Zavod: ZEMRIS
Kolegij: Operacijski sustavi 2
Student: **Dinko Korunić, 0036355514**
Voditelji: prof. dr. sc. Leo Budin, dr. sc. Marin Golub

Blowfish simetrični kriptografski algoritam

(teorija i praktična implementacija)

1. Općenito

Blowfish je simetrični algoritam razvijen 1993. godine od Bruce Schneiera, nezavisnog savjetnika i kriptografa i danas se smatra jednim od najpopularnijih alternativa DES algoritmu. Osnovne karakteristike su jednostavnost implementiranja i velika brzina izvođenja, kao i mali memorijski zahtjevi - dotični algoritam se može izvršavati čak na platformama sa manje od 5KB radne memorije. Osim toga, spomenimo i da je dužina ključa proizvoljne dužine te može biti čak 448 bitova; dok sam algoritam koristi 128-bitne ključeve i 16 rundi. Ako ga uspoređujemo sa DES-om, također možemo vidjeti da se koriste S-boxovi i XOR funkcija, no koristi se i binarno zbrajanje. I još jedna razlika od DES-a: Blowfish ne koristi statičke S-boxove, već ih dinamički generira kružno ponavljajući Blowfish algoritam pri čemu je potrebno 512 interacija BF algoritma da bi se stvorili podključevi i S-boxovi - iz ovoga je očit nedostatak BF algoritma za aplikacije gdje se tajni ključ često mijenja.

2. Specifikacije

Dužina ključa: minimalno 32, maksimalno 448, standardno 128 (umnošci od 8 bitova)
Veličina bloka: 8 bajtova
Sigurnosni komentar: tzv. slabi ključevi (Serge Vaudenay, "On the weak keys of Blowfish", Fast Software Encryption, Third International Workshop, Volume 1008 of Lecture Notes in Computer Science (B. Preneel, ed.), pp. 286-297. Springer-Verlag, 1995.) ne vrijede za puni (16-rundi) BF
Varijante: Blowfish-Direct ili Blowfish-ISK (ključevi su specificirani kao P1..P18, zatim S1, S2, S3 i S4, pri čemu je svaki zapis 4 bajta u zapisan big-endian slijedu)

3. Detaljni opis

Blowfish je u osnovi Feistel mreža, te se ostvaruje iteriranjem jednostavne enkripcijske funkcije 16 puta, pri čemu je veličina bloka 64 bita i ključ može biti bilo koje dužine do uključno 448 bitova. Nakon relativno složenog inicijalizacijskog procesa (i praktički neizbježnog ako se mijenja ključ) slijedi relativno jednostavan i brz proces enkripcije. Sam algoritam se pokazao vrlo upotrebljivim u nizu primjena na različitim platformama (specijalni hardver, veliki/srednji/mali mikroprocesori): enkripcija datoteka sa podacima i kontinuiranih tokova podataka, generiranju slučajnih bitova, enkripciji paketno orijentiranih podataka (npr. ATM paket), hashing.
Sam algoritam se sastoji iz dva dijela - tzv. key-expansion part u kojem se ključ (do 448 bitova) dijeli na nekoliko vektora podključeva (ukupno do 4168 bajtova), te sam proces enkripcije podataka. Kao što smo već spomenuli, enkripcija podataka se dešava u 16-ciklusnoj Feistel mreži, pri čemu se svaki ciklus sastoji od permutacije ovisne o ključu i supstitucije koja je ovisna i o ključu i o samom podatku. Same operacije se sastoje isključivo od XOR-ova i binarnih zbrajanja na 32-bitnim riječima, te adresiranja četiriju indeksiranih vektora podataka.

3.1. Podključevi

BF koristi veliki broj podključeva koji moraju biti izračunati prije same enkripcije i dekripcije.

1. P-vektor se sastoji od 18 32-bitnih podključeva, redom:
   P1, P2, ..., P18
2. uz to postoji 4 32-bitna S-boxa od kojih svaki ima 256 zapisa
   S1,0, S1,1, ..., S1,255
   S2,0, S2,1, ..., S2,255
   S3,0, S3,1, ..., S3,255
   S4,0, S4,1, ..., S4,255

3.2. Enkripcija i dekripcija

Kao što je već rečeno, BF algoritam je Feistel mreža koja se sastoji od 16 ciklusa. Ulaz je 64bitni podatak "x", a sam pseudokod enkripcijskog algoritma slijedi:

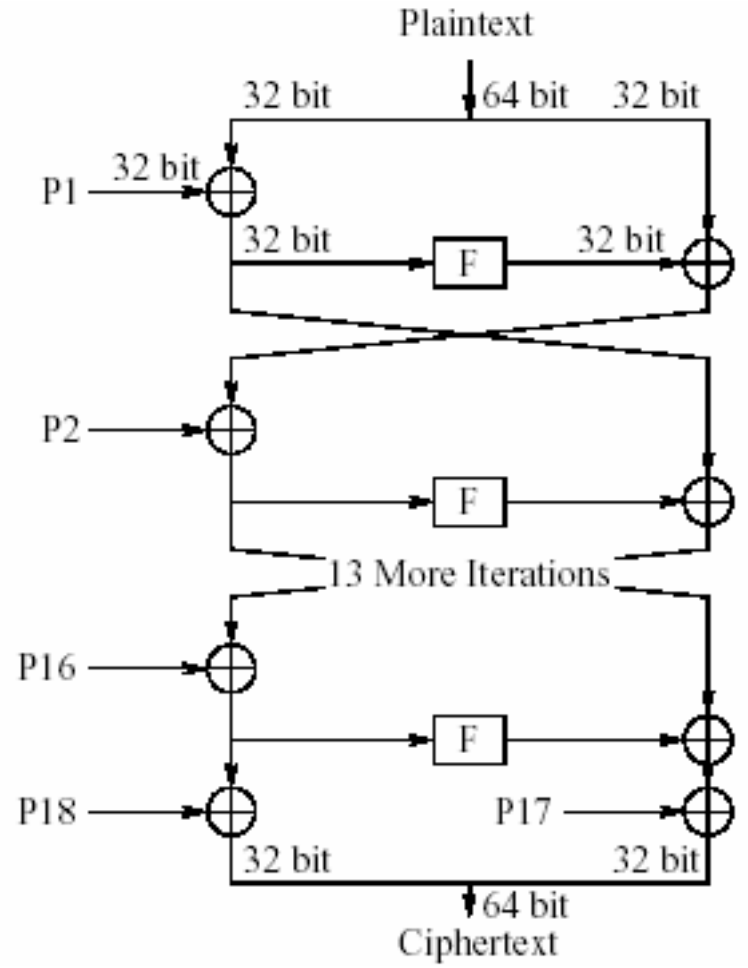
```
podijeli "x" u dvije 32-bitne polovine: xL i xR
dok je i = 1 do 16 čini
   xL = xL XOR Pi
   xR = F(xL) XOR xR
   zamijeni xL i xR
zamijeni xL i xR (vraćamo nazad zadnju zamjenu)
xR = xR XOR P17
xL = xL XOR P18
spoji xL i xR
```

Sama funkcija F je također vrlo jednostavnog pseudokoda:

```
podijeli xL u 4 8-bitne četvrtine: a, b, c, d
F(xL) = ((((S1,a + S2,b) mod 2^32) XOR S3,c) + S4,d) mod
2^32
```

Što se pak tiče dekripcije, algoritam je identičan s time da se P1, P2, ..., P18 koriste u obrnutom redoslijedu:

```
podijeli "x" u dvije 32-bitne polovine: xL i xR
dok je i = 18 do 2 čini
   xL = xL XOR Pi
   xR = F(xL) XOR xR
   zamijeni xL i xR
zamijeni xL i xR (vraćamo nazad zadnju zamjenu)
xR = xR XOR P2
xL = xL XOR P1
spoji xL i xR
```



3.3. Generiranje podključeva

Svi podključevi se izračunavaju samim BF algoritmom. Algoritam je slijedeći:

1. incijalizira se prvo P-vektor, a zatim i sva četiri S-boxa sa unaprijed definiranim znakovnim nizom - a riječ je o heksadecimalnom zapisu broja PI (bez vodeće znamenke 3.):
   P1 = 0x243f6a88
   P2 = 0x85a308d3
   P3 = 0x13198a2e
   P4 = 0x03707344
2. čini se XOR P1 sa prvim 32-bita ključa, XOR P2 sa drugim 32-bitna ključa i tako dalje za sve bitove ključa (moguće do P14); ovo se kružno izvršava sve dok cijeli P-vektor ne bude XOR-an sa bitovima ključa (ako je A 64-bitni ključ onda mu je ekvivalentni AA, AAA itd. kojim se XOR-a)

3. enkriptiramo zero-string sa BF algoritmom koristeći ključeve iz koraka 1. i 2.
4. zamijenimo P1 i P2 sa rezultatom koraka 3.
5. enkriptiramo rezultat koraka 3 sa BF algoritmom i modificiranim podključevima
6. zamijenimo P3 i P4 sa rezultatom koraka 5.
7. nastavljamo proces zamjenjujući sve zapise u P-vektoru i zatim sva 4 S-boxa slijedno iz dobivenih (stalno mijenjajućih) rezultata

Ukupno je potrebno 521 iteracija za stvaranje svih potrebnih podključeva - iz ovog je očito da je bolje ovako dobivene podključeve spremiti nakon proračuna, nego svaki put izračunavati kad se ukaže potreba.

4. Opis same softverske implementacije

U našem slučaju logičan odabir je bio C jezik zbog svoje portabilnosti i brzine. Osim toga, u njemu je bilo iznimno jednostavno vršiti potrebne bitovne operacije (shift right kao i bitovni XOR). Sam algoritam je praktički trivijalno implementirati, a za incijalno punjenje P-vektora i S-boxova korišteni su konstante dostupne na adresi http://www.counterpane.com/constants.txt, dok su testni vektori korišteni u ispitivanju ispravnosti same implementacije dostupni na adresama http://www.counterpane.com/vectors.txt i http://www.counterpane.com/vectors2.txt.

5. Literatura

- "Ritter's Crypto Glossary and Dictionary of Technical Cryptography"
- Counterpane Labs: Blowfish Paper (Description of a New Variable-Length Key, 64-Bit Block Cipher (Blowfish))
- http://www.counterpane.com/blowfish.html
- C kodovi: referentni kod iz SSLeay 0.6.6, Bruce Schneier implementacija, Paul Kocher implementacija
- već spomenuti testni vektori i konstante dobiveni iz Counterpane Labs

6. Primjer funkcioniranja programa

```
Polazno:        aaccbbdd 1626280a
Enkriptirano:   c0031fc2 feab3bc6
Dekriptirano:   aaccbbdd 1626280a
```