Sveučilište u Zagrebu
Fakultet elektrotehnike i računarstva
Zavod za elektroničke sustave i obradbu informacija

Seminarski rad iz kolegija
Sustavi za praćenje i vođenje procesa

Algoritmi za izračunavanje sažetka

Marko Štrkalj

Zagreb, lipanj 2005.g.

Sažetak

Uvod

Algoritmi za izračunavanje sažetka poruke (engl. *hash functions*) igraju temeljnu ulogu u modernoj kriptografiji. Algoritmi za izračunavanje sažetka uzimaju poruku kao ulaz i proizvode izlaz koji nazivamo sažetak poruke. Preciznije algoritam za izračunavanje sažetka h preslikava niz proizvoljne duljine u niz utvrđene duljine, recimo n bitova. Ovaj algoritam kao funkcija je preslikavanje više u jedan, a to implicira da je postojanje sukoba (parova ulaza s istim izlazom) neizbježno.Osnovna ideja algoritama za izračunavanje sažetka poruke je dobivanje sažetog predstavnika ulaznog niza koji se koristi kao poistovjećenje tog niza. Algoritmi za izračunavanje sažetka se koriste za provjeru integriteta podataka kod algoritama digitalnog potpisivanja, gdje se zbog više razloga prvo izračunava sažetak poruke, i zatim se sažetak, kao predstavnik poruke, potpisuje umjesto originalne poruke. Drugačiji razred algoritama za izračunavanje sažetka, zvani kodovi za autentifikaciju poruke (*engl. Message Authentication Codes* ili skraceno *MAC*), omogućuje autentifikaciju poruke simetričnim tehnikama. MAC algoritam je algoritam za generiranje kratkog niza podataka koji se koristi za provjeru autentičnosti poruke. MAC algoritam sastoji se od algoritma za izračunavanje sažetka poruke i tajnog kljuca koji kao rezultat daju MAC (ponekad znan kao *tag*). Kodovi za autentifikaciju poruka se računaju i provjeravaju koristeći isti ključ, tako da ga samo oni kojima je poruka namijenjena mogu provjeriti. MAC-ovi se mogu koristiti za provjeru integriteta podataka i simetričnu autentifikaciju izvora podataka, kao i za identifikaciju u shemama simetričnog ključa.

Definicija algoritma za izračunavanje sažetka

Definicija 1: Algoritam za izračunavanje sažetka (u neograničenom smislu) je funkcija h koja ima, kao minimum, slijedeća dva svojstva:

1. **kompresija** – h preslikava ulaz x proizvoljne konacne duljine, na izlaz $h(x)$ zadane duljine n .
2. **lakoća izračuna** – s danom funkcijom h i ulazom x , $h(x)$ je lagano izračunati.

Na najvišoj razini algoritmi za izračunavanje sažetka se mogu podijeliti u dva razreda:
• Algoritam za izračunavanje sažetka bez kljuca (MDC)
- funkcija ima samo jedan ulazni parametar, poruku.
• Algoritmi za izračunavanje sažetka s kljucem (MAC)
- funkcija ima dva različita ulazna parametra, poruku i tajni ključ

Algoritmi za izračunavanje sažetka bez ključa

Da olakšamo daljnje definicije, navedena su tri potencijalna svojstva (uz kompresiju i jednostavnost izračuna), za algoritam za izračunavanje sažetka bez ključa s ulazima x , x' i izlazima y , y' .

• **jednosmjeran** – za sve unaprijed definirane izlaze, računski je neizvedivo pronaći bilo kakav ulaz koji bi računanjem sažetka dao taj izlaz.

• **slaba otpornost na kolizije** – računski je neizvedivo pronaći neki drugi ulaz koji ima isti izlaz kao bilo koji specificirani ulaz.

• **jaka otpornost na kolizije** – računski je neizvedivo pronaći dva različita ulaza x , x' koji daju isti izlaz iz funkcije sažetka npr. $h(x) = h(x')$. (Primjetite da je ovdje slobodan izbor oba ulaza).

Jednosmjerni algoritam za izračunavanje sažetka (*engl. One way hash function, kratica OWHF*) je algoritam za izračunavanje sažetka h kao prema definiciji 1. sa sljedećim dodatnim svojstvima: jednosmjernost, slaba otpornost na kolizije.

Algoritam za izračunavanje sažetka otporan na kolizije (*engl. Collision resistant hash function, kratica CRHF*) je funkcija za izračunavanje sažetka h kao prema definiciji 1. sa slijedecim dodatnim svojstvima: slaba otpornost na kolizije, jaka otpornost na kolizije. Iako u praksi algoritam za izračunavanje sažetka otporan na kolizije gotovo uvijek ima i dodatno svojstvo jednosmjernosti, zbog tehnickih uvjeta ovo svojstvo nije uvjetovano ovdje.

Dodatna svojstva jednosmjernih algoritama za izračunavanje sažetka:

• **međusobna nepovezanost** – Ne bi smjela postojati veza izmedu ulaznih i izlaznih bitova. Povezano s time, poželjno je svojstvo lavine slicno onima kod dobrih algoritama za kriptiranje blokova gdje svaki ulazni bit utjece na svaki izlazni bit.

• ***near-collision*** **otpornost** - Trebalo bi biti teško pronaći bilo koja dva ulaza x, $x0$ takva da se $h(x)$ i $h(x0)$ razlikuju u samo nekoliko bitova.

• **lokalna jednosmjernost** – Trebalo bi biti jednako teško ponovo dobiti bilo koji podniz kao i ponovno dobivanje cijelog ulaza. Osim toga, čak i ako je poznat dio ulaza trebalo bi biti teško pronaći ostatak.

Generalni model iterativnih algoritama za izračunavanje sažetka

Većina algoritama za izračunavanje sažetka bez ključa su dizajnirani kao iterativni procesi koji sažimaju ulaze proizvoljne duljine obrađujući uzastopne blokove ulazne poruke fiksne duljine. Ulaz u funkciju za izračunavanje sažetka x proizvoljne konačne duljine je podijeljen na r -bitne blokove fiksne duljine x_i . To pretprocesiranje uobičajeno uključuje pridodavanje dodatnih bitova (podloge) što je potrebno za dobivanje ukupne duljine koja je višekratnik duljine bloka r i često uključuje (zbog sigurnosnih razloga) blok ili djelomični blok koji predstavlja duljinu u bitovima neproširene poruke. Svaki blok x_i tada služi kao ulaz za algoritam za izračunavanje sažetka fiksne duljine f , kompresijskoj funkciji od h , koja računa novi prijelazni rezultat duljine n bitova, kao funkcija prethodnog prijelaznog rezultata i slijedećeg ulaznog bloka x_i .

Neka H_i označava prijelaznu vrijednost nakon koraka i, općeniti postupak za iterativnu funkciju za izračunavanje sažetka s ulazom $x = x_1 x_2 \ldots x_t$ možemo oblikovati kao što slijedi:

$$H_0 = IV;\ H_i = f(H_{i-1}, x_i);\ 1 = x = t;\ h(x) = g(H_t)$$

H_{i-1} služi kao n -bitna varijabla ulančavanja izmedu koraka $i-1$ i i , H_0 je unaprijed određena početna vrijednost ili inicijalizirajuća vrijednost (IV). Opcionalna izlazna transformacija g se koristi u zadnjem koraku za preslikavanje n -bitne varijable ulančavanja u m -bitni rezultat $g(H_t)$; g je često istovjetno preslikavanje $g(H_t) = H_t$. Određene algoritmi za izračunavanje sažetka se razlikuju po načinu pretprocesiranja, kompresijskoj funkciji i izlaznoj transformaciji.

MD5 Algoritam

MD5 algoritam kao ulaz uzima poruku proizvoljne duljine i kao izlaz daje 128-bitni otisak ili sažetak poruke ulazne poruke. Može se pretpostaviti da je računski nemoguće proizvesti dvije poruke koje bi imale jednaki sažetak poruke ili proizvesti bilo kakvu poruku iz unaprijed zadanog sažetka poruke. MD5 algoritam je namijenjen za korištenje u aplikacijama za digitalno potpisivanje, gdje treba "komprimirati" veliku datoteku na siguran način prije nego se kriptira s privatnim ključem kriptosustava s javnim ključem kao RSA. Vjerojatnost da dvije poruke imaju isti sažetak je reda 2^{64} operacija, a težina otkrivanja originalne poruke iz sažetka je reda 2^{128} operacija.

Opis MD5 Algoritma

Ulaz je poruka duljine b -bitova. Broj b je proizvoljan pozitivni cijeli broj; b može biti nula, ne mora biti djeljiv s osam i može biti proizvoljno velik. Bitovi poruke zapisani su kao:

$$m_0 m_1 \ldots m_{\{b-1\}}$$

Kako bi izračunali sažetak poruke izvodi se slijedećih pet koraka.

1. Korak - Proširivanje poruke

Poruka je proširena tako da je njena duljina (u bitovima) kongruentna 448 mod 512. Drugim riječima, poruka je proširena tako da je za 64 bita prekratka da bude višekratnik od 512. Proširivanje poruke se uvijek izvodi, čak i ako je poruka vec kongruentna 448 mod 512.

Proširivanje se izvodi kao što slijedi: jedan bit '1' se dodaje poruci, a zatim se dodaje onoliko bitova '0' koliko je potrebno da poruka bude kongruentna 448 mod 512. Sveukupno gledajući, dodaje s najmanje jedan a najviše 512 bitova.

2. Korak - Proširivanje duljine

Rezultatu prethodnog koraka dodaje se 64-bitni zapis broja b (duljina poruke prije proširivanja poruke). U slučaju da je b veci od 2^{64} , onda se koriste samo 64 manje značajnih bitova (ti bitovi se dodaju kao dvije 32-bitne riječi i dodaje se prvo manje značajnija riječ).

U ovoj točki poruka ima duljinu koja je višekratnik od 512. Istovremeno, poruka ima duljinu koja je višekratnik od 16 (32-bitnih) rijeci. Neka $M[0 \ldots N-1]$ oznacava rijeci rezultirajuce poruke, gdje je N višekratnik od 16.

3. Korak - Inicijalizacija MD spremnika

Za izračunavanje sažetka poruke koristi se spremnik velicine četiri riječi (*A, B, C, D*). Svaki od *A, B, C, D* je 32-bitni registar. Registri su inicijalizirani na slijedeće heksadecimalne vrijednosti (manje značajni bajtovi prvi):

rijec *A* : *01 23 45 67*

rijec *B* : *89 ab cd ef*

rijec *C* : *fe dc ba 98*

rijec *D* : *76 54 32 10*

4. Korak - Obrada poruke po blokovima duljine 16-riječi

Prvo definiramo četiri pomoćne funkcije od kojih svaka kao ulaz uzima tri 32-bitne riječi i na izlazu daje jednu 32-bitnu riječ.

F(X,Y,Z) = XY v not(X) Z

G(X,Y,Z) = XZ v Y not(Z)

H(X,Y,Z) = X xor Y xor Z

I(X,Y,Z) = Y xor (X v not(Z))

Ovaj korak koristi tablicu od 64 elementa *T[1 ... 64]* sastavljene iz sinusne funkcije. Neka *T[i]* označava *i*-ti element tablice, koji je jednak cjelobrojnom dijelu umnoška *4294967296* i *abs(sin(i)),* gdje je *i* izražen u radijanima.

Pseudokod algoritma obrade poruke :

/* Obradi svaki blok od 16-riječi*/

For i = 0 to N/16-1 do

/* Kopiraj blok *i* u X. */

For j = 0 to 15 do

Set X[j] to M[i*16+j].

end /* kraj petlje varijable j */

/* Pohrani A kao AA, B kao BB, C kao CC, i D kao DD. */

AA = A

BB = B

CC = C

DD = D

/* 1.Korak */

/* Neka [abcd k s i] predstavlja operaciju

a = b + ((a + F(b,c,d) + X[k] + T[i]) <<< s). */

/* Obavi slijedecih 16 operacija. */

[ABCD 0 7 1] [DABC 1 12 2] [CDAB 2 17 3] [BCDA 3 22 4]

[ABCD 4 7 5] [DABC 5 12 6] [CDAB 6 17 7] [BCDA 7 22 8]

[ABCD 8 7 9] [DABC 9 12 10] [CDAB 10 17 11] [BCDA 11 22 12]

[ABCD 12 7 13] [DABC 13 12 14] [CDAB 14 17 15] [BCDA 15 22 16]

/* 2.Korak */

/* Neka [abcd k s i] predstavlja operaciju

a = b + ((a + G(b,c,d) + X[k] + T[i]) <<< s). */

/* Obavi slijedecih 16 operacija. */

[ABCD 1 5 17] [DABC 6 9 18] [CDAB 11 14 19] [BCDA 0 20 20]

[ABCD 5 5 21] [DABC 10 9 22] [CDAB 15 14 23] [BCDA 4 20 24]

[ABCD 9 5 25] [DABC 14 9 26] [CDAB 3 14 27] [BCDA 8 20 28]

[ABCD 13 5 29] [DABC 2 9 30] [CDAB 7 14 31] [BCDA 12 20 32]

/* 3.Korak */

/* Neka [abcd k s t] predstavlja operaciju

a = b + ((a + H(b,c,d) + X[k] + T[i]) <<< s). */

/* Obavi slijedecih 16 operacija. */

[ABCD 5 4 33] [DABC 8 11 34] [CDAB 11 16 35] [BCDA 14 23 36]

[ABCD 1 4 37] [DABC 4 11 38] [CDAB 7 16 39] [BCDA 10 23 40]

[ABCD 13 4 41] [DABC 0 11 42] [CDAB 3 16 43] [BCDA 6 23 44]

[ABCD 9 4 45] [DABC 12 11 46] [CDAB 15 16 47] [BCDA 2 23 48]

/* 4.Korak */

/* Neka [abcd k s t] predstavlja operaciju

a = b + ((a + I(b,c,d) + X[k] + T[i]) <<< s). */

/* Obavi slijedecih 16 operacija. */

[ABCD 0 6 49] [DABC 7 10 50] [CDAB 14 15 51] [BCDA 5 21 52]

[ABCD 12 6 53] [DABC 3 10 54] [CDAB 10 15 55] [BCDA 1 21 56]

[ABCD 8 6 57] [DABC 15 10 58] [CDAB 6 15 59] [BCDA 13 21 60]

[ABCD 4 6 61] [DABC 11 10 62] [CDAB 2 15 63] [BCDA 9 21 64]

/* Zatim obavi slijedeca zbrajanja. (Zbroji svaki registar s

vrijednošcu koju je imao prije ovog bloka.) */

A = A + AA

B = B + BB

C = C + CC

D = D + DD

end /* kraj petlje varijable i */

5. Korak - Izlaz. Sažetak poruke kao rezultat je sadržaj registara *A, B, C, D* i to tako da započinje s manje značajnijim bitovima od *A* , a završava značajnijim bitovima od *D* .

Algoritmi za izračunavanje sažetka s ključem MAC

Algoritmi za izračunavanje sažetka s ključem (*engl. Message authentification code*) su familija funkcija h_k parametriziranih tajnim ključem k , sa slijedećim svojstvima:

1. **lakoća izračuna** - za poznatu funkciju h_k , sa zadanim vrijednostima k i ulaza x , $h_k(x)$ je lagano izračunati. Taj rezultat se zove MAC vrijednost ili MAC.

1. **kompresija** - h_k preslikava ulaz proizvoljne konačne duljine na izlaz $h_k(x)$ određene duljine n.

Nadalje, s danim opisom familije funkcija h , za svaku fiksnu dozvoljivu vrijednost k (tzv. nepoznatu napadaču), vrijedi slijedeće svojstvo:

1. **računska otpornost** - sa zadanim nijednim ili više tekst-MAC parova (x_i ; h_k (x_i)), računski je neizvedivo izračunati bilo koji tekst-MAC par (x ; h_k (x)) za bilo koji novi ulaz $x \neq xi$ (uključujuci moguće za h_k (x) = h_k (x_i) za neki i).

Ako računska otpornost ne drži, MAC algoritam je podložan MAC krivotvorenju. Dok računska otpornost povlači svojstvo neoporavka ključa (mora biti računski neizvedivo saznati k , s poznatim jednim ili više parova tekst-MAC (x_i ; h_k (x_i)) za taj k), neoporavak ključa ne povlači računsku otpornost (ne treba se svaki put saznati ključ kako bi krivotvorili nove MAC-ove).

Postoje četiri vrste MAC-ova:

1. bezuvjetno siguran - temelji se na kriptiranju jednostrukim podloškom. Kriptirani tekst poruke autentificira sam sebe, budući da nitko drugi nema pristup jednostrukom podlošku. Medutim, mora postojati određena zalihost u poruci. Bezuvjetno siguran MAC možemo dobiti uporabom jednokratnog tajnog ključa.

2. temeljen na hash funkciji – koriste ključ ili ključeve u vezi hash funkcijom da proizvede MAC koji se pridodaje poruci.

3. temeljen na algoritmima za kriptiranje tokova – U svom algoritmu, dokazano siguran algoritam se koristi za dijeljenje poruke u dva podtoka i svaki podtok se propušta kroz LFSR, MAC je konačno stanje dva LFSR-a.

4. temeljeni na algoritmima za kriptiranje blokova – DES-CBC MAC je širom korišten S.A.D. i internacionalni standard. Osnovna ideja je se poruka kriptira koristeći DES u CBC modu i da se zadnji izlazni blok koristi kao MAC poruke.

HMAC

Pružanje načina provjere integriteta informacije koja se prenosi ili pohranjuje na nepouzdanim medijima je primarna potreba u svijetu otvorenog računarstva i komunikacije. Mehanizmi koji pružaju takvu provjeru integriteta temeljenu na tajnom ključu uobičajeno zovemo "kodovi za autentifikaciju poruke" (MAC). Tipično, MAC-ovi se koriste izmedu dvije strane koje dijele tajni ključ kako bi provjerili valjanost informacije izmijenjene između tih strana. HMAC se može koristiti u kombinaciji s bilo kojom iterativnom funkcijom za izračunavanje sažetka poruke. MD5 i SHA-1 su primjeri takvih funkcija. HMAC jednako tako koristi tajni ključ za računanje i provjeru MAC-ova. Glavni ciljevi takve konstrukcije su:

- uporaba, bez promjena, dostupnih algoritama za izračunavanje sažetka. Posebno, algoritmi za izračunavanje sažetka koje se dobro izvode u softveru i za koje je kod slobodno i širom dostupan.

- sačuvati izvorne performanse algoritma za izračunavanje sažetka bez uvođenja značajnih degradacije performansi.
- uporaba i rukovanje ključevima na jednostavan način.
- posjedovanje razumljive kriptografske analize otpornosti mehanizma autentifikacije temeljene na razumnim pretpostavkama o hash funkciji ispod.
- dozvoliti laganu zamjenu algoritma za izračunavanje sažetka u slučaju da su pronađeni ili zahtijevani brži ili sigurniji algoritmi za izračunavanje sažetka.

HMAC-MD5

HMAC-MD5 je HMAC algoritam temeljen na MD5 hash funkciji.

Opis algoritma**:**

Neka niz znakova 'text' označava podatke na koje će biti primijenjen HMAC-MD5 algoritam, a *K* neka označava tajni ključ autentifikacije poruke kojeg dijele sudionici. Ključ *K* može biti proizvoljne duljine do duljine bloka algoritma za izračunavanje sažetka, tj. do 64 bajta za MD5 (medutim, 16 bajtova je minimalna preporučena duljina za ključeve).

Definiramo dva stalna i različita niza *ipad* i *opad* kao što slijedi ('i' i 'o' mnemonici za unutarnji i vanjski):

ipad = bajt 0x36 ponovljen 64 puta

opad = bajt 0x5C ponovljen 64 puta

Da izračunamo HMAC-MD5 od podataka *'text'* izvodimo

MD5(K XOR opad, MD5(*K XOR ipad* , *text*))

To jest potrebno je:

• dodati nule na kraj *K* tako da se dobije 64-bajtni niz znakova (npr. Ako je *K* duljine 16 bajtova, dodaju se 48 nula bajtova *0x00*)

• izračunati vrijednost XOR funkcije između niza dobivenog u koraku 1) i *ipad* -a

• dodati niz podataka *'text'* 64-bitnom nizu dobivenog u koraku 2)

• primijeniti MD5 algoritam na niz generiran u koraku 3)

• izračunati vrijednost XOR funkcije između niza dobivenog u koraku 1) i *opad* -a

• pridodati rezultat MD5 algoritma iz koraka 4) na 64-bajtni niz dobiven u koraku 5)

• primijeniti MD5 algoritam na niz generiran u koraku 6) i uzeti rezultat s izlaza

Ključevi

Ključevi za HMAC-MD5 mogu biti proizvoljne duljine (ključevi dulji od 64 bajta se prvo hashiraju koristeći MD5 i zatim rezultirajućih 16 bajtova koristimo kao ključ za HMAC-MD5). Medutim, ključevi kraći od 16 bajtova se ne preporučaju jer bi smanjili sigurnost algoritma. Ključevi dulji od 16 bajtova su prihvatljivi, ali dodatna duljina ne bi značajnije povećala snagu algoritma. (Dulji ključ je preporučljiv ako se smatra da je slučajnost ključa slaba.). Ključevi trebaju biti izabrani slučajno, ili koristeći kriptografski jak generator pseudoslučajnih brojeva sa slučajnom početnom vrijednosti. Preporuča se periodička i što učestalija promjena kljuceva.

Ostali algoritmi

Uz MD5, SHA-1 je najčešće korišten algoritam za izračunavanje. Uz SHA-1 imamo cijelu familiju SHA algoritama, SHA-256, SHA-384, i SHA-512 koji se razlikuju po duljini sažetka poruke. RIPEMD-160 (*engl. RACE Integrity Primitives Evaluation Message Digest*) je algoritam za izračunavanje sažetka duljine sažetka 160 bita. Razvijen je u Europi u otvorenoj akademskoj zajednici. Dizajn je sličan MD4 familiji algoritama i sličan je po brzini i snazi SHA algoritmu. Također postoje i verzije s duljinama sažetka od 128, 256 i 320 bitova. Među poznatije algoritme još spada i algoritam Snefru.

Primjer algoritma za izračunavanje sažetka temeljen na modularnoj aritmetici je MASH-1 (*engl. Modular Arithmetic Secure Hash algorithm 1*). MASH-1 koristi RSA sličan modul M; čija duljina utječe na sigurnost. M takoder odreduje veličinu bloka i duljinu sažetka (npr. 1025 bitni modul kao rezultat daje 1024 bitni sažetak). Budući da je algoritam relativno nov njegova sigurnost nije u potpunosti poznata.

Literatura:

- *http://mapmf.pmfst.hr/*