mondial, serviciul în efortul de a combate înșelătoria.

Phishing-ul poate viza și informații confidențiale ale companiei. Prin persoanele țintă (trimiterea unui email tuturor persoanelor de la o anumită companie presupusă de la departamentul IT, de exemplu), hoții au obținut cu succes accesul la nume de utilizatori și parole corporative. Folosind aceste informații, hackerii ar putea să se infiltreze și să acceseze rețeaua corporativă și, la rândul lor, informații confidentiale despre cienți sau utilizatori, care poate prezenta nu numai probleme de răspundere juridică, ci și probleme de conformitate cu reglementările.

16.4 pharming

Programul automat dăunător care se află în așteptare până când un utilizator se conectează la un site web țintă (în principal bănci, alte instituții financiare și site-uri de e-commerce) utilizează o nouă schemă numita „pharming”. Ca phishing-ul, acești hoți își propun sa fure informații confidențiale despre cont. Spre deosebire de phishing, această metodă nu se bazează pe email-uri false pentru a ademeni victimele nesatisfăcute, de fapt, este aproape nedetectabil. Pharming folosește viruși de tip cal troian care modifică comportamentul browserelor web. Utilizatorul încearcă să acceseze un site bancar online sau unul dintre celelalte site-uri țintă care declanșează de fapt browserul către un site fraudulos. Odată ce o mașină este infectată, utilizatorul poate tasta adresa URL corectă și poate ajunge în continuare tot pe site-ul fraudulos.

16.5 site-uri web atacate

Hackerii pot transforma un site web într-unul malițios. Când site-urile web sunt atacate, site-urile, în sine, devin vectori de atac și sunt folosiți pentru distribuirea codului malițios. Când serverul web a unei companii este compromis, clienții (sau potențiali clienți) sunt infectați în mod involuntar cu codul malițios atunci când pur și simplu vizitează site-ul, aceste infecții apar fără ca clientul să fie nevoit sa ruleze programe sau să deschidă vreun atașament.

16.6 site-uri web defrișate (spoofed)

Infractorii cibernetici valorifică încrederea consumatorilor în anumite produse și marci și folosesc această încredere pentru a păcălii utilizatorii în divulgarea infromatiilor confidențiale despre cont. Un scenariu tipic implică trimiterea utilizatorilor a unui site înșelator, rugându-i să facă click pe un link pentru a-și actualiza informațiile despre cont. HTML-ul din email-uri arată convingător și familiar. Mulți utilizatori respectă cu ușurință „cererea băncii lor” oferind informațiile contului de pe site-urile concetate – site-uri care par valide, dar sunt, de fapt, frauduloase.

Indiferent dacă utilizatorii cad sau nu victime ale acestor planuri, aceștia devin precauți și suspectează comunicațiile de pe site-uri de e-commnerce sau site-uri bancare, și acum sunt mai puțin susceptibile să se angajeze în tranzacții online. Aceste temeri – deși justificate pot avea un impact asupra comerțului global.

chapter 17 viruși

chapter 18 troieni

18.1 definiție

Un cal troian (uneori prescurtat ca troian), este un malware care nu se auto-reproduce, care pare să îndeplinească o funcție de dorit pentru utilizator, dar în schimb facilitează accesul neautorizat la sistemul informatic al utilizatorului. Termenul este derivat din povestea Calului troian în mitologia greacă.

Un program aparent inocent conceput pentru a eluda caracteristicile de securitate ale unui sistem. Metoda obișnuită de introducere a unui cal troian este prin donarea unui program sau a unei părți dintr-un program unui utilizator al sistemului a cărui securitate trebuie încălcată. Codul donat va îndeplini în mod evident o funcție utilă; destinatarul nu va ști că codul are alte efecte, cum ar fi scrierea unei copii a numelui său de utilizator și a parolei într-un fișier a cărui existență este cunoscută doar donatorului și din care donatorul va colecta ulterior orice date au fost scrise

18.2 purpose and operation

Trojan horses are designed to allow a hacker remote access to a target computer system. Once a Trojan horse has been installed on a target computer system, it is possible for a hacker to access it remotely and perform various operations. The operations that a hacker can perform are limited by user privileges on the target computer system and the design of the Trojan horse.

Operations that could be performed by a hacker on a target computer system include:

- Use of the machine as part of a botnet (i.e. to perform spamming or to - perform Distributed Denial-of-service (DDoS) attacks)
- Data theft (e.g. passwords, credit card information, etc.)
- Installation of software (including other malware)
- Downloading or uploading of files
- Modification or deletion of files
- Keystroke logging
- Viewing the user's screen
- Wasting computer storage space

Trojan horses require interaction with a hacker to fulfill their purpose, though the hacker need not be the individual responsible for distributing the Trojan horse. In fact, it is possible for hackers to scan computers on a network using a port scanner in the hope of finding one with a Trojan horse installed, that the hacker can then use to control the target computer.

18.3 installation and distribution

Trojan horses can be installed through the following methods:

1. **Software downloads** (i.e., a Trojan horse included as part of a software application downloaded from a file sharing network)
2. **Websites containing executable content** (i.e., a Trojan horse in the form of an ActiveX control)
3. **Email attachments**
4. **Application exploits** (i.e., flaws in a web browser, media player, messaging client, or other software that can be exploited to allow installation of a Trojan horse). Also, there have been reports of

compilers that are themselves Trojan horses. While compiling code to executable form, they include code that causes the output executable to become a Trojan horse.

Users can be tricked into installing Trojan horses by being enticed or frightened. For example, a Trojan horse might arrive in email described as a computer game. When the user receives the mail, they may be enticed by the description of the game to install it. Although it may in fact be a game, it may also be taking other action that is not readily apparent to the user, such as deleting files or mailing sensitive information to the attacker. As another example, an intruder may forge an advisory from a security organization, such as the CERT Coordination Center, that instructs system administrators to obtain and install a patch.

Other forms of *"social engineering"* can be used to trick users into installing or running Trojan horses. For example, an intruder might telephone a system administrator and pose as a legitimate user of the system who needs assistance of some kind. The system administrator might then be tricked into running a program of the intruder's design.

Software distribution sites can be compromised by intruders who replace legitimate versions of software with Trojan horse versions. If the distribution site is a central distribution site whose contents are mirrored by other distribution sites, the Trojan horse may be downloaded by many sites and spread quickly throughout the Internet community.

Because the Domain Name System (DNS) does not provide strong authentication, users may be tricked into connecting to sites different than the ones they intend to connect to. This could be exploited by an intruder to cause users to download a Trojan horse, or to cause users to expose confidential information.

Intruders may install Trojan horse versions of system utilities after they have compromised a system. Often, collections of Trojan horses are distributed in toolkits that an intruder can use to compromise a system and conceal their activity after the compromise, e.g., a toolkit might include a Trojan horse version of ls which does not list files owned by the intruder. Once an intruder has gained administrative access to your systems, it is very difficult to establish trust in it again without rebuilding the system from known-good software. A Trojan horse may be inserted into a program by a compiler that is itself a Trojan horse.

Finally, a Trojan horse may simply be placed on a web site to which the intruder entices victims. The Trojan horse may be in the form of a Java applet, JavaScript, ActiveX control, or other form of executable content.

18.4 removal

Antivirus software is designed to detect and delete Trojan horses, as well as preventing them from ever being installed. Although it is possible to remove a Trojan horse manually, it requires a full understanding of how that particular Trojan horse operates. In addition, if a Trojan horse has possibly been used by a hacker to access a computer system, it will be difficult to know what damage has been done and what other problems have been introduced. In situations where the security of the computer system is critical, it is advisable to simply erase all data from the hard disk and reinstall the operating system and required software.

18.5 current use

Due to the growing popularity of botnets among hackers, Trojan horses are becoming more common. According to a survey conducted by BitDefender from January to June 2009, "Trojan-type malware is on the rise, accounting for 83-percent of the global malware detected in the wild".

Trojan horses can be particularly effective when offered to systems staff who can run code in highly privileged modes. Two remedies are effective: no code should be run unless its provenance is absolutely certain; no code should be run with a higher level of privilege than is absolutely essential.

18.6 solutions

The best advice with respect to Trojan horses is to ***avoid them*** in the first place. System administrators (including the users of single-user systems) should take care to verify that every piece of software that is installed is from a trusted source and has not been modified in transit. When digital signatures are provided, users are encouraged to validate the signature (as well as validating the public key of the signer). When digital signatures are not available, you may wish to acquire software on tangible media such as CDs, which bear the manufacturer's logo. Of course, this is not foolproof either. Without a way to authenticate software, you may not be able to tell if a given piece of software is legitimate, regardless of the distribution media. Software developers and software distributors are strongly encouraged to use cryptographically strong validation for all software they produce or distribute. Any popular technique based on algorithms that are widely believed to be strong will provide users a strong tool to defeat Trojan horses. Anyone who invests trust in digital signatures must also take care to validate any public keys that may be associated with the signature. It is not enough for code merely to be signed -- it must be signed by a trusted source. Do not execute anything sent to you via unsolicited electronic mail. Use caution when executing content such as Java applets, JavaScript, or Active X controls from web pages. You may wish to configure your browser to disable the automatic execution of web page content. Apply the principle of least privilege in daily activity: do not retain or employ privileges that are not needed to accomplish a given task. For example, do not run with enhanced privilege, such as "root" or "administrator," ordinary tasks such as reading email. Install and configure a tool such as Tripwire® that will allow you to detect changes to system files in a cryptographically strong way.

Educate your users regarding the danger of Trojan horses. Use firewalls and virus products that are aware of popular Trojan horses. Although it is impossible to detect all possible Trojan horses using a firewall or virus product (because a Trojan horse can be arbitrary code), they may aid you in preventing many popular Trojan horses from affecting your systems.

Review the source code to any open source products you choose to install. Open source software has an advantage compared to proprietary software because the source code can be widely reviewed and any obvious Trojan horses will probably be discovered very quickly. However, open source software also tends to be developed by a wide variety of people with little or no central control. This makes it difficult to establish trust in a single entity. Keep in mind that reviewing source code may be impractical at best, and that some Trojan horses may not be evident.

Adopt the use of cryptographically strong mutual authentication systems, such as ssh, for terminal emulation, X.509 public key certificates in web servers, S/MIME or PGPfor electronic mail, and kerberos for a variety of services. Avoid the use of systems that trust the domain name system for authentication, such as telnet, ordinary http (as opposed to https), ftp, or smtp, unless your network is specifically designed to support that trust. Do not rely on timestamps, file sizes, or other file attributes when trying to determine if a file contains a Trojan horse.

Exercise caution when downloading unauthenticated software. If you choose to install software that has not been signed by a trusted source, you may wish to wait for a period of time before installing it in order to see if a Trojan horse is discovered.

chapter 19 reguli practice de protecție

19.1 definiție

„Un

19.2 clasificare

Există

chapter 20 carduri inteligente

20.1 definition

ing

20.2 classification

There are sever

chapter 21 biometrică

21.1 definition

“An exploit

chapter 22 comerțul electronic

22.1 file system

For file systems, the access control lists are operating system specific data structures which specify individual or group rights to certain objects, like files, directories or processes.

chapter 23 sisteme de plăți și transfer electronic

În acest capitol

23.1 generalități

Data

23.2 visa, mastercard, american express, discover

DNSSEC

23.3 transfer SWIFT

Data

Bibliografie

[5-TEN-2021] – Five key cybersecurity trends for 2021 - https://www.forbes.com/sites/forbestechcouncil/2021/12/30/five-key-cybersecurity-trends-for-2021/

[AME-TEN] – Amenințări și tendințe în securitatea cibernetică în 2020 - https://onlinedegrees.sandiego.edu/top-cyber-security-threats/

[AVA-HIS] – Istoria securității cibernetice - https://blog.avast.com/history-of-cybersecurity-avast

[DATS] – Data security Wikipedia - https://en.wikipedia.org/wiki/Data_security

[DESW] – DES Wikipedia - https://en.wikipedia.org/wiki/Data_Encryption_Standard

[DNSI] – DNSSEC – what is it and why is it important - https://www.icann.org/resources/pages/dnssec-what-is-it-why-important-2019-03-05-en

[IBAN] – IBAN, SWIFT, SEPA - https://riscograma.ro/9794/iban-sepa/

[KSF] – the keccak sponge family functionhttp://keccak.noekeon.org/specs_summary.html

[SHA1C] - https://phys.org/news/2017-02-cwi-google-collision-industry-standard.html

[PKIS] – Totul despre PKI - https://www.ssl2buy.com/wiki/public-key-infrastructure