POGLAVLJE 1

Osnove

1.1 TERMINOLOGIJA

Pošiljalac i primalac

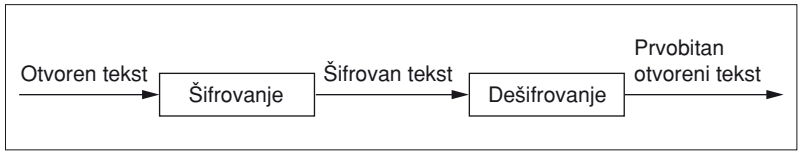
Pretpostavimo da pošiljalac želi da pošalje poruku primaocu. Osim toga, ovaj pošiljalac želi da poruku pošalje sigurno: želi da obezbedi da prisluškivač kanala (njuškalo) ne može da pročita poruku.

Poruke i šifrovanje

Poruka je **otvoren tekst** (engl. *plaintext*) (ili čist tekst, engl. *cleartext*). **Šifrovanje** (engl. *encryption)* proces je maskiranja poruke koji za rezultat ima sakrivanje njene sadržine. Šifrovana poruka je **šifrat** (engl. *ciphertext*). **Dešifrovanje** (engl. *decryption*) jeste proces pretvaranja šifrata u otvoreni tekst (slika 1.1).

(U standardu ISO 7498-2 koriste se termini „encipher" i „decipher" jer su u nekim kulturama termini „encrypt" i „decrypt" uvredljivi pošto se odnose na grobnicu – crypt – i posmrtne ostatke.)

Kriptografija (engl. *cryptography*) jeste umetnost i nauka čuvanja bezbednosti poruka. Kriptografijom se bave **kriptografi** (engl. *cryptographers*). **Kriptoanalitičari** (engl. *cryptoanalysts*) bave se **kriptoanalizom** (engl. *cryptanalysis*), umetnošću provaljivanja šifrata; tj. pogleda „iza maske". **Kriptologija** (engl. *cryptology*) grana je matematike koja obuhvata kriptografiju i kriptoanalizu. Kriptologijom se bave **kriptolozi** (engl. *cryptologists*). Savremeni kriptolozi uglavnom izučavaju teoretsku matematiku – jer to moraju da rade.



Slika 1.1 Šifrovanje i dešifrovanje.

Otvoren tekst je označen sa *M* (od engl. *message*), ili sa *P* (od engl. *plaintext*). On može biti niz bitova, tekstualni dokument, bitmapirana slika, digitalizovani glas, digitalni video... bilo šta. Za računar, *M* su samo binarni podaci. (U ostalim poglavljima ove knjige obrađeni su binarni podaci i kompjuterska kriptografija.) Otvoren tekst može biti namenjen slanju ili skladištenju. U svakom slučaju, *M* je poruka koju treba šifrovati.

Šifrat je označen sa *C*. On takođe predstavlja binarne podatke: nekada iste veličine kao *M*, nekada veće. (Kombinovanjem šifrovanja i kompresije, *C* može biti manje od *M*. U svakom slučaju, to se ne postiže šifrovanjem.) Funkcija šifrovanja, *E*, obrađuje, *M* i daje rezultat *C*. Matematički zapisano:

$$E(M) = C$$

U obrnutom procesu, funkcija dešifrovanja *D* obrađuje *C*, a rezultat je *M:*

$$D(C) = M$$

Pošto je cilj šifrovanja i dešifrovanja poruke dobijanje originalnog otvorenog teksta, mora da važi sledeća jednakost:

$$D(E(M)) = M$$

Provera identiteta, integritet i nemogućnost poricanja

Osim što obezbeđuje poverljivost, kriptografija često obavlja i druge zadatke:

- **Provera identiteta (engl. *authentication*)** Trebalo bi da primalac poruke može da utvrdi njeno poreklo; uljez ne bi trebalo da je u mogućnosti da se predstavlja kao neko drugi.
- **Integritet (engl. *integrity*)** Trebalo bi da primalac može da potvrdi kako poruka nije izmenjena u tranzitu; uljez ne bi trebalo da je u mogućnosti da podmetne lažnu poruku kao pravu.
- **Nemogućnost poricanja (engl. *nonrepudiation*)** Ne bi trebalo da pošiljalac može kasnije poricati da je poslao poruku.

Ovo su osnovni preduslovi društvenih odnosa između korisnika računara, i analogni su odnosu licem u lice. Da je neko onaj ko kaže da jeste... da su nečiji akreditivi – bilo da je u pitanju vozačka dozvola, doktorska diploma, ili pasoš – važeći... da je dokument za koji pretpostavljamo da potiče od određene osobe, zaista i došao od te osobe. Sve ove preduslove omogućuju provera identiteta, integritet i nemogućnost poricanja.

Algoritmi i ključevi

Kriptografski algoritam (engl. *cryptographic algorithm*), poznat i kao **šifra** (engl. *cipher*), matematička je funkcija koja se koristi za šifrovanje i dešifrovanje. (U osnovi su to dve srodne funkcije: jedna za šifrovanje, a druga za dešifrovanje.)

Ako se sigurnost algoritma zasniva na tajnosti načina na koji algoritam radi, onda je u pitanju **ograničeni** (engl. *restricted*) algoritam. Ograničeni algoritam ima istorijsku ulogu, ali je potpuno neadekvatan po današnjim standardima. Ne mogu ga koristiti velike ili promenljive grupe korisnika, jer kad god bi neko od korisnika napustio grupu, svi ostali bi morali da pređu na drugi algoritam. Ako neko slučajno oda tajnu, svi ostali moraju da promene algoritam.

Da stvar bude gora, nemoguće je obaviti kontrolu kvaliteta ili standardizovati ograničeni algoritam. Svaka grupa korisnika mora da ima svoj jedinstveni algoritam. Takva grupa ne može koristiti najnoviji hardver i softver; prisluškivač kanala (njuškalo) može da kupi isti proizvod i nauči algoritam. Članovi grupe moraju da pišu svoje algoritme i implementacije. Ako u grupi nema dobrog kriptografa, onda neće znati ni da li je njihov algoritam siguran.

Uprkos ovim velikim nedostacima, ograničeni algoritmi su veoma popularni za aplikacije niske bezbednosti. Korisnici ili ne shvataju, ili ih nije briga za bezbednosne probleme svojstvene njihovom sistemu.

Savremena kriptografija ovaj problem rešava pomoću **ključa** (engl. *key*), čija je oznaka *K*. Ključ može imati bilo koju vrednost. Raspon mogućih vrednosti ključa naziva se **prostor ključeva** (engl. *keyspace*). Za šifrovanje i dešifrovanje koristi se ključ (tj. oni zavise od ključa i ta je činjenica označena indeksom *K*), tako da funkcije sada postaju:

$$E_K(M) = C$$
$$D_K(C) = M$$

Ove funkcije imaju sledeće svojstvo (slika 1.2):

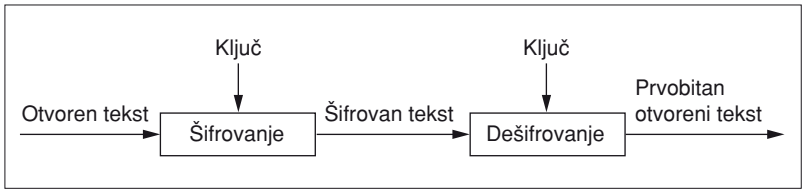
$$D_K(E_K(M)) = M$$

Neki algoritmi koriste posebne ključeve za šifrovanje i dešifrovanje (slika 1.3). Ključ za šifrovanje K_1, drugačiji od odgovarajućeg ključa za dešifrovanje K_2. U ovom slučaju:

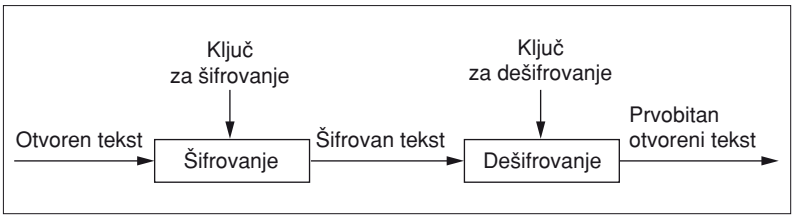
$$E_{K_1}(M) = C$$
$$D_{K_2}(C) = M$$
$$D_{K_2}(E_{K_1}(M)) = M$$

Sigurnost ovih algoritama zasnovana je na ključu (ili ključevima), a ne na detaljima algoritma. To znači da algoritam može da se objavi i analizira. Proizvodi koji koriste algoritam mogu se masovno proizvoditi. Nije od značaja ako prisluškivač kanala (njuškalo) zna vaš algoritam; ako nema vaš ključ, ne može da pročita vaše poruke.

Kriptosistem (engl. *cryptosystem*) predstavlja algoritam sa svim mogućim otvorenim tekstovima, šifratima i ključevima.



Slika 1.2 Šifrovanje i dešifrovanje pomoću ključa.



Slika 1.3 Šifrovanje i dešifrovanje pomoću dva različita ključa.

Simetrični algoritmi

Postoje dva opšta tipa algoritama zasnovanih na ključu: simetrični algoritmi i algoritmi s javnim ključem. **Simetrični algoritmi** (engl. *symmetric algorithms*), poznati i kao konvencionalni algoritmi (engl. *conventional algorithms*), jesu algoritmi kod kojih ključ za šifrovanje može biti izveden iz ključa za dešifrovanje, i obrnuto. Mnogi simetrični algoritmi imaju iste ključeve za šifrovanje i dešifrovanje. Pri korišćenju ovih algoritama, poznatih i kao algoritmi s tajnim ključem (engl. *secret-key algorithms*) ili algoritmi s jednim ključem (engl. *single-key algorithms, one-key algorithms*), potrebno je da se pošiljalac i primalac slože oko upotrebe ključa pre nego što otpočnu s bezbednom komunikacijom. Sigurnost simetričnog algoritma počiva na ključu; otkrivanje ključa znači da svako može da šifruje i dešifruje poruke. Dokle god imamo potrebu za tajnom komunikacijom, dotle i ključ mora ostati tajan.

Šifrovanje i dešifrovanje simetričnim algoritmom izraženi su kao:

$$E_K(M) = C$$
$$D_K(C) = M$$

Simetrični algoritmi se mogu podeliti u dve kategorije. Prva grupa radi na otvorenom tekstu bit po bit (nekad bajt po bajt); njih nazivamo **algoritmi toka** (engl. *stream algorithms*) ili **šifre toka** (engl. *stream ciphers*). Druga grupa radi na otvorenom tekstu u grupama bitova. Grupe bitova nazivamo **blokovi** (engl. *blocks*), dok se algoritmi zovu **blokovski algoritmi** (engl. *block algorithms*) ili **blokovske šifre** (engl. *block ciphers*). U savremenim kompjuterskim algoritmima, tipična veličina bloka je 64 bita – dovoljno velika da spreči analizu, i dovoljno mala da bude obradiva. (Pre pojave računara, kriptografski algoritmi su radili sa otvorenim tekstom po principu slovo po slovo. To možete zamisliti kao algoritam toka koji radi s tokom znakova.)

Algoritmi s javnim ključem

Algoritmi s javnim ključem (engl. *public-key algorithms*), poznati i kao asimetrični algoritmi (engl. *asymmetric algorithms*), osmišljeni su tako da se za šifrovanje i dešifrovanje koriste različiti ključevi. Osim toga, ključ za dešifrovanje ne može (bar ne u razumnom vremenskom roku) biti izveden tj. izračunat iz ključa za šifrovanje. Algoritam se zove „s javnim ključem" jer ključ za šifrovanje može biti javno objavljen: svako može da iskoristi ključ za šifrovanje, ali samo osoba koja ima odgovarajući ključ za dešifrovanje može da dešifruje poruku. U ovakvim sistemima, ključ za šifrovanje obično se naziva **javni ključ** (engl. *public key*), dok se ključ za dešifrovanje

često naziva **privatni ključ** (engl. *private key*). Privatni ključ se takođe naziva i tajni ključ, ali da ih ne bismo mešali sa simetričnim algoritmima, u ovoj knjizi ih nećemo pominjati pod tim imenom.

Šifrovanje javnim ključem K izražava se kao:

$$E_K(M) = C$$

Iako su javni i privatni ključ različiti, dešifrovanje odgovarajućim privatnim ključem izražava se kao:

$$D_K(C) = M$$

Nekada će poruke biti šifrovane privatnim ključem a dešifrovane javnim; tako se poruka digitalno potpisuje (engl. *digital signing*) – videti odeljak 2.6. Uprkos mogućoj zabuni, ove operacije se izražavaju na sledeći način:

$$E_K(M) = C$$
$$D_K(C) = M$$

Kriptoanaliza

Suština kriptografije je očuvanje otvorenog teksta (ili ključa, ili i jednog i drugog) tajnim i sigurnim od upada prisluškivača kanala/njuškala (koji su poznati i kao protivnici, napadači, presretači, nametljivci, uljezi, ili jednostavno – neprijatelji). Pretpostavlja se da prisluškivači kanala imaju kompletan pristup komunikacionim kanalima između pošiljaoca i primaoca.

Kriptoanaliza je nauka koja bez ključa određuje otvoreni tekst na osnovu šifrata. Uspešnom primenom kriptoanalize moguće je rekonstruisati otvoreni tekst ili ključ. Upotrebom kriptoanalize moguće je pronaći slabosti u kriptosistemu i na kraju doći do već pomenutih rezultata. (Otkrivanje ključa nekriptoanalitičkim metodama nazivamo **kompromitovanje**, engl. *compromise*.)

Pokušaj kriptoanalize naziva se **napad** (engl. *attack*). Osnovna postavka kriptoanalize, koju je prvi objavio Holanđanin A. Kerkhof (A. Kerckhoffs) u XIX veku, jeste da tajnost mora biti potpuno zasnovana na ključu [794]. Kerkhof je pretpostavio da kriptoanalitičar zna sve pojedinosti kriptografskog algoritma i njegove primene. (Naravno, pretpostavljamo da CIA nema običaj da daje Mosadu podatke o svojim kriptografskim algoritmima, iako ih verovatno Mosad ipak sazna.) Mada u stvarnosti kriptoanalitičari nemaju uvek tako detaljnu informaciju, ipak je dobro pretpostaviti da je imaju. Ukoliko drugi ne mogu da provale algoritam čak i kada znaju kako on radi, onda sigurno neće moći ako to ne znaju.

Postoje četiri opšta tipa kriptoanalitičkog napada. Naravno, za sva četiri tipa pretpostavlja se da kriptoanalitičar ima sve potrebno znanje o šifrovanju koje koristi algoritam:

1. **Napad „samo šifrat“** (engl. *ciphertext-only attack*) Kriptoanalitičar ima šifrate nekoliko poruka šifrovanih istim algoritmom za šifrovanje. Zadatak kriptoanalitičara je da rekonstruiše otvoreni tekst što većeg broja poruka, ili – još bolje – da otkrije ključ (ili ključeve) korišćene za šifrovanje poruka kako bi dešifrovao druge poruke šifrovane istim ključem.

Dato: $C_1 = E_k(P_1)$, $C_2 = E_k(P_2)$, ... $C_i = E_k(P_i)$

Izračunati: Ili P_1, P_2, ... P_i; k; ili algoritam
da bi se dobilo P_{i+1} iz $C_{i+1} = Ek(P_{i+1})$

2. **Napad „poznat otvoreni tekst“** (engl. *known-plaintext attack*) Kriptoanalitičar ima pristup šifratima za nekoliko poruka, kao i otvorenim tekstovima tih poruka. Njegov je posao da izračuna ključ (ili ključeve) korišćen za šifrovanje poruka, ili algoritam, kako bi dešifrovao svaku novu poruku šifrovanu istim ključem (ili ključevima).

   Dato: P_1, $C_1 = E_k(P_1)$, P_2, $C_2 = E_k(P_2)$, ... P_i, $C_i = E_k(P_i)$

   Izračunati: Ili k, ili algoritam
   da bi se dobilo P_{i+1} iz $C_{i+1} = Ek(P_{i+1})$

3. **Napad „odabran otvoreni tekst“** (engl. *chosen-plaintext attack*) Kriptoanalitičar ne samo da ima pristup šifratu i pripadajućem otvorenom tekstu nekoliko poruka, već i bira otvoreni tekst koji će biti šifrovan. Ovo je moćnije od napada „poznat otvoreni tekst“, jer kriptoanalitičar može da odabere specifične blokove otvorenog teksta za šifrovanje i tako dođe do više informacija o ključu. Njegov je posao da izračuna ključ (ili ključeve) korišćen za šifrovanje poruka, ili da odredi algoritam pomoću koga bi dešifrovao svaku novu poruku šifrovanu istim ključem (ili ključevima).

   Dato: P_1, $C_1 = E_k(P_1)$, P_2, $C_2 = E_k(P_2)$, ... P_i, $C_i = E_k(P_i)$,
   gde kriptoanalitičar odabira P_1, P_2, ... P_i

   Izračunati: Ili k, ili algoritam da bi se dobilo P_{i+1} iz $C_{i+1} = E_k(P_{i+1})$

4. **Napad „prilagodljiv odabrani otvoreni tekst“** (engl. *adaptive-chosen-plaintext attack*) Ovo je poseban slučaj napada „odabran otvoreni tekst“. Ne samo da kriptoanalitičar može da odabere šifrovan otvoreni tekst, već može i da modifikuje ono što je odabrao na osnovu rezultata prethodnih šifrovanja. Prilikom napada „odabran otvoreni tekst“, kriptoanalitičar može da odabere za šifrovanje jedan veliki blok otvorenog teksta; u napadu „prilagodljiv odabrani otvoreni tekst“, on može da odabere manji blok otvorenog teksta, a zatim još jedan na osnovu rezultata prvog, i tako dalje.

Pored nabrojanih, postoje još bar tri tipa kriptoanalitičkog napada.

5. **Napad „odabrani šifrat“** (engl. *chosen-ciphertext attack*) Kriptoanalitičar može da odabere šifrat za dešifrovanje i ima pristup dešifrovanom otvorenom tekstu. Na primer, kriptoanalitičar ima pristup **modulu** (engl. *tamperproof box*) koji obavlja automatsko dešifrovanje. Njegov posao je da izračuna ključ.

   Dato: C_1, $P_1 = D_k(C_1)$, C_2, $P_2 = D_k(C_2)$, ... C_i, $P_i = D_k(C_i)$

   Izračunati: k

Ovaj napad je u prvom redu primenljiv na algoritme s javnim ključem i biće obrađen u odeljku 19.3. Napad „odabrani šifrat" ponekad je efikasan i protiv simetričnog algoritma. (Napad „odabran otvoreni tekst" i „odabrani šifrat" zajedno se nazivaju **napad „odabrani tekst"** (engl. *chosen-text attack*.)

6. **Napad „odabrani ključ"** (engl. *chosen-key attack*) Ne znači da kriptoanalitičar može da izabere ključ prilikom ovog napada, već treba da zna nešto o odnosu između različitih ključeva. Ovaj napad je čudan i nejasan, nije naročito praktičan i biće obrađen u odeljku 12.4.
7. **Kriptoanaliza pretnjom**, **krađom ili ucenom** (engl. *rubber-hose cryptanalysis*) Kriptoanalitičar preti, ucenjuje, ili muči nekoga dok ne dobije ključ. Podmićivanje se nekada naziva i napad tipa „kupovina ključa" (engl. *purchase-key attack*). Ovo su veoma opasni napadi i često su najbolji način za razbijanje algoritma.

Napadi „poznat otvoreni tekst" i napadi „odabran otvoreni tekst" mnogo su češći nego što biste mogli pretpostaviti. Nije tako retko da kriptoanalitičar nabavi otvoreni tekst šifrovane poruke, ili da podmiti nekoga da šifruje izabranu poruku. Možda nećete ni morati da podmitite bilo koga; ako date poruku nekom ambasadoru, kasnije ćete verovatno otkriti da je poruka šifrovana i poslata u njegovu zemlju na razmatranje. Mnoge poruke imaju standardne početke i krajeve koji su verovatno poznati kriptoanalitičarima. Šifrovan izvorni kôd je posebno ranjiv zbog redovnog pojavljivanja rezervisanih reči: #define, struct, else, return. Na iste probleme nailazimo i u šifrovanom izvršnom kodu: funkcije, petlje i tako dalje. Napadi „poznat otvoreni tekst" (čak i napadi „odabran otvoreni tekst") uspešno su korišćeni protiv Nemaca i Japanaca za vreme Drugog svetskog rata. Knjige Dejvida Kana (David Khan) [794,795,796] sadrže primere ovih napada iz istorije.

I ne zaboravite Kerkohovu hipotezu: Ako snaga vašeg kriptosistema počiva na činjenici da napadač ne zna kako vaš algoritam funkcioniše – propali ste. Ukoliko verujete da će tajnost strukture vašeg algoritma povećati sigurnost vašeg kriptosistema više nego slobodna analiza akademske zajednice, varate se. Ukoliko mislite da niko neće rastaviti vaš izvorni kôd i obrnutim procesom (reverznim inženjeringom) detaljno proučiti vaš algoritam, naivni ste. (Ovo se dogodilo 1994. sa algoritmom RC4 – videti odeljak 17.1.) Najbolji algoritmi su oni objavljeni. I pored toga što su ih godinama napadali najbolji svetski kriptografi, oni još uvek nisu provaljeni. (Američka Agencija za nacionalnu bezbednost drži svoje algoritme u tajnosti, ali za njih rade najbolji svetski kriptografi, a za vas ne. Osim toga, oni među sobom diskutuju o svojim algoritmima, oslanjajući se pri tome na profesionalnu procenu kolega radi otkrivanja mogućih slabosti u radu.)

Kriptoanalitičari nemaju uvek pristup algoritmima, kao npr. kada su SAD provalile japanski diplomatski kôd PURPLE za vreme Drugog svetskog rata [794] – ali uglavnom imaju. Ako se algoritam koristi za komercijalni bezbednosni softver, samo je pitanje vremena i novca kada će program biti rastavljen, a algoritam otkriven. Ukoliko se algoritam koristi za vojne komunikacione sisteme, samo je pitanje vremena i novca za kupovinu (ili krađu) opreme i detaljno proučavanje algoritma.

Oni koji tvrde da poseduju neprovaljivu šifru samo zato što sami ne mogu da je provale, ili su geniji ili budale. Nažalost, ima više ovih drugih. Čuvajte se ljudi koji hvale vrline svojih algoritama a pri tom odbijaju da ih objave; verovati njihovim algoritmima slično je poverenju u lažni lek.

Dobri kriptografi se oslanjaju na profesionalnu procenu kolega kako bi razlikovali dobar algoritam od lošeg.

Sigurnost algoritama

Različiti algoritmi pružaju različite nivoe sigurnosti, u zavisnosti od toga koliko ih je teško provaliti. Ako je cena provaljivanja algoritma viša od vrednosti šifrovanih podataka, verovatno ste bezbedni. Ukoliko je vreme za provaljivanje algoritma duže od vremena tokom kojeg je neophodno da šifrovani podaci ostanu tajna, verovatno ste bezbedni. Ako je količina podataka šifrovanih jednim ključem manja od količine podataka neophodnih za provaljivanje algoritma, verovatno ste bezbedni.

Kažem „verovatno" jer uvek postoji mogućnost da bude novih otkrića u oblasti kriptoanalize. S druge strane, vrednost većine podataka opada s vremenom. Važno je da vrednost podataka ostane niža od troškova potrebnih za razbijanje njihovog obezbeđenja.

Lars Knudsen je odredio sledeće kategorije provaljivanja algoritma. Dati su redom od najozbiljnijih ka manje ozbiljnim [858]:

1. **Potpuna provala** (engl. *total break*) Kriptoanalitičar dolazi do ključa, K, $D_K(C) = P$.
2. **Opšti zaključak** (engl. *global deduction*) Kriptoanalitičar nalazi alternativni algoritam, C, kojem odgovara $D_K(C)$, a da pri tom nema K.
3. **Trenutni (lokalni) zaključak** (engl. *instance deduction, local deduction*) Kriptoanalitičar otkriva otvoreni tekst presretnutog šifrata.
4. **Zaključak na osnovu informacije** (engl. *information deduction*) Kriptoanalitičar dolazi do neke informacije o ključu ili otvorenom tekstu. To može biti samo nekoliko bitova ključa, neka informacija o vrsti otvorenog teksta i slično.

Algoritam je bezuslovno siguran (engl. *unconditionally secure*) ako, bez obzira na količinu šifrata koju kriptoanalitičar ima, nema dovoljno informacija za obnovu otvorenog teksta. U stvari, samo je jednokratna beležnica (engl. *one-time pad*) – videti odeljak 1.5 – neprovaljiva i uz beskonačne resurse. Svi drugi kriptosistemi su provaljivi napadom „samo šifrat", jednostavnim isprobavanjem svih mogućih ključeva, jedan po jedan, i proverom da li rezultujući otvoreni tekst ima smisla. Ovo se naziva **napad grubom silom** (engl. *brute-force attack*) – videti odeljak 7.1.

Kriptografija se više bavi kriptosistemima koji su nepraktični za provaljivanje računanjem. Algoritam se smatra **računski sigurnim** (engl. *computationally secure*) – ili snažnim – ako ne može da se provali raspoloživim sredstvima, bilo postojećim ili budućim. Šta su tačno „raspoloživa sredstva", slobodno se tumači.

Složenost napada (odeljak 11.1) može se izmeriti na razne načine:

1. **Složenost podataka** (engl. *data complexity*) Količina ulaznih podataka potrebna za napad.

2. **Složenost obrade** (engl. *processing complexity*) Vreme potrebno za izvođenje napada. Ovo često nazivamo **radni faktor** (engl. *work factor*).
3. **Potrebe skladištenja** (engl. *storage requirements*) Količina memorije potrebne za napad.

Za procenu složenosti napada, obično se uzima onaj faktor čija je vrednost najmanja. U nekim napadima pravi se kompromis između ove tri vrednosti: napad može biti brži ukoliko se koristi veći prostor za skladištenje.

Složenost se izražava eksponentom broja 2. Ako je složenost obrade algoritma 2^{128}, onda je potrebno 2^{128} operacija za njegovo provaljivanje. (Ove operacije mogu biti komplikovane i oduzeti dosta vremena.) Ipak, ako imate dovoljno brz računar da izvede milion operacija svake sekunde, i odvojite milion paralelnih procesora za obavljanje zadatka, još uvek će vam biti potrebno više od 10^{19} godina da biste otkrili ključ. To je milijardu puta više od starosti univerzuma.

Složenost napada na algoritam je konstantna (naravno, dok neki kriptoanalitičar ne osmisli bolji napad), ali brzina računara nije. Kompjuterska tehnologija je u proteklih pola veka zabeležila fenomenalan napredak, pa nema razloga za pomisao da se takav trend neće nastaviti. Mnogi kriptoanalitički napadi su savršeni za paralelno povezane mašine: zadatak se može razbiti na milijarde segmenata, a procesori ne moraju da komunicaraju između sebe. Proglasiti algoritam sigurnim samo zato što je nepraktičan za provaljivanje s današnjom tehnologijom, u najboljem slučaju je riskantno. Dobri kriptosistemi su nepraktični za provaljivanje čak i kada se uzme u obzir razvoj kompjuterske tehnologije još mnogo godina u budućnosti.

Istorijski termini

Istorijski gledano, reč **kôd** (engl. *code*) odnosi se na kriptosistem koji koristi lingvističke jedinice: reči, fraze, rečenice itd. Na primer, reč „OCELOT" može biti šifrat fraze „SKRENI LEVO ZA 90 STEPENI", reč „LIZALICA" može biti šifrat za „SKRENI DESNO ZA 90 STEPENI", a sintagma „PRIČA BEZ KRAJA" može biti šifrat za reč „HAUBICA". Kodovi ovog tipa nisu razmatrani u ovoj knjizi; pogledajte [794,795]. Kodovi su korisni jedino u posebnim situacijama. Šifre su korisne uvek. Ako vaš kôd nema odrednicu „MRAVOJEDI", onda to ne možete reći. Za razliku od toga, šifrom možete izraziti svaku reč.

1.2 STEGANOGRAFIJA

Steganografija (engl. *steganography*) služi za skrivanje tajnih poruka u drugim porukama, tako da je i samo postojanje tajne skriveno. U većini slučajeva pošiljalac napiše bezazlenu poruku, a onda sakrije tajnu poruku na istom listu papira. Trikovi korišćeni tokom vremena su: nevidljivo mastilo, mali ubodi iglom na odabranim slovima, male razlike pri pisanju slova rukom, oznake načinjene olovkom na slovima otkucanim pisaćom mašinom, šabloni koji pokrivaju veći deo poruke osim nekoliko slova, i tako dalje.

U novije vreme, ljudi skrivaju poruke u slikama. Najmanje značajan bit svakog bajta slike zamenite bitovima poruke. Slika se neće bitnije promeniti – u većini grafičkih formata prikazuje se više boja nego što ljudsko oko može da razazna – pa poruka može biti otpakovana na kraju svog puta. Na opisan način možete ubaciti poruku od 64 kilobajta u crnobelu fotografiju rezolucije 1024 × 1024. Nekoliko besplatnih programa obezbeđuje ovakve usluge.

Oponašajuće funkcije (engl. *mimic functions*) Pitera Vejnera (Peter Wayner) maskiraju poruke. Ove funkcije modifikuju poruku tako da njihov statistički profil liči na nešto drugo: npr. na odeljak sa oglasima *Njujork tajmsa*, Šekspirov komad, ili diskusionu grupu na Internetu [1584,1585]. Ovaj vid steganografije neće nikoga prevariti, ali može da obmane velike kompjutere koji pretražuju Internet u potrazi za zanimljivim porukama.

1.3 SUPSTITUCIONE ŠIFRE I TRANSPOZICIONE ŠIFRE

Pre pojave računara, kriptografija se sastojala od algoritama zasnovanih na slovima. Različiti kriptografski algoritmi su ili zamenjivali slova jedno drugim, ili su transpozicionirali slova. Bolji algoritmi su radili obe operacije, svaku mnogo puta.

Situacija je danas složenija, ali je filozofija ostala ista. Osnovna razlika je to što danas algoritmi rade s bitovima, a ne sa slovima. To je u stvari samo promena veličine abecede: za engleski jezik, od 26 na 2 elementa. Mnogi dobri algoritmi još uvek kombinuju elemente supstitucije i transpozicije.

Supstitucione šifre

Supstituciona šifra (engl. *substitution cipher*) svako slovo u otvorenom tekstu zamenjuje drugim slovom u šifratu. Primalac obrće supstituciju u šifratu da bi rekonstruisao otvoreni tekst.

U klasičnoj kriptografiji postoje četiri vrste supstitucionih šifara:

- **Obična supstituciona šifra** (engl. *simple substitution*), ili **monoalfabetska šifra** (engl. *monoalphabetic cipher*), jeste šifra u kojoj je svako slovo otvorenog teksta zamenjeno odgovarajućim slovom u šifratu. Kriptogrami u novinama su obične supstitucione šifre.
- **Homofonična supstituciona šifra** (engl. *homophonic substitution cipher*) slična je običnoj supstitucionoj šifri; razlika je samo u tome što jedno slovo iz otvorenog teksta može biti zamenjeno jednim ili nekolikim slovima u šifratu. Na primer, „A“ može da odgovara broju 5, 13, 25 ili 56, „B“ može odgovarati broju 7, 19, 31 ili 42 i tako dalje.
- **Poligramska supstituciona šifra** (engl. *polygram substitution cipher*) šifruje blokove znakova u grupama. Na primer, „ABA“ može odgovarati „RTQ“, „ABB“ može odgovarati „SLL“, i tako dalje.
- **Polialfabetska supstituciona šifra** (engl. *polyalphabetic substitution cipher*) sastoji se od više običnih supstitucionih šifara iz različitih abeceda. Na primer, može se upotrebiti pet različitih običnih supstitucionih šifara, a abeceda se menja na proizvoljnim mestima u otvorenom tekstu.

U obične supstitucione šifre spada i poznata **Cezarova šifra** (engl. *Caesar Cipher*); ona svaki znak otvorenog teksta zamenjuje znakom koji se nalazi tri mesta udesno, po modulu 26 („A" se zamenjuje sa „D", „B" sa „E", „W" sa „Z",..., „X" sa „A", „Y" sa „B", i „Z" sa „C"). Ona je u stvari još jednostavnija, zato što je abeceda šifrata rotiran otvoreni tekst a ne nasumična promena.

ROT13 je jednostavan program za šifrovanje koji se često nalazi u UNIX sistemima; on takođe spada u obične supstitucione šifre. „A" se zamenjuje sa „N", „B" sa „O" itd. Svako slovo je rotirano za 13 mesta.

Šifrovanjem dokumenta dva puta sa ROT13, dobija se originalni dokument.

$$P = \text{ROT13}\,(\text{ROT13}\,(P))$$

ROT13 nije namenjen sigurnosti; često se koristi u Usenet forumima da bi se sakrio potencijalno uvredljiv tekst, izbeglo odavanje rešenja zagonetke i tako dalje.

Obične supstitucione šifre mogu se lako provaliti jer šifra ne krije osnovne frekvencije različitih slova otvorenog teksta. Potrebno je samo 25 slova engleske abecede da bi dobar kriptoanalitičar mogao da rekonstruiše otvoreni tekst [1434]. Algoritam za rešavanje šifara ovog tipa može se naći u [578,587,1600,78,1475,1236,880]. Dobar računarski algoritam je [703].

Homofonične supstitucione šifre koristile su se još 1401. godine u vojvodstvu Mantova [794]. Njih je mnogo teže provaliti nego obične supstitucione šifre, ali ni one ne skrivaju sve statističke osobine jezika otvorenog teksta.

Šifra se lako može otkriti napadom „poznat otvoreni tekst". Napad „samo šifrat" je teži, ali je za njega računaru potrebno samo nekoliko sekundi. Pogledati detalje u [1261].

U poligramskoj supstitucionoj šifri, grupe slova su šifrovane zajedno. Plejferovu (Playfair) šifru, izmišljenu 1854, koristili su Britanci za vreme Prvog svetskog rata [794]. Ona šifruje parove slova, a ne pojedinačne znakove. Kriptoanaliza ove šifre obrađena je u [587,1475,880]. Hilova (Hill) šifra je još jedan primer poligramske supstitucione šifre [732]. Nekad se Hafmanovo (Huffman) kodiranje koristi kao šifra; ovaj način poligramskog supstitucionog šifrovanja nije siguran.

Polialfabetsku supstitucionu šifru izmislio je 1568. Leon Batista (Leon Battista) [794]. Šifru je koristila vojska Severa za vreme Američkog građanskog rata. Uprkos činjenici da ih je lako provaliti [819,577,587,794] (naročito pomoću računara), mnogi komercijalni proizvodi za sigurnost računara koriste šifre ovog oblika [1387,1390,1502]. Detalje za provaljivanje ove šeme, na primeru WordPerfecta, možete naći u [135,139]. Vižnerova (Vigenère) šifra, prvi put objavljena 1586, i Boforova (Beaufort) šifra, takođe su primeri polialfabetske supstitucione šifre.

Polialfabetske supstitucione šifre imaju više ključeva s jednim slovom, a svaki od njih se koristi za šifrovanje jednog slova u otvorenom tekstu. Prvi ključ šifruje prvo slovo otvorenog teksta, drugi šifruje drugo slovo otvorenog teksta i tako dalje. Nakon što se svi ključevi iskoriste, pristupa se njihovom recikliranju. Ako postoji 20 jednoslovnih ključeva, onda će svako dvadeseto slovo biti šifrovano istim slovom. Ovo se naziva **ciklus** (engl. *period*) šifre. U vreme klasične kriptografije, šifre s dužim ciklusima bilo je znatno teže provaliti od šifara s kraćim ciklusima. Danas postoje kompjuterske tehnike kojima se lako mogu otkriti supstitucione šifre s veoma dugim ciklusima.

Polialfabetska supstituciona **šifra „trčeći ključ“** (engl. *running-key cipher*) – poznata i pod imenom šifra „knjiga“ (engl. *book cipher*) koristi jedan tekst za šifrovanje drugog. Iako je u ovoj šifri ciklus jednak dužini celog teksta, lako ju je provaliti [576,794].

Transpozicione šifre

U **transpozicionoj šifri** (engl. *transposition cipher*), elementi otvorenog teksta ostaju isti ali im se menja raspored. U jednostavnoj **stubičnoj transpozicionoj šifri** (engl. *simple columnar transposition cipher*), otvoreni tekst je napisan horizontalno na parčetu papira s kvadratićima fiksne širine, a šifrat se čita vertikalno (slika 1.4). Pri dešifrovanju se šifrat piše vertikalno, na papiru s kvadratićima iste širine, a zatim se otvoreni tekst čita horizontalno.

Kriptoanaliza ovih šifara razmotrena je u [587,1475]. Pošto su slova u šifratu ista kao i u otvorenom tekstu, analizom učestalosti šifrata otkriće se da je verovatnoća pojave svakog slova približno jednaka kao u engleskom jeziku. To daje veoma dobar trag kriptoanalitičaru, koji onda može da iskoristi razne tehnike kako bi odredio pravi poredak slova i tako dobio otvoreni tekst. Obrada šifrata drugom transpozicionom šifrom u velikoj meri povećava sigurnost. Postoje još komplikovanije šifre, ali je sve njih moguće provaliti pomoću računara.

Nemačka šifra ADFGVX, korišćena za vreme Prvog svetskog rata, transpoziciona je šifra kombinovana sa običnom supstitucionom. Algoritam je bio veoma složen za svoje vreme, ali ga je provalio francuski kriptoanalitičar Žorž Penvan (Georges Painvin) [794].

Iako mnogi savremeni algoritmi koriste transpoziciju, to je nezgodno jer je za nju potrebno mnogo memorije i ponekad mora da bude ograničena dužina poruke. Supstitucija se koristi mnogo češće.

Rotor mašine

Dvadesetih godina XX veka izumljene su razne sprave za automatizaciju procesa šifrovanja. Mnoge od njih su se zasnivale na konceptu **rotora** – mehaničkog točka za obavljanje raznih vrsta supstitucije.

Rotor mašina (engl. *rotor machine*) ima tastaturu i niz rotora, i radi na principu Vižnerove šifre. Svaki rotor sadrži proizvoljnu permutaciju engleske abecede, ima 26 pozicija, i obavlja običnu supstituciju. Na primer, rotor može biti podešen da „A“ zamenjuje sa „F“, „B“ sa „U“, „C“ sa „L“, i tako dalje. Izlazne iglice jednog rotora povezane su sa ulaznim iglicama sledećeg.

Otvoren tekst: COMPUTER GRAPHICS MAY BE SLOW BUT AT LEAST IT'S EXPENSIVE.

```
COMPUTERGR
APHICSMAYB
ESLOWBUTAT
LEASTITSEX
PENSIVE
```

Šifrovan tekst: CAELP OPSEE MHLAN PIOSS UCWTI TSBIV EMUTE RATSG YAERB TX

Slika 1.4 Stubična transpoziciona šifra.

Na primer, na mašini sa 4 rotora prvi rotor može da zamenjuje „A" sa „F", drugi može da menja „F" sa „Y", treći „Y" sa „E", dok četvrti može da menja „E" sa „C"; „C" bi u tom slučaju bio izlazni šifrat. Neki rotori se onda pomeraju da bi sledeći put supstitucija bila drugačija.

Kombinacija nekoliko rotora i zupčanika koji ih pomeraju, čini da rezultat rada mašine bude siguran. Pošto se rotori kreću različitim brzinama, ciklus za mašinu sa n rotora iznosi 26^n. Neke rotor mašine imaju nekoliko pozicija za svaki rotor – što predstavlja dodatnu glavobolju za kriptoanalitičare.

Najpoznatiji rotor uređaj je Enigma. Enigmu su koristili Nemci za vreme Drugog svetskog rata. Do zamisli su u Evropi došli Artur Šerbijus (Arthur Scherbius) i Arvid Gerhard Dam (Arvid Gerhard Damm). Uređaj je patentirao Artur Šerbijus u Sjedinjenim Državama [1383]. Nemci su za ratnu upotrebu značajno poboljšali osnovnu konstrukciju.

Nemačka Enigma je upotrebljavala tri rotora, po izboru – od ukupno pet, komutator koji je neznatno permutovao otvoreni tekst, i reflektujući rotor zbog kojeg je svaki rotor po dva puta operisao sa svakim slovom otvorenog teksta. Tokom rata, prešlo se na izbor tri rotora od osam, dok je mornarica koristila četiri rotora od osam. Koliko god da je Enigma bila komplikovana, provaljena je za vreme Drugog svetskog rata. Poljski kriptografi su prvo provalili nemačku Enigmu, a zatim objasnili Britancima kako. Nemci su modifikovali Enigmu kako je rat napredovao, a Britanci su nastavili da analiziraju nove verzije. Objašnjenja kako rade rotor šifre i kako su provaljene, naći ćete u [794,86,448,498,446,880,1315,1587,690]. Dva fascinantna izveštaja o razbijanju Enigme su [735,796].

Dodatna literatura

Ovo nije knjiga o klasičnoj kriptografiji, pa se neću više zadržavati na pomenutim temama. Dve izuzetne knjige iz vremena pre pojave računara jesu [587,1475]; [448] predstavlja modernu kriptoanalizu nekih mašina za šifrovanje. Doroti Dening (Dorothy Denning) razmatra mnoge od ovih šifara u [456], dok u [880] daje prilično kompleksnu matematičku analizu tih istih šifara. Još jedan stariji kriptografski tekst koji govori o analognoj kriptografiji jeste [99]. Članak [579] sadrži dobar pregled ove tematike. Istorijske knjige o kriptografiji Dejvida Kana, takođe su izuzetne [794,795,796].

1.4 OBIČAN XOR ALGORITAM

XOR je operacija „ekskluzivno ili" : '^' u C-u, odnosno $\oplus$ u matematičkoj notaciji. To je standardna operacija s bitovima:

$$0 \oplus 0 = 0$$
$$0 \oplus 1 = 1$$
$$1 \oplus 0 = 1$$
$$1 \oplus 1 = 0$$

Imajte u vidu da:

$$a \oplus a = 0$$
$$a \oplus b \oplus b = a$$

Običan XOR algoritam je u stvari prava bruka; on nije ništa drugo do Vižnerova polialfabetska šifra. Ovde je pomenut samo zato što dominira u komercijalnim softverskim paketima, bar u onima iz svetova MS-DOS-a i Mekintoša [1502,1387]. Nažalost, ako je u opisu zaštitnog softvera istaknuto da sadrži vlasnički (engl. *proprietary*) algoritam za šifrovanje – koji je uz to i dosta brži od DES-a – velike su šanse da je u pitanju samo varijanta ovoga:

```
/* Upotreba:  kriptografski ključ ulazna_datoteka izlazna_datoteka */

void main (int argc, char *argv[ ])
{
     FILE *fi, *fo;
     char *cp;
     int c;

     if ((cp = argv[1]) && *cp!='\0')  {
         if ((fi = fopen(argv[2], "rb")) != NULL)  {
             if ((fo = fopen(argv[3], "wb")) != NULL)  {
                  while ((c = getc(fi)) != EOF)  {
                       if (!*cp) cp = argv[1];
                       c ^= *(cp++);
                       putc(c,fo);
                  }
                  fclose(fo);
             }
             fclose(fi);
         }
     }
}
```

Ovo je simetrični algoritam. Šifrat se dobija kao rezultat operacije XOR izvršene nad ključem i otvorenim tekstom. Pošto obrada iste vrednosti XOR-om dva puta, rekonstruiše original, šifrovanje i dešifrovanje se obavljaju na isti način:

$$P \oplus K = C$$
$$C \oplus K = P$$

Ovde nema prave sigurnosti. Ova vrsta šifrovanja lako se provaljuje, čak i bez računara [587,1475]. S računarom je potrebno samo nekoliko sekundi.

Pretpostavite da je otvoreni tekst na engleskom i da je dužina ključa bilo koji mali broj bajtova. Evo kako ga provaliti:

1. Pronađite dužinu ključa pomoću procedure poznate pod imenom **brojanje podudarnosti** (engl. *counting coincidences*) [577]. Izvršite operaciju XOR nad šifratom i njim samim pomerenim za nekoliko bajtova, i izbrojte jednake bajtove. Ako je pomeraj jednak umnošku vrednosti ključa, onda će nešto preko 6% bajtova biti jednako. Ukoliko nije, onda će manje od 0,4% biti jednako (uz pretpostavku da ASCII tekst šifruje slučajan ključ; ostali otvoreni tekstovi imaće različite brojeve). Ovo nazivamo **indeks podudarnosti** ili **indeks koincidencije** (engl. *index of coincidence*). Najmanji pomeraj kojim se dobija umnožak dužine ključa, jeste sama dužina ključa.

2. Pomerite šifrat za tu dužinu i na njega primenite operaciju XOR. Time se uklanja ključ i ostaje otvoreni tekst koji je rezultat primene operacije XOR na otvoreni tekst pomeren za dužinu ključa. Pošto engleski sadrži 1,3 bita prave informacije po bajtu (odeljak 11.1), ostaje dosta redundanse za određivanje jedinstvenog dešifrovanja.

Uprkos navedenom, zapanjuje koliko proizvođača softvera hvali ovu igračku od algoritma da je „siguran skoro kao DES“ [1387]. Na kraju je i američka NSA odobrila industriji mobilnih telefona da ovaj algoritam (sa 160-bitnim ponavljajućim „ključem“) koristi za obezbeđivanje privatnosti razgovora. XOR može sprečiti vašu mlađu sestru da pročita vaše dokumente, ali neće zaustaviti kriptoanalitičara duže od nekoliko minuta.

1.5 JEDNOKRATNA BELEŽNICA

Verovali ili ne, savršena šema za šifrovanje postoji. Zove se **jednokratna beležnica** (engl. *one-time pad*) i 1917. godine izmislili su je major Džozef Moborn (Joseph Mauborgne) i Džilber Vernam (Gilbert Vernam) iz AT&T-a [794]. (U stvari, jednokratna beležnica je specijalan slučaj **šeme „prag“** (engl. *threshold scheme*); pogledajte odeljak 3.7.) U suštini, jednokratna beležnica je veliki neponavljajući skup istinski nasumičnih ključeva, napisanih na parčetu papira i zalepljenih zajedno u beležnicu. Na samom početku to je bila teleprinterska traka za jednokratnu upotrebu. Pošiljalac koristi svako slovo ključ iz beležnice da bi šifrovao tačno jedno slovo iz otvorenog teksta. Otvoreni tekst se šifruje tako što se svaki njegov znak sabere sa znakom ključa po modulu 26.

Svako slovo ključ koristi se jednom, za samo jednu poruku. Pošiljalac šifruje poruku a onda uništava iskorišćene stranice beležnice ili iskorišćeni deo trake. Primalac ima identičnu beležnicu i koristi svaki ključ iz beležnice da bi dešifrovao svako slovo šifrata, jedno po jedno. Po dešifrovanju poruke, primalac uništava stranice iz beležnice ili deo trake. Nova poruka – nova slova ključevi. Na primer, ako je poruka:

```
ONETIMEPAD
```

a raspored ključeva iz beležnice je

```
TBFRGFARFM
```

Onda je šifrat

```
IPKLPSFHGQ
```

jer

O + T mod 26 = I
N + B mod 26 = P
E + F mod 26 = K
itd.

Šema je apsolutno sigurna ako pretpostavimo da prisluškivač komunikacionih kanala nema pristup jednokratnoj beležnici korišćenoj za šifrovanje poruke. Dati šifrat poruke može odgovarati bilo kojem otvorenom tekstu slične dužine.

Pošto je svaka sekvenca ključeva moguća (setite se, slova ključevi su generisani nasumično), protivnik nema informaciju neophodnu za kriptoanalizu šifrata. Raspored ključeva može isto tako biti:

```
POYYAEAAZX
```

što bi dešifrovanjem dalo:

```
SALMONEGGS
```

ili

```
BXFGBMTMXM
```

što bi dešifrovanjem dalo:

```
GREENFLUID
```

Na ovom mestu moramo da ponovimo: pošto je svaki otvoreni tekst poruke podjednako moguć, ne postoji način na koji bi kriptoanalitičar otkrio koji je otvoreni tekst poruke tačan. Slučajna sekvenca ključa dodata nenasumičnom otvorenom tekstu rezultira potpuno slučajnim šifratom, i nema procesorske snage koja to može da promeni.

Važno je upozoriti da se slova ključevi moraju generisati nasumice. Bilo koji napad na ovaj sistem bio bi uperen protiv metoda koje su korišćene za generisanje slova ključeva. Korišćenje pseudoslučajnih generatora brojeva ne dolazi u obzir, jer oni uvek imaju nenasumične osobine. Bezbedno je ako koristite pravi izvor slučajnih brojeva – to je mnogo teže nego što izgleda na prvi pogled; pogledajte odeljak 17.14.

Druga važna pojedinost je da više nikada ne smete koristiti isti redosled ključeva. Nikada. Čak i ako upotrebljavate višegigabajtnu beležnicu, kriptoanalitičar može da rekonstruiše otvoreni tekst ako ima više šifrata čiji se ključevi preklapaju. On postavlja svaki par šifrata jedan prema drugom, i broji poklapanja na svakoj poziciji. Ako su dobro poravnati, procenat poklapanja iznenada skače – tačan procenat zavisi od jezika kojim je pisan otvoreni tekst. Posle je kriptoanaliza laka. Slična je indeksu koincidencije, ali sa samo dva „ciklusa“ za upoređivanje [904]. Ne radite ovo.

Princip jednokratne beležnice može se lako preneti na binarne podatke. Umesto da se jednokratna beležnica sastoji od slova, koristite jednokratno beleženje bitova. Umesto unošenja otvorenog teksta u jednokratnu beležnicu, koristite XOR. Da biste dešifrovali poruku, istom jednokratnom beležnicom i operacijom XOR obradite šifrat. Sve ostalo ostaje isto a sigurnost je savršena.

Sve ovo zvuči dobro, ali ima nekoliko problema. Pošto bitovi ključeva moraju da budu nasumični, i uz to ne mogu više nikada da budu ponovo iskorišćeni, dužina ključa mora biti jednaka dužini poruke. Jednokratna beležnica može da odgovara za nekoliko kratkih poruka, ali nikada neće raditi s komunikacionim kanalom od 1,544 Mb/s. Možete da smestite 650 megabajta slučajnih bitova na CD-ROM, ali postoje problemi. Prvo, želite tačno dve kopije slučajnih bitova, ali CD-ROM je isplativ

samo za veliku količinu. Drugo, želite da uništite iskorišćene bitove. CD-ROM nije moguće izbrisati, jedino se može fizički uništiti disk. Digitalna traka je mnogo bolji medijum za ovu namenu.

Čak i ako rešite distribuciju ključeva i problem skladištenja, morate obezbediti da pošiljalac i primalac budu savršeno sinhronizovani. Ukoliko primalac kasni za bit (ili je nekoliko bitova ispušteno tokom slanja), poruka neće imati smisla. Pored toga, ako je nekoliko bitova promenjeno prilikom slanja (a da nijedan bit nije dodat ili odstranjen – što je mnogo verovatnije da će se dogoditi zbog šuma), samo će ti bitovi biti dešifrovani pogrešno. Ali, s druge strane, jednokratne beležnice ne pružaju proveru identiteta.

Jednokratne beležnice se i danas primenjuju u onim komunikacionim kanalima uskog propusnog opsega u kojima je potrebno obezbediti ultravisok nivo bezbednosti. Govorilo se da je vruća linija između SAD i bivšeg Sovjetskog Saveza bila (da li je još aktivna?) šifrovana jednokratnom beležnicom. Mnogo je sovjetskih poruka agentima bilo šifrovano jednokratnim beležnicama. Te poruke su i dan-danas bezbedne i uvek će biti. Nije bitno koliko dugo superračunari rade na problemu. Čak i ako dođu vanzemaljci sa Andromede sa svojim ogromnim brodovima i nezamislivo snažnim kompjuterima, neće biti u stanju da pročitaju sovjetske špijunske poruke šifrovane jednokratnim beležnicama (osim ako mogu da se vrate kroz vreme i nabave te jednokratne beležnice).

1.6 Računarski algoritmi

Postoji mnogo kriptografskih algoritama. Najčešće se koriste sledeća tri:

- DES (Data Encryption Standard) najpopularniji je kompjuterski algoritam za šifrovanje. DES je američki i međunarodni standard. U pitanju je simetrični algoritam; isti ključ se koristi za šifrovanje i dešifrovanje.
- RSA (ime je dobio po svojim tvorcima – Rivestu, Shamiru i Adlemanu) najpopularniji je algoritam s javnim ključem. Može se koristiti za šifrovanje i digitalno potpisivanje.
- DSA (Digital Signature Algorithm, koristi se kao deo DSA-a – Digital Signature Standard) predstavlja još jedan algoritam s javnim ključem. Ne može se koristiti za šifrovanje, već samo za digitalno potpisivanje.

Ova knjiga se bavi navedenim algoritmima.

1.7 Veliki brojevi

Kroz celu knjigu koristim mnogo velikih brojeva da bih opisao razne oblasti kriptografije. Pošto je veoma lako izgubiti iz vida ove brojeve i šta oni predstavljaju, u tabeli 1.1 objašnjeni su neki od njih.

TABELA 1.1
Veliki brojevi

Fizički analog	Broj
Šanse da poginete od udara groma (dnevno)	1 u 9 milijardi (2^{33})
Šanse da osvojite glavnu nagradu na američkoj lutriji	1 u 4,000,000 (2^{22})
Šanse da osvojite glavnu nagradu na američkoj lutriji i da poginete od udara groma u istom danu	1 u 2^{55}
Šanse da se udavite (u Americi, godišnje)	1 u 59,000 (2^{16})
Šanse da poginete u automobilskoj nesreći (u Americi, 1993)	1 u 6100 (2^{13})
Šanse da poginete u automobilskoj nesreći (u Americi, tokom celog života)	1 u 88 (2^{7})
Vreme do sledećeg ledenog doba	14,000 (2^{14}) godina
Vreme dok Sunce ne postane super nova	10^{9} (2^{30}) godina
Starost planete	10^{9} (2^{30}) godina
Starost univerzuma	10^{10} (2^{34}) godina
Broj atoma u planeti	10^{51} (2^{170})
Broj atoma u Suncu	10^{57} (2^{190})
Broj atoma u galaksiji	10^{67} (2^{223})
Broj atoma u univerzumu (ne računajući tamnu materiju)	10^{77} (2^{265})
Zapremina univerzuma	10^{84} (2^{280}) cm^3
Ako je univerzum zatvoren:	
Ukupni vek univerzuma	10^{11} (2^{37}) godina 10^{18} (2^{61}) sekundi
Ako je univerzum otvoren:	
Vreme dok se male zvezde ne ohlade	10^{14} (2^{47}) godina
Vreme dok se planete ne odvoje od zvezda	10^{15} (2^{50}) godina
Vreme dok se zvezde ne odvoje od galaksija	10^{19} (2^{64}) godina
Vreme do deformacije orbite usled gravitacionog zračenja	10^{20} (2^{67}) godina
Vreme dok crne rupe ne ispare po Hokingu	10^{64} (2^{213}) godina
Vreme dok se temperatura celog univerzuma ne spusti na apsolutnu nulu	10^{65} (2^{216}) godina
Vreme dok sva materija ne pređe u gvožđe	$10^{10^{26}}$ godina
Vreme dok se sva materija ne uruši u crnu rupu	$10^{10^{76}}$ godina

Ovi brojevi su procene reda veličine i potiču iz raznih izvora. Mnogi brojevi iz oblasti astrofizike objašnjeni su u radu Frimana Dajsona (Freeman Dyson) „Time Without End: Physics and Biology in an Open Universe" („Vreme bez kraja: fizika i biologija u otvorenom univerzumu"), objavljenog u časopisu *Reviews of Modern Physics,* v. 52, n. 3, July 1979, pp. 447–460). Smrtni slučajevi kao posledica saobraćajnih nesreća, 163 smrtna slučaja na milion ljudi i uz prosečan životni vek od 69,7 godina, izračunati su prema statističkim podacima iz američkog Ministarstva saobraćaja.