Autentificirana enkripcija

Biskup, Lucija

SVEUČILIŠTE U ZAGREBU
PRIRODOSLOVNO–MATEMATIČKI FAKULTET
MATEMATIČKI ODSJEK

Lucija Biskup

AUTENTIFICIRANA ENKRIPCIJA

Diplomski rad

Voditelj rada:
Izv. prof. dr. sc. Filip Najman

Zagreb, Travanj, 2018

Ovaj diplomski rad obranjen je dana _____ pred ispitnim povjerenstvom u sastavu:

1. _____, predsjednik
2. _____, član
3. _____, član

Povjerenstvo je rad ocijenilo ocjenom _____.

Potpisi članova povjerenstva:

1. _____
2. _____
3. _____

Sadržaj

Uvod

Ljudi se od davnina bave problematikom slanja poruka nesigurnim kanalima i oduvijek žele sigurno komunicirati. Razvojem tehnologije, a posebno interneta, potreba za sigurnošću prijenosa podataka raste. Bezuvjetno povjerenje se može postići samo ako se komunikacija odvija licem u lice. U svim drugim slučajevima, podaci su izloženi različitim prijetnjama. Dakle, u svijetu rasprostranjenih i udaljenih komunikacija, povjerenje ovisi o primjeni pouzdanih mehanizama. Stoga, ako želimo postići privatnost poslanih podataka, potreban nam je mehanizam za zaštitu povjerljivosti poslane poruke. Međutim, tajna informacija je podložna i različitim vrstama modifikacije i pogrešaka u prijenosu. Da bi se od njih zaštitili, trebamo integrirati dodatni sigurnosni mehanizam koji jamči autentičnost. U cilju da se spriječe ovakvi napadi, autorizirane strane komunikacije se dogovore da će se koristiti kriptografskim tehnikama da bi osigurale sigurnost i postigle kriptografske ciljeve. Kriptografija je znanstvena disciplina koja se bavi proučavanjem metoda za slanje poruka u takvoj formi koja je čitljiva samo onima kojima je informacija namijenjena, dok će za ostale biti neupotrebljiva.

Mi ćemo se u ovom radu fokusirati na postizanje dva najvažnija cilja kriptografije, a to su pružanje povjerljivosti i autentičnosti poruka koje se šalju preko nesigurnog kanala. Povjerljivost osigurava da su podaci dostupni samo onim stranama koje su ovlaštene za njihovo pribavljanje, dok autentičnost provjerava je li sadržaj poruke tijekom transporta od izvora do odredišta ostao nepromijenjen. Zaštita povjerljivosti općenito ne osigurava integritet poruka. Kroz povijest su se ova dva aspekta sigurnosti komunikacije proučavala odvojeno, te su se ostvarivala zasebnim metodama. Međutim, u praksi je često potrebno istovremeno osigurati nevedene ciljeve. Većina aplikacija zahtijeva autentičnost kao dodatak povjerljivosti. Kao što su ciljevi privatnosti i integriteta poruka različiti, tako se razlikuju i tehnike i alati za njihovo postizanje. Povjerljivost se najčešće ostvaruje metodama enkripcije, dok se autentičnost provjerava izračunavanjem autentifikacijskog koda poruke (*eng. Message Authentication Code*).

U prvom poglavlju ćemo proučavati metode enkripcije. Prvo ćemo ukratko opisati kriptografske algoritme i navesti njihove načine djelovanja. Zatim ćemo klasificirati

vrste napada na enkripcijske sheme prema tipu informacije koju napadač posjeduje, te ćemo dati definicije sigurnosti kriptosustava pod pojedinim napadom.

U drugom poglavlju se bavimo metodama za postizanje integriteta poruke. Prvo ćemo opisati hash-funkcije i navesti ćemo sva potrebna svojstva koja bi one trebale posjedovati. Zatim ćemo opisati posebnu klasu hash-funkcija s ključem, tzv. kodove za autentifikaciju poruke (MAC). Rezultat njihovog izračunavanja je niz bitova koji se dodaje na orginalnu poruku da bi se osigurala autentifikacija pošiljatelja i integritet poruke.

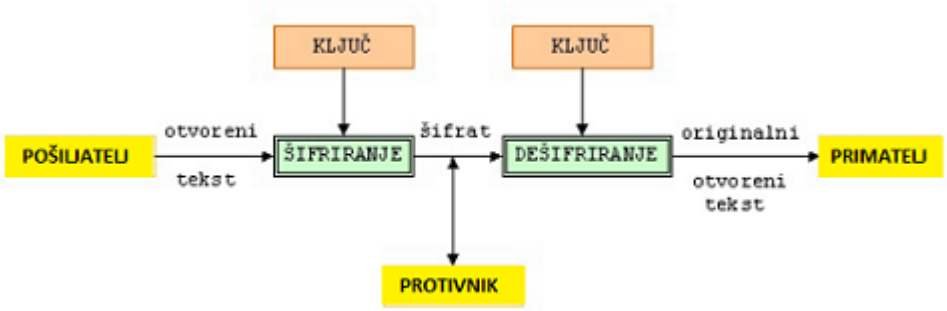
Treće poglavlje opisuje pojam autentificirane enkripcije. Autentificirana enkripcija je oblik enkripcije koji, osim povjerljivost, osigurava integritet i autentičnost poslane poruke. Prvo opisujemo tri kombinacije enkripcijske sheme i MAC-a, a potom ispitujemo njihovu sigurnost. Pokazati će se da jedino kombinacija *šifriraj-pa-autentificiraj* zadovoljava kriterije sigurnosti. Na kraju navodimo neke algoritme autentificirane enkripcije. Ti algoritmi su relativno nedavno nastali. Pošto su efikasni i lakše upravljaju ključevima, već se učestalo koriste.

Diplomski rad napravljen je u sklopu aktivnosti Projekta KK.01.1.1.01.0004 - Znanstveni centar izvrsnosti za kvantne i kompleksne sustave te reprezentacije Liejevih algebri.

Poglavlje 1

Osnovni enkripcijski pojmovi

U daljnjem razmatranju, dvije osobe koje komuniciraju putem komunikacijskog kanala nazivamo *pošiljatelj* i *primatelj*. Originalnu poruku koju pošiljatelj šalje primatelju nazivamo *otvoreni tekst*. Pošiljatelj transformira otvoreni tekst koristeći unaprijed dogovoreni ključ te tako dobiva šifriranu poruku koju šalje putem komunikacijskog kanala. Taj se postupak zove *šifriranje* ili *enkripcija*, a dobiveni rezultat *šifrat*. Primatelj zna ključ te pomoću njega može dešifrirati poruku i dobiti originalni tekst. Taj postupak dekodiranja poruke, odnosno vraćanja poruke iz njenog kriptiranog oblika u originalni oblik nazivamo *dešifriranje* ili *dekripcija*. Osobu koja prisluškuje komunikacijski kanal nazivamo *protivnik* ili *napadač*.



Slika 1.1: Osnovni pojmovi

Ako otvoreni tekst označimo s m, a šifrat s c, tada funkcija enkripcije Enc djeluje na m i vraća c:

$$\mathrm{Enc}(m) = c.$$

U obrnutom procesu, funkcija dekripcije Dec djeluje na c i vraća m:

$$\mathrm{Dec}(c) = m.$$

Kako je cijeli smisao enkriptiranja, a zatim dekriptiranja poruke dobiti ponovno otvoreni tekst, sljedeći izraz mora biti zadovoljen:

$$\mathrm{Dec}(\mathrm{Enc}(m)) = c.$$

U nastavku ćemo otvoreni tekst m promatrati kao element skupa $\{0,1\}^*$, gdje nam operator * predstavlja Kleenijevu zvijezdu, odnosno $\{0,1\}^* = \bigcup_{k=0}^{\infty} \{0,1\}^k$.

1.1 Osnovni kriptografski algoritmi

U prošlosti, prije nego što su računala ušla u široku upotrebu, većina kriptografskih metoda šifriranja se bazirala na tajnosti šifre. Međutim, takvi algoritmi su bili dosta nepouzdani, stoga su se morali pronaći neki drugi načini šifriranja. Današnje se metode šifriranja zasnivaju na uporabi ključa. Ključ je najvažniji dio u pravilnom enkriptiranju i dekriptiranju poruka.

Ovisno o načinu korištenja ključa, razvile su se dvije klase algoritama, **simetrični** i **asimetrični** algoritmi kriptiranja. Simetrični algoritmi koriste isti ključ za enkripciju i dekripciju, dok asimetrični algoritmi koriste različite ključeve.

Simetrični algoritmi

Simetrični algoritmi koriste isti ključ za enkripciju i dekripciju podataka. Sigurnost simetričnih algoritama ovisi o sigurnosti samog algoritma i dužini ključa. Najpoznatiji simetrični algoritam je DES (*eng. Data Encryption Standard*), koji je razvio IBM-ov tim kriptografa 1976. godine. DES šifrira otvoreni tekst duljine 64 bita, koristeći ključ duljine 56 bitova. Algoritam je zasnovan na tzv. *Feistelovoj šifri*. Jedna od glavnih ideja je alternirana uporaba supstitucija i transpozicija kroz više iteracija. Bio je standard sve do 2000. godine kad ga je zamijenio AES (*eng. Advanced Encryption Standard*), koji rukuje 128-bitnim blokovima i ključevima duljine 128, 192 i 256 bita.

Glavni nedostatak simetričnih algoritama je taj što se mora osigurati razmjena ključeva preko nesigurnih komunikacijskih kanala. Ukoliko je ključ kompromitiran, napadač može dekriptirati sve poruke enkriptirane tim ključem. Također, sa svakim novim korisnikom mreže naglo raste broj ključeva. Za mrežu koja ima n korisnika treba imati $\frac{n\cdot(n-1)}{2}$ ključeva. Tako primjerice 100 korisnika zahtjeva 4950 ključeva.

Simetrična kriptografija je podijeljena na blok šifre i protočne šifre.

- **Blok šifre** Kod blok šifriranja enkripcija se vrši po blokovima podataka, tj. uzimaju se blokovi od više bitova (64, 128, 196, 256 ...), te se enkriptiraju kao cjelina. U praksi se blok šifre koriste češće od protočnih šifri.

- **Protočne šifre** Šifriranje pomoću protočnih šifri radi tako da se enkripcija poruke vrši bit po bit, dok se dekripcija najčešće vrši inverznim enkriptiranjem, tj. algoritam je isti, ali se podključevi enkripcije koriste obrnutim redoslijedom.

Asimetrični algoritmi

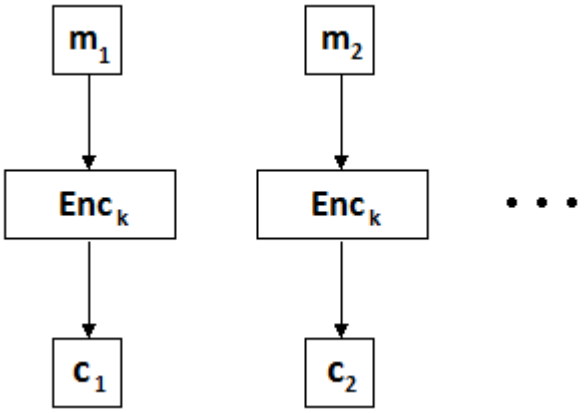
Ove algoritme nazivamo još i algoritmi javnog ključa. Oblikovani su tako da se ključ koji se koristi za enkripciju razlikuje od ključa koji se koristi za dekripciju. Enkripcijski ključ se objavi javno te ga bilo tko može iskoristiti radi enkriptiranja poruke, ali poruka može biti dešifrirana samo s primateljevim privatnim ključem. Asimetrični algoritmi funkcioniraju na sljedeći način: osoba A objavi svoj javni ključ preko nekog medija. Osoba B koja osobi A želi poslati poruku, šifrira tu svoju poruku s ključem koji je osoba A javno objavila te joj pošalje takvu poruku. Jedino osoba A sa svojim privatnim ključem može dešifrirati poruku poslanu od osobe B.

Prednost asimetričnih algoritama u odnosu na simetrične je u tome da nema potrebe za pohranjivanjem mnogo ključeva. Svaka strana treba čuvati samo svoj privatni ključ, a javni ključevi drugih strana se mogu dobiti kada zatrebaju. Također, strane koje komuniciraju ne moraju dijeliti tajni ključ što omogućuje tajno komuniciranje čak i kad je sva komunikacija među strankama nadzirana. Međutim, enkripcija javnim ključem zahtjeva mnogo vremena za obavljanje operacija šifriranja i dešifriranja pa se kod slanja duljih poruka češće koriste simetrični algoritmi.

Načini djelovanja algoritama za kriptiranje podataka

- **ECB** (*eng. Electronic Code Book*): ECB je najjednostavniji način šifriranja sa blok šiframa gdje se svaki blok otvorenog teksta šifrira pomoću istog ključa. Poruka se rastavi na blokove dužine b bita ($b = 64$ ako se radi o DES standardu). Ako zadnji blok nije dovoljne dužine, on se nadopuni dodatnim bitovima. Zatim se pomoću istog ključa šifrira jedan po jedan blok. Blokovi se šifriraju neovisno jedan o drugom pa se šifriranje može paralelizirati. Također, greška u jednoj blok šifri utječe na dekripciju samo tog bloka, dok na ostale blokove nema utjecaj. ECB je vrlo siguran za prijenos kratkih poruka, dok je kod prijenosa dužih poruka sigurnost nešto manja. Razlog tome je što isti blokovi u otvorenom tekstu daju iste šifrate pa napadač može jednostavno analizirati blokove koji se ponavljaju i pomoću toga doći do otvorenog teksta. Stoga se ECB najčešće koristi za razmjenu ključeva za šifriranje.

- **CBC** (*eng. Cipher Block Chaining*): Glavna ideja režima ulančavanja blok šifri je da šifrat c_i bloka m_i ne zavisi samo od bloka m_i nego i od svih prethodnih blokova otvorenog teksta. Na trenutni blok otvorenog teksta se primjeni operacija



Slika 1.2: ECB [1]

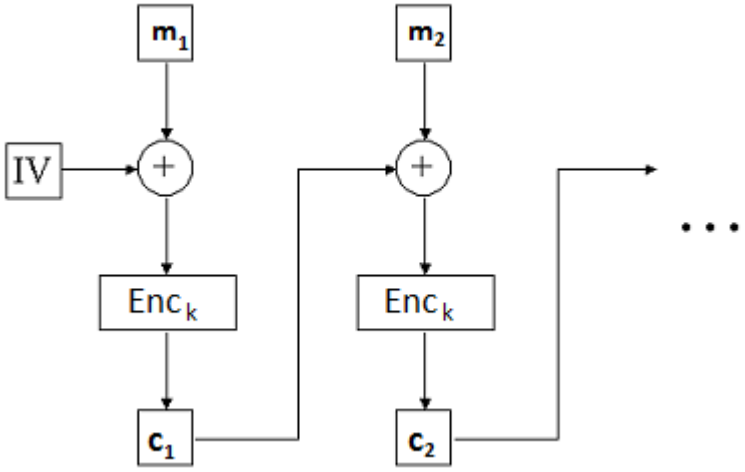
XOR sa šifratom prethodnog bloka, a zatim se dobiveni rezultat šifrira pomoću ključa k. Na taj način se identičnim blokovima u otvorenom tekstu pridružuju različiti šifrati, što vodi povećanju sigurnosti. Za prvi blok otvorenog teksta ne postoji prethodni šifrat. Zbog toga za c_0 uzimamo da je jednak IV, gdje je IV vektor inicijalizacije. Taj vektor nam omogućuje da učinimo svaku CBC enkripciju nedeterminističkom. Radi se o nizu proizvoljno odabranih bitova čija je duljina jednaka duljini bloka. Ta vrijednost mora biti poznata i pošiljatelju i primatelju, a oni je mogu razmijeniti koristeći se npr. ECB-om.

Dakle, konstrukcija CBC-a je slijedeća:

$$c_i = \mathrm{Enc}_k(c_{i-1} \oplus m_i), \quad i \geq 1$$

$$c_0 = IV$$

Ako šifriramo niz blokova jednom sa IV, a drugi put sa IV' gdje je $IV \neq IV'$, dva šifrata koje dobijamo kao rezultat izgledaju potpuno nepovezano za napadača. Stoga se preporuča što češća izmijena vektora inicijalizacije. Postoji mnogo različitih načina za dobivanje i usaglašavanje vrijednosti vektora inicijalizacije. To može biti nasumično izabran broj koji autorizirane strane razmijene prije prijenosa same poruke. Alternativno, za vektor inicijalizacije se može uzet neka vrijednost poznata poošiljatelju i primatelju koja se poveća svaki puta kada nova sesija započne. Nedostatak ovog modusa je u tome što se šifriranje ne može paralelizirati jer algoritam mora završiti obradu jednog bloka da bi mogao preći na idući.



Slika 1.3: CBC [1]

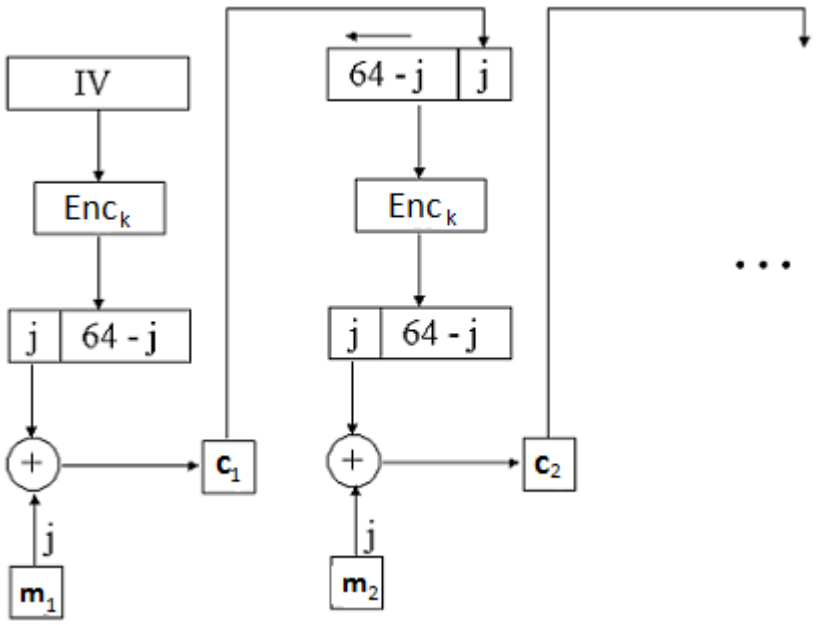
- **CFB** (*eng. Cipher Feedback*): Za razliku od prethodna dva načina gdje se koriste blokovne šifre, ovdje koristimo protočnu šifru. Odjednom se obrađuje j bitova. Prvo šifriramo vektor inicijalizacije IV. Zatim, na prvih j bitova dobivenog izlaza primijenimo XOR operaciju sa m_1 i tako dobijemo c_1. U idućem koraku se za ulaz uzima prethodni ulazni podatak pomaknut za j mjesta ulijevo, a na desni kraj stavimo c_1. Postupak nastavljamo sve dok se sve jedinice otvorenog teksta ne šifiraju. Budući da se kod protočnih šifri poruke ne trebaju proširivati, šifrat će biti jednako dug kao i otvoreni tekst.

- **OFB** (*eng. Output Feedback*): Jedina razlika u odnosu na CFB je da se ulazni podatak za funkciju enkripcije šalje odmah nakon primjene Enc_k u prethodnom koraku, odnosno prije primjene operacije XOR. Na taj način se greške u transmisiji ne propagiraju, što znači da npr. greška u bloku c_1 utječe samo na blok m_1.

- **CTR** (*eng. Counter*): Koristi se niz brojača $x_1, x_2, ...$ koji moraju biti u parovima različiti. To se postiže tako da se prvom brojaču pridruži neka inicijalna vrijednost, a zatim se ostali brojači povećaju za 1: $x_i = x_1 + (i-1) \mod 2^b$ (b je duljina bloka). Niz šifrata se dobiva na sljedeći način:

  $$c_i = m_i \oplus e_k(x_i).$$

  U CTR modu se šifriranje može jednostavno paralelizirati, što je kod ulančanih modusa bio problem jer bi algoritam morao prvo završiti obradu jednog bloka



Slika 1.4: CFB [1]

da bi mogao preći na idući.

1.2 Kriptografski napadi

Napadi su zlonamjerne akcije koje izvode neautorizirani članovi komunikacijskog modela, tzv. napadači. Uspješan napad na kriptografski sustav podrazumijeva pronalaženje praktičnog načina da napadač od šifriranog teksta dobije otvoreni tekst. Mi ćemo promatrati sigurnost od polinomijalnih napada čije se vrijeme trajanja, kao i vjerojatnost uspjeha, mogu prikazati kao funkcije od $n \in \mathbf{N}$. Kada autorizirane strane komunikacije iniciraju enkripciju, one izaberu neku vrijednost za n koju nazivamo sigurnosni parametar. Pretpostavlja se da je vrijednost tog parametra poznata i napadaču.

Općenito, enkripcijsku shemu ćemo smatrati sigurnom od određene vrste napada ako za sve polinomijalne protivnike vrijedi da je vjerojatnost uspješnog napada zanemariva. Sada ćemo iskazati formalnu definiciju enkripcijske sheme i zanemarive funkcije.

Definicija 1.2.1. *[2] Shema enkripcije s privatnim ključem je trojka polinomijalnih algoritama* (Gen, Enc, Dec) *takvih da:*

1. *Algoritam za generiranje ključa* Gen *uzima kao ulaz sigurnosni parametar* 1^n *i vraća ključ* k. *Pišemo:* $k \leftarrow \text{Gen}(1^n)$. *Bez smanjenja općenitosti možemo pretpostavti da za svaki* k *izlaz od* $\text{Gen}(1^n)$ *vrijedi* $|k| \geq n$.

2. *Algoritam enkripcije* Enc *uzima kao ulaz ključ* k *i otvoreni tekst* $m \in \{0,1\}^*$ *i vraća šifrat* c. *Pišemo:* $c \leftarrow \text{Enc}_k(m)$

3. *Algoritam dekripcije* Dec *uzima kao ulaz ključ* k *i šifrat* c *i vraća poruku* m. *Pretpostavljamo da je* Dec *deterministički pa pišemo:* $m := \text{Dec}_k(c)$.

Želimo da za svaki n, *svaki* k *izlaz od* $\text{Gen}(1^n)$ *i svaki* $m \in \{0,1\}^*$ *vrijedi*

$$\text{Dec}_k(\text{Enc}_k(m)) = m.$$

Definicija 1.2.2. *[2] Za funkciju* f *kažemo da je* ***zanemariva*** *ako za svaki polinom* p *postoji* $N \in \mathbf{N}$ *takav da za svaki* $n \in \mathbf{N}, n > N$ *vrijedi* $f(n) < \dfrac{1}{p(n)}$.

Razlikujemo četiri vrste kriptoanalitičkih napada prema tipu informacije koju napadač posjeduje:

- **Samo šifrat**

  Napadač posjeduje samo šifrate od nekoliko poruka šifriranih pomoću istog algoritma. U tom slučaju on želi otkriti orginalne poruke čiji su to šifrati ili ključ pomoću kojeg su te poruke šifrirane.

- **Poznat otvoreni tekst**

  Napadač posjeduje šifrat neke poruke i njemu odgovarajući otvoreni tekst. Tada on provodi analizu danih podatka s ciljem pronalaženja ključa koji se koristio za šifriranje. Ovakvoj vrsti napada su podložne klasične šifre, dok su moderne šifre otpornije.

- **CPA**

  Kratica CPA dolazi od engleskog izraza *chosen - plaintext attacks*, što se može prevesti kao napadi odabranim otvorenim tekstovima. Kod CPA napada, napadač ima mogućnost odabira teksta koji će biti šifriran pomoću ključa k. Zatim, on može tako šifrirani tekst slati dvjema stranama putem kanala koji može nadzirati. Također, može i promatrati poruke koje putuju tim kanalom, a bile su šifrirane istim ključem k.

  Pretpostavimo da napadač zna da je poruka m jedna od dvije mogućnosti m_0, m_1. Kažemo da je kriptosustav CPA-siguran ako napadač ne može odlučiti

koja od poruka m_0, m_1 odgovara početnoj poruci m bilo kakvim postupkom čija je vjerojatnost veća od vjerojatnosti nasumičnog pogađanja. Navesti ćemo i formalnu definiciju.

U formalnoj definiciji CPA napade modeliramo tako da protivniku $\mathcal{A}$ omogućimo pristup tzv. "crnoj kutiji". Protivnik odabire poruke koje će se kriptirati u "crnoj kutiji" koristeći ključ k. Međutim, ključ k nije poznat protivniku. Dakle, protivnik $\mathcal{A}$ šalje "crnoj kutiji" svoju poruku m, a dobiva kriptirani tekst $c \leftarrow \mathrm{Enc}_k(m)$. Protivnik $\mathcal{A}$ ima pravo pristupa "crnoj kutiji" koliko god puta želi.

Promotrimo sljedeći eksperiment. Neka je $\Pi = (\mathrm{Gen}, \mathrm{Enc}, \mathrm{Dec})$ shema enkripcije gdje je Gen funkcija za generiranje ključa, Enc funkcija koja šifrira otvoreni tekst i Dec funkcije koja dešifrira kriptirani tekst. Neka je $\mathcal{A}$ protivnik te neka je n parametar sigurnosti.

Eksperiment CPA neprimjetnosti $\mathrm{PrivK}^{\mathrm{cpa}}_{\mathcal{A},\Pi}(n)$**:**

1. Ključ k se generira koristeći funkciju $\mathrm{Gen}(1^n)$.
2. Protivniku $\mathcal{A}$ je dan ulaz 1^n i pristup "crnoj kutiji" u kojoj se enkriptiraju podaci sa ključem k, te za izlaz vraća dvije poruke m_0, m_1 koje su jednake duljine.
3. Uniformno se odabire bit $b \in \{0, 1\}$, zatim se izračuna kriptirani tekst $c \leftarrow \mathrm{Enc}_k(m_b)$ te se taj kriptirani tekst daje protivniku $\mathcal{A}$.
4. Protivnik $\mathcal{A}$ nastavlja pristupati "crnoj kutiji" te na kraju vraća bit b'.
5. Izlaz eksperimenta je definiran kao 1 ako je $b = b'$ i tada kažemo da je protivnik $\mathcal{A}$ uspio. Ako je $b \neq b'$, izlaz eksperimenta je 0 i kažemo da protivnik $\mathcal{A}$ nije uspio.

Definicija 1.2.3. *[2] Shema enkripcije* $\Pi = (\mathrm{Gen}, \mathrm{Enc}, \mathrm{Dec})$ *s privatnim ključem je CPA-sigurna ako za sve polinomijalne protivnike* $\mathcal{A}$ *postoji zanemariva funkcija* negl *takva da je*

$$P\big[\,\mathrm{PrivK}^{\mathrm{cpa}}_{\mathcal{A},\Pi}(n) = 1\big] \leq \frac{1}{2} + \mathrm{negl}(n).$$

Iz gornje definicije je vidljivo da nijedna deterministička enkripcijska shema nije CPA-sigurna.

- **CCA**

  Kratica CCA dolazi od izraza *eng. chosen - ciphertext attacks*, što možemo prevesti kao napadi odabranim šifratima. Kod CCA napada protivniku je omogućen pristup "crnoj kutiji" u kojoj se proizvoljne poruke mogu enkriptirati i dekriptirati. Cilj napadača je otkriti ključ za dešifriranje. Ovaj napad je tipičan za kriptosustave s javnim ključem.

  Promotrimo sljedeći eksperiment. Neka je $\Pi = (\mathrm{Gen}, \mathrm{Enc}, \mathrm{Dec})$ shema enkripcije sa privatnim ključem gdje je Gen funkcija za generiranje ključa, Enc funkcija koja šifrira otvoreni tekst i Dec funkcije koja dešifrira kriptirani tekst. Neka je $\mathcal{A}$ protivnik te neka je n parametar sigurnosti.

  Eksperiment CCA neprimjetnosti $\mathrm{PrivK}^{\mathrm{cca}}_{\mathcal{A},\Pi}(n)$**:**

  1. Ključ k se generira koristeći funkciju $\mathrm{Gen}(1^n)$.
  2. Protivniku $\mathcal{A}$ je dan ulaz 1^n i pristup "crnoj kutiji" u kojoj se poruke mogu enkriptirati i dekriptirati. Protivnik daje za izlaz dvije poruke m_0, m_1 koje su jednake duljine.
  3. Uniformno se odabire bit $b \in \{0, 1\}$, zatim se izračuna kriptirani tekst $c \leftarrow \mathrm{Enc}_k(m_b)$ te se taj kriptirani tekst daje protivniku $\mathcal{A}$.
  4. Protivnik $\mathcal{A}$ nastavlja pristupati "crnoj kutiji" te na kraju vraća bit b'.
  5. Izlaz eksperimenta je 1 ako je $b = b'$ i tada kažemo da je protivnik $\mathcal{A}$ uspio. Ako je $b \neq b'$, izlaz eksperimenta je 0 i kažemo da protivnik $\mathcal{A}$ nije uspio.

  Definicija 1.2.4. *[2] Shema enkripcije* $\Pi = (\mathrm{Gen}, \mathrm{Enc}, \mathrm{Dec})$ *s privatnim ključem je CCA-sigurna ako za sve polinomijalne protivnike* $\mathcal{A}$ *postoji zanemariva funkcija* negl *takva da je*

  $$P\big[\,\mathrm{PrivK}^{\mathrm{cca}}_{\mathcal{A},\Pi}(n) = 1\big] \leq \frac{1}{2} + \mathrm{negl}(n).$$

  U gornjem eksperimentu protivniku je omogućen neograničen pristup "crnoj kutiji", ali uz uvjet da protivnik ne zatraži dekriptiranje šifrata $c \leftarrow \mathrm{Enc}_k(m_b)$. Bez tog uvjeta, navedenu sigurnost bi bilo nemoguće postići.

Poglavlje 2

Osiguravanje integriteta

U prethodnom poglavlju smo opisali kako je moguće postići privatnost prilikom komunikacije otvorenim kanalima. Međutim, pojam sigurnosti nije samo u činjenici da napadač ne zna poslanu poruku. Ponekad je čak i od veće važnosti osigurati integritet poslane poruke, odnosno primatelj se mora moći uvjeriti od koga je poruku primio. Integritet podataka je snažno povezan s autentifikacijom izvora podataka. Ako se poruka mijenja, njen integritet je ugrožen, a samim time i izvor pošljatelja više nije ovjeren. Dakle, integritet poruka i autentifikacija izvora neodvojivo su vezani jedno za drugo. Kršenje jednog od njih automatski vodi kršenju drugog. Stoga, autentifikacijski mehanizmi implicitno osiguravaju i autentifikaciju i integritet podataka.

Ukoliko su poruke preduge, korištenje kriptiranja za potpisivanje cijele poruke postaje veoma nepraktično. Velika dužina poruke iziskuje dosta resursa i troši puno vremena na kriptiranje. Zato se uvodi mogućnost da se umjesto potpisivanja cijele poruke, potpisuje samo njen sadržaj. Pošiljatelj formira skraćenu verziju poruke, odnosno njen sadržaj kojeg potpisuje i šalje dalje komunikacijskim kanalom. Osoba koja primi tako skraćenu poruku provjerava njen potpis. Svaka promjena u izvornoj poruci izaziva i promjenu u sadržaju, što se odražava i na promjenu potpisa. Na taj način se minimizira mogućnost zlouporabe. Sadržaj poruke se kreira pomoću hash-funkcija za sažimanje.

2.1 Hash-funkcije

Kriptografske funkcije za izračunavanje sažetka poruke (hash-funkcije) igraju temeljnu ulogu u modernoj kriptografiji. Hash-funkcija ili hash-algoritam je funkcija za sažimanje i identificiranje podataka. Takav sažetak naziva se hash-vrijednost ili sažetak, a proces izračunavanja te vrijednosti naziva se hashiranje (*eng. hashing*).

Hash-funkcija h nizovima znakova proizvoljne konačne duljine pridružuje nizove znakova fiksne duljine.

Osnovna ideja hash-funkcija je dobivanje sažetog predstavnika ulaznog podatka koji se koristi kao poistovjećenje tog podatka. Jedna od najvažnijih primjena kriptografskih hash-funkcija je očuvanje integriteta poruke. Računanjem sažetaka prije i poslije prijenosa poruke preko mreže možemo utvrditi da li je poruka mijenjana. Glavno svojstvo hash-funkcija je determinističnost, što znači da za dva različita sažetka dobivena istom funkcijom mora vrijediti i da su ulazi bili različiti.

Kriptografske hash-funkcije se smatraju nesigurnima ako je moguće pronaći ulazni podatak za koji algoritam daje traženu hash-vrijednosti ili ako je moguće naći dva različita podatka koji će nakon hashiranja dati isti sažetak. Ako bi napadač bio u mogućnosti napraviti bilo što od prethodno navedenog, sigurnost podataka bi bila ugrožena.

Sada ćemo navesti neke od poželjnih kriptografskih osobina koje bi svaka hash-funkcija trebala posjedovati:

- Jednosmjernost:
  Hash-funkcija $h : \{0,1\}^* \to \{0,1\}^n$ je jednosmjerna ako danu hash-vrijednost $d \in \{0,1\}^n$ nije moguće naći ulaznu poruku $m \in \{0,1\}^*$ takvu da je $h(m) = d$.

- Slaba otpornost na koliziju:
  Kolizija predstavlja situaciju u kojoj za dva različita ulaza, hash-funkcija izračuna identične izlaze. Takva situacija se javlja kao posljedica fiksne dužine hash vrijednosti, a varijabilne dužine ulaza. Budući da sažetak uvijek ima istu duljinu, takvih postoji samo ograničen broj, dok ulaza ima beskonačno mnogo. Na taj način jednom hash sažetku može biti pridruženo više ulaza.

  Hash-funkcija $h : \{0,1\}^* \to \{0,1\}^n$ je slabo otporna na koliziju ako za zadanu poruku $m_1 \in \{0,1\}^*$ i zadanu hash-vrijednost $d \in \{0,1\}^n$ takvu da je $h(m_1) = d$ nije moguće naći drugu ulaznu poruku $m_2 \in \{0,1\}^*$ takvu da je $h(m_1) = h(m_2) = d$.

- Jaka otpornost na koliziju:
  Za hash-funkciju $h : \{0,1\}^* \to \{0,1\}^n$ kažemo da je jako otporna na koliziju ako nije moguće naći dvije poruke m_1 i $m_2 \in \{0,1\}^*$ $(m_1 \neq m_2)$ takve da je $h(m_1) = h(m_2)$.

Kada kažemo "nije moguće", mislimo na to da se zahtijeva napor daleko veći od raspoloživih resursa.

Primijetimo da je jaka otpornost na kolizije jači zahtjev od slabe otpornosti. U definiciji slabe otpornosti, jedna od poruka je fiksirana, dok kod jake otpornosti na kolizije obje poruke mogu biti izabrane na proizvoljan način. Dakle, ako hash-funkcija posjeduje jaku otpornost na kolizije, onda ima i slabu otpornost na kolizije, dok obratno ne vrijedi. Od kriptografskih hash-funkcija se uglavnom očekuje da se ponašaju kao slučajna funkcija, odnosno da je nemoguće predvidjeti bilo koji bit izlaza za dani ulaz. Dakle, ulazi koji se razlikuju u samo jednom bitu imaju u potpunosti različite hash-vrijednosti. Ovo se još zove *efekt lavine*, što znači da i male promjene ulaza izazivaju značajne promjene na izlazu.

Na najvišoj razini, hash-funkcije možemo podijeliti u dvije velike klase: hash-funkcije bez ključa koje za ulazni parametar imaju samo poruku, i hash-funkcije s ključem koje za ulaz uz poruku imaju i tajni ključ.

Većina hash-funkcija bez ključa su dizajnirane kao iterativni procesi koji hashiraju ulazne podatke proizvoljne duljine na način da obrađuju uzastopne ulazne blokove koji su fiksne veličine. Hash ulaz m koji je proizvoljne konačne duljine, podijeljen je na n-bitne blokove m_i fiksne duljine. Ako zadnji blok nema dovoljnu duljinu, nadopunjuje se dodatnim bitovima.

Neka je f hash-funkcija fiksne veličine i neka h_i označava međurezultat poslije faze i. Iterativni proces za hash-funkciju s ulazom $m = m_1 m_2 ... m_s$ može se modelirati na sljedeći način:

$$h_0 = IV; \quad h_i = f(h_{i-1}, m_i), \quad 1 \leq i \leq s; \quad h(m) = g(h_s).$$

h_i predstavlja n-bitno ulančavanje između faze $i-1$ i faze i, a h_0 je vektor inicijalizacije (početna vrijednost) čija je duljina jednaka duljini bloka. Funkcija g je funkcija transformacije, najčešće identiteta, $g(h_s) = h_s$.

MD4 algoritam

MD4 (*eng. Message Digest algorithm 4*) je 128-bitna hash-funkcija. Dizajnirao ju je Ronald Rivest 1990. godine s ciljem da njeno "slamanje" zahtijeva brute-force trud. To znači da je za pronalazak poruka sa istom hash-vrijednosti potrebno oko 2^{64} operacija, a za pronalazak poruke koja daje unaprijed određenu hash-vrijednost potrebno oko 2^{128} operacija. Danas je poznato da MD4 ne ispunjava te zahtjeve budući da su pronađene kolizije u 2^{20} računanja funkcije kompresije. Zbog toga se MD4 ne preporuča za korištenje kao hash-funkcija s jakom otpornošću na koliziju. Unatoč navedenim nedostacima, ovaj algoritam je imao veliki utjecaj na razvoj drugih hash-funkcija poput MD5 i SHA-1.

Pretpostavimo da nam je za ulaz dana poruka m duljine b bitova, $b \geq 0$. Koraci algoritma su sljedeći:

1. korak

   Proširimo poruku m tako da njena duljina u bitovima plus 64 bude djeljiva sa 512. Poruka se uvijek proširuje neovisno o tome je li uvjet djeljivosti zadovoljen. Proširivanje se vrši tako da bit '1' dodamo na kraj poruke, a zatim dodajemo nule sve dok duljina poruke plus 64 bita nije djeljiva sa 512.

2. korak

   Duljinu b orginalne poruke m prikažemo sa 64 bita koja dodamo na kraj proširene poruke iz prethodnog koraka. Ako je orginalna poruka dulja od 2^{64} znakova, tada se na kraj poruke dodaje samo nižih 64 bita. Bitovi se dodaju kao dvije 32-bitne riječi na način da se prvo doda riječ koja sadrži niže bitove, a zatim riječ koja sadrži više bitove. Označimo sa l broj 512-bitnih blokova u rezultirajućoj poruci. Tako formatirani ulaz se sastoji od $16l$ 32-bitnih riječi $m = m_0 m_1 ... m_{16l-1}$.

3. korak: Inicijalizacija MD međuspremnika

   Za izračunavanje sažetka poruke koristimo spremnik od četiri 32-bitne riječi (A, B, C, D). Njihove početne vrijednosti su:

   A: 01 23 45 67

   B: 89 ab cd ef

   C: fe dc ab 98

   D: 76 54 32 10

4. korak: Procesiranje poruke u blokovima od 16 riječi

   Prvo definiramo 3 pomoćne funkcije f, g, h koje imaju za ulaz tri 32-bitne riječi, a kao izlaz daju jednu riječ duljine 32 bita.

   $f(u, v, w) = uv \vee \neg uw$

   $g(u, v, w) = uv \vee uw \vee vw$

   $h(u, v, w) = u \oplus v \oplus w$

   Za svaki i od 0 do $l - 1$ izvršavamo sljedeće:

   Sačuvamo vrijednosti od A, B, C, D u posebnim varijablama

   $$AA = A$$

   $$BB = B$$

   $$CC = C$$

   $$DD = D$$

Kopiramo i-ti blok od 16 32-bitnih riječi u privremeni spremnik:

$$X[j] \leftarrow m_{16i+j}, \quad 0 \leq j \leq 15$$

.

Zatim ih obradimo u sljedeća tri kruga:

- 1. krug
  Označimo sa $[ABCD\ k\ s]$ operaciju:

  $$A = (A + f(B, C, D) + X[k]) \quad \ll \quad s,$$

  gdje smo sa '$\ll$' označili rotaciju ulijevo za s bitova.
  Izvršimo sljedećih 16 operacija:
  $[ABCD\ 0\ 3]$ $[DABC\ 1\ 7]$ $[CDAB\ 2\ 11]$ $[BCDA\ 3\ 19]$
  $[ABCD\ 4\ 3]$ $[DABC\ 5\ 7]$ $[CDAB\ 6\ 11]$ $[BCDA\ 7\ 19]$
  $[ABCD\ 8\ 3]$ $[DABC\ 9\ 7]$ $[CDAB\ 10\ 11]$ $[BCDA\ 11\ 19]$
  $[ABCD\ 12\ 3]$ $[DABC\ 13\ 7]$ $[CDAB\ 14\ 11]$ $[BCDA\ 15\ 19]$
- 2. krug
  Označimo sa $[ABCD\ k\ s]$ operaciju:

  $$A = (A + g(B, C, D) + X[k]) + 5a827999 \quad \ll \quad s$$

  Vrijednost 5a827999 je heksadecimalna 32-bitna konstanta koja predstavlja drugi korijen iz 2.
  Izvršimo sljedećih 16 operacija:
  $[ABCD\ 0\ 3]$ $[DABC\ 4\ 5]$ $[CDAB\ 8\ 9]$ $[BCDA\ 12\ 13]$
  $[ABCD\ 1\ 3]$ $[DABC\ 5\ 5]$ $[CDAB\ 9\ 9]$ $[BCDA\ 13\ 13]$
  $[ABCD\ 2\ 3]$ $[DABC\ 6\ 5]$ $[CDAB\ 10\ 9]$ $[BCDA\ 14\ 13]$
  $[ABCD\ 3\ 3]$ $[DABC\ 7\ 5]$ $[CDAB\ 11\ 9]$ $[BCDA\ 15\ 13]$
- 3. krug
  Označimo sa $[ABCD\ k\ s]$ operaciju:

  $$A = (A + h(B, C, D) + X[k]) + 6ed9eba1 \quad \ll \quad s$$

  Vrijednost 6ed9eba1 je heksadecimalna 32-bitna konstanta koja predstavlja drugi korijen iz 3.
  Izvršimo sljedećih 16 operacija:

$[ABCD\ 0\ 3]$ $[DABC\ 8\ 9]$ $[CDAB\ 4\ 11]$ $[BCDA\ 12\ 15]$
$[ABCD\ 2\ 3]$ $[DABC\ 10\ 9]$ $[CDAB\ 6\ 11]$ $[BCDA\ 14\ 15]$
$[ABCD\ 1\ 3]$ $[DABC\ 9\ 9]$ $[CDAB\ 5\ 11]$ $[BCDA\ 13\ 15]$
$[ABCD\ 3\ 3]$ $[DABC\ 11\ 9]$ $[CDAB\ 7\ 11]$ $[BCDA\ 15\ 15]$

Na kraju svaki registar uvećamo za vrijednost koju je imao na početku izvršavanja ovog bloka programa:

$A = A + AA$

$B = B + BB$

$C = C + CC$

$D = D + DD$

5. korak: Kraj

   Sažetak poruke je $ABCD$.

Primjer 1. *Sada navodimo primjere hash-vrijednosti za prethodno opisani algoritam MD4:*

- *MD4("") = 31d6cfe0d16ae931b73c59d7e0c089c0*
- *MD4("autentificirana enkripcija") = 68a48a409ca060cb5ad074165f23939a*
- *MD4("abcc") = 7bc4cad0a8abf618af464e7a0fb075ae*
- *MD4("abcd") = 41decd8f579255c5200f86a4bb3ba740*

Iz gornjih primjera je vidljivo da se prilikom promjene samo jednog slova ulazne poruke, pripadne hash-vrijednosti u potpunosti razlikuju.

2.2 MAC

Pružanje načina provjere integriteta informacije koja se prenosi je primarna potreba u svijetu otvorenog računarstva i komunikacije. Mehanizme koji pružaju takvu provjeru integriteta temeljenu na tajnom ključu uobičajeno zovemo kodovi za autentifikaciju poruke (MAC, *eng. Message Authentication Code*). Tipično, MAC-ovi se koriste između dvije strane koje dijele tajni ključ kako bi provjerili valjanost informacije izmijenjene između tih strana.

MAC je algoritam za generiranje kratkog niza podataka koji se koristi za provjeru autentičnosti poruke. Sastoji se od funkcije za izračunavanje sažetka poruke i tajnog

ključa koji kao rezultat daju MAC-oznaku. MAC-oznaka predstavlja niz bitova koji se dodaju na orginalnu poruku da bi se osigurala autentifikacija pošiljatelja i integritet poruke. Kodovi za autentifikaciju poruka se računaju i provjeravaju koristeći isti ključ, tako da ga samo oni kojima je poruka namijenjena mogu provjeriti. Kada jedan član komunikacije želi poslati poruku m drugom članu, on generira MAC-oznaku t baziranu na poruci i djeljenom ključu te šalje poruku m zajedno sa oznakom t primatelju. Nakon što primi (m, t) primatelj verificira je li t ispravna oznaka poslane poruke m. Osnovna pretpostavka je da će MAC-računanje vratiti netočan rezultat ako je poruka izmijenjena tijekom prijenosa.

Definicija 2.2.1. *[2] Kod za autentifikaciju poruka (MAC) je trojka polinomnih algoritama* $(\mathrm{Gen}, \mathrm{Mac}, \mathrm{Vrfy})$ *takvih da:*

1. *Algoritam za generiranje ključa* Gen *uzima kao ulaz* 1^n *i vraća ključ* k *t.d.* $|k| > n$.

2. *Algoritam za generiranje MAC-oznake* Mac *uzima kao ulaz ključ* k *i poruku* $m \in \{0,1\}^*$ *i vraća oznaku* t. *Pišemo:* $t \leftarrow \mathrm{Mac}_k(m)$.

3. *Algoritam za verifikaciju* Vrfy *uzima kao ulaz ključ* k, *poruku* m *i MAC-oznaku* t *i vraća bit* b, $b = \mathrm{Vrfy}_k(m, t)$. *Ako je* $b = 1$, *onda je poruka ispravna, a ako je* $b = 0$ *onda je poruka izmijenjena u prijenosu.*

Želimo da za svaki n, *za svaki* k *izlaz od* $\mathrm{Gen}(1^n)$ *i za svaki* $m \in \{0,1\}^*$, *vrijedi* $\mathrm{Vrfy}_k(m, \mathrm{Mac}_k(m)) = 1$.

Ako je $(\mathrm{Gen}, \mathrm{Mac}, \mathrm{Vrfy})$ *takav da je za svaki* k *izlaz od* $\mathrm{Gen}(1^n)$ *algoritam* Mac *definiran samo za poruke* $m \in \{0,1\}^{l(n)}$ *(i* Vrfy_k *vraća 0 za svaki* $m \notin \{0,1\}^{l(n)}$*), tada kažemo da je* $(\mathrm{Gen}, \mathrm{Mac}, \mathrm{Vrfy})$ *MAC fiksne duljine za poruke duljine* $l(n)$.

Sigurnost MAC-a

Intuitivna ideja koja stoji iza definicije sigurnosti MAC-a je da nijedan napadač u polinomom vremenu ne može generirati važeću MAC-oznaku za nijednu novu poruku koja prethodno nije bila poslana putem danog komunikacijskog kanala. Prilikom autentifikacije poruke napadač koji nadzire komunikacijski kanal može vidjeti sve poruke koje se šalju kao i njihove odgovarajuće MAC-oznake. Napadaču je omogućen pristup "crnoj kutiji" u kojoj se na temelju predane poruke generira MAC-oznaka. Sigurnost sheme smatramo narušenom ako napadač može dobiti poruku m i MAC-oznaku t tako da vrijedi:

- t je važeća MAC-oznaka poruke m

- napadač nije prethodno iz "crne kutije" zatražio MAC-oznaku za poruku m

Uspjeh prvog slučaja znači da je napadač uspio poslati važeću poruku jednoj od autoriziranih strana komunikacije. Drugi uvjet je potreban jer napadač može jednostavno kopirati prethodno poslanu poruku. Takav napad se zove *napad odgovorom* i njega ne smatramo narušavateljem sigurnost MAC-a.

Navesti ćemo i formalnu definiciju sigurnost za MAC. Prvo promotrimo sljedeći eksperiment:

Eksperiment autentifikacije poruke $\text{Mac-forge}_{\mathcal{A},\Pi}(n)$**:**

1. Ključ k se generira koristeći algoritan $\text{Gen}(1^n)$.
2. Protivniku $\mathcal{A}$ je dan ulaz 1^n i pristup "crnoj kutiji" u kojoj se generiraju MAC-oznake. Napadač vraća par (m, t). Neka je $\mathcal{Q}$ skup svih upita koje $\mathcal{A}$ šalje "crnoj kutiji".
3. Izlaz eksperimenta je 1 ako i samo ako vrijedi $\text{Vrfy}_k(m, t) = 1$ i $m \notin \mathcal{Q}$.

Definicija 2.2.2. *[2] Kod za autentifikaciju poruke* $\Pi = (\text{Gen}, \text{Mac}, \text{Vrfy})$ *je siguran ako za sve polinomijalne protivnike* $\mathcal{A}$ *postoji zanemariva funkcija* negl *takva da je*

$$P\big[\,\text{Mac-forge}_{\mathcal{A},\Pi}(n) = 1\big] \leq \text{negl}(n).$$

.

MAA algoritam

MAA je prilagođeni MAC algoritam za 32-bitne strojeve. Vrijeme izvršavanja algoritma je proporcionalno duljini ulazne poruke, a pokazalo se da je MAA dvostruko sporiji od MD4 algoritma.

Pretpostavimo da nam je za ulaz dana poruka m duljine $32j$ bitova, $1 \leq j \leq 10^6$ i 64-bitni tajni ključ $Z = Z[1]...Z[8]$. Koraci algoritma su sljedeći:

1. Proširimo ključ Z do šest 32-bitnih vrijednosti X, Y, V, W, S, T.

   - Prvo zamijenimo bilo koji bajt 0x00 ili 0xff u Z na sljedeći način:
     za svaki i od 1 do 8 izvršavamo sljedeće:
     $P \leftarrow 0$
     ako je $Z[i] =$ 0x00 ili 0xff, onda
     $$P \leftarrow P + 1,\ Z[i] \leftarrow Z[i] \vee P$$

- Neka J predstavlja prva četiri, a K zadnja četiri bajta od Z. Zatim izračunamo sljedeće:
  $X \leftarrow J^4(\mod 2^{32}-1) \oplus J^4(\mod 2^{32}-2)$
  $Y \leftarrow [K^5(\mod 2^{32}-1) \oplus K^5(\mod 2^{32}-2)](1+P)^2(\mod 2^{32}-2)$
  $V \leftarrow J^6(\mod 2^{32}-1) \oplus J^6(\mod 2^{32}-2)$
  $W \leftarrow K^7(\mod 2^{32}-1) \oplus K^7(\mod 2^{32}-2)$
  $S \leftarrow J^8(\mod 2^{32}-1) \oplus J^8(\mod 2^{32}-2)$
  $T \leftarrow K^9(\mod 2^{32}-1) \oplus K^9(\mod 2^{32}-2)$
- Obradimo tri dobivena para (X,Y), (V,W), (S,T) kako bi uklonili bajtove 0x00 i 0xff (isto kao u prvoj točki). Definiramo konstante:
  $A = 02040801$
  $B = 00804021$
  $C =$ bfef7fdf
  $D =$ 7dfefbff.

2. Inicijaliziramo vektor rotacije $v \leftarrow V$ i varijable: $H_1 \leftarrow X$, $H_2 \leftarrow Y$. Dodati blokove S i T u poruku $m = m_1...m_t$.

3. Obraditi svaki 32-bitni blok $m_i, 1 \leq i \leq t$ na sljedeći način:
   $v \leftarrow (v \hookleftarrow 1), \quad U \leftarrow (v \oplus W)$
   $t_1 \leftarrow (H_1 \oplus x_i) \times_1 (((H_2 \oplus x_i) + U) \vee A) \wedge C$
   $t_2 \leftarrow (H_2 \oplus x_i) \times_2 (((H_1 \oplus x_i) + U) \vee B) \wedge D$
   $H_1 \leftarrow t_1, \quad H_2 \leftarrow t_2,$
   gdje $\times_i$ označava množenje $\mod 2^{32}-i$ za $i = 1, 2$,
   $+$ označava zbrajanje $\mod 2^{32}$,
   $\hookleftarrow 1$ predstavlja rotaciju ulijevo za jedan bit.

4. Dobivena MAC-vrijednost je: $H = H_1 \oplus H_2$.

2.3 HMAC

HMAC algoritam izgrađen je oko algoritma za izračunavanje sažetka poruke. Kao i svaki drugi MAC algoritam koristi se za provjeru integriteta i autentičnost poruka. Svaka iterativna hash funkcija može se koristiti za izračunavanje HMAC oznake.

Definirajmo dva konstantna različita stringa ipad i opad:

$$\text{ipad} = \underbrace{00110110, 00110110, ..., 00110110}_{\text{ASCII znak 0x36 ponovljen b puta}}$$

$$\text{opad} = \underbrace{01011100, 01011100, ..., 01011100}_{\text{ASCII znak 0x5C ponovljen b puta}}$$

Koraci algoritma su sljedeći:

1. Simetrični ključ k se proširi nulama s lijeve strane tako da dužina ključa bude b bajtova, gdje je b ulazna širina bloka hash-funkcije.
2. Izračuna se vrijednost XOR funkcije između konstante *ipad* i proširenog ključa iz prvog koraka.
3. Nizu dobivenom u prethodnom koraku se dodaje poruka $M = (M_1, M_2, ..., M_n)$.
4. Na niz dobiven u prethodnom koraku primijeni se hash-funkcija h.
5. Na prošireni ključ iz prvog koraka se primjeni XOR operacija sa konstantom opad.
6. Nizu dobivenom u prethodnom koraku dodaje se niz iz 4. koraka.
7. Primijeni se hash funkcija h na dobiveni niz podataka iz prethodnog koraka.

Prethodnu konstrukciju možemo izraziti kao:

$$\text{HMAC}_k(M) = h[(k^+ \oplus opad)||h[(k^+ \oplus ipad)||x]]$$

2.4 CBC-MAC

CBC-MAC temelji se na CBC načinu rada algoritma enkripcije i vrlo je korišten u praksi.

Poruku m prvo podijelimo na blokove $m_i, i = 1, ..., n$. Pomoću tajnog ključa k i početne vrijednosti $c_0 = 0^n$ računamo prvu iteraciju MAC algoritma na sljedeći način:

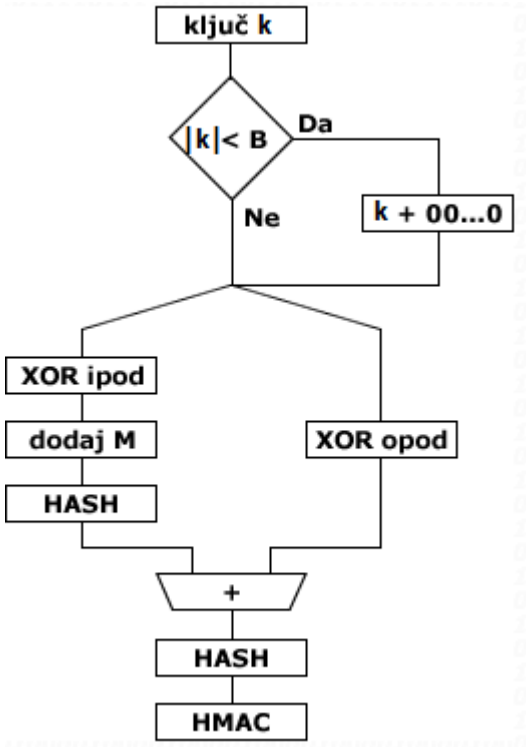
$$c_1 = e_k(m_i \oplus c_{i-1})$$

Za poruku $m = m_1 m_2 m_3 ... m_n$, MAC-oznaka t je izlaz posljednjeg kriptiranog bloka:

$$t = MAC_k(m) = c_n$$

Dužina izlaza je određena dužinom bloka šifre koja se ovdje koristi. Vrijednosti $c_1, c_2, c_3, ..., c_{n-1}$ se ne šalju, već one samo služe kao međurezultati u izračunavanju konačne MAC vrijednosti $t = c_n$.

Verifikacija obuhvaća ponavljanje operacija koje se koriste za generiranje MAC-oznake. Tako dobivenu MAC-oznaku (označimo ju sa t') uspoređujemo sa orginalnom



Slika 2.1: Struktura HMAC algoritma

MAC-oznakom t. Ako je $t' = t$ poruku smatramo ispravnom, inače su MAC-oznaka i/ili poruka promijenjeni tijekom prijenosa.

Zbog XOR miješanja, vrijednost posljednjeg kriptiranog bloka ovisi o svim blokovima podataka. Zbog korištenja algoritma enkripcije vrijednost posljednjeg kriptiranog bloka ovisi i o ključu. Prema tome, neispravna MAC-oznaka znači ili izmjene u podacima, odnosno povredu integriteta, ili korištenje pogrešnog ključa enkripcije što uzrokuje povreda autentičnosti.

Razlike u odnosu na CBC:

1. CBC-MAC je deterministčan (nema vektor inicijalizacije) što je ključno za njegovu sigurnost.

2. Kod CBC načina djelovanja svaki blok šifri $c_1, c_2, c_3, ..., c_n$ moramo iskoristiti da bismo mogli dekriptirati poruku. Kod CBC-MAC algoritma za verifikaciju koristimo samo rezultat zadnjeg bloka. Štoviše, kada bi CBC-MAC izmijenili tako da koristi sve blokove, narušili bismo njegovu sigurnost.

Poglavlje 3

Autentificirana enkripcija

Pokazali smo kako enkripcijom poruke možemo postići privatnost. Zatim smo pokazali kako pomoću kodova za autentifikaciju poruke možemo ostvariti autentifikaciju i integritet podataka. Međutim, često zahtijevamo i privatnost i integritet istovremeno. Mogli bismo pomisliti kako bilo koja kombinacija sigurne enkripcijske sheme i bilo kojeg sigurnog MAC-a osigurava ova dva svojstva. Nažalost, to nije uvijek tako.

Neka k_1 označava ključ za enkripciju i neka k_2 bude MAC ključ. Postoje tri uobičajena načina za kombiniranja šifriranja i autentifikacije poruke. To su:

- **Šifriraj-i-autentificiraj**

  U ovoj metodi šifiranje i autentifikacija poruke se odvijaju paralelno neovisno jedno o drugom, tj. za dani otvoreni tekst m, pošiljatelj šalje poruku (c, t) gdje je

  $$c \leftarrow \mathrm{Enc}_{k_1}(m) \quad i \quad t \leftarrow \mathrm{Mac}_{k_2}(m).$$

  Kod dekripcije primatelj prvo dešifrira šifrat c da dobije otvoreni tekst m i zatim verificira MAC-oznaku t. Ako je $\mathrm{Vrfy}_{k_2}(m, t) = 1$, primatelj vraća m. Inače, vraća $\perp$ što znači da poruka nije valjana.

- **Autentificiraj-pa-šifriraj**

  Prvo se izračunava MAC-oznaka, zatim se dobivena MAC-oznaka dodaje na otvoreni tekst m te se oni zajedno šifriraju. Dakle, pošiljatelj šalje poruku c dobivenu kao:

  $$t \leftarrow \mathrm{Mac}_{k_2}(m) \quad i \quad c \leftarrow \mathrm{Enc}_{k_1}(m||t),$$

  gdje smo sa '||' označili konkatenaciju (spajanje) dva stringa. MAC-oznaka t se šalje zajedno sa šifriranom porukom. Primatelj dešifrira c i verificira MAC-oznaku t na poruci m. Ako je $\mathrm{Vrfy}_{k_2}(m, t) = 1$, primatelj vraća m. Inače, vraća $\perp$.

- **Šifriraj-pa-autentificiraj**

  Poruka m se prvo šifrira i onda se MAC-oznaka t izračunava za šifriranu poruku. Poruka koja se šalje je par (c, t) gdje je

  $$c \leftarrow \mathrm{Enc}_{k_1}(m) \quad i \quad t \leftarrow \mathrm{Mac}_{k_2}(c).$$

  Primatelj prvo verificira MAC-oznaku t. Zatim, ako je MAC-oznaka validna, dekriptira šifrat c.

Sada ćemo analizirati svaku od navedenih metoda pod pretpostavkom da koriste proizvoljnu CPA-sigurnu shemu enkripcije i proizvoljan siguran MAC. Da bismo mogli analizirati sigurnost gornjih metoda, moramo prvo definirati što ćemo smatrati sigurnom komunikacijom. Modelirat ćemo pojam sigurnog komunikacijskog kanala i onda ćemo dokazati da dana metoda zadovoljava tu definiciju.

Neka je $\Pi_E = (\mathrm{Gen}_E, \mathrm{Enc}, \mathrm{Dec})$ proizvoljna enkripcijska shema i neka je $\Pi_M = (\mathrm{Gen}_M, \mathrm{Mac}, \mathrm{Vrfy})$ proizvoljan kod za autentifikaciju poruke. *Shema prijenosa poruke* $\Pi' = (\mathrm{Gen}', \mathrm{EncMac}', \mathrm{Dec}')$ dobivena kao kombinacija Π_E i Π_M je trojka algoritama takva da:

- Algoritam za generiranje ključa Gen' uzima kao ulaz 1^n, izvršava se $\mathrm{Gen}_E(1^n)$ i $\mathrm{Gen}_M(1^n)$ da se dobiju ključevi k_1 i k_2. Izlaz je ključ (k_1, k_2).

- Algoritam za prijenos poruke EncMac' uzima kao ulaz ključeve (k_1, k_2) i poruku m i vraća vrijednost c koja se dobije proizvoljnom kombinacijom Enc_{k_1} i Mac_{k_2}.

- Dekripcijski algoritam Dec' uzima kao ulaz ključeve (k_1, k_2) i vrijednost c i primjenjuje proizvoljnu kombinaciju Dec_{k_1} i Vrfy_{k_2}. Izlaz od Dec' je ili orginalna poruka m ili simbol $\perp$.

Želimo da za svaki n, za svaki par ključeva (k_1, k_2) izlaz od $\mathrm{Gen}'(1^n)$ i za svaki $m \in \{0,1\}^*$ vrijedi

$$\mathrm{Dec}'_{k_1,k_2}(\mathrm{EncMac}'_{k_1,k_2}(m)) = m$$

Primjetimo da Π' zadovoljava sintaksu simetrične enkripcije, ali ne zadovoljava sintaksu za MAC. Stoga uvodimo specifičnu definiciju. Promotrimo sljedeći eksperiment koji je definiran za shemu prijenosa poruke Π', napadača $\mathcal{A}$ i sigurnosni parametar n:

Eksperiment sigurnog prijenosa poruke $\mathrm{Auth}_{\mathcal{A},\Pi'}(n)$**:**

1. Ključ $k = (k_1, k_2)$ se generira koristeći funkciju $\mathrm{Gen}'(1^n)$.

2. Protivniku $\mathcal{A}$ je dan ulaz 1^n i pristup algoritmu za prijenos poruke EncMac'_k. Napadač vraća izlaz c. Neka je $\mathcal{Q}$ skup svih upita koje napadač šalje algoritmu za prijenos poruke.

3. Neka je $m := \text{Dec}'_k(c)$. Izlaz eksperimenta definiramo da je jednak 1 ako i samo ako je $m \neq \perp$ i $m \notin \mathcal{Q}$.

Definicija 3.0.1. *[2] Shema prijenosa poruke* Π *postiže autentificiranu komunikaciju ako za sve polinomijalne protivnike* $\mathcal{A}$ *postoji zanemariva funkcija* negl *takva da je*

$$P\big[\,\text{Auth}_{\mathcal{A},\Pi}(n) = 1\big] \leq \text{negl}(n)$$

Sada možemo definirati sigurnost sheme prijenosa poruke.

Definicija 3.0.2. *Shema prijenosa poruke* $\Pi' = (\text{Gen}', \text{EncMac}', \text{Dec}')$ *je sigurna ako je CCA-sigurna enkripcijska shema i ujedno postiže autentificiranu komunikaciju.*

Sada analiziramo sigurnost prethodno opisanih metoda.

Enkriptiraj-i-autentificiraj

Kao što smo već naveli, u ovoj metodi se enkripcija i autentifikacija odvijaju odvojeno. Za zadanu poruku m, vrijednost koja se prenosi je (c, t) gdje je

$$c \leftarrow \text{Enc}_{k_1}(m) \quad i \quad t \leftarrow \text{Mac}_{k_2}(m)$$

Ova kombinacija općenito nije sigurna jer ne osigurava privatnost. Budući da siguran MAC ne mora nužno podrazumijevati privatnost, moguće je da MAC-oznaka sadrži informacije o orginalnoj poruci. Naprimjer, ako je $(\text{Gen}_M, \text{Mac}, \text{Vrfy})$ siguran MAC, tada je i shema definirana kao $\text{Mac}'_k(m) = (m, \text{Mac}_k(m))$ također sigurna.

Autentificiraj-pa-enkriptiraj

Prvo se izračunava MAC-oznaka $t \leftarrow \text{Mac}_{k_2}(m)$, zatim se enkriptira $m||t$ i onda se rezultat $\text{Enc}_{k_1}(m \| \text{Mac}_{k_2}(m))$ prenosi putem komunikacijskog kanala.Pokazat ćemo da ni ova kombinacija nije nužno sigurna. Koristit ćemo sljedeću shemu enkripcije:

- Neka je funkcija $\text{Transform}(m)$ takva da svaku nulu iz m transformira u 00, a svaku jedinicu iz m transformira u 01 ili 10. Inverz ove funkcije parsira ovu poruku na parove bitova i onda mapira 00 u 0, a 01 ili 10 u 1. Ako se pojavi par 11, rezultat je $\perp$. Primjerice, $\text{Transform}^{-1}(1001) = 11$, a $\text{Transform}^{-1}(1011) = \perp$

- Neka je $\mathrm{Enc}_k(m) = \mathrm{Enc}'_k(\mathrm{Transform}(m))$, gdje je Enc' CTR način enkripcije.

Sada ćemo pokazati da metoda *Autentificiraj-pa-enkriptiraj* sa gornjom enkripcijskom shemom i proizvoljnim MAC-om nije sigurna u slučaju CCA napada. Napad je uspješan ako napadač zna je li dani šifrat validan, čak i ako ga ne može dešifrirati u potpunosti.

Promotrimo sljedeći CCA napad: Za zadani šifrat

$$c = \mathrm{Enc}'_{k_1}(\mathrm{Transform}(m||\,\mathrm{Mac}_{k_2}(m)))$$

napadač promijeni prva dva bita drugog bloka od c (prvi blok je početna vrijednost) i provjerava je li tako dobiveni šifrat validan. Ako je prvi bit orginalne poruke m jednak 1, modificirani šifrat je validan. Budući da je prvi bit poruke m jednak 1, znači da su prva dva bita od $\mathrm{Transform}(m)$ 01 ili 10, pa promjena tih bitova ne utječe na orginalnu poruku. Ovaj primjer pokazuje da metoda *Autentificiraj-pa-enkriptiraj* općenito nije sigurna.

Enkriptiraj-pa-autentificiraj

Poruka m se prvo enkriptira i onda se računa MAC iz šifrata. Poruka je par (c, t) gdje je

$$c \leftarrow \mathrm{Enc}_{k_1}(m) \quad i \quad t \leftarrow \mathrm{Mac}_{k_2}(c).$$

Teorem 3.0.3. *[2] Neka je Π_E CPA-sigurna shema enkripcije s privatnim ključem i neka je Π_M siguran kod za autentifikaciju poruke. Tada je kombinacija*

$$(\mathrm{Gen}', \mathrm{EncMac}', \mathrm{Dec}')$$

koja je dobivena koristeći enkriptiraj-pa-autentificiraj način sigurna shema prijenosa poruke.

3.1 Algoritmi autentificirane enkripcije

U prethodnom odjeljku smo opisali kako se kombinacijom dvaju algoritama sa dva zasebna tajna ključa može ostvariti povjerljivost i autentičnost poruke. Problem korištenja dva algoritma upravo rješava pojava algoritama autentifikacijske kriptografije, koja umjesto dva zasebna algoritma sa dva ključa koristi jedan algoritam s jednim ključem koji objedinjuje funkcije koje pružaju povjerljivost i integritet poruke.

- **GCM**

  GCM je algoritam autentificirane enkripcije dizajniran kako bi osigurao autentičnost i povjerljivost podataka. Objedinjuje dvije funkcije, autentifikacijsku enkripciju i autentifikacijsku dekripciju. Jedna od korisnih karakterika GCM algoritma je da podatak o dužini poruke nije zahtjevan unaprijed, već se dužina može računati u tijeku pristizanja i procesuiranja podataka.

  GCM koristi CTR način šifriranja. Šifra ima veličinu bloka 128 bitova poput AES-a. Dakle, prvom brojaču se pridruži neka inicijalna vrijednost, a zatim se ostali brojači x_i povećaju za 1. Niz šifrata c_i se dobiva na sljedeći način:

  $$c_i = m_i \oplus e_k(x_i).$$

  Za autentifikaciju je potrebno prvo generirati potključ autentifikacije $H = \mathrm{Enc}_k(0)$. Zatim se računaju posredni parametri autentifikacije g_i na sljedeći način:

  $$g_0 = H$$

  $$g_i = (g_{i-1} \oplus c_i)xH, \quad 1 \leq i \leq n$$

  Na kraju dobivamo MAC-oznaku $t = (g_n \times H) \oplus \mathrm{Enc}_k(c_0)$. Množenje je u $GF(2^{128})$ s ireducibilnim polinomom $P(x) = x^{128} + x^7 + x^2 + x + 1$.

- **OCB**

  OCB (*eng. Offset Codebook*) je način rada autentifikacijske enkripcije za blokovske kriptografske algoritme. Dizajniran je da istovremeno pruža povjerljivost i autentičnost podataka. OCB postiže autentificiranu enkripciju u približno jednakom vremenu u kojem CTR postiže samo privatnost što ga čini iznimno efikasnom metodom zaštite podataka. Trenutno postoje 3 verzije: OCB1, OCB2 i OCB3.

  OCB enkripcija se odvija tako što se računa XOR svakog bloka poruke i brojača, prije i nakon enkripcije. Blokovi se mogu enkriptirati i dekriptirati paralelno, neovisno jedni o drugima. Autentifikacija izvora se postiže tako što se računa XOR kontrolne sume izračunate nad blokovima poruke i MAC-a izračunatog nad dodatnim informacijama, pa se tako dobiveni rezultat enkriptira. OCB je veoma efikasan, ali je zaštićen licencom, što spriječava njegovo šire korištenje.

Bibliografija

[1] A. Dujella i M. Maretić, *Kriptografija*, Element, 2007.

[2] J. Katz i Y. Lindell, *Introduction to Modern Cryptography*, CRC Press, 2007.

[3] D.R. Stinson, *Cryptography. Theory and practice*, CRC Press, 2006.

Sažetak

Dva najvažnija cilja kriptografije su pružanje povjerljivosti i autentičnosti, odnosno integriteta poruka koje se šalju putem nesigurnog komunikacijskog kanala. Povjerljivost osigurava da je poruka čitljiva samo osobama kojima je namijenjena, dok za ostale treba biti neupotrebljiva. Osiguravanje povjerljivost se postiže metodama enkripcije pomoću kojih se izvorni tekst pretvara u šifrirani tekst, te se takav šalje komunikacijskim kanalom. Integritet podataka je svojstvo koje osigurava da podaci nisu izmijenjeni ili krivotvoreni od strane neovlaštenih entiteta. Integritet se najčešće postiže pomoću kodova za autentifikaciju poruka, tzv. MAC-ova. MAC je algoritam za generiranje kratkog niza podataka koji se dodaje na orginalnu poruku kako bi se osigurala autentifikacija pošiljatelja i integritet poslane poruke.

Mnoge aplikacije koriste algoritme enkripcije i MAC zajedno, svaki algoritam sa svojim ključem, kako bi ostvarile navedene ciljeve. Međutim, ne osigurava svaka kombinacija sigurne enkripcijske sheme i bilo kojeg sigurnog MAC-a tražena svojstva. Pokazalo se da jedino metoda *šifriraj-pa-autentificiraj* u kojoj se poruka prvo šifrira, a zatim se izračunava MAC za tako šifriranu poruku, osigurava oba svojstva.

U novije vrijeme, javlja se ideja za pružanjem povjerljivosti i integriteta podataka koristeći samo jedan algoritam s jednim ključem. Problem korištenja dva algoritma upravo rješava pojava autentificirane enkripcije, koja umjesto dva zasebna algoritma sa dva ključa koristi jedan algoritam s jednim ključem koji objedinjuje funkcije koje pružaju povjerljivost i integritet poruke.

Summary

The two most important goals of cryptography are providing confidentiality, authenticity and integrity of messages that are sent through insecure communication channel. Confidentiality ensures that the message is readable only by the original recipient, while for the others it's ineligible. Confidentiality is achieved by encryption schemes where the plaintext is converted into ciphertext and in this form it is sent through communication channel. Data integrity is a feature that ensures that data is not altered or forged by unauthorized parts of communication. Integrity is usually achieved by message authentication codes, MACs. MAC is an algorithm for generating a short data string which is added to the original message to ensure the authentication of the sender and the integrity of the sent message.

Many applications use encryption algorithms and MAC together, each algorithm with its own key, to accomplish these goals. However, not every combination of secure encryption schemes and a secure message authentication code can provide required properties. It can be shown that only the method textit encrypt-then-authenticate, where the message is first encrypted and then a MAC tag is computed over the encrypted message, provides both properties.

More recently, the idea of providing confidentiality and data integrity is emerging using only one single-key algorithm. The problem of using two algorithms is solved with the phenomenon of authenticated encryption, which instead of two separate algorithms uses one single-key algorithm that combines functions that provide both confidentiality and integrity of the message.

Životopis

Rođena sam 3.12.1991.g. u Splitu. Osnovnu školu sam pohađala u Janjini na poluotoku Pelješcu. Tijekom tog razdoblja, sudjelovala sam na nekolicini matematičkih natjecanja gdje se razvio moj interes za matematikom. Nakon osnovne škole, upisala sam Opću gimnaziju u Dubrovniku koju sam završila sa odličnim uspjehom.

Godine 2010. sam upisala Preddiplomski studij matematike na Prirodoslovno-matematičkom fakultetu u Zagrebu koji sam završila 2015. godine i stekla zvanje sveučilišne prvostupnice matematike. Iste godine sam upisala Diplomski studij Matematike i računarstva na Prirodoslovno-matematičkom fakultetu u Zagrebu. Od rujna 2017. godine radim kao programer u informatičkoj firmi Laus CC u Dubrovniku, gdje se nadam da ću i nastaviti usvajati nova znanja i vještine.