Univerza v Ljubljani
Fakulteta za računalništvo in informatiko

Študijski program: Podiplomski magistrski program Informacijski sistemi in odločanje

Predmet: Kriptografija in računalniška varnost

Predavatelj: prof. dr. Aleksandar Jurišić

Seminarska naloga

HMAC: Uporaba kriptografskih zgoščevalnih funkcij za avtentikacijo sporočil

Študent: Boštjan Oblak (bostjanob@email.si)
Datum oddaje naloge: 2. september 2002

1. Kazalo

1. KAZALO 1
2. UVOD 2
3.1 MAC (MESSAGE AUTHENTICATION CODE) 2
3.2 KRIPTOGRAFSKE ZGOŠČEVALNE FUNKCIJE 3
3.3 VPELJAVA KLJUČA V ITERATIVNE KRIPTOGRAFSKE ZGOŠČEVALNE FUNKCIJE 4
4.1 VGNEZDENA KONSTRUKCIJA NMAC (NESTED MAC) 5
4.2 KONSTRUKCIJA HMAC 6
5.1 VARNOST NMAC 8
5.2 VARNOST HMAC 10
5.3 NAPADI NA NMAC/HMAC 11
6. UPORABA HMAC 12
7. ZAKLJUČEK 12
VIRI: 13

2. Uvod

Preverjanje celovitosti in avtentičnosti podatkov je v današnjem času v računalniških sistemih in omrežjih izrednega pomena. Namen preverjanja je zaščita pred spreminjanjem poslanih podatkov in overjanje pošiljatelja. Preverjanje je potrebno predvsem takrat, ko se podatki prenašajo po komunikacijskem kanalu, ki ni varen, je pa dostopen široki množici uporabnikov.

Zelo priljubljen način za ugotavljanje avtentičnosti podatkov v številnih internetnih protokolih in aplikacijah sloni na uporabi kriptografskih zgoščevalnih funkcij (npr. MD5 in SHA-1). Številne metode, ki za avtentikacijo uporabljajo kriptografske zgoščevalne funkcije, pa so zgrajene na "ad hoc" način, zato so tudi nedorečene v smislu varnosti. Da bi odpravili to pomanjkljivost, so Mihir Bellare, Ran Canetti in Hugo Krawczyk predlagali na kriptografskih zgoščevalnih funkcijah temelječo enostavno in učinkovito konstrukcijo HMAC (Keyed-Hashing for Message Authentication), za katero je mogoče pokazati, da je z izbiro kriptografsko primerne iterativne zgoščevalne funkcije tudi varna. Še preden pa se spustimo v podrobno razlago konstrukcije HMAC in njene varnosti, si bližje poglejmo njena osnovna gradnika – avtentikacijsko shemo MAC in kriptografske zgoščevalne funkcije.

3.1 MAC (Message authentication code)

Avtentikacijska shema MAC je široko uporabljena metoda za ugotavljanje avtentičnosti sporočil. Njegova uporaba pošiljatelju in prejemniku, ki si izmenjujeta sporočila preko komunikacijskega kanala, omogoča preveriti, če se sporočilo na poti od pošiljatelja k prejemniku ni spremenilo (celovitost podatkov) in če je sporočilo dejansko poslal pošiljatelj (overjanje pošiljatelja).

Postopek MAC zahteva uporabo skrivnega ključa K, ki si ga delita osebi, ki si želita izmenjavati sporočila. Na podlagi skritega ključa K in sporočila M pošiljatelj S izračuna avtentikacijsko kodo $MAC_K(M)$ (imenovano tudi odtis) sporočila, ki se odpošlje skupaj s sporočilom. Ob sprejemu sporočila M' prejemnik R izračuna njegovo avtentikacijsko kodo $MAC_K(M')$, ki se primerja s prejeto $MAC_K(M)$. Če predpostavimo, da nihče razen pošiljatelja in prejemnika ne pozna skrivnega ključa, potem ujemanje izračunane in prejete avtentikacijska koda prejemniku zagotavlja, da se sporočilo ni spremenilo in da je sporočilo lahko poslal le pošiljatelj. Pravzaprav so napadalcu, ki prisluškuje na komunikacijski poti, znani izmenjani pari sporočil M_1, M_2, … in njim ustreznih avtentikacijskih kod $MAC_K(M_1)$, $MAC_K(M_2)$, … med pošiljateljem in prejemnikom, tako da bi napadalec lahko izmenjani par zamenjal z že znanim parom, vendar se temu lahko enostavno izognemo npr. tako, da vsakemu sporočilu na začetek dodamo enolično označbo. Da bi napadalec podtaknil sporočilo, ki še ni bilo izmenjano, pa mora ugotoviti tudi avtentikacijsko kodo tega sporočila oziroma drugače povedano razbiti mora MAC.

Razbitje avtentikacijske sheme MAC je mogoče s tem, da najdemo skrivni ključ ali pravilno avtentikacijsko kodo za neznano sporočilo na podlagi že poznanih parov sporočil, ki si jih lahko napadalec tudi sam izbere, in njihovih avtentikacijskih kod. Pravimo tudi, da je avtentikacijska shema MAC (ε,t,q,L)-varna, če se napadalec, ki ne pozna skrivnega ključa K, pozna pa avtentikacijske kode $MAC_K(M_1)$, $MAC_K(M_2)$, …, $MAC_K(M_q)$ za največ q sporočil M_1, M_2, …, M_q po njegovi izbiri, pri čemer so dolžine sporočil manjše ali enake od L, v času t ne more dokopati do pravilne avtentikacijske kode $MAC_K(M)$ za poljubno sporočilo $M \notin \{M_1, M_2, ..., M_q\}$ in s tem razbiti avtentikacijsko shemo MAC, razen z verjetnostjo boljšo ali enako kot ε. Verjetnost za razbitje bi morala biti v primeru "varne" MAC zanemarljivo majhna ali

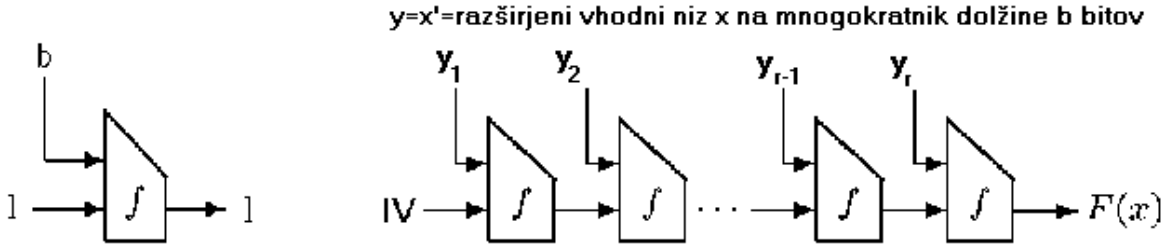
pa bi za razbitje potrebovali ogromno časa ali ogromno število parov sporočil z avtentikacijskimi kodami.

Za izračun avtentikacijskih kod so se običajno uporabljale bločne šifre kot npr. DES v CBC načinu. Algoritem DES "priredimo" za MAC tako, da enkripiramo celotno sporočilo in nato uporabimo zadnjih nekaj bitov enkripiranega sporočila za avtentikacijsko kodo. Alternativna in v zadnjem času tudi vedno bolj priljubljena možnost za MAC so kriptografske zgoščevalne funkcije kot npr. MD5, SHA-1 ali RIPEMD-160. Razlogi za uporabo kriptografskih zgoščevalnih funkcij za MAC ležijo v višji hitrosti kriptografskih zgoščevalnih funkcij napram bločnim šifram, pa tudi programske knjižnice za kriptografske zgoščevalne funkcije so široko dostopne in brez omejitev v smislu izvoza, patentov in licenc. Vendar pa kriptografske zgoščevalne funkcije ne temeljijo na skritem ključu, zato jih je potrebno za uporabo v avtentikacijski shemi MAC ustrezno prirediti.

3.2 Kriptografske zgoščevalne funkcije

Že iz samega imena je razvidno, da kriptografske zgoščevalne funkcije zgostijo vhodne nize različnih dolžin v niz z vnaprej določeno krajšo dolžino. Zgostitev je takšna, da je težko najti dva vhodna niza x in x', za katera sta zgostitvi $F(x)$ in $F(x')$ z zgoščevalno funkcijo F enaki. Tej lastnosti kriptografskih zgoščevalnih funkcij pravimo odpornost na trke. K tej lastnosti do določene mere pripomore tudi lastnost "naključnosti" kriptografskih zgoščevalnih funkcij, ki se odraža v mešanju bitov vhodnega niza, neodvisnosti zgostitve od vhodnega niza, nepredvidljivosti zgostitve, če je znan samo del vhodnega niza, itd. Kriptografske zgoščevalne funkcije se predvsem uporabljajo v kombinaciji z digitalnimi podpisi in iz tega razloga tudi ne vključujejo skritega ključa. Druga področja uporabe vključujejo generatorje naključnih števil, bločne šifre in avtentikacijske sheme MAC.

Za večino v praksi uporabljenih kriptografskih zgoščevalnih funkcij (npr. MD5, RIPEMD-160, SHA-1, …) se uporablja iterativna konstrukcija. Osnova iterativne konstrukcije je zgoščevalna funkciji f, ki na vhodu sprejme en blok dolžine b (običajno 512) bitov vhodnega niza in nastavitveni niz dolžine l bitov, zgostitev pa je dolžine l (npr. 128 bitov pri MD5, 160 bitov pri SHA-1 in RIPEMD-160) bitov (slika 3.1 levo). Zgostitev in mešanje bitov vhodnega niza se pri zgoščevalni funkciji f opravi s kombinacijo bitnih pomikov, seštevanj in odštevanj ter operacij konjunkcije in (ekskluzivne) disjunkcije, negacije, itd.



slika 3.1: zgoščevalna funkcija f (levo) in iterativna konstrukcija zgoščevalne funkcije F s povezavo zgoščevalnih funkcij f v verigo (desno)

Iterativna konstrukcija zahteva, da vhodni niz x najprej razširimo na mnogokratnik dolžine b bitov. Če želimo, da bo iterativna zgoščevalna funkcija F odporna na trke, mora biti razširitev vhodnega niza injektivna funkcija; torej morata biti različna vhodna niza tudi po razširitvi različna. V razširitvi je tipično vključena binarna predstavitev $|x|$ dolžine vhodnega

niza pred razširitvijo, pred katero dodamo še zadostno število (običajno) ničelnih bitov, tako da je skupna dolžina deljiva z *b*. Možna razširitev vhodnega niza *x*, ki se uporablja v praksi za mnogo iterativnih kriptografskih zgoščevalnih funkcij (npr. MD5, SHA-1, RIPEMD-160,…), je podana z enačbo 3.1.a.

$$\begin{aligned} &\textit{pogoj: } |x|<2^{64} \\ &d=(447-|x|) \bmod 512 \\ &l=\text{"binarna predstavitev } |x|\text{; pri tem je } |l|=64\text{"} \\ &y=x'=x \,||\, 1 \,||\, 0^d \,||\, l \text{ ; kjer z operacijo } || \text{ označujemo stik nizov} \end{aligned} \qquad (3.1.a)$$

Razširjeni vhodni niz $y=x'$ razbijemo na podnize $y_1, y_2,..., y_r$ dolžine b bitov. Zgoščevalne funkcije f v iterativni konstrukciji povežemo v verigo (slika 3.1 desno), tako da je nastavitveni niz zgoščevalne funkcije f v koraku i pravzaprav rezultat zgostitve bloka y_{i-1} v koraku $i-1$ z zgoščevalno funkcijo f. V prvem koraku za nastavitveni niz uporabimo vhodni vektor IV, ki je vnaprej določen in javen. Rezultat zgostitve $h_r=f(h_{r-1},y_i)$ zgoščevalne funkcije f v zadnjem koraku r je hkrati tudi zgostitev $F(x)=h_i$ vhodnega niza x. Opisani postopek prikazuje tudi spodnja enačba (*3.1.b*).

$$\begin{aligned} h_0 &= IV \\ h_1 &= f(h_0,y_1) \\ h_2 &= f(h_0,y_2) \\ &\ldots \\ &\ldots \\ &\ldots \\ h_r &= f(h_{r-1},y_r) \\ F(x) &= h_r \end{aligned} \qquad (3.1.b)$$

Razlog za uporabo iterativne konstrukcije izhaja iz dejstva, da v kolikor je osnovna zgoščevalna funkcija f odporna na trke, potem je tudi opisana iterativna konstrukcija osnovnih zgoščevalnih funkcij f, to je iterativna zgoščevalna funkcija F, odporna na trke. S tem problem iskanja zgoščevalne funkcije odporne na trke za vhodne nize različnih dolžin prevedemo na lažji problem iskanja zgoščevalne funkcije odporne na trke za vhodne nize vnaprej določenih krajših dolžin.

3.3 Vpeljava ključa v iterativne kriptografske zgoščevalne funkcije

Avtentikacijska shema MAC zahteva uporabo skrivnega ključa, same iterativne kriptografske zgoščevalne funkcije F pa običajno skrivnih ključev ne uporabljajo. Iterativne zgoščevalne funkcije lahko prilagodimo za uporabo skrivnega ključa na več načinov. S prilagoditvijo dobimo iterativne zgoščevalne funkcije s ključem F_K in sicer je najenostavneje ključ dodati na začetek in/ali konec vhodnega niza (sporočila) x:

- ključ K dodamo pred sporočilo x: $F_K(x)=F(K \,||\, x)$
- ključ K dodamo na konec sporočila x: $F_K(x)=F(x \,||\, K)$
- ključ K_1 dodamo na začetek, ključ K_2 na konec sporočila x: $F_{(K_1,K_2)}(x)=F(K_1||x||K_2)$

Druga možnost je, da namesto javnega in vnaprej določenega vhodnega vektorja IV iterativne kriptografske zgoščevalne funkcije uporabimo skrivni ključ K, kar pomeni, da v enačbi *3.1.b*

korak $h_0 = IV$ zamenjamo s $h_0 = K$ in tako v zadnjem koraku namesto $F(x)$ dobimo $F_K(x)$. Nenazadnje je pri vpeljavi ključa možna tudi kombinacija dodajanja enega ključa K_2 v sporočilo in zamenjave vhodnega vektorja z drugim ključem K_1, npr. $F_{(K_1,K_2)} = F_{K_1}(x \,||\, K_2)$.

Zaradi enostavnejše analize predvsem varnosti je najbolj smiselno vpeljati ključ v vhodni vektor iterativne zgoščevalne funkcije. Poleg tega lahko iterativno zgoščevalno funkcijo, ki namesto javnega vhodnega vektorja IV uporablja skrivni ključ $K_1=f(K_2)$, "simuliramo" z iterativno zgoščevalno funkcijo s ključem K_2, ki je dodan pred sporočilo in ima dolžino b bitov, če osnovna zgoščevalna funkcija f uporabljena v iterativni konstrukciji zgosti nize dolžine b bitov. Po drugi strani pa zamenjava javnega vhodnega vektorja s skrivnim ključem prinese s seboj potrebo po spremembi programske kode iterativne zgoščevalne funkcije.

S tem ko vpeljemo ključ v vhodni vektor iterativne zgoščevalne funkcije, lahko definiramo družino funkcij $\{F_K\}$ iterativne zgoščevalne funkcije F na množici vseh možnih ključev $K \in \{0,1\}^l$ dolžine l, kjer je l dolžina zgostitve osnovne zgoščevalne funkcije f uporabljene v iterativni konstrukciji. Eden izmed elementov družine funkcij $\{F_K\}$ je tudi funkcija $F_{K=IV}$, torej običajna iterativna zgoščevalna funkcija, ki uporablja javen in vnaprej določen vhodni vektor IV.

Omenili smo že, da od kriptografskih zgoščevalnih funkcij zahtevamo odpornost na trke, kar velja tudi za iterativne zgoščevalne funkcije s ključi. Pravimo, da je družina funkcij $\{F_K\}$ iterativne zgoščevalne funkcije F na množici ključev $K \in \{0,1\}^l$ dolžine l (ε,t,q,L)-šibka brez trčenj, če verjetnost, da napadalec, ki ne pozna skrivnega ključa K, pozna pa zgostitve $F_K(M_1)$, $F_K(M_2)$, ..., $F_K(M_q)$ za največ q sporočil (vhodnih nizov) M_1, M_2, ..., M_q po njegovi izbiri, pri čemer so dolžine sporočil manjše ali enake od L, v času t najde dve različni sporočili M in M' z enako zgostitvijo $F_K(M)=F_K(M')$, ni večja od ε. Zaradi zamenjave vhodnega vektorja s skrivnim ključem ima napadalec pri iskanju enakih zgostitev pred seboj težjo nalogo kot v primeru, ko je vhodni vektor IV javen in vnaprej določen, saj si parov sporočilo in ustreznih zgostitev teh sporočil ne more izračunati sam. Iz tega razloga je tudi lažje doseči, da postane verjetnost enake zgostitve za različni sporočili zanemarljivo majhna, oziroma da postane potrebni čas in/ali število potrebnih sporočil z ustreznimi zgostitvami neobvladljiv(o).

4.1 Vgnezdena konstrukcija NMAC (Nested MAC)

Osnova avtentikacijske sheme NMAC je iterativna kriptografska zgoščevalna funkcija F_K z zgostitvijo dolžine l bitov in dolžino bloka b bitov v iterativni konstrukciji, ki namesto vhodnega vektorja IV uporablja skrivni ključ dolžine l bitov. Iterativna zgoščevalna funkcija F_K je v konstrukciji NMAC uporabljena dvakrat, enkrat kot zunanja s ključem K_1 (F_{K_1}) in drugič kot notranja vgnezdena funkcija s ključem K_2 (F_{K_2}). In sicer vhodni niz x najprej zgostimo z notranjo iterativno kriptografsko zgoščevalno funkcijo F_{K_2} in tako dobimo zgostitev $F_{K_2}(x)$ dolžine l bitov, ki je vhodni niz - ki pa ga je še potrebno razširiti na dolžino bloka b bitov - za zunanjo iterativno kriptografsko zgoščevalno funkcijo F_{K_1}, rezultat zunanje iterativne kriptografske zgoščevalne funkcije F_{K_1}, pa je avtentikacijska koda $NMAC_{(K_1,K_2)}(x)$ dolžine l bitov za vhodni niz x. Opisani postopek je strnjen v spodnji enačbi (*4.1*).

$$NMAC_{(K_1,K_2)}(x) = F_{K_1}(F_{K_2}(x)) \qquad (4.1)$$

Konstrukcija NMAC je navkljub svoji enostavnosti tudi zelo učinkovita. Časovna zahtevnost notranje iterativne kriptografske zgoščevalne funkcije F_K je pri istem vhodnem nizu enaka kot časovna zahtevnost same iterativne kriptografske zgoščevalne funkcije F, ki ne uporablja ključa, zunanja iterativna kriptografska zgoščevalna funkcija F_K pa zahteva samo eno iteracijo v iterativni konstrukciji in če zanemarimo čas potreben za razširitev vhodnega niza $F_{K_2}(x)$, je njena časovna zahtevnost enaka zahtevnosti osnovne kriptografske zgoščevalne funkcije f_K. Zunanjo iterativno kriptografsko zgoščevalno funkcijo F_K s ključem K_1 zato v primeru, ko je razširitev l bitnega niza dolžine b bitov, lahko enačimo z osnovno kriptografsko zgoščevalno funkcija f_K s ključem K_1, če za obe funkciji razširitev definiramo na enak način. Ključa K_1 in K_2 uporabljena v konstrukciji NMAC imata dolžino l bitov, z vidika varnosti pa je poleg njune dolžine, ki pa jo določa uporabljena iterativna kriptografska zgoščevalna funkcija F, pomembno tudi, da sta med seboj neodvisna in naključna.

NMAC terja zaradi zamenjave vhodnega vektorja IV s skrivnim ključem v iterativni kriptografski zgoščevalni funkciji poseg v njeno programsko kodo. Iz tega razloga se v praksi kot prilagoditev NMAC uporablja HMAC z iterativno kriptografsko zgoščevalno funkcijo z javnim in vnaprej določenim vhodnim vektorjem IV. Po drugi strani se sama NMAC zaradi lažje analize v primerjavi s HMAC uporablja v teoriji; teoretične izsledke, ki se npr. nanašajo na varnost, pa lahko v večini primerov prenesemo tudi na HMAC.

4.2 Konstrukcija HMAC

Konstrukcija HMAC sloni na iterativni kriptografski zgoščevalni funkciji F z zgostitvijo dolžine l bitov, dolžino bloka b bitov v iterativni konstrukciji in z javnim ter vnaprej določenim vhodnim vektorjem IV, medtem ko se skrivni ključ potreben za avtentikacijo doda v vhodni niz (sporočilo). Za razliko od NMAC uporablja HMAC samo en ključ, kar poenostavi rokovanje s ključi. Ključ je lahko poljubne dolžine, vendar pa se ključi daljši od b bitov pred uporabo zgostijo. Konstrukcija HMAC predvideva za dobljeno avtentikacijsko kodo z dolžino l bitov tudi njeno skrajšanje na njenih prvih t bitov. V celoti gledano določimo avtentikacijsko kodo $HMAC_K(x)_t$ dolžine t bitov, kjer je število t deljivo z 8, na podlagi vhodnega niza x, ključa K, katerega razširitev na b bitov označimo s K^+ in iterativne zgoščevalne funkcije F z zgostitvijo dolžine l bitov ter dolžino bloka b bitov, kjer sta števili l in b deljivi z 8, s spodnjo enačbo (*4.2*):

$$NMAC_{(K_1=f(K^+ XOR\ opad),\ K_2=f(K^+ XOR\ ipad))}(x) = HMAC_K(x)_l$$

$$HMAC_K(x)_t = F(K^+ XOR\ opad \,||\, F(K^+ XOR\ ipad \,||\, x))_t \qquad (4.2)$$

$$b \bmod 8 = l \bmod 8 = t \bmod 8 = 0 \quad ipad = 0x\underbrace{36 \quad \ldots \quad 36}_{\frac{b}{8}} \quad opad = 0x\underbrace{5c \quad \ldots \quad 5c}_{\frac{b}{8}}$$

Posamezni koraki algoritma HMAC pa so tako naslednji:

0. Niz *ipad* (*opad*) je bajt *0x36* (*0x5c*) ponovljen tolikokrat, da dobimo blok dolžine *b* bitov oziroma *b/8* bajtov.
1. Ključ *K* daljši od *b* bitov zgostimo z iterativno zgoščevalno funkcijo *F* in s tem dobimo nov ključ dolžine *l* bitov: $K=F(K)$.
2. Ključu *K* krajšemu od *b* bitov - kar je lahko posledica zgostitve na *l* bitov v koraku 1 - dodamo na konec toliko ničelnih bitov, da ima razširjeni ključ K^+ dolžino *b* bitov (npr. ključu *K* z dolžino 20 bajtov (160 bitov) dodamo na konec 44 ničelnih bajtov (*0x00*), če je dolžina bloka 64 bajtov (512 bitov)). Za ključ *K* z dolžino *b* bitov je "razširjeni" ključ K^+ kar enak ključu *K*.
3. Operacija ekskluzivne disjunkcije (XOR) med razširjenim ključem K^+ in *ipad*, s katero dobimo niz dolžine *b* bitov: $K^+ XOR ipad$.
4. Vhodni niz *x* dodamo na konec niza, ki je rezultat operacije ekskluzivne disjunkcije v koraku 3: $K^+ XOR ipad \,||\, x$.
5. Z iterativno zgoščevalno funkcijo *F* zgostimo niz, ki ga dobimo v koraku 4, in kot rezultat dobimo zgostitev dolžine *l* bitov: $F(K^+ XOR ipad \,||\, x)$.
6. Operacija ekskluzivne disjunkcije (XOR) med razširjenim ključem K^+ in *opad*, s katero dobimo niz dolžine *b* bitov: $K^+ XOR opad$.
7. Zgostitev, ki je rezultat koraka 5, dodamo na konec niza, ki je rezultat operacije ekskluzivne disjunkcije v koraku 6: $K^+ XOR opad \,||\, F(K^+ XOR ipad \,||\, x)$.
8. Z iterativno zgoščevalno funkcijo *F* zgostimo niz, ki ga dobimo v koraku 7, in kot rezultat dobimo zgostitev dolžine *l* bitov: $F(K^+ XOR opad \,||\, F(K^+ XOR ipad \,||\, x))$.
9. Avtentikacijsko kodo vhodnega niza *x* predstavlja prvih *t* levih bitov zgostitve dobljene v koraku 8: $HMAC_K(x)_t = F(K^+ XOR opad \,||\, F(K^+ XOR ipad \,||\, x))_t$.

Že v enačbi *4.2* je nakazano, da je HMAC, ki je opredeljena s ključem *K*, iterativno zgoščevalno funkcijo *F* z zgostitvijo dolžine *l* bitov ter avtentikacijsko kodo z dolžino *l* bitov, le posebna oblika avtentikacijske sheme NMAC z zgoščevalno funkcijo F_K s ključema $K_1=f(K^+ XOR opad)$ ter $K_2=f(K^+ XOR ipad)$, kjer je *f* osnovna zgoščevalna funkcija uporabljena v iterativni konstrukciji funkcije *F* ter tudi F_K. S tem ko za ključa K_1 oziroma K_2 vzamemo zgostitev bloka $K^+ XOR opad$ oziroma $K^+ XOR ipad$ z osnovno zgoščevalno funkcijo *f*, si zagotovimo zaradi lastnosti naključnosti zgoščevalne funkcije *f*, da sta tudi ključa naključna. Po drugi strani sta tudi bloka *ipad* in *opad* namenoma izbrana tako, da je med njima Hammingova razdalja velika, da sta rezultata (ključa K_1 in K_2) mešanja bitov $K^+ XOR opad$ in $K^+ XOR ipad$ z osnovno zgoščevalno funkcijo *f*, med seboj čim bolj neodvisna. HMAC ohranja tudi učinkovitost in enostavnost NMAC. Tako zahteva izračun avtentikacijske kode z algoritmom HMAC, če pri tem zanemarimo čas potreben za ekskluzivno disjunkcijo K^+ (razširitve ključa *K*) z blokom *ipad* in z blokom *opad* ter morebitno zgostitev predolgega ključa *K*, v primerjavi z izračunom le-te v NMAC konstrukciji dodatno samo dva koraka, ki ustrezata eni iteraciji v iterativni konstrukciji notranje funkcije F_K in eni iteraciji v iterativni konstrukciji zunanje funkcije F_K.

Avtorji avtentikacijske sheme HMAC so z opisano konstrukcijo sledili ciljem, ki naj bi pripomogli k uporabi kriptografskih zgoščevalnih funkcij (preko njihove avtentikacijske sheme) za MAC v praksi. Ti cilji so:

- uporaba zgoščevalnih funkcij v avtentikacijski shemi brez spreminjanja
- možnost enostavne nadomestitve v avtentikacijski shemi uporabljene iterativne zgoščevalne funkcije z novo, če se pojavi potreba po hitrejši oziroma varnejši iterativni zgoščevalni funkciji
- enostavno rokovanje s ključi
- hitrost izračuna avtentikacijske kode sporočila naj bi bila primerljiva s hitrostjo zgostitve sporočila z iterativno zgoščevalno funkcijo uporabljeno v avtentikacijski shemi
- možnost sklepanja o varnosti avtentikacije na podlagi uporabljene zgoščevalne funkcije

Da bi zadostili prvima dvema ciljema, obravnava HMAC uporabljeno (iterativno) zgoščevalno funkcijo kot črno škatlo, saj se pri implementaciji algoritma HMAC obstoječa programska koda uporabljene zgoščevalne funkcije brez kakršnega koli spreminjanja vključi kot poseben programski modul. Iz tega razloga je v avtentikacijski shemi HMAC mogoča tudi enostavna zamenjava določene kriptografske zgoščevalne funkcije z drugo hitrejšo oziroma varnejšo (npr. zamenjava MD5 z varnejšo SHA-1), saj je v algoritmu HMAC potrebno le nadomestiti programski modul za staro zgoščevalno funkcijo s programskim modulom za novo zgoščevalno funkcijo.

HMAC uporablja samo en ključ, kar pripomore k enostavnemu rokovanju s ključi. Tudi čas potreben za izračun avtentikacijske kode sporočila z algoritmom HMAC je za dolga sporočila praktično enak kot čas, ki bi bil potreben za zgostitev istega sporočila z iterativno zgoščevalno funkcijo F uporabljeno v algoritmu HMAC, saj algoritem HMAC pravzaprav doda samo še tri izvajanja osnovne zgoščevalne funkcije f. Pri krajših sporočilih je razlika bolj opazna, vendar si v primeru, ko za več krajših sporočil uporabimo enak ključ za izračun avtentikacijske kode, lahko izognemo dvema izvajanjema osnovne zgoščevalne funkcije f z vnaprejšnjim izračunom zgostitev $f(K^+ XOR\ opad)$ in $f(K^+ XOR\ ipad)$ ter temu ustrezno prilagoditvijo algoritma HMAC.

Uporaba avtentikacijske sheme je smiselna le, če je avtentikacija sporočil varna. Varnost HMAC je odvisna med drugim tudi od uporabljene kriptografske iterativne zgoščevalne funkcije. Kot bomo videli v naslednjem razdelku, je z izbiro kriptografsko primerne iterativne zgoščevalne funkcije tudi HMAC varna. Drugače povedano, če lahko pokažemo, da HMAC za izbrano iterativno zgoščevalno funkcijo ni varna, potem ta iterativna zgoščevalna funkcija ni primerna za uporabo ne samo v HMAC pač pa nasploh v kriptografiji.

5.1 Varnost NMAC

Varnost NMAC je v neposredni povezavi z (iterativno) zgoščevalno funkcijo f_K (F_K), ki je osnova konstrukcije NMAC. Velja, da je NMAC $(\varepsilon_F+\varepsilon_f,t,q,L)$-varna MAC, če je iterativna zgoščevalna funkcija F_K (ε_F,t,q,b)-šibka brez trčenj ter je iterativna zgoščevalna funkcija F_K za vhodna sporočila krajša od b bitov (ε_f,t,q,b)-varna MAC oziroma drugače povedano je (ε_f,t,q,b)-varna MAC osnovna zgoščevalna funkcija f_K z dolžino bloka b bitov. To pomeni, da pri danih omejitvah v smislu časa t, ki je na razpolago napadalcu, in števila izbranih sporočil q, za katere napadalec pozna avtentikacijsko kodo oziroma zgostitev,

verjetnost uspešnega napada na NMAC ni večja od vsote verjetnosti, da napadalec najde dve različni sporočili z enako zgostitvijo z iterativno zgoščevalno funkcijo F_K (ε_F) ali pa se dokoplje do pravilne avtentikacijske kode oziroma zgostitve z osnovno zgoščevalno funkcijo f_K za poljubno neizbrano sporočilo (ε_f).

Za dokaz trditve iz prejšnjega odstavka predpostavimo, da je verjetnost uspešnega napada na NMAC vsaj ε_N (torej je NMAC *(ε_N,t,q,L)*-varna MAC) in je iterativna zgoščevalna funkcija *F (ε_F,t,q,b)*-šibka brez trčenj. Nato pokažemo, da lahko napadalec, ne da bi poznal ključa K_1 in K_2 , samo na podlagi $f_{K_1}(F_{K_2}(M))$ in avtentikacijskih kod $NMAC_{(K_1,K_2)}(M_i)$ ter zgostitev $F_{K_2}(M_i)$ za q izbranih sporočil M_i za $1<=i<=q$ v času t z verjetnostjo $\varepsilon_f \geq \varepsilon_N - \varepsilon_F$ najde za poljubno sporočilo M pravilno avtentikacijsko kodo oziroma zgostitev z osnovno zgoščevalno funkcijo f_K s ključem K_1, kadar je $M \neq F_{K_2}(M_i)$ za $1<=i<=q$.

Napad izgleda tako, da si napadalec izbere poljuben ključ K_2 in q poljubnih sporočil M_i ($1<=i<=q$). Za ta sporočila so mu na voljo tako avtentikacijske kode $NMAC_{(K_1,K_2)}(M_i)$ kot tudi zgostitve $F_{K_2}(M_i)$. Napadalcu avtentikacijske kode $NMAC_{(K_1,K_2)}(M_i)$ služijo za izračun možne avtentikacijske kode $NMAC_{(K_1,K_2)}(M)$ za sporočilo $M \neq M_i$, kjer je $1<=i<=q$. $NMAC_{(K_1,K_2)}(M)$ je za sporočilo $F_{K_2}(M)$ hkrati tudi možna avtentikacijska koda oziroma zgostitev z osnovno zgoščevalno funkcijo f_K s ključem K_1. Napad na avtentikacijsko shemo, ki jo določa osnovna zgoščevalna funkcija f_K, se s tem, ko sporočilu $F_{K_2}(M)$ določimo možno avtentikacijsko kodo $NMAC_{(K_1,K_2)}(M)$, zaključi.

Ločimo dva primera, v katerih opisani napad ne uspe. Kadar avtentikacijska koda $NMAC_{(K_1,K_2)}(M)$ za sporočilo M ni pravilna, pride do neujemanje med dobljeno in pravilno avtentikacijsko kodo ($NMAC_{(K_1,K_2)}(M)$ in $f_{K_1}(F_{K_2}(M))$). Četudi pa se avtentikacijski kodi $NMAC_{(K_1,K_2)}(M)$ in $f_{K_1}(F_{K_2}(M))$ ujemata, je lahko $F_{K_2}(M)$ enaka $F_{K_2}(M_i)$ za $1<=i<=q$ in tudi v tem primeru je napad neuspešen, zato ker je bila avtentikacijska koda za sporočilo $F_{K_2}(M) = F_{K_2}(M_i)$ napadalcu že podana. Oba primera se med seboj izključujeta, zato je verjetnost $1-\varepsilon_f$, da je opisani napad neuspešen, navzgor omejena z vsoto verjetnosti obeh primerov. Verjetnost neujemanja je največ enaka verjetnosti $1-\varepsilon_N$ neuspešnega napada na NMAC. Verjetnost drugega primera je navzgor omejena z vrednostjo ε_F, saj je iterativna zgoščevalna funkcija F_K *(ε_F,t,q,b)*-šibka brez trčenj. Iz povedanega sledi, da je $1-\varepsilon_f \leq 1-\varepsilon_N+\varepsilon_F$ oziroma $\varepsilon_N - \varepsilon_F \leq \varepsilon_f$. Torej je $\varepsilon_N \leq \varepsilon_F + \varepsilon_f$ in v najslabšem primeru je $\varepsilon_N = \varepsilon_F + \varepsilon_f$, tako da je v tem primeru NMAC *($\varepsilon_F+\varepsilon_f$,t,q,L)*-varna MAC in s tem je trditev dokazana.

Napadalec, ki mu uspe razbiti NMAC, lahko v enakem času in z enakim številom poznanih zgostitev za izbrana sporočila "razbije" tudi v NMAC uporabljeno iterativno zgoščevalno funkcijo na enega izmed naslednjih načinov:

1. Napadalec najde dve različni sporočili z enako zgostitvijo, četudi je vhodni vektor *IV* iterativne zgoščevalne funkcije naključen in skrit.

2. Napadalec lahko predvidi zgostitev sporočila z osnovno zgoščevalno funkcijo, četudi je vhodni vektor *IV* osnovne zgoščevalne funkcije naključen, skrit in napadalcu nepoznan.

Spomnimo se, da od kriptografskih zgoščevalnih funkcij zahtevamo predvsem odpornost na trke. Če napadalcu uspe napad na prvi način, je za uporabljeno iterativno zgoščevalne funkcijo odpornost na trke postavljena pod vprašaj. To drži toliko bolj, ker je

vhodni vektor *IV* iterativno zgoščevalne funkcije zamenjan s skritim ključem in zaradi te zamenjave ima napadalec pri iskanju enakih zgostitev pred seboj težjo nalogo kot v primeru, ko je vhodni vektor *IV* javen in vnaprej določen, saj si parov sporočilo in ustreznih zgostitev teh sporočil ne more izračunati sam. S tem odpade tudi možnost, da bi napad na iterativno zgoščevalno funkcijo s paradoksom rojstnih dnevov, s katerim se običajno lotevamo iskanja enakih zgostitev za različni sporočili, paralelizirali. Zato je zaradi skrivnega ključa uporaba iterativne zgoščevalne funkcije lahko v NMAC varna, četudi je za isto funkcijo v primeru javnega in vnaprej določenega vhodnega vektorja *IV* njena odpornost na trke vprašljiva.

Napad na drugi način preverja lastnost "naključnosti" zgoščevalne funkcije, ki jo v določeni meri tudi pričakujemo od kriptografske zgoščevalne funkcije. Uspešen napad bodisi s požrešnim iskanjem skrivnega ključa bodisi z napadom s paradoksom rojstnih dnevov, kaže na predvidljivost zgostitve, čeprav je vhodni vektor *IV* skrit in nepoznan. To meče slabo luč na lastnost "naključnosti" (iterativne) zgoščevalne funkcije in tudi lastnost "odpornosti" na trke, h kateri naj bi pripomogla tudi "naključnost" (iterativne) zgoščevalne funkcije.

Razbitje NMAC tako pomeni, da v NMAC uporabljena iterativna zgoščevalna ni varna in tudi drugače ni primerna za uporabo v kriptografiji. Po drugi strani pa je lahko uporaba iterativne zgoščevalne funkcija, ki sicer ni varna, v NMAC sprejemljiva.

5.2 Varnost HMAC

Varnost avtentikacijske sheme HMAC temelji na varnosti NMAC. Ključa, ki ju uporablja NMAC, morata biti naključna in med seboj neodvisna. Takšna morata biti tudi ključa K_1 in K_2, ki se določita na podlagi osnovne zgoščevalne funkcije *f* uporabljene v HMAC, da lahko zaključke o varnosti NMAC prenesemo na HMAC. V ta namen mora osnovna zgoščevalna funkcija *f* zagotavljati zadovoljivo "naključnost" zgostitev vhodnih nizov. Ker pa zgostitve z osnovno zgoščevalno funkcijo napadalcu niso vidne, ni potrebno, da bi bile zahteve po "naključnosti" osnovne zgoščevalne funkcije zelo stroge. Po drugi strani lastnost "naključnosti" zgoščevalne funkcije ne bi smela biti težko doseči tudi zato, ker se mnoge izmed kriptografskih zgoščevalnih funkcij v praksi uporabljajo za generiranje naključnih števil. Napadi, ki uspejo razbiti HMAC ne pa tudi NMAC, so sicer možni, a kažejo na precejšnje pomanjkljivosti v zvezi z lastnostjo "naključnosti" osnovne zgoščevalne funkcije, ki je uporabljena v konstrukciji HMAC oziroma NMAC. Če pa je (iterativna) kriptografska zgoščevalna funkcija varna v smislu odpornosti na trke in ima tudi lastnosti "naključnosti", potem sta varni obe avtentikacijski shemi NMAC in HMAC.

Eden izmed možnih napadov na HMAC je tudi iskanje skrivnega ključa s požrešnim algoritmom. Ker so v konstrukciji HMAC podprti tudi ključi z dolžinami različnimi od *l* bitov, kjer je *l* dolžina zgostitve iterativne zgoščevalne funkcije uporabljene v HMAC, je dobro vedeti, kakšne naj bodo dolžine ključev. Pravilo je, da imajo ključi dolžino večjo ali enako kot *l/2* bitov, vendar pa ključi z dolžino večjo kot *l* bitov v primerjavi s ključi dolžine *l* bitov ne pripomorejo dosti k varnosti HMAC, saj se ključ v algoritmu HMAC tako ali tako zgosti z osnovno zgoščevalno funkcijo na dolžino *l* bitov. Ravno tako je potrebno poskrbeti, da se ključi periodično zamenjujejo.

Algoritem HMAC predvideva, da imajo poleg ključev tudi avtentikacijske kode lahko dolžino različno od *l* bitov. Avtentikacijska koda je lahko krajša od *l* bitov, vendar naj bi bila vsaj dolžine *80* bitov in večja ali enaka kot *l/2* bitov. S tem ko je avtentikacijska koda krajša, je napadalcu na voljo manj podatkov za razbitje HMAC, kar je prednost. Slabost pa je, da je napadalcu potrebno predvideti manjše število bitov v avtentikacijski kodi.

5.3 Napadi na NMAC/HMAC

Najmočnejši poznan napad na avtentikacijski shemi NMAC/HMAC je napad s paradoksom rojstnih dnevov. Možna oblika tega napada na NMAC, ki temelji na iterativni zgoščevalni funkciji F z zgostitvijo dolžine l bitov, poteka na sledeči način. Napadalec si priskrbi avtentikacijske kode $NMAC_{(K_1,K_2)}$ za mnogo različnih sporočil, ki pa morajo vsa imeti enako dolžino. Število sporočil mora biti tolikšno, da imata vsaj dve sporočili M_p in M_q izmed vseh enako avtentikacijsko kodo $NMAC_{(K_1,K_2)}(M_p) = NMAC_{(K_1,K_2)}(M_q)$. Zatem si napadalec priskrbi še avtentikacijsko kodo $NMAC_{(K_1,K_2)}(M_p||B)$ za prvo izmed teh dveh sporočil (M_p), ki mu na konec doda poljuben neprazen niz B ($M_p'=M_p||B$). Če sta enaki avtentikacijski kodi $NMAC_{(K_1,K_2)}(M_p)$ in $NMAC_{(K_1,K_2)}(M_q)$ za sporočili M_p in M_q posledica enakih zgostitev teh dveh sporočil $F_{K_2}(M_p)$ in $F_{K_2}(M_q)$ z notranjo iterativno zgoščevalno funkcijo F_K uporabljeno v NMAC, potem je avtentikacijska koda $NMAC_{(K_1,K_2)}(M_q||B)$ za sporočilo $M_q'=M_q||B$ enaka avtentikacijski kodi $NMAC_{(K_1,K_2)}(M_p||B)$, saj sta tudi zgostitvi nizov M_p' in M_q' z notranjo iterativna zgoščevalno funkcijo F_K enaki. S tem je avtentikacijska shema NMAC razbita, razbitje pa postane verjetno šele, ko si napadalec priskrbi avtentikacijske kode za vsaj $2^{l/2}$ sporočil.

Opisani napad je možno prilagoditi tudi za HMAC, vendar pa je povsem neuporaben, če v konstrukciji HMAC uporabimo varno kriptografsko zgoščevalno funkcijo z zgostitvijo dolžine $l \geq 128$. V primeru ko HMAC uporablja iterativno zgoščevalno funkcijo, kot je npr. MD5 (zgostitev dolžine $l=128$ bitov in dolžina bloka b=512 bitov), si mora napadalec od oseb, ki posedujejo skrivni ključ, priskrbeti avtentikacijske kode pridobljene z enim in istim ključem za 2^{64} sporočil. Na prenos vseh potrebnih podatkov (sporočil in avtentikacijskih kod) pa bi v tem primeru napadalec preko 1Gbit/sec povezave moral čakati več kot 250 tisoč let. Če pa napad s paradoksom rojstnih dnevov morebiti uspe, je to jasen znak, da v avtentikacijski shemi HMAC uporabljena iterativna zgoščevalna funkcija ni varna in posledično tudi drugače ni primerna za uporabo v kriptografiji.

Napada s paradoksom rojstnih dnevov se poslužujemo tudi za "razbijanje" iterativne zgoščevalne funkcije z javnim in vnaprej določenim vhodnim vektorjem IV, s tem da najdemo enaki zgostitvi dveh različnih sporočil za to iterativno zgoščevalno funkcijo. Takšen napad na iterativno zgoščevalno funkcijo z zgostitvijo dolžine l ravno tako zahteva izračun oziroma poznavanje vsaj $2^{l/2}$ zgostitev različnih sporočil. V tem primeru verjetnost, da med temi zgostitvami najdemo dve enaki, ni več zanemarljiva. Ker je vhodni vektor IV iterativne zgoščevalne funkcije javen in vnaprej določen, si napadalec lahko sam izračunava avtentikacijske kode za sporočila, možna pa je tudi paralelizacija napada, tako da je čas potreben za "razbitje" iterativne zgoščevalne funkcije v primerjavi s časom potrebnim za razbitje avtentikacijske sheme HMAC, ki uporablja isto iterativno zgoščevalno funkcijo, mnogo krajši. Uporaba običajne iterativne zgoščevalne funkcije z zgostitvijo dolžine le 128 bitov, kot je npr. MD5, je ob današnji tehnologiji in napadu s paradoksom rojstnih dnevov, ki izkorišča paralelnost, tako vprašljiva. Kot smo videli, pa je uporaba iste iterativne zgoščevalne funkcije v avtentikacijski shemi HMAC povsem varna.

Poleg napada s paradoksom rojstnih dnevov, je možno razbiti HMAC s požrešnim iskanjem skrivnega ključa. Takšen napad terja za ključe z dolžino enako ali daljšo kot l bitov, kjer je l dolžina zgostitve iterativne zgoščevalne funkcije uporabljene v konstrukciji HMAC, časovno zahtevnost reda 2^l operacij. V primeru, ko je ključ krajši in ima dolžino $l'<l$ bitov, pa je časovna zahtevnost reda $2^{l'}$ operacij. Od tod tudi priporočilo, da naj bi imeli ključi dolžino večjo ali enako kot $l/2$ bitov, da preprečimo to vrsto napada.

6. Uporaba HMAC

Avtentikacija z algoritmom HMAC je predvidena ali pa se že uporablja v mnogih novejših internetnih protokolih, kot so npr. SSH (Secure Shell), IPSec (IP Secure), SHTTP (Secure HTTP), TLS (Transport Layer Security), SET (Secure Electronic Transaction)… Konstrukcija HMAC je v teh protokolih nadomestila kriptografsko manj varne avtentikacijske sheme, ponekod pa celo zgoščevalne funkcije, ki sploh niso uporabljale ključev. Malce bolj podrobno si poglejmo, kje je mesto HMAC v protokolih SSH in IPSec.

V prvi različici protokola SSH (SSH-1) se je kontrolna vsota za preverjanje integritete prenašanih podatkov izračunavala brez ključev preko algoritma CRC-32. V drugi različici tega protokola (SSH-2) pa je dodatno podprta avtentikacijska shema HMAC (HMAC-SHA1), ki temelji na kriptografski zgoščevalni funkcijo SHA1. Poleg avtentikacijske sheme HMAC-SHA1 predvideva protokol SSH-2 tudi podporo za avtentikacijski shemi HMAC (HMAC-MD5 in HMAC-RIPEMD160), temelječi na kriptografskih zgoščevalnih funkcijah MD5 in RIPEMD160. Programske rešitve (npr. OpenSSH), ki temeljijo na protokolu SSH, pa so celovit paket, ki zagotavlja varnost v smislu integritete podatkov in overjanja pošiljatelja pri oddaljenem delu na računalniku, s tem da ponuja kriptografsko varno zamenjavo za telnet, rlogin, rsh, rcp…

Na nižjem nivoju kot ga naslavlja protokol SSH deluje protokol IPSec, ki določa šifriranje in avtentikacijo na nivoju omrežnega komunikacijskega sloja (IP), česar zdajšnja četrta različica internetnega protokola IPv4 ne omogoča. Za preverjanje nespremenjenosti podatkov in overjanje je v protokolu IPSec predvideno avtentikacijsko zaglavje (AH – Authentication Header). V začetku je bila za avtentikacijo predlagana avtentikacijska shema, ki temelji na kriptografski zgoščevalni funkciji MD5 in ključu, ki je dodan pred sporočilo in na njegov konec, vendar pa je bila ta avtentikacijska shema kasneje iz varnostnih razlogov zamenjana s HMAC- SHA1 in HMAC-MD5.

7. Zaključek

Na koncu se lahko vprašamo, kje so možnosti za izboljšavo algoritma HMAC in kakšne so alternative uporabi HMAC. Sama hitrost algoritma HMAC je predvsem odvisna od uporabljene kriptografske zgoščevalne funkcije, enako velja za varnost te avtentikacijske sheme. Izboljšavo zato lahko prinesejo nove hitrejše oziroma varnejše kriptografske zgoščevalne funkcije. Kot možna alternativa in bodoča zamenjava za HMAC pa se največkrat omenja novejša avtentikacijska shema UMAC, ki je hitrejša in jo je mogoče paralelizirati, hkrati pa je tudi varna.

VIRI:

[1] M. Bellare, R. Canetti and H. Krawczyk, "HMAC: Keyed-Hashing for Message Authentication,", RFC2104 February 1997.

[2] M. Bellare, R. Canetti, and H. Krawczyk, "Keyed Hash Functions and Message Authentication", Proceedings of Crypto'96, Lecture Notes in Computer Science Vol, 1109, pp. 1-15, 1996.

[3] William Stallings, "The HMAC Algorithm: Key hashing for message authentication", Dr. Dobb's Journal, April 1999.

[4] D. Stinson, "Cryptography: Theory and Practice", CRC Press, 2nd. ed. 2002.

[5] D. J. Barrett, R. E. Silverman: "SSH, the Secure Shell: The Definitive Guide", O'Reilly, 2001.

[6] S. Kent, R. Atkinson, "IP Authentication Header", RFC2402, November 1998.