Zgoščevalne funkcije v kriptografiji

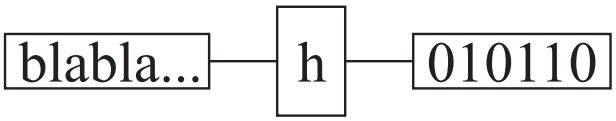
Projekt iz Kriptografije in računalniške varnosti

Tomaž Zaplotnik
mentor: dr. Aleksandar Jurišič

15. Julij 2002

1 Uvod

Zgoščevalne funkcije danes v kriptografiji zavzemajo precej pomembno mesto in se uporabljajo na veliko različnih področjih. Osnovna ideja zgoščevalnih funkcij je, da preslikajo vhod poljubne dolžine v izhod fiksne dolžine (slika 1). V tem sestavku bomo najprej predstavili nekaj osnovnih definicij, ki določajo, kaj zgoščevalne funkcije sploh so, ter nekaj različnih tipov in področij uporabe. Sledi kratka predstavitev družine zgoščevalnih funkcij MD in njene lastnosti. V nadaljevanju se bomo posvetili predvsem varnosti zgoščevalnih funkcij ter napadom nanje, saj je v kriptografiji to najbolj zanimivo in preučevano področje. Na koncu je podan še kratek pregled že razbitih ali vsaj ne več zadostno varnih zgoščevalnih funkcij iz družine MD.



Slika 1:
Ponazoritev zgoščevalne funkcije.

2 Definicije

Definicija 1 *Funkcija* $h : \{0,1\}^* \longrightarrow \{0,1\}^n$*, ki preslika sporočilo* x *poljubne dolžine v niz* y *fiksne dolžine* n*, se imenuje enosmerna zgoščevalna funkcija (ang. One-Way Hash Function ali krajše OWHF), če zadošča naslednjim pogojem:*

1. *Opis in parametri funkcije* h *so javno znani, tako da za izračun* h *ne potrebujemo nobene skrivne informacije.*

2. *Funkcija* h *je lahko izračunljiva, kar pomeni, da iz znanega* x *hitro in brez težav dobimo* $h(x)$*.*

3. *Iz poznanega* y *težko najdemo sporočilo* x*, da velja* $h(x) = y$*. Z drugimi besedami to pomeni, da je iskanje inverza od* h *računsko zahtevno. Ta problem imenujemo tudi problem iskanja praslike (ang. preimage).*

4. *Iz poznanih* x *in* $h(x)$ *težko najdemo nek* $x' \neq x$*, da velja* $h(x') = h(x)$*. Ta problem imenujemo tudi problem iskanja druge praslike (ang. second preimage).*

Definicija 2 *Funkcija* $h : \{0,1\}^* \longrightarrow \{0,1\}^n$*, ki preslika sporočilo* x *poljubne dolžine v niz* y *fiksne dolžine* n*, je varna pred trki (ang. Collision Resistant Hash Function ali krajše CRHF), če zadošča vsem pogojem iz definicije 1 in pogoju:*

1. *Težko je najti dve različni sporočili x in x', da velja $h(x) = h(x')$.*

Definicija 3 *Funkcija $h : \{0,1\}^m \times \{0,1\}^* \longrightarrow \{0,1\}^n$, ki preslika ključ K fiksne dolžine m in sporočilo x poljubne dolžine v niz y fiksne dolžine n, se imenuje zgoščevalna funkcija s ključem ali tudi MAC (ang. Message Authentication Code), če zadošča naslednjim pogojem:*

1. *Opis in parametri funkcije h so javno znani, tako da za izračun h ne potrebujemo nobene skrivne informacije.*

2. *Iz poznanega K in x je lahko izračunati $h(K, x)$.*

3. *Če ne poznamo ključa K, iz znanega y težko najdemo x, da velja $h(K, x) = y$. Prav tako težko najdemo dve različni sporočili x in x', tako da bo $h(K, x) = h(K, x')$.*

4. *Če poznamo mnogo parov (x_i, y_i), kjer je $y_i = h(K, x_i)$, težko najdemo skrivni ključ K.*

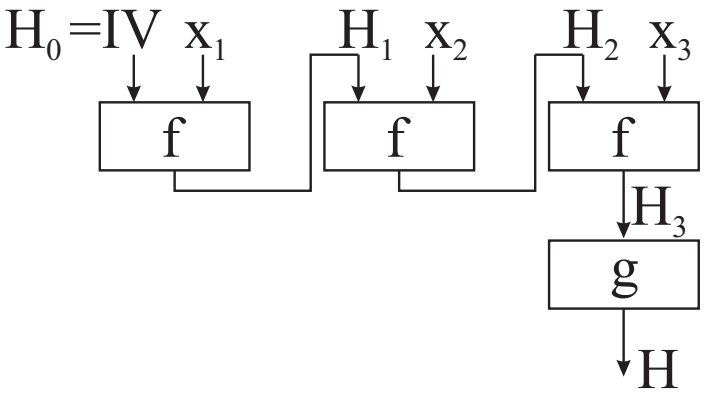
5. *Če ne poznamo ključa K, težko določimo $h(K, x)$ za poljubno sporočilo x, tudi če poznamo mnogo parov $(x_i, h(K, x_i))$, in seveda $x \neq x_i$.*

3 Tipi in uporaba zgoščevalnih funkcij

V splošnem zgoščevalne funkcije konstruiramo s pomočjo kompresijskih funkcij $f : \{0,1\}^k \longrightarrow \{0,1\}^l, \quad l < k$, ki niz fiksne dolžine preslikajo v drug krajši niz fiksne dolžine. Vrednost $h(x)$ dobimo z iteracijo tako, da sporočilo x razbijemo na bloke x_i in potem na vsakem bloku uporabimo funckijo f, pri čemer upoštevamo tudi rezultat prejšnje iteracije. Takim zgoščevalnim funkcijam pravimo iterirane zgoščevalne funkcije. Poenostavljeno shemo kaže slika 2. Načinov, kako dobimo uporabno kompresijsko funkcijo, je veliko, omenili bomo le nekaj najbolj pogostih [2, 4].

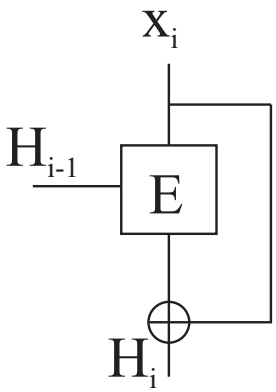
3.1 Zgoščevalne funkcije izpeljane iz bločnih šifer

Če se hočemo izogniti konstrukciji zgoščevalnih funkcij popolnoma od začetka, lahko za osnovo uporabimo že obstoječe bločne šifre. Kompresijsko funkcijo f lahko sestavimo iz poljubne enkripcijske funkcije $E_K(M)$. Preprost primer je na sliki 3, v praksi pa se uporabljajo bolj zapletene sheme [2, 1]. Na žalost velja, da obstaja relativno malo uporabnih zgoščevalnih funkcij izpeljanih iz bločnih šifer. Slabosti so namreč počasnost, ker so enkripcijski algoritmi dokaj zapleteni, in pa šibke točke takih konstrukcij, ki s samo enkripcijo nimajo zveze ter jih lahko izkoristi napadalec.



Slika 2:

Iterirana zgoščevalna funkcija. Sporočilo x razbijemo na bloke x_i fiksne dolžine. Začetna vrednost (ang. Initial Value ali IV) je poljuben in javno znan bitni niz, funkcija g pa pomeni poljubno izhodno transformacijo, ki je dodana zaradi večje varnosti. Rezultat zgoščevalne funkcije je kar izhod H.



Slika 3:

Uporaba bločnih šifer pri konstrukciji zgoščevalnih funkcij. Enkripcijsko funkcijo E uporabimo kot kompresijsko funkcijo, rezultat prejšnje iteracije H_{i-1} pa služi kot ključ.

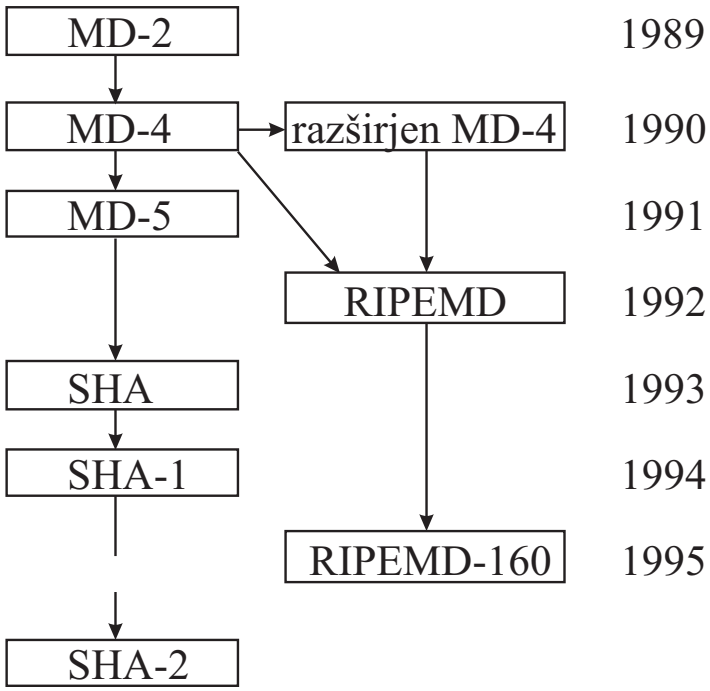
3.2 Zgoščevalne funkcije temelječe na modularni aritmetiki

Take zgoščevalne funkcije izkoriščajo dejstvo, da je v teoriji števil težko rešiti problema faktorizacije in diskretnega logaritma. Dva napogostejša kriptosistema, ki temeljita na modularni aritmetiki, sta RSA kriptosistem z javnimi ključi in

kriptosistem ElGamal [4]. Prednosti zgoščevalnih funkcij, ki uporabljajo modularno aritmetiko, so v tem, da včasih lahko dokažemo prevedbo problema iskanja trka na problem faktorizacije ali diskretnega logaritma, slabosti pa v počasnosti (potenciranje velikih števil) in možnosti izkoriščanja matematične strukture za napade. Najbolj znani zgoščevalni funkciji tega tipa sta MASH-1 in MASH-2 (ang. Modular Arithmetic Secure Hash) [1].

3.3 Druge zgoščevalne funkcije

V praksi se najbolj uporabljajo zgoščevalne funkcije, ki ne temeljijo na kakšnih posebnih že znanih konceptih, ampak na pazljivo konstruiranih kompresijskih funkcijah. Najpogostejša je družina zgoščevalnih funkcij MD (ang. Message Digest), ki je sestavljena, kot kaže slika 4. To družino sestavljajo funkcije MD-2, MD-4, MD-5, SHA, SHA-1 (ang. Secure Hash Algorithm) ter RIPEMD in RIPEMD-160 [2]. Pri takih funkcijah je težko matematično dokazati njihovo varnost pred napadi, kar pa še zdaleč ne pomeni, da niso uporabne. Za MD-2 velja, da je bila že razbita in je v nadaljevanju ne bomo več omenjali, ker se ne uporablja več. Nekaj lastnosti družine MD je zbranih v tabeli 1.



Slika 4:

Družina zgoščevalnih funkcij MD-4. V desnem stolpcu so letnice nastanka teh funkcij.

algoritem	n	št. krogov
MD-4	128	3
razš. MD-4	256	2×3
MD-5	128	4
SHA in SHA-1	160	4
RIPEMD	128	2×3
RIPEMD-160	160	2×5

Tabela 1:

Lastnosti zgoščevalnih funkcij iz družine MD-4. V prvem stolpcu je dolžina rezultata funkcije, v drugem pa število krogov, ki se uporabijo pri izračunu.

3.4 Uporaba zgoščevalnih funkcij

Zgoščevalne funkcije v kriptografske namene uporabljamo na mnogih področjih, največ za zagotavljanje integritete podatkov in avtentikacijo sporočil ter pri digitalnih podpisih. Pri zagotavljanju integritete podatkov je rezultat zgoščevalne funkcije varno shranjen in zaščiten pred spreminjanjem, kadarkoli pa lahko preverimo, če so bili podatki spremenjeni, tako da izračunemo zgoščevalno funkcijo ter rezultat primerjamo s tistim v bazi. Eno področje uporabe je tudi zagotavljanje redundance podatkov, kjer rezultat zgoščevalne funkcije (kontrolna vsota ali ang. Checksum) pripnemo na konec sporočila ter tako lahko odkrivamo morebitne napake pri prenosu (ang. Manipulation Detection Codes ali krajše MDC) [1]. Za avtentikacijo sporočil in digitalne podpise uporabljamo zgoščevalne funkcije s ključem (MAC) [4]. Uporabljamo jih še za zaščito gesel pri prenosu preko mreže (prenašamo samo rezultat zgoščevalne funkcije, ne pa celega gesla), generiranju naključnih binarnih nizov (kot so na primer kriptografski ključi) ter konstrukciji novih kriptografskih sistemov.

4 Varnost zgoščevalnih funkcij

Za zgoščevalno funkcijo h velja, da je varna, kadar so naslednji problemi težko rešljivi [2]:

1. Za dan rezultat zgoščevalne funkcije y najdi tak x, da bo $y = h(x)$. To je problem iskanja praslike in če želimo, da bo iz samega rezultata težko ugotoviti, kakšno je sporočilo, ki da tak rezultat, potem je očitno, da mora

biti ta problem težko rešljiv.

2. Za dano sporočilo x in rezultat zgoščevalne funkcije $y = h(x)$ najdi tak x', da bo $y = h(x) = h(x')$. To je problem iskanja druge praslike in če hočemo potencialnemu napadalcu preprečiti podtikanje sporočil, ki imajo enak rezultat kot originalno sporočilo, potem moramo zahtevati, da je ta problem težko rešljiv.

3. Najdi par sporočil x in x', da bo $h(x) = h(x')$. To je problem iskanja trčenj in očitno je med vsemi tremi naštetimi problemi najlažji, saj iščemo katerikoli par sporočil, ki ima enak rezultat zgoščevalne funkcije.

Dokazati je možno, da lahko problem iskanja trčenj prevedemo na problem iskanja praslike ali problem iskanja druge praslike [4]. To pomeni, da, če obstaja učinkovit algoritem za iskanje praslike oz. druge praslike, potem obstaja tudi algoritem za iskanje trčenj, ki je istega reda računske zahtevnosti. Od tod sledi, da je zgoščevalna funkcija, ki je varna pred trki, varna tudi pred iskanjem praslike in druge praslike. To lahko z drugimi besedami povemo tudi takole: razred CRHF je podmnožica razreda OWHF. Omeniti velja, da pri vseh možnih načinih uporabe v praksi ni vedno potrebno, da je zgoščevalna funkcija varna pred trčenji, vedno pa seveda zahtevamo, da je enosmerna. Kljub temu se v splošnem napadi skoraj vedno osredotočajo na problem iskanja trkov in zgoščevalna funkcija, ki ni varna pred trki, se šteje za razbito.
Nekaj prostora velja posvetiti še besedni zvezi "težko rešljiv". Problem (iskanja praslike, druge praslike ali trkov) je težko rešljiv, če ne obstaja napad, ki bi porabil bistveno manj časa, kot napad z grobo silo (ang. brute force attack). Zahtevnost napada z grobo silo je odvisna samo od dolžine rezultata zgoščevalne funkcije (in seveda opreme, ki je na voljo), prav nič pa od samega algoritma funkcije.

5 Napadi na zgoščevalne funkcije

5.1 Napadi z grobo silo

Ogledali si bomo dve vrsti napada, in sicer iskanje druge praslike ter iskanje trkov:

1. Pri iskanju druge praslike napad z grobo silo pomeni enostavno preskušanje velike množice sporočil in primerjanje rezultata s podano vrednostjo. Če je n dolžina rezultata zgoščevalne funkcije, c vrednost investicije v opremo v dolarjih, k število poskusov in t število sporočil, ki jih obdelujemo naenkrat, potem je približna empirična formula za verjetnost, da najdemo drugo prasliko [1]:

$$p = \frac{c \cdot k \cdot t}{2^n}.$$

   V praksi je za $n > 90$ ta napad neizvedljiv v nekem realnem času.

2. Pri iskanju trkov napad z grobo silo pomeni uporabo paradoksa rojstnih dnevov. Naključno izberemo r sporočil in na njih uporabimo zgoščevalno funkcijo. Če je n dolžina rezultata in p verjetnost, da najdemo trk, potem velja [1]:

$$p > 50\% \Longrightarrow r \approx 2^{n/2}.$$

Te vrste napad je v praksi neizvedljiv če je $n > 160$.

Tabela 2 kaže dolžino rezultata n, da je zgoščevalna funkcija še varna pred napadom z grobo silo v času enega leta. Podana je tudi projekcija v leto 2015.

napadalec	investicija	Rezultat hash funkcije			
		2nd preimage		trki	
		2000	2015	2000	2015
osamljeni hacker	$400	70	80	107	127
majhno podjetje	$10K	75	85	117	137
oddelek korporacije	$300K	86	96	139	159
veliko podjetje	$10M	91	101	149	169
obveščevalna agencija	$300M	96	106	154	174

Tabela 2:

Dolžina rezultata zgoščevalne funkcije n, ki še zagotavlja varnost pred napadom z grobo silo v času enega leta, v odvisnosti od investicije [1]. Pri iskanju druge praslike je število sporočil, ki jih obdelujemo naenkrat, enako 65536.

5.2 Drugi napadi na zgoščevalne funkcije

Podali bomo le kratko zgodovino napadov na družino MD-4 [1]:

- Merkle je leta 1990 našel trke pri prvih dveh krogih MD-4.
- den Boer in Bosselaers sta leta 1991 našla trke pri drugem in tretjem krogu MD-4.
- den Boer in Bosselaers sta leta 1993 našla psevdo-trke pri MD-5. Psevdo-trki pravzaprav niso pravi trki, saj se začetni vrednosti zgoščevalne funkcije razlikujeta.
- Vaudenay je leta 1994 našel trke pri prvih dveh krogih MD-4 in skorajšnje trke pri MD-5. Skorajšnji trk je trk, kjer se rezultata zgoščevalne funkcije razlikujeta samo v enem bitu.

- NSA je leta 1994 našla napad na SHA, ki pa ni bil nikoli razkrit.
- Dobbertin je v letih 1995 do 1997 našel:
  - trke pri drugem in tretjem krogu RIPEMD ter pri prvem in drugem krogu RIPEMD.
  - trke pri MD-4.
  - trke pri razširjeni MD-4 kompresijski funkciji z naključno začetno vrednostjo.
  - trke pri MD-5 kompresijski funkciji z naključno začetno vrednostjo.
  - prasliko pri prvih dveh krogih MD-4.
- Chabaud in Joux sta leta 1998 našla trke pri SHA za $r = 2^{61}$, kar je skoraj milijonkrat manj, kot napad z grobo silo ($r = 2^{80}$).

5.3 Povzetek napadov na družino MD-4

algoritem	trki	
	hash h	compress f
MD-4	1s – 10h	<1s
razš. MD-4		1h
MD-5		10h
RIPEMD		meseci
SHA	< groba sila	

Tabela 3:

Čas potreben za iskanje trkov pri zgoščevalnih funkcijah iz družine MD-4 oziroma njihovih kompresijskih funkcijah.

Tabela 3 kaže čas, ki je potreben za iskanje trkov pri zgoščevalnih funkcijah iz družine MD. Ker v praksi velja, da zgoščevalna funkcija ni več varna, če pri njej ali pri njeni kompresijski funkciji najdemo trke, lahko sklepamo naslednje [3]:

1. Funkcije MD-2, MD-4, SHA in RIPEMD se štejejo za razbite in se ne uporabljajo več.
2. Funkcija MD-5 ne ustreza več zahtevam o varnosti, ker je njena kompresijska funkcija razbita. Uporaba v novih aplikacijah se odsvetuje. Zgoščevalna funkcija s ključem, ki uporablja MD-5 (HMAC), zaenkrat še ni ogrožena, saj

uporaba skrivnega ključa onemogoča uporabo istih algoritmov za razbitje [3].

6 Zaključek

Za zaključek naj omenimo nekaj odprtih vprašanj, ki so prisotna pri zgoščevalnih funkcijah v kriptografiji. Predvsem še vedno premalo vemo o varnosti zgoščevalnih funkcij ter o tem, kako konstruirati varne zgoščevalne funkcije. Osnovni problem je, kako najti kompresijsko funkcijo, ki bo varna pred trki, saj je to prvi potrebni pogoj, da bo tudi zgoščevalna funkcija varna pred trki in torej tudi pred drugimi napadi (iskanje praslike in druge praslike). Pojavlja se tudi problem pri varnosti iteriranih aplikacij, kjer lahko z nepravim načinom iteracije v sistem vnesemo šibke točke, ki jih napadalec lahko izkoristi. Vsekakor je absolutno varna zgoščevalna funkcija, ki bo tudi praktično uporabna, zaenkrat še stvar prihodnosti.

Literatura:

[1] Bart Preneel, The State of Cryptographic Hash Functions, 6. Spanish Meeting on Cryptology and Information Security, 2000 (http://www.esat.kuleuven.ac.be/~preneel/slides.html)

[2] S. Bakhtiari, R. Safavi-Naini, J. Pieprzyk, Cryptographic Hash Functions: A Survey, Technical Report 1995, University of Wollongong (http://citeseer.nj.nec.com/bakhtiari95cryptographic.html)

[3] Hans Dobbertin, The Status of MD-5 After a Recent Attack, CryptoBytes, 1996, vol. 2

[4] Douglas Stinson, Cryptography: Theory and Practice, CRC Press Inc., 2002