SIGURNOSNI MEHANIZMI U APLIKACIJAMA ZA KOMUNIKACIJU

Režić, Zvonimir Sokol

SVEUČILIŠTE U SPLITU

SVEUČILIŠNI ODJEL ZA STRUČNE STUDIJE

Specijalistički diplomski stručni studij Elektrotehnike

ZVONIMIR SOKOL REŽIĆ

ZAVRŠNI RAD

SIGURNOSNI MEHANIZMI U APLIKACIJAMA ZA KOMUNIKACIJU

Split, veljača 2023.

SVEUČILIŠTE U SPLITU

SVEUČILIŠNI ODJEL ZA STRUČNE STUDIJE

Specijalistički diplomski stručni studij Elektrotehnike

Predmet: Sigurnost mreža i usluga

ZAVRŠNI RAD

Kandidat: Zvonimir Sokol Režić

Naslov rada: Sigurnosni mehanizmi u aplikacijama za komunikaciju

Mentor: Marko Meštrović, mag. ing. el., pred.

Split, veljača 2023.

Sadržaj

Sažetak

Sigurnosni mehanizmi u aplikacijama za komunikaciju

Sigurnosni mehanizmi unutar aplikacija za komunikaciju započeli su razvojem kriptografije, znanstvene discipline kojoj je zadatak razvijati metode slanja poruka ili informacija u obliku koji će biti razumljiv samo pošiljatelju ili primatelju te poruke. Na taj način, informacije sadržane u porukama su zaštićene od čitanja i mijenjanja trećih strana. Razvojem aplikacija za komunikaciju koje koristimo putem računala, tableta i mobilnih uređaja za razmjenjivanje poruka, slika i datoteka između korisnika, želimo da privatnost sudionika u razgovoru bude na što višem nivou. E2EE (engl. end to end encryption) metoda omogućuje kriptiranje sadržaja koji se nalazi unutar poruke (tekstualni zapis, videozapis, slika, audiozapis itd.), odnosno da sadržaj sigurno stigne do odredišta bez da ga itko, čak ni pružatelj komunikacijske usluge, pročita ili promijeni. Korištenje end to end enkrcipije započinje 2014. godine unutar aplikacija Signal i WhatsApp te se razvija i dalje, a s povećanjem snage računala, metode postaju sve naprednije u čitanju naših poruka. Danas većina komunikacijskih protokola koristi SHA-256 funkciju koja nudi dovoljno zaštite da sačuva naše informacije od trećih strana.

Ključne riječi: Kriptografija, E2E enkripcija, WhatApp, Signal, SHA-256

Summary

Security mechanisms in communication applications

Safety mechanisms implemented in communication applications started with development of cryptography, practice and study of techniques which will develop methods for sending messages or information which will be readable only to sender and receiver of the message. In that case information that was inside the message were kept secured from reading or changing by third party. Development of communication applications which are used over computers, tablets and mobile phones through we exchange messages, pictures, files between users, we want that privacy between users be on highest level possible. E2EE (end to end encryption) method enables encryption of content that is kept inside message (text, video, pictures, audio record, etc.) to safely comes to receiver without even the service provider can not see or change sent message. End to end encryption started to be used in 2014 by Signal and WhatsApp applications, its developing constantly because method to intercept or change our messages are getting stronger and stronger as power of computers rise. Today most of communications protocols are using SHA-256 function, which offers enough of protection to keep our information safe from third party.

Key words: Cryptography, E2E encryption, WhatsApp, Signal, SHA-256

1. UVOD

Zadatak ovog rada je obrada sigurnosnih mehanizama u aplikacijama za komunikaciju koje omogućavaju siguran prijenos poruke od pošiljatelja do primatelja bez da itko, osim njih, može pročitati ili promijeniti poruku. Kriptiranje informacija koje se nalaze u poruci omogućeno je pomoću simetričnog i nesimetričnog načina kriptiranja, čije će prednosti i mane biti obrađene u ovom radu. E2E enkripcija je namijenjena da zaštiti podatke koji se prenose, na način da se sačuva izvorni oblik poruke i spriječi čitanje podataka od trećih strana. Svaka poslana poruka je digitalno potpisana, što štiti izvornu autentičnost poruke, te će svaka eventualna izmjena sadržaja nakon slanja poruke biti detektirana. Osobe koje nisu ni pošiljatelj ni primatelj poruke mogu presresti poruke, ali one će im biti prikazane tek kao skup alfanumeričkih znakova iz kojih je nemoguće pročitati izvorni sadržaj. MITM (engl. man in the middle) napadi, jedan su od načina koji se koristi da bi se poruke čitale i promijenile. MITM ne pokušava dešifrirati algoritam, već „glumi“ primatelja poruke tako da svoj javni ključ razmjenjuje s pošiljateljem, nakon čega će poruka biti kriptirana ključem koji je poznat napadaču. U ovom radu bit će opisan razvoj E2E enkripcije, OTR protokola te razne vrste napada kojem su, navedeni mehanizmi, izloženi, kao i aplikacije u kojima se koriste te njihov način primjene. Dolje navedena slika predstavlja primjenu E2E enkripcije kod najpoznatije aplikacije za komunikaciju danas, WhatsApp.

Slika 1.1. Primjena E2E enkripcije u WhatsApp aplikaciji [8]

2. KRIPTOGRAFIJA

Kriptografija je prevođenje jasnog (razgovijetnog) teksta u kriptirani (nerazgovijetan) tekst. Kriptirani tekst može dešifrirati samo prijemna strana, no ako u međuvremenu treća strana ipak dospije do kriptiranog dijela, tekst postaje besmislen te je nemoguće dobiti njegov izvorni oblik. Kriptografija se koristi stoljećima i to najviše u vojne i diplomatske svrhe (očuvanje vojnih i državnih tajni). Razni su državni i vojni vrhovi koristili brojne metode kriptiranja teksta tako da neprijatelj, ukoliko dođe do poslane poruke, ne sazna način kriptiranja izvornog teksta te samim time i ne uspije dekriptirati navedenu poruku. S vremenom, pojedine su metode kriptiranja zastarjele te se javila potreba za novim metodama i načinima šifriranja teksta. Kriptografija kao znanstvena disciplina počinje s razvojem 1949. godine od strane C.E. Shannona koji je utemeljio njene osnove. Ona je danas jedna od najvažnijih znanstvenih disciplina za očuvanje privatnosti korisnika kod komunikacije komunikacijskim mrežama. Primjene kriptografije su mnogobrojne, a neke od njih su: očuvanje osobnih informacija, bankovnih podataka, poslovnih podataka, vojnih i diplomatskih tajni itd. Danas se većina informacija prenosi putem računala i mobilnih uređaja kao niz bitova ili znakova (znak je predstavljen skupom bitova) koji imaju normiran način kodiranja. Sav sadržaj koji se nalazi unutar poruke kriptira se na način da je svaki abecedni znak, broj ili interpunkcijski znak, predstavljen jednim bajtom koji sadrži osam bitova, te se s navedenim bajtom može prikazati 256 različitih kombinacija, a samim time i 256 različitih znakova. Moderno kriptiranje razdjeljuje nizove bitova proizvoljne duljine te ih kriptira raznim algoritmima koji garantiraju tajnost sadržaja. Pod pojmom algoritma za kriptiranje i dekriptiranje teksta koriste se matematičke funkcije (ključevi) čiji parametri prikazuju konkretan način kriptiranja datog teksta. Metode kriptografije možemo podijeliti na simetrične i asimetrične.

Moderna kriptografija brine o četiri stavke, a to su: tajnost, integritet, poricanje i autentifikacija. Tajnost garantira da će samo pošiljatelj i primatelj poruke moći razumjeti njen sadržaj, dok nam integritet omogućava da detektiramo promjene u sadržaju poruke koje su se dogodile prilikom prijenosa putem mreže. Nadalje, poricanje ne dopušta pošiljatelju i kreatoru poruke da poriče izvor i pošiljatelja poruke, dok autentifikacija potvrđuje identitet pošiljatelja i primatelja te izvor i odredište poruke. Kriptografske *hash* funkcije pretvaraju različite količine podataka primjenom algoritama u podatke s fiksnom duljinom, a da bi bila korisna u kriptografiji, iz *hash* funkcije mora biti vrlo teško ili

gotovo nemoguće izvući originalnu poruku. Nekada je mnogo važnije potvrditi integritet poruke, nego poruku držati tajnom. Najuobičajenija *hash* funkcija je SHA-1 koja proizvodi 160 bitova dugu *hash* vrijednost za podatkovne vrijednosti do 2**64 bita, dok je funkcija SHA-256 najzaštićenija. SHA-256 je dio SHA-2 familije algoritama. Broj 256 označava finalnu vrijednost koja se dobije nakon što se tekstovi različitih veličina provuku kroz SHA-256 algoritam. Bez obzira na duljinu ulaznog teksta, veličina izlazne *hash* vrijednosti je uvijek 256 bita.

Prednosti SHA-256 funkcije:

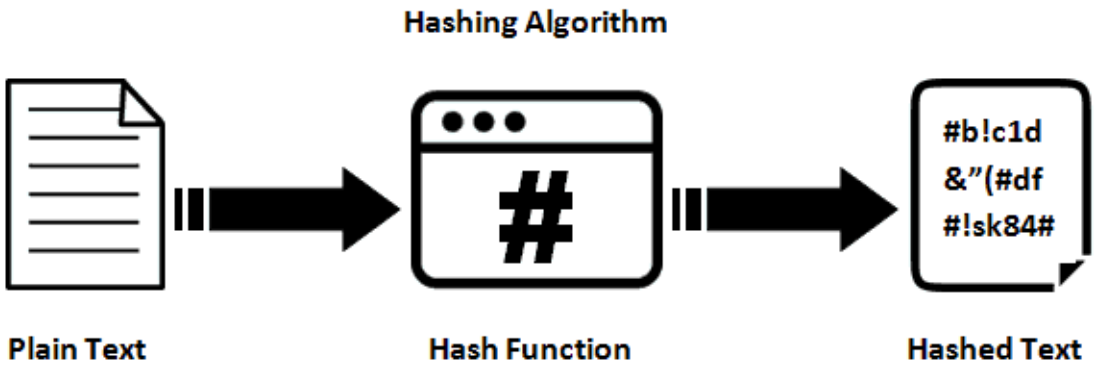
- Jednosmjerni algoritam, koji za rekonstruiranje ulazne vrijednosti treba potrošiti mnogo vremena i resursa. Napadač bi trebao pogoditi kombinaciju nula i jedinica iz 2256 različitih mogućnosti.
- Duljina izlazne vrijednosti, vrijednost *hash* funkcija za SHA-256 iznosi 256 bita, SHA-128 128 bita itd. Veće izlazne vrijednosti traže puno više resursa i memorije, ali nude i veću razinu zaštite.
- SHA-256 algoritam još nije probijen.
- Ako promijenimo samo jedno slovo na ulazu, na izlazu dobivamo drastično različit rezultat.

Proces dobivanja *hash* izlazne vrijednosti sastoji se od 5 segmenta:

1. Prvi se segment sastoji od dodavanja dodatnih bitova tako da originalna poruka bude 64 bita kraća od bilo kojeg višekratnika broja 512. Ako se operacija vrši po višekratniku 1024, tada poruka mora iznositi $1024 - 64 = 960$. No, ako poruka iznosi manje od 960 bitova, npr. 900 bitova, tada se preostalih 60 bitova puni proširenim nizom koji počinje s vrijednošću 1, dok su preostali bitovi do 960 ispunjeni nulama.
2. Drugi segment duljinu originalne poruke prikazuje u 64 bita te ih dodaje rezultatu dobivenom u prethodnom koraku.
3. Treći segment predstavlja inicijalizaciju spremnika, odnosno kreiranje 8 različitih vrijednosti za izračunavanje sažetka poruke te je potrebno kreirati 64 različita ključa, K[0] do K[63].
4. Nadalje, cijela se poruka rastavlja u više blokova i to po 512 bitova pojedinačno. Svaki se blok provodi kroz 64 operacije tako da izlazna vrijednost jedne operacije

bude ulazna vrijednost druge operacije (slika 2.1.). Dok je vrijednost K[i] u svim operacijama kreirana, vrijednost W[i] je drugi izlaz koji je individualno izračunat za svaki blok te je ovisan o broju ponavljanja koje se trenutno obrađuju.

5. Sa svakim ponavljanjem, izlaz jednog bloka služi kao ulaz drugog bloka te se operacija nastavlja sve dok se ne dobije 512 bitni blok, koji se smatra izlaznom *hash* funkcijom. Izlaz je duljine 256 bita, kako je i samo ime algoritma.

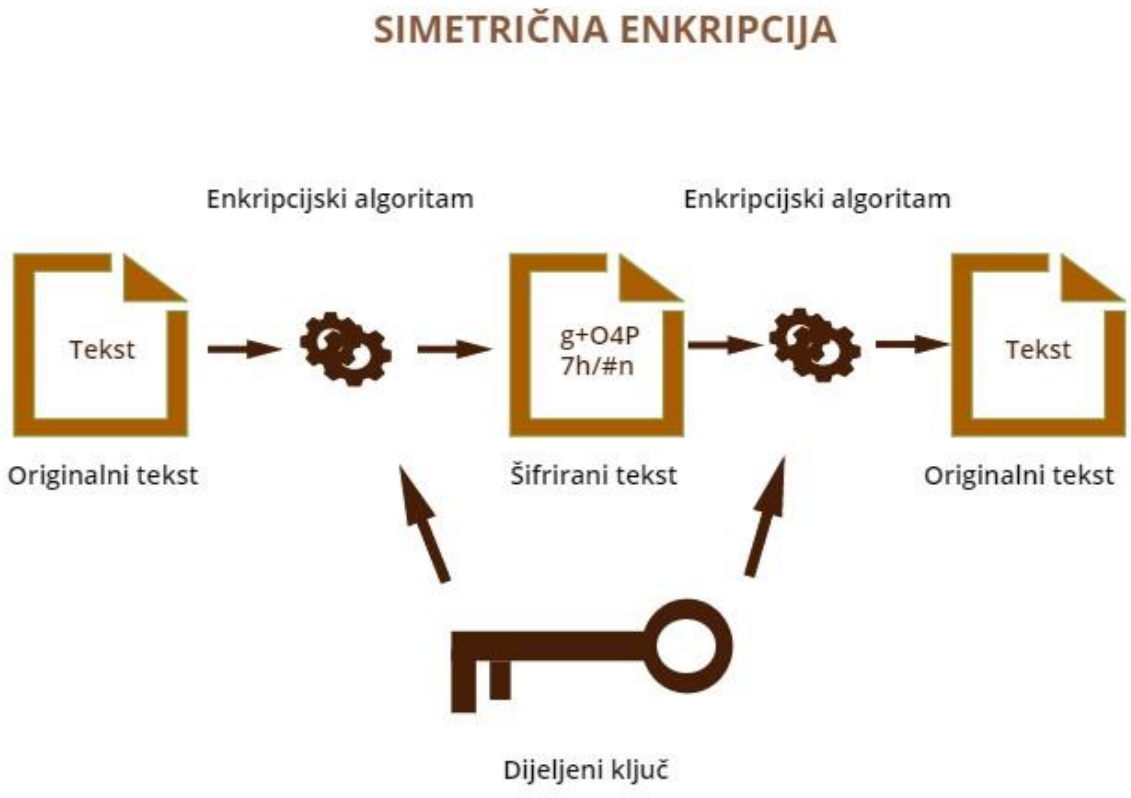


Slika 2.1. Proces *hash* algoritma [9]

2.1. Simetrični sustavi kriptografije

Kod simetričnih sustava kriptografije (tajni ključ) koriste se isti ključevi za kriptiranje i dekriptiranje podataka. Ključevi mogu biti identični ili ih može dijeliti jednostavna transformacija. Pošiljatelj i primatelj imaju identičnu kopiju ključa koju drže privatnom i ne dijele je ni sa kim te su jedine osobe koje mogu pročitati kriptiranu poruku. Korištenje identičnog ključa uvelike olakšava i ubrzava proces enkripcije, prvenstveno iz razloga što se koristi manje memorije, a takav tip enkripcije koristi se i kod kriptiranja veće količine podataka. Simetrični ključ radi na principu da kriptiramo bit po bit ili kompletne blokove informacija.

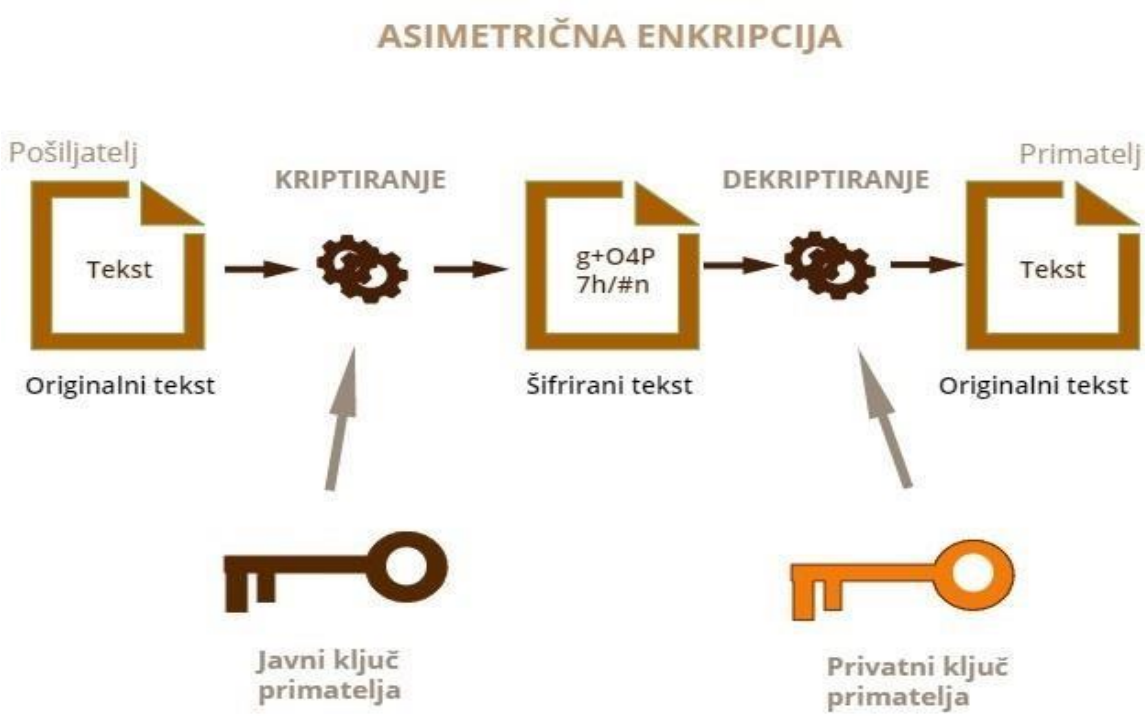
Kriptiranjem kompletnih blokova moguće je kriptirati kompletnu poruku odjednom ili je podijeliti u blokove pa kriptirati svaki zasebno. Predefinirane duljine ključeva koje se mogu koristiti su 128, 192 ili 256 bita. Primjeri kriptiranja simetričnim ključem su DES (*Data Encryption Standard*), *Triple* DES, TLS/SSL protokol i AES (*Advanced Encryption Standard*).



Slika 2.2. Simetrična enkripcija [11]

2.2. Asimetrični sustavi kriptografije

Asimetrični sustav kriptografije (javni ključ) je proces enkripcije između dvije strane, ali, za razliku od simetričnog ključa koji koristi jedan ključ, asimetrični ključ koristi dva ključa za obavljanje procesa. Prvi ključ je javni ključ, koji radi enkripciju podataka prije slanja na mrežu, dok je drugi, privatni ključ, ključ koji dekriptira podatke na prijemnoj strani. Javni i privatni ključ matematički su povezani. Jedan ključ je dostupan svakom sudioniku razgovora, dok je drugi ključ tajni. Asimetrični ključ žrtvuje brzinu kriptiranja za sigurnost zbog čega se za kriptiranje većih podataka koristi simetrični. U slučaju da želimo sigurniji prijenos, tada ipak koristimo asimetrični ključ.



Slika 2.3. Asimetrična enkripcija [11]

2.3. Diffie-Hellman razmjena ključeva

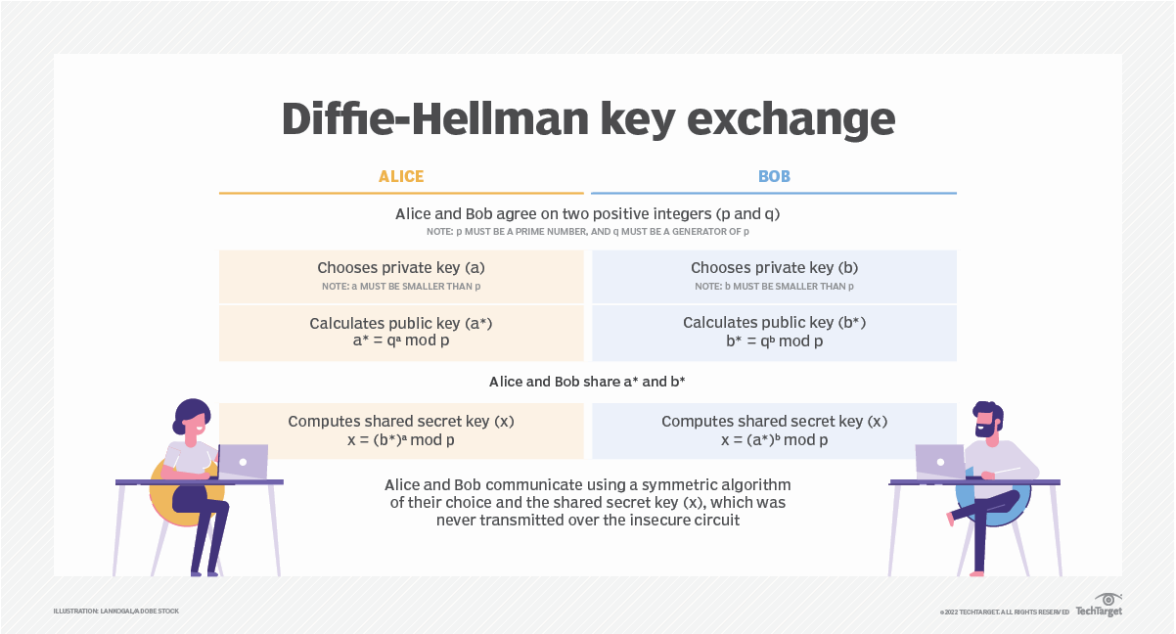
Diffie-Hellman razmjena ključeva je metoda digitalne enkripcije koja omogućava sigurnu razmjenu kriptografskih ključeva između dvaju sudionika razgovora preko javnog kanala. Sudionici razgovora koriste simetričnu metodu za kriptiranje i dekriptiranje poruka. Osmišljen 1976. godine od strane Whitfield Diffie i Martin Hellmana, jedan je od prvih primjera kriptografije javnim ključevima.

Diffie-Hellman razmjena ključeva zahtijeva puno resursa za generiranje ključeva. Komponente ključeva se ne prenose direktno te informacije tijekom razmjene ključeva se ne razmjenjuju. Sudionici zajedno generiraju ključ, ali nemaju informacije jedan o drugom, zbog čega potencijalni napadač ima puno zahtjevniji matematički zadatak prilikom razbijanja šifre.

DH razmjena ključeva ima cilj ostvariti siguran kanal za generiranje i dijeljenje ključa za simetričnu enkripciju. Koristi se za enkripciju, autentifikaciju lozinki i „forward secrecy protokol“. Autentifikacija lozinki je korištena za sprječavanje MITM (*man in the middle*) napada. *Forward secrecy* protokoli kreiraju novi par ključeva za svaku sesiju. DH razmjena ključeva uobičajeno se nalazi u sigurnosnim protokolima kao što su TLS (*Transport Layer Security*), SSH (*Secure Shell*) i IP (*IPsec*).

Implementacija DH razmjene ključeva započinje u trenutku kada se dva krajnja korisnika međusobno dogovore oko pozitivnog cijelog broja p (prosti broj) i q (generator broja p). Nešto više o radu DH razmjene ključeva bit će prikazano u 4. poglavlju.

Najveća mana DH sustava je manjak autentifikacije. Ako komunikacija koristi DH zasebno, ranjiva je na MITM napade. Da bi se provjerili identiteti sudionika u razgovoru, DH razmjenu ključeva, u praksi, potrebno je koristiti s nekom od metoda za autentifikaciju kao što je primjerice digitalni potpis. DH razmjena ključeva je također ranjiva na *logjam* napade, što je specifično za TLS protokol. *Logjam* napadi smanjuju razinu sigurnosti TLS konekcije na 512-bitnu kriptografiju omogućujući tako napadaču da čita i mijenja podatke koji su prošli kroz vezu. U slučaju da je DH razmjena ključeva implementirana s 2,048-bitnim ključem, može biti sigurna jer *logjam* napadi neće raditi na navedenom ključu.

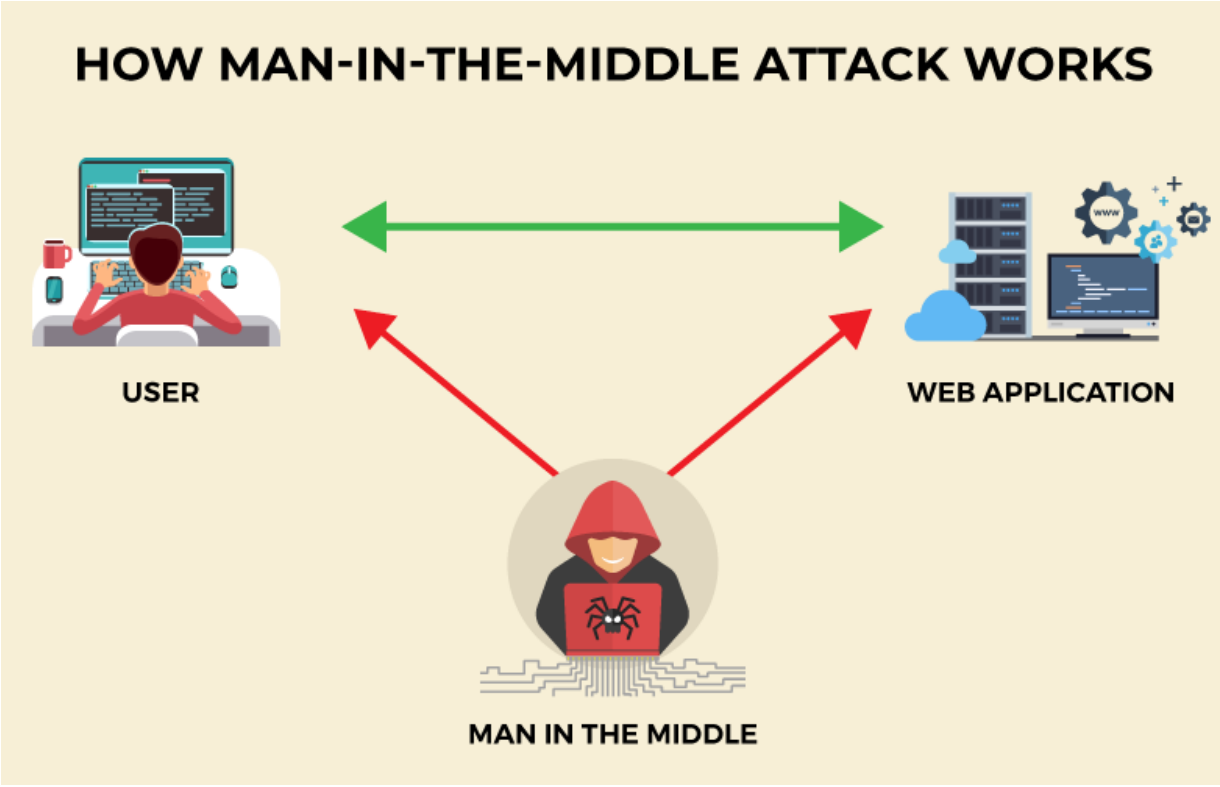


Slika 2.4. Diffie-Hellman razmjena ključeva [12]

Ako dvije osobe, nazovimo ih Alice i Bob, žele komunicirati i dijeliti osjetljive informacije putem javne mreže (u knjižnici, kafiću, restoranu i dr.), korištenjem DH razmjene ključeva za enkripciju osjetljivih informacija mogu izbjeći prisluškivanje od trećih strana ili hakera. Alice i Bob će tada odabrati tajni ključ na koji će se primijeniti funkcija za kreiranje javnog ključa. Rezultat je podijeljen, ali ne i funkcija. U slučaju da je treća strana prisluškivala razgovor, iz rezultata je gotovo nemoguće dobiti funkciju kojim je izveden javni ključ. Alice i Bob zatim primjenjuju novu funkciju koristeći rezultat dobiven od druge strane (Alice od Boba i Bob od Alice), njihov tajni ključ i originalan prosti broj. Na kraju, Alice i Bob generiraju zajednički javni ključ koji napadač ne može razbiti, nakon čega mogu slobodno komunicirati bez prisluškivanja trećih strana.

2.4. MITM (*man in the middle*) napad

MITM napad je generalni izraz za situaciju kada se počinitelj pozicionira u razgovor između korisnika i aplikacije s ciljem prisluškivanja razgovora ili imitacije jednog od sudionika razgovora, dok isti nisu svjesni da je u tijeku krađa informacija. Cilj napada je krađa privatnih informacija kao što su korisnička imena, lozinke, informacije o bankovnim računima i brojevi kreditnih kartica. Mete su najčešće financijske aplikacije kao što su PBZ mobilno bankarstvo, Aircash, KEKS mobile pay i dr. Informacije koje je počinitelj sakupio tijekom napada može koristiti u različite svrhe kao što su krađa identiteta, neodobreni prijenos novčanih sredstava ili pak promjena lozinke na servisima koje koristimo.



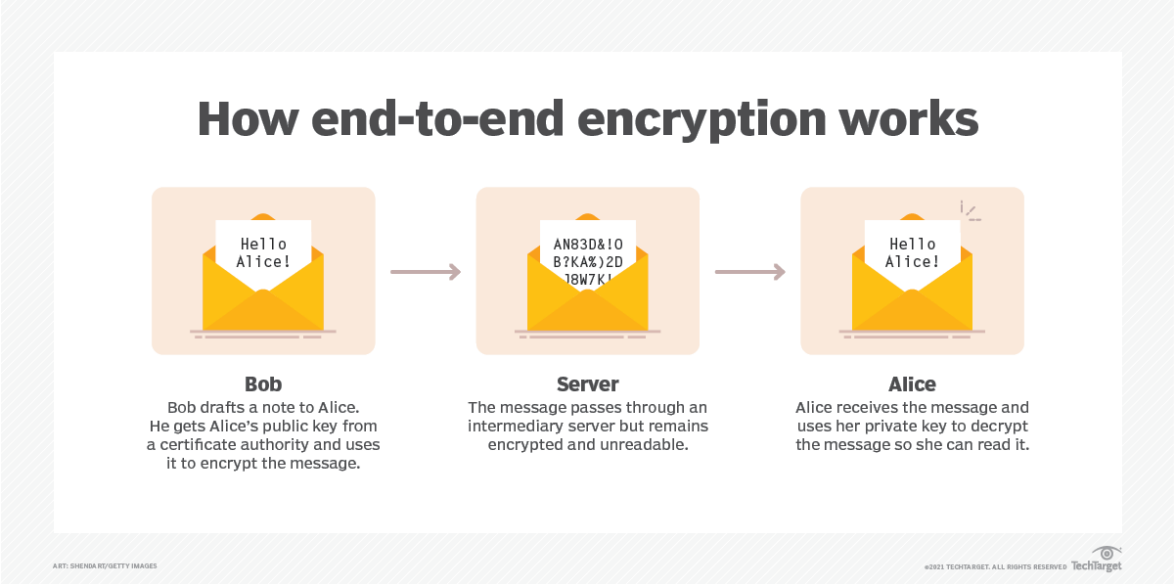
Slika 2.5. MITM (*man in the middle*) napad [13]

Prvi korak MITM napada je presretanje korisničkog prometa prije nego stigne na odredišnu točku. Najčešći, ali i najjednostavniji način je da napadač napravi WiFi žarišnu točku dostupnom bez potrebne lozinke i s imenom koje odgovara mjestu gdje se nalazi. Kada se žrtva spoji na navedenu žarišnu točku, napadač može pratiti svaki njen online korak. Osim navedenog načina, napadač se može poslužiti i podvalom IP adrese, lažne ARP poruke i DNS podvalom. Nakon što napadač dobije tražene podatke, mora ih dekriptirati bez da obavijesti korisnika ili aplikaciju.

3. END-TO-END ENKRIPCIJA

E2EE mehanizam služi za siguran prijenos poruka između dvaju ili više sudionika. Prilikom slanja poruke od pošiljatelja do primatelja, E2EE mehanizam štiti našu poruku od trećih strana koje se ne nalaze u našem razgovoru i kojima nije upućena poruka. On čuva izvornu poruku na način da, ukoliko se poruka tijekom prijenosa od pošiljatelja do primatelja promijeni, sustav to detektira i obavještava sudionike. Za enkripciju i dekripciju poruke koristi se metoda javnih ključeva u kojoj su ključevi pohranjeni kod krajnjih korisnika.

Enkripcija javnim ključem koristi javni ključ, koji se može dijeliti s drugima, i privatni ključ, kojeg koristi samo vlasnik. Enkripcija i dekripcija se provode na način da poruku koju želimo poslati kao pošiljatelj kodiramo korištenjem javnog ključa primatelja poruke. Na prijemnoj strani, primatelj poruku, kako bi je pročitao, dekodira vlastitim privatnim ključem. Na taj način samo vlasnik privatnog ključa može dešifrirati poruku, dok ostali, neželjeni primatelji, vide samo beskorisnu *hash* poruku.



Slika 3.1. Primjer E2E enkripcije [3]

Mnogi servisi za razmjenu poruka podatke, koji se prenose između korisnika, spremaju na servere koji su u posjedu komunikacijskih pružatelja usluga, no takvi su podaci zaštićeni samo u prijenosu. Iako server štiti podatke od neovlaštenih sudionika, mana ove metode je to što pošiljatelj može vidjeti informaciju i to u slučajevima gdje je potrebna zaštita u svim točkama prijenosa, što ponekad može biti nepoželjno. U slučaju E2E enkripcije čitljiva

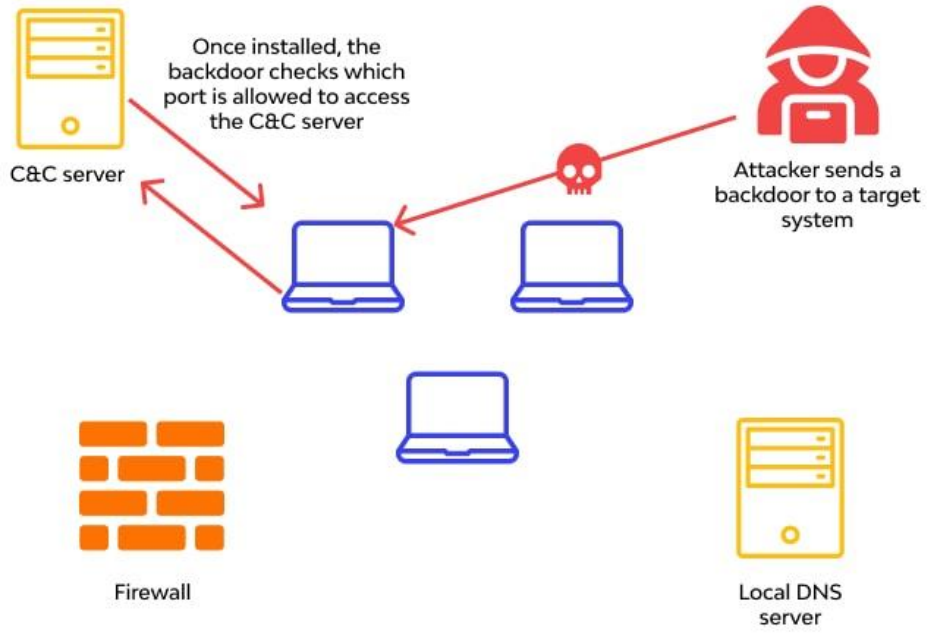
poruka je vidljiva samo onim sudionicima koji imaju ključ za dekriptiranje nečitljivog teksta. Drugim riječima, E2E enkripcija sprječava neovlaštenim sudionicima da čitaju ili mijenjaju podatke koji se prenose te se koristi u slučajevima gdje je privatnost sudionika razgovora na prvom mjestu, bez obzira jesu li oni prenosili tekst, slike, audiozapise, datoteke ili dr. Nadalje, E2E enkripcija štiti važne podatke kao što su: osobni podaci, bankovni računi, e-mail adrese, šifre servisa koje koristimo na internetu itd., te štiti od *cyber* napada. Stvorena je kao centraliziran sustav upravljanja korisničkim pravima i pruža detaljnu kontrolu nad time tko ima pristup kojim informacijama. Zajedno sa sustavom upravljanja ključevima, organizacije mogu zaštiti svoje podatke na svim razinama.

E2E enkripcija se koristi:

- Za sigurno komuniciranje u aplikacijama za komunikaciju kao što s WhatsApp i Signal te za digitalne radio postaje kao što je TETRA. E2E enkripciju koriste i E-mail pružatelji usluga kao što su ProtonMail i Tutanota.
- Za zaštitu lozinki. Aplikacije koje pružaju usluge zaštite lozinki su: 1Password, BitWarden, Dashlane i LastPass. U slučaju zaštite lozinki, korisnik je jedini sudionik s ključem i nalazi se na oba kraja komunikacije. Aplikacije koriste SHA-256 algoritam za zaštitu lozinki te do sada nikada nisu bile hakirane.
- Spremanje podataka može zaštiti podatke na korisničkom uređaju uz pomoć opcije za zaštitu podataka spremanjem na oblak (*cloud*), čuvajući tako podatke od svih ostalih sudionika pa čak i od servera *cloud* servisa.

End to end enkripcija počinje s kriptografijom, metodom za zaštitu podataka koja čitljivi tekst pretvara u nečitljiv, zvan *ciphertext*. Samo korisnici koji posjeduju tajni ključ mogu vidjeti podatke u čitljivom formatu. U E2EE, pošiljatelj ili kreator podataka kriptira podatke, a samo ih namijenjeni primatelj može dekriptirati. Enkripcija javnim ključem, tzv. asimetrična, kriptira i dekriptira podatke koristeći dva različita ključa. Javni ključ služi za kriptiranje podataka koji se šalju vlasniku javnog ključa. Kada primi kriptirani sadržaj, vlasnik javnog ključa dekriptira poruku vlastitim privatnim ključem kojeg samo on posjeduje. E2E enkripcija ima zadatak da kriptira podatke u prijenosu (podaci nisu kriptirani samo na krajnjim točkama), da zaštiti podatke od hakera, MITM (*man in the middle*) napada i da zaštiti od upada na „zadnja vrata“ koje kreiraju kompanije kako bi alternativnim putem pristupili podacima, otvarajući tako hakerima mogućnost više za

dolazak do željenih podataka. Hakeri također mogu samostalno instalirati alternativni ulaz kojim će zaobići sigurnosni sistem.



Slika 3.2. *Backdoor* napad [14]

Prilikom komunikacije putem *online* servisa kao što su WhatsApp, Telegram i Viber nailazimo na posrednika u komunikaciji koji je, u većini slučajeva, server te pripada telekomunikacijskoj kompaniji. Korištenje privatnog ključa, u E2E enkripciji, sprječava posrednike (servere) od čitanja poruka koje se prenose. Licenca koja osigurava da je javni ključ vjerodostojan ključ, kreiran od strane primatelja poruke, izdana je od strane CA (*Certificate Authority*).

3.1. CA (*Certificate Authority*) digitalni certifikat

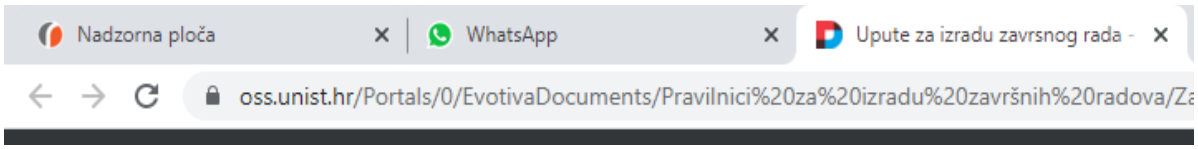
CA je vjerni digitalni certifikat koji povezuje entitet s javnim ključem. Mrežni ih preglednici koriste kako bi identificirali podatke koje dobivamo s mrežnih servera te kako osigurali da je traženi podatak isporučen od strane željenog pošiljatelja. Kao pružatelj usluge certificiranja ključeva, CA je neophodan i pouzdan temelj internetske infrastrukture javnog ključa ili PKI (*Public Key Infrastructure*). Glavni cilj CA certifikata je potvrditi pouzdanost i autentičnost mrežne stranice, domene i organizacije kako bi korisnici znali s kim komuniciraju na mreži te može li se navedenom entitetu vjerovati tijekom prijenosa podataka. Kada je CA certifikat dodijeljen mrežnoj stranici, korisnici znaju da su povezani

sa službenom stranicom, a ne s prividnom, koja je stvorena od strane trećih entiteta s ciljem krađe podataka.

Glavne uloge CA certifikata:

- Izdaje digitalni certifikat.
- Pomaže uspostaviti sigurnu vezu između entiteta koji žele komunicirati putem interneta.
- Potvrđuje vjerodostojnost domene mrežne stranice ili organizacije potvrđivanjem njihovog identiteta.
- Održava listu digitalnih certifikata.

SSL/TLS certifikat potvrđuje autentičnost i sigurnost stranice ili organizacije, odnosno sigurnu vezu između mrežne stranice i entiteta. Potvrdu sigurnosti veze možemo vidjeti u našem mrežnom pretraživaču kao ikonu lokota koja se uvijek nalazi do imena domene stranice koje pretražujemo.

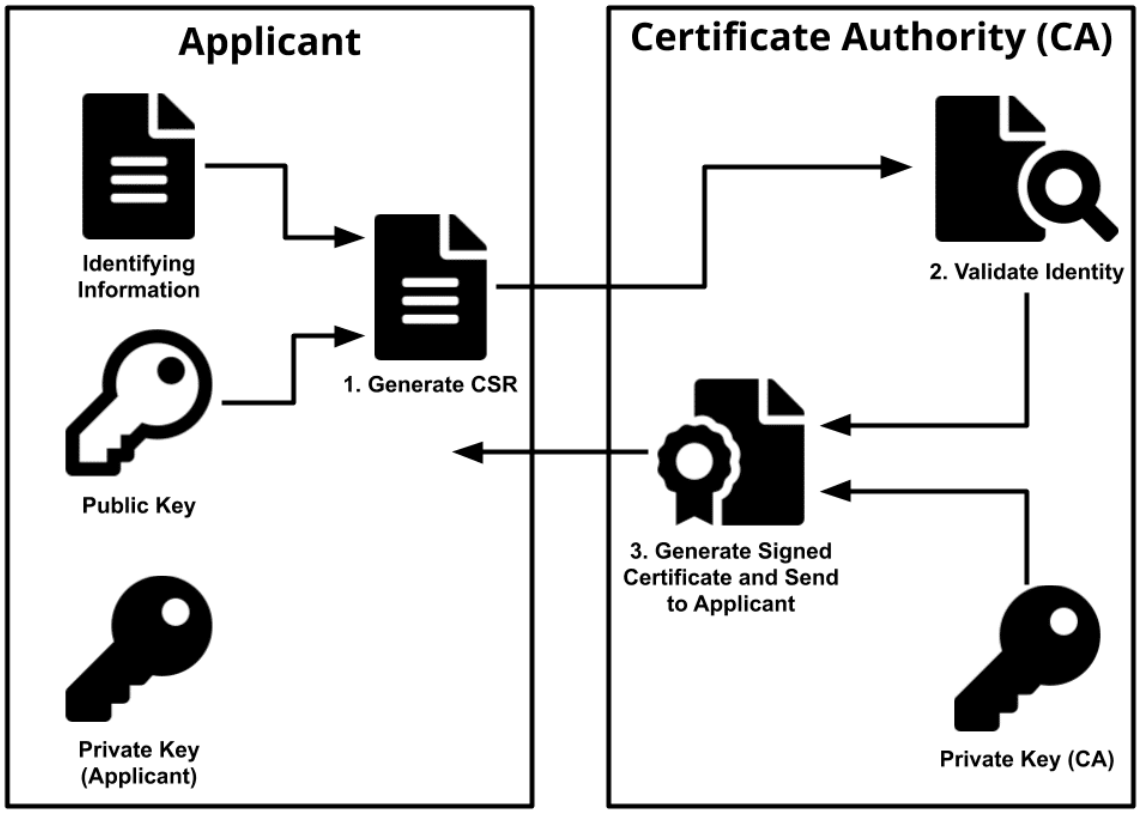


Slika 3.3. Potvrda autentične i sigurne stranice ikonom lokota.

SSL/TLS protokoli, kao važne komponente PKI-a, moraju sadržavati digitalni certifikat za rad. U tom trenutku nastupa CA. Entitet (organizacija ili osoba) može zatražiti izdavanje digitalnog certifikata od strane CA. Prvo se generira par ključeva koji sadrže sljedeće:

- Privatni ključ koji se drži tajnim i ne dijeli ni sa kim (niti sa CA).
- Javni ključ, spomenut u digitalnom certifikatu kojeg izdaje CA, te se također generira zahtjev za izdavanje certifikata CSR (*certificate signing request*), kriptirani tekst koji specificira informacije koje će biti uključene u digitalni certifikat, kao što su: naziv domene, alternativni ili dodatni nazivi domene, naziv organizacije, kontakt (e-mail adresa, kontakt telefon itd.)

Informacije uključene u CSR ovise o namjeni korištenja certifikata i razini valjanosti. Proces kreiranja digitalnog certifikata uvijek se odrađuje na serveru gdje se certifikat i instalira. Potraživač, nakon generiranja CSR, šalje CA koji provjerava informacije unutar CSR-a i potvrđuje identitet potraživača, nakon čega CA generira digitalni certifikat te ga digitalno potpisuje s privatnim ključem, a zatim šalje potraživaču.



Slika 3.4. Koraci CA autorizacije [15]

3.2. Digitalni potpis

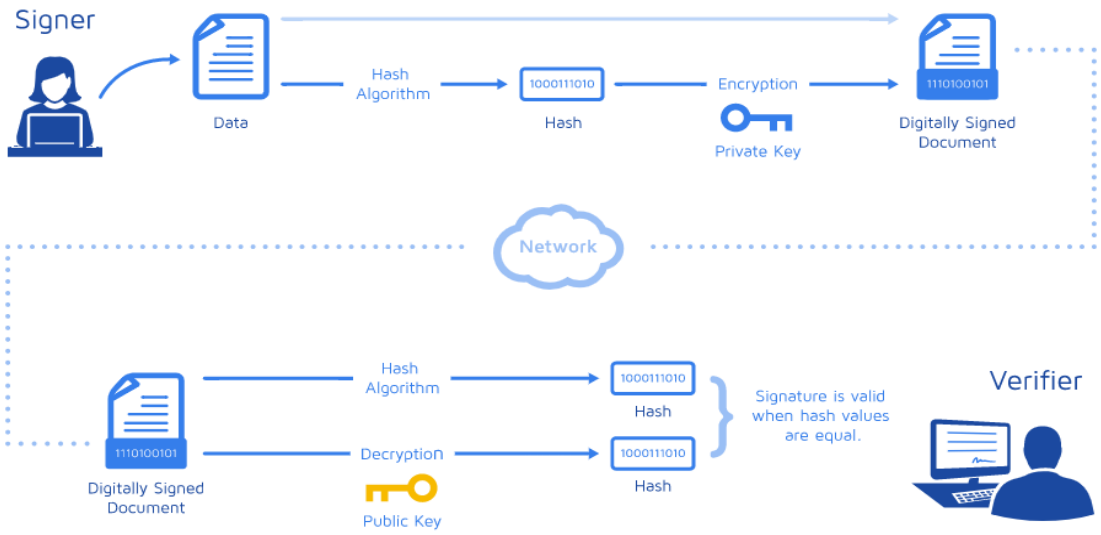
Digitalni potpis je forma kriptirane poruke koja povezuje kreatora dokumenta sa samim dokumentom. Digitalni potpisi koriste PKI (*Public Key Infrastructure*) standard kako bi pružili visoku razinu sigurnosti i univerzalnosti. PKI koristi digitalni potpis za verifikaciju identiteta.

Digitalni potpisi, kao i ručni potpisi, jedinstveni su za svakog potpisivača. PKI zahtijeva, od pružatelja usluge digitalnog potpisa, da generira dva ključa koristeći matematički algoritam, jedan javni i jedan privatni. Kada potpisnik elektronički potpiše dokument, potpis je kreiran korištenjem privatnoga ključa potpisnika, koji se uvijek nalazi kod istog.

Matematički algoritam koji djeluje poput šifre te stvara podatke koji odgovaraju potpisanom dokumentu kriptirajući iste, naziva se funkcija raspršivanja ili *hash* funkcija. Potpis također sadrži vrijeme u kojem je dokument potpisan te, ako se dokument nakon potpisa promijeni, digitalni potpis je nevažeći.

Primjer digitalnog potpisa:

Ana potpisuje ugovor za prodaju dionica koristeći svoj privatni ključ. Kupac prima dokument zajedno s kopijom Aninog javnog ključa. Ako javni ključ ne može dekriptirati potpis, to znači da potpis nije Anin ili je, u međuvremenu, bio promijenjen. U tom slučaju, potpis je nevažeći. Kako bi se zaštito integritet digitalnog potpisa, PKI zahtijeva da su ključevi kreirani, provjereni i spremljeni u sigurnim uvjetima od strane CA.



Slika 3.5. Proces provjere digitalnog potpisa [16]

Pošiljatelj primjenjuje *hash* algoritam na poruku te dobiva *hash* funkciju koja se kriptira pomoću privatnog ključa, a rezultat je digitalno potpisani dokument. Na prijemnoj strani, digitalno potpisani dokument se dekriptira pomoću javnog ključa te se istovremeno na dokument primjenjuje *hash* funkcija. U slučaju da su dobiveni ostaci jednaki, digitalni potpis je važeći.

4. PRIMJENA E2E ENKRIPCIJE NA PRIMJERU ALPIKACIJE WHATSAPP

WhatsApp, jedan od najpoznatijih i najkorištenijih servisa za razmjenu poruka, također koristi E2E enkripciju kao metodu zaštite podataka od neželjenih napadača. Svaka poruka koja je poslana željenom primatelju, kriptirana je na uređaju pošiljatelja. Kriptirana poruka putuje kroz mrežu i servere te dolazi na uređaj željenog primatelja, na čijem se uređaju i dekriptira.

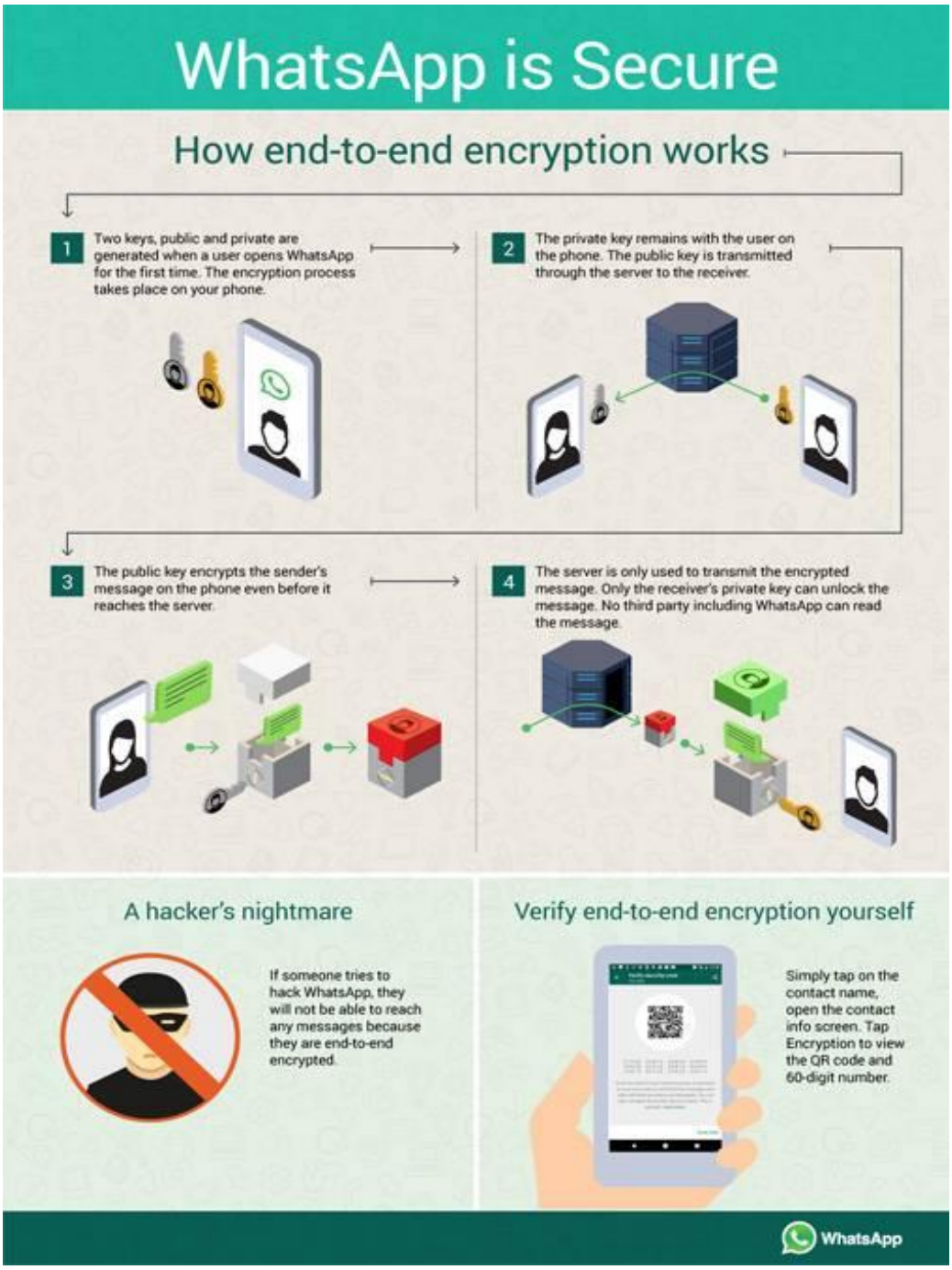
Primjena E2EE metode bit će pobliže opisana u nastavku, uz pomoć tri imena (Alice, Bob i Mallory).

Alice i Bob žele razmijeniti poruke. Mallory pak želi prisluškivati njihov razgovor i doći do željenih informacija. Također želi sudjelovati u razgovoru tako da, pretvarajući se da je Alice, šalje poruke Bobu te, pretvarajući se da je Bob, šalje poruke Alice.

Naš sustav mora imati sljedeća svojstva:

- Povjerljivost - Mallory ne smije imati uvid u poruke koje su si Alice i Bob međusobno slali.
- Integritet - U slučaju da Bob primi Aliceinu poruku, Bob može provjeriti je li poruka promijenjena od trećih strana tj. od strane Mallory.
- Autentičnost - Kada Bob primi Aliceinu poruku, može biti siguran da je ona došla od Alice, a ne od Mallory.

Valja spomenuti još jedno važno svojstvo, a to je poricanje. Ako netko oporavi poruke koje su Alice i Bob razmjenjivali u prošlosti, navedene poruke ne mogu biti povezane s navedenim sudionicima. Povjerljivost se može postići koristeći sistem kriptiranja/dekriptiranja (detaljnije pojašnjen u 5. poglavlju), a integritet i autentičnost mogu biti postignuti kroz digitalan potpis.



Slika 4.1. Implementacija E2E enkripcije u WhatsApp [1]

WhatsApp koristi nekoliko vrsta ključeva:

Javni ključevi:

- Par identifikacijskih ključeva - predstavljaju dugoročni par ključeva, generiraju se prilikom instalacije aplikacije, a generirani su pomoću Curve 25519 algoritma.
- Potpisani par ključeva - srednjoročni su par ključeva, potpisani od strane identifikacijskog ključa, koji se periodično rotiraju. Generirani su pomoću Curve 25519 algoritma.
- Jednokratni par ključeva - stvoren od strane Curve 25519 algoritma, generiran je prilikom instalacije te se po potrebi obnavlja.

Ključevi sesije:

- *Root* ključ (32-bitna vrijednost korištena za generiranje lančanih ključeva).
- Lančani ključ (32-bitna vrijednost korištena za generiranje ključeva poruka).
- Ključ poruke (80-bitna vrijednost koja je korištena za dekriptiranje sadržaja poruke, od čega su 32-bita korištena za AES-256 ključ, 32-bita za HMAC-SHA256 ključ i 16 bita za IV(inicijalizacijski vektor)).

Ostali ključevi:

- Tajni ključ za povezivanje (32-bitna vrijednost koja je generirana na povezanom uređaju te mora biti poslana preko sigurnog kanala do primarnog uređaja), koristi se za provjeru HMAC algoritma s primarnog uređaja. Prijenos ključa od povezanog do primarnog uređaja obavlja se skeniranjem QR koda.

WhatsApp koristi Signal protokol otvorenog koda koji je razvijen od strane Open Whisper Systems (Signal je također razvijen od strane iste kompanije). Signal protokol koristi metode kao što su Diffie Hellman, predključevi, Curve25519, AES u CBC modu, *Double Ratchet* algoritam i HMAC_SHA256.

4.1. Predključevi

Predključevi su Curve25519 parovi, generirani na uređaju tijekom instalacije. Sadrži jedan potpisani par i nekoliko jednokratnih ključeva. Javni ključ identiteta i par javnih ključeva potpisani su dugoročnim tajnim ključem identiteta (Curve25519 privatni ključ odgovara javnom ključu identiteta) i poslani na server tijekom registracije. Navedene ključeve server pohranjuje zajedno s javnim ključem identiteta. Curve25519 pruža 128-bitnu zaštitu (ključ duljine 256 bita), a dizajniran je za korištenje s eliptičnom krivuljom Diffie-Hellman protokola (ECDH), jednom od najbržih eliptičnih krivulja (ECC).

4.2. Implementacija DH protokola unutar WhatsApp-a

Diffie Hellman protokol dopušta dvama entitetima da razmjenjuju tajne poruke preko javnog kanala (Mallory može prisluškivati razgovor između Alice i Boba), a funkcionira na temelju modula prostih brojeva na sljedeći način:

Alice i Bob dogovaraju parametre protokola, veliki prosti broj p i generator g (generator multiplikativne grupe po modulu p). Generator je broj između 1 i $[p-1]$ i ima svojstvo da svaki element u nizu $[1-p]$ može biti prikazan kao $g^k \, mod_p$, k iznosi $[0, p-2]$.

1. Alice odabire nasumičan broj x koji se nalazi između 1 i $[p-1]$, koji je njen privatni ključ.
2. Bob odabere nasumičan broj y između 1 i $[p-1]$, koji je njegov privatni ključ.
3. Alice pošalje Bobu $x\_p = g^x \, mod_p$, Alicein javni ključ.
4. Bob pošalje Alice $y\_p = g^y mod_p$, Bobov javni ključ.
5. Alice računa $ss\_a = y\_p^x \, mod_p$.
6. Bob računa $ss\_b = x\_p^y \, mod_p$.

Nakon kompletiranja protokola, Alice i Bob su podijelili tajne $ss\_a = ss\_b = g^{(xy)} \, mod_p$. Mallory na mreži vidi x_p i y_p te je računalno neizvedivo odrediti podijeljene tajne između Alice i Boba bez znanja x ili y broja.

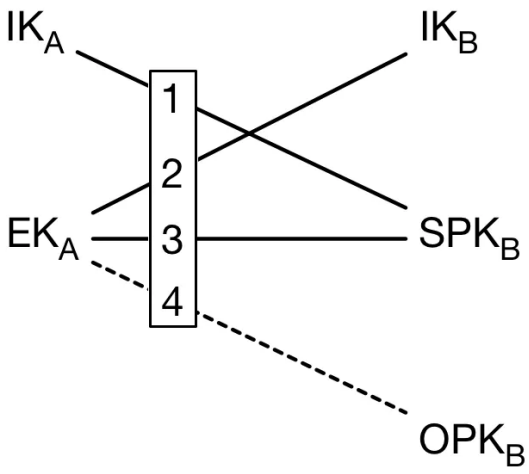
Nadalje, važno je spomenuti i prošireni Diffie Hellman protokol pod nazivom X3DH. X3DH je ekstenzija Diffie Hellman protokola s asinkronim postavkama. Pretpostavimo da Alice želi uspostaviti zajednički ključ s Bobom za slanje kriptirane poruke prema istom. Razmjena traje sigurno dok je Bob na mreži, no kada je Bob van mreže Alice ga mora

čekati da dođe kako bi nastavili siguran razgovor. Kao što je već objašnjeno u gore navedenoj razmjeni ključeva, za Alice i Boba, dakle, nije sigurno da razgovaraju jedno s drugim. Moguće je da je Mallory u međuvremenu uspostavila MITM napad te da prisluškuje razgovor cijelo vrijeme. Oba navedena problema rješava X3DH koristeći siguran server i predključeve. Alice i Bob registriraju potpisane predključeve (potpisane s dugoročnim privatnim ključevima) na pouzdanom serveru. Svaki put kada jedan od njih želi uspostaviti zajedničku tajnu prvi dohvaća paket potpisanih predključeva.

Prepostavimo da je Alice pošiljatelj.

Alice provodi DH operaciju dohvaćanja javnog ključa s privatnim ključem (operacija 5. u prethodnom nizu) tri ili četiri puta (ovisno o predključevima dobivenim sa servera).

1. Alicein dugoročni privatni ključ (IKa) i Bobov potpisani predključ (SPKb).
2. Alicein kratkotrajni privatni ključ (EKa) (dobiven od para generiranih specifično za ovu namjenu te izbrisanih nakon operacije) i Bobov dugoročni potpisani predključ (SPKb).
3. Alicein kratkotrajni privatni ključ (EKa) i Bobov dugoročni javni ključ (IKb).
4. Alicein kratkotrajni privatni ključ (EKa) i Bobov jednokratni javni ključ (OPKb). (navedeni se korak izvodi tek ako paket predključeva sadrži samo jedan jednokratni ključ)



Slika 4.2. Proces dobivanja glavnog ključa [7]

Izlazi su iz navedenih koraka kombinirani kako bi se dobila glavna tajna (*master secret*). Ključevi, izvedeni iz glavne tajne, korišteni su u *Double Ratchet* algoritmu za slanje kriptiranih poruka Bobu, a sve poruke uključuju Alicein kratkotrajni javni ključ, dugoročni identifikacijski ključ i informaciju o tome koji se Bobov jednokratni javni ključ koristi (u slučaju 4. koraka). Poslana poruka može biti primljena od strane Boba ako je na mreži ili može biti pohranjena na server s kojeg Bob kasnije može preuzeti poruku. Kada Bob preuzme poruku sa servera, on može izvesti istu glavnu tajnu koristeći svoj privatni ključ i Alicein javni ključ. Navedeni postupak omogućuje Alice i Bobu da se međusobno autentificiraju i izvedu glavnu tajnu kao temelj za stvaranje ključa.

4.3. HMAC_SHA256

HMAC_SHA256 je ključna kriptografska *hash* funkcija. Osim vrijednosti koja će biti šifrirana, navedena funkcija dodaje tajni ključ u podatke poruke te šifrira rezultat sa *hash* funkcijom, a dobiveni rezultat opet miješa s tajnim ključem te ponovno primjenjuje *hash* funkciju. Izlaz je duljine 256 bita. Navedena funkcija može odrediti je li poruka, poslana preko nesigurnog kanala, neovlašteno mijenjana. Pošiljatelj i primatelj poruke moraju podijeliti tajni ključ kako bi se navedeno utvrdilo. Pošiljatelj izračunava *hash* vrijednost za originalne podatke te šalje originale podatke i *hash* vrijednost u jednoj poruci, nakon čega primatelj računa *hash* vrijednost na dobivenoj poruci i provjerava je li dobivena vrijednost jednaka prenesenoj vrijednosti. Svaka promjena podataka rezultirala bi s nejednakosti u rezultatima. Da bi mogao mijenjati poruku, napadač mora znati tajni ključ koji je podijeljen između primatelja i pošiljatelja navedene poruke.

4.4. Dupli Ratchet algoritam

Ratchet je ime za uređaj koji se kreće u samo jednom smjeru. Dupli Ratchet koristi dva kriptografska Ratcheta, izvodi nove ključeve iz trenutačnih te zaboravlja stare ključeve. Dupli Ratchet koristi Diffie Hellman Ratchet i Hash Ratchet. Svaki put kada DH Ratchet ide naprijed, tajna između pošiljatelja i primatelja je uspostavljena korištenjem, već objašnjenog, DH algoritma. Tajna se koristi za kreiranje nova dva ključa (lančanog i *root* ključa). Hash Ratchet ide naprijed koristeći funkciju derivacije ključa (KDF), lančani ključ za generiranje ključa kod kriptiranja poslane poruke i lančani ključ koji će biti korišten za

sljedeći pomak. Navedeni postupak pruža korisno svojstvo protokolu zvanom *forward secrecy*. Primjer toga je kada Alice ili Bob kompromitiraju svoje ključeve u budućnosti, a prethodne poruke iz njihovog razgovora ne mogu se dekriptirati jer su ključevi izbrisani.

Svi navedeni primjeri zaštite rade zajedno kako bi klijentu pružili najveću razinu zaštite na mreži. U sljedećem primjeru bit će objašnjen način na koji protokoli rade zajedno. Navedeni se koraci pojavljuju kada Alice želi poslati poruku Bobu koristeći WhatsApp:

- Registracija klijenta s WhatsApp serverom (mobilne aplikacije na Aliceinom i Bobovom uređaju).

Aliceina strana:

- Sesija se uspostavlja od strane Alice koristeći X3DH.
- Alice izračunava glavnu tajnu i, koristeći DH Ratchet korak, izvodi *root* ključ i lančani ključ za korištenje u Hash Ratchetu.
- Alice izvodi ključ poruke i sljedeći lančani ključ koristeći Hashing Ratchet.
- Alice kriptira poruku koristeći ključ poruke (AES256 u CBC modu).
- Svaki put kada Alice pošalje poruku, Hashing Ratchet ide naprijed.
- Svaki put kada dobije Bobov odgovor, koji uključuje novi javni ključ u zaglavlju, Alice nastavlja naprijed s DH Ratchetom te računa novi *root* i lančani ključ.

Na Bobovoj strani:

- Kada dobije prvu Aliceinu poruku, Bob dovršava postavljanje sesije izvođenjem glavne tajne, *root* ključa, lančanog ključa i ključa poruke.
- Koristi ključ poruke za dekriptiranje poruke.
- Ako želi poslati poruku, kreira novi kratkotrajni par ključeva, DH Ratchet gura naprijed koristeći *root* ključ i kratkotrajni privatni ključ te ga mijenja s lančanim ključem i ključem poruke.
- Novi je ključ poruke izveden iz lančanog ključa, a poruka je kriptirana.
- Kriptirana je poruka poslana zajedno s Bobovim kratkotrajnim ključem u zaglavlju.

U slici ispod (pretpostavljajući da je Alicein mobitel), poruke u bijelim oblacima predstavljaju primljene poruke, dok poruke u zelenim oblacima predstavljaju poslane. Svaki uzastopni bijeli ili zeleni oblak može predočiti kretnje Ratchet-a.

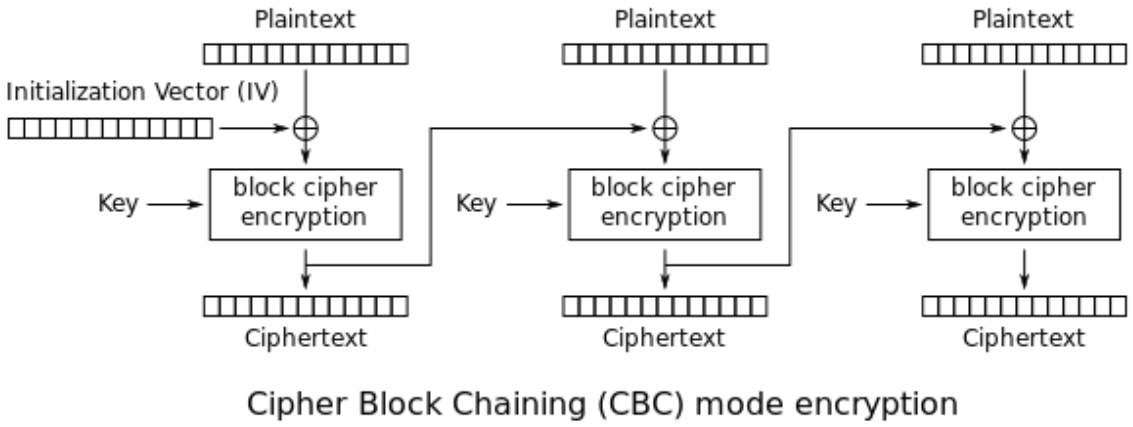
- 2 uzastopna bijela oblaka: Bobov Hashing Ratchet ide naprijed jedan korak (novi ključ poruke i novi lančani ključ su kreirani)
- 2 uzastopna zelena oblaka: Bobov Hashing Ratchet ide naprijed jedan korak (novi ključ poruke i lančani ključ)
- Bijeli pa zeleni oblak: Alicien DH Ratchet ide naprijed jedan korak (izvodi novi *root* i lančani ključ)
- Zeleni pa bijeli oblak: Bobov DH Ratchet ide naprijed jedan korak (izvodi novi *root* ključ i lančani ključ)

Slika 4.3. Alicein i Bobov razgovor [7]

Prilikom slanja, medijski se privitci i dokumenti kriptiraju i učitavaju na Blob spremnik. U slučaju da Alice želi vidjeti audiozapis, sliku ili video koji joj je Bob poslao, pokazivač, na lokaciji gdje je datoteka spremljena, je kriptiran i poslan gore navedenim postupkom.

Kada pošiljatelj kreira grupu s više članova te pošalje prvu poruku, on (ili ona) generira ključ pošiljatelja koji je poslan svim članovima u grupi koristeći, već spomenuti, postupak jedan na jedan sa svakim članom grupe. Sljedeće se poruke u grupi kriptiraju koristeći novi ključ poruke, izveden iz Hash Ratchet-a. Svi članovi grupe posjeduju ključeve jedni drugih, te ih, po potrebi, šalju Ratchet pošiljatelju za dekriptiranje poruke. Kada jedan od članova napusti grupu, proces razmjene ključeva, kao i prilikom kreiranja grupe, provodi se iznova.

CBC (*Cipher Block Chaining*) mod je shema za kriptiranje podataka koja koristi čisti tekst. Vrši XOR (ekskluzivno ili) operaciju sa IV (inicijalizacijski vektor), a dobiveni kod kriptira se pomoću ključa K. S tim dobivenim kriptiranim kodom izvodi se XOR operacija s čistim tekstom.



Slika 4.4. CBC (*Cipher Block Chaining*) mod [10]

4.5. Telefoniranje kod WhatsApp aplikacije

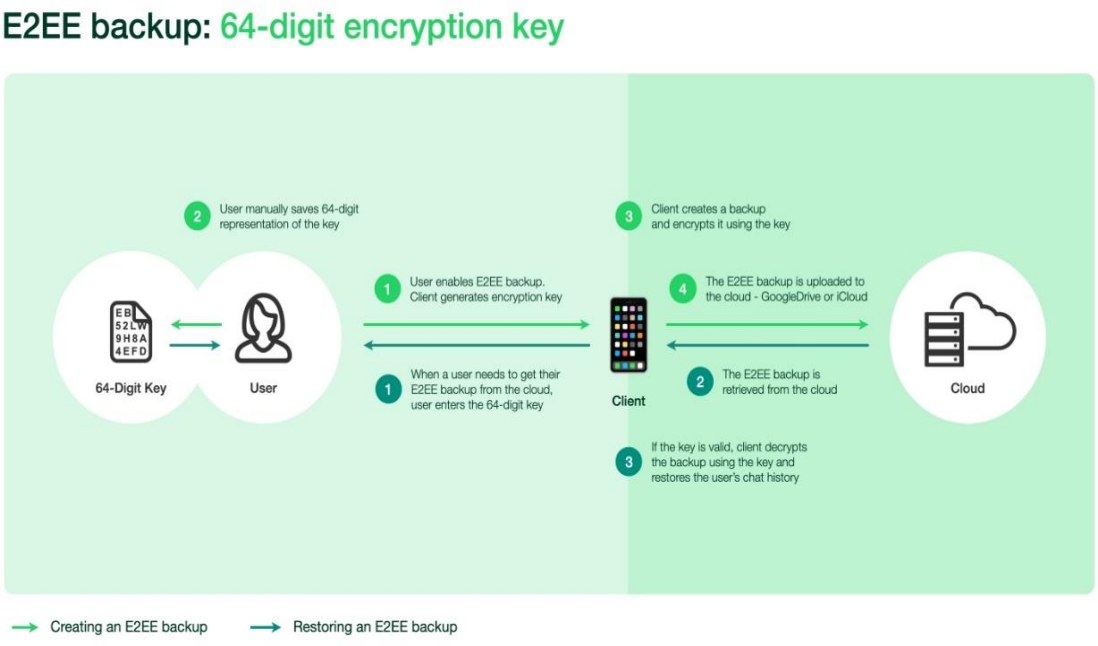
Telefoniranje je u realnom vremenu. Kada Alice želi uspostaviti poziv sa Bobom, Alice kreira nasumičnu SRTP tajnu. Tajna je poslana Bobu te, u momentu kada on odgovori na poziv, započinje kriptirana sesija. SRTP (*Secure Real-Time Transport Protocol*) je ekstenzija RTP-a (*Real-Time Transport Protocol*) koja pruža dodatne sigurnosne značajke kao što su poruke autentifikacije i povjerljivost. RTP koristi autentifikaciju i kriptiranje u svrhu minimiziranja rizika od napada. Navedene se značajke mogu individualno aktivirati ili deaktivirati. Kao iznimka, protokol Secure RTCP (*RTP Control Protocol*) mora imati uvijek aktiviranu značajku autentifikacije. Secure RTP koristi AES (*Advanced Encryption Standard*) kao temelj svog sigurnosnog sustava. Secure RTP je fleksibilan protokol te se vrlo lako može prilagoditi novim sigurnosnim algoritmima.

4.6. Cloud pohrana

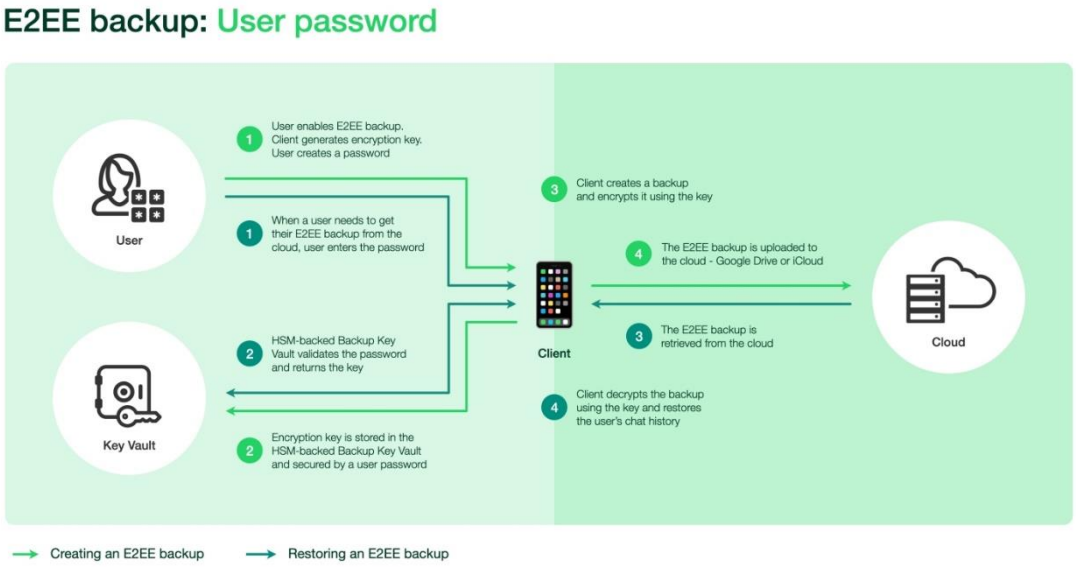
Još jedna od važnih značajki E2E enkripcije koja je implementirana u WhatsApp aplikaciju je pohrana (*backup*) razgovora, datoteka, videozapisa itd. WhatsApp je nudio opciju učitavanja željenih datoteka na *cloud* servise kao što su Google Drive i iCloud te nema pristup pohranjenim datotekama iz razloga što su zaštićene od strane korištenog *cloud* servisa. Navedeni servisi nisu pružali dovoljnu razinu zaštite da bi sačuvali naše datoteke, što je više puta i dokazano. Implementacijom E2EE pohrane, WhatsApp i *cloud* servis nemaju pravo pristupa pohranjenim datotekama ili ključu kojim je kriptirana pohrana. Kako bi omogućio E2EE backup, WhatsApp je kreirao novi sistem kriptiranog spremišta koji radi s Android i iOS mobilnim uređajima. Omogućenom E2EE pohranom, pohrana će biti kriptirana s jedinstvenim, nasumično generiranim, ključem. Korisnici sami mogu birati između zaštite s navedenim ključem ili zaštitu pomoću lozinke. Ukoliko korisnik odabere lozinku kao opciju, ključ se pohranjuje u rezervnom trezoru ključeva koji je kreiran na bazi HSM-a (*Hardware Security Modul*), specijaliziranog i sigurnog sustava koji se može koristiti za pohranu enkripcijskih ključeva. Kada vlasnik računa želi pristupiti svojoj pohrani to može učiniti uz pomoć enkripcijskog ključa ili uz pomoć svoje lozinke za povrat enkripcijskog ključa iz rezervnog trezora te na taj način dekriptirati svoju pohranu. HSM rezervni trezor je odgovoran za provjeru pristupne lozinke i ograničavanje broja neuspjelih pokušaja kod upisivanja iste. Nakon nekoliko neuspjelih pokušaja, ključ postaje trajno nedostupan. Navedene sigurnosne mjere pružaju zaštitu pokušaja dohvaćanja ključa *brute-*

force metodom. *Brute-force* metoda je iscrpno pretraživanje i isprobavanje svih mogućih kombinacija lozinki te uvijek pronalazi rješenje ako ono postoji, iako vrijeme i resursi potrebni za rješavanje problema rastu proporcionalno s brojem mogućih rješenja.

ChatD, WhatsApp servis, nadzire konekcije na liniji klijent-server i implementira protokol koji za zadatak ima slanje ključeva prema i od WhatsApp servera. Klijent i HSM trezor ključeva razmjenjuju kriptiranu poruku, dok je sadržaj navedene poruke dostupan ChatD servisu. HSM trezor ključeva pruža visoku sigurnost za enkripcijske ključeve u pohrani. Pohrana je generirana kao kontinuirani tok podataka koji je kriptiran korištenjem simetrične metode s generiranim ključem. S omogućenom E2E enkripcijom, nakon kriptiranja, osim na uređaju, pohrana može biti pohranjena i na *cloud* servise (Google Drive ili iCloud). WhatsApp ima preko dvije milijarde korisnika i jedan od najzahtjevnijih izazova je da HSM trezor funkcionira pouzdano. Kako bi WhatsApp osigurao stalnu dostupnost usluge HSM trezora, podaci su razmješteni u više podatkovnih centara kako bi, u slučaju pada jednog od centara, podaci ostali i dalje dostupni.



Slika 4.5. Korištenje ključa za očuvanje sigurnosti pohrane [2]



Slika 4.6. Korištenje lozinke za očuvanje sigurnosti pohrane [2]

Kada vlasnik računa odluči koristiti osobnu lozinku za zaštitu pohrane, HSM trezor ključeva će pohraniti njegov ključ. Kada klijent želi preuzeti svoju pohranu, prvo mora unijeti lozinku, koja se kriptira, a zatim provjerava od strane HSM trezora. Nakon što je lozinka provjerena, HSM trezor šalje enkripcijski ključ WhatsApp klijentu te kada klijent preuzme ključ, moguće je dekriptirati podatke pohrane. Ako je klijent odabrao 64-bitni ključ za zaštitu svoje pohrane, ključ za dekripciju i pristup podacima pohrane unosi ručno.

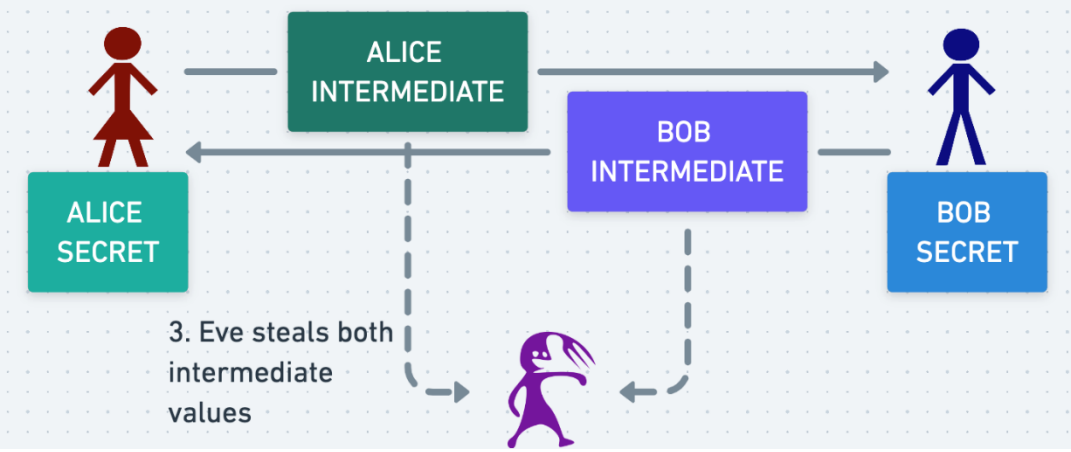
WhatsApp pruža visoku razinu zaštite te koristi kombinaciju više sigurnosnih algoritama da bi naše poruke i podaci koje razmjenjujemo bili zaštićeni. U skladu s novim sigurnosnim mehanizmima, napadači konstantno unaprjeđuju svoje metode zbog čega je potrebno iznova stvarati nove algoritme koji su zahtjevniji za „razbiti“. Osim WhatsApp *end to end* enkripcije, kanal između WhatsApp klijenta i WhatsApp servera je također siguran. Zaglavlja u kojima se ključevi razmjenjuju dobivaju dodatnu razinu zaštite uz zaštitu koju poruka ima kao cijela. Danas je u svijetu prisutan mit da WhatsApp, ali i druge društvene aplikacije nadziru naše poruke i datoteke koje prenosimo, a dokaz tome su reklame koje se nerijetko pojavljuju među novostima tih mreža, a često su povezane s razgovorima koje smo imali s prijateljima preko WhatsApp mreže. WhatsApp E2E enkripcija je implementirana, kao što i sam naziv govori, od kraja do kraja. Naše poruke putuju preko WhatsApp servera te su cijelo vrijeme kriptirane pa, prema tome, WhatsApp ne može nadzirati niti čitati razmijenjene poruke. Pretpostavimo da je protokol točno

implementiran, WhatsApp serveri i dalje znaju koji WhatsApp korisnik komunicira s kojim, koliko često i kada je zadnja komunikacija ostvarena. Ako je Vaš mobilni uređaj istovremeno spojen i s Facebook profilom i s WhatsApp aplikacijom, WhatsApp može iskoristiti tu informaciju za povezivanje Facebook aktivnosti Vas i Vaših prijatelja te, na račun toga, servirati relevantne oglase koji će Vas, moguće, iznenaditi.

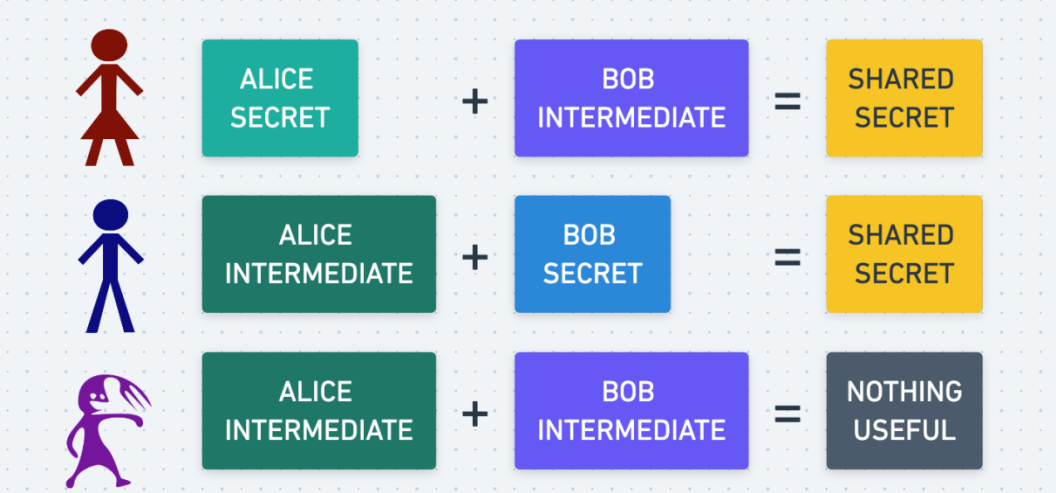
5. OTR (OFF THE RECORD MESSAGING)

OTR (*Off the record messaging*) je protokol koji se koristi za kriptiranje instant poruka. OTR koristi kombinaciju AES algoritma simetričnog ključa sa 128-bitnom duljinom, Diffie-Hellman razmjenu ključeva sa 1536-bitnim veličinama grupe i SHA-1 hash funkciju. Uz autentifikaciju i enkripciju, OTR pruža unaprijed tajnost i fleksibilnu enkripciju podataka. Poruke se kriptiraju korištenjem privremenih AES ključeva koji su kreirani pomoću Diffie-Hellman protokola. Ako napadač dođe u posjed ključeva, kompromitirana je samo poslana poruka, dok su prethodno poslane poruke i dalje zaštićene. Nadalje, poruke u komunikaciji nemaju digitalni potpis, što znači da nakon završene komunikacije svatko može kreirati poruku i pretvarati se da je jedan od pošiljatelja ili primatelja, dok za vrijeme komunikacije primatelj može biti siguran da poruka dolazi od izvornog pošiljatelja. Verzija OTR 3.1 protokola podržava međusobnu autentifikaciju korisnika korištenjem tajne podijeljene kroz SM (*Socialist millionaire problem*) protokol koji omogućava dvama stranama da provjere međusobni identitet koristeći dijeljenje tajne, izbjegavajući MITM napad korištenjem manualne usporedbe ključeva preko vanjskog kanala. OTR protokol koristi dvije faze. Prva faza je autentifikacija u kojoj se razmjenjuju ključevi dobivajući dijeljeni sesijski ključ, dok se druga faza odnosi na obnavljanje sesijskog ključa kod razmjene poruka na IM aplikacijama.

Za dobivanje dijeljenog tajnog ključa, OTR koristi Diffie Hellman razmjenu ključeva. Alice i Bob uzimaju dva slučajna broja, x i y, te ih razmjenjuju koristeći formulu $X = g^x$ i $Y = g^y$ preko javnog kanala. Obje strane mogu izračunati vrijednost zajedničke tajne, dok tajne dvaju strana ostaju zaštićene. Tajna vrijednost g^{xy} je dobivena od strane Alice formulom Y^xi od strane Boba X^y. Zajednički je ključ K generiran provlačenjem g^{xy} kroz *hash* funkciju $K = H(g^{xy})$ te su eksponenti izbrisani. Iako napadač može vidjeti vrijednosti X i Y, vjeruje se da se iz tih vrijednosti ne mogu izvući izvorne tajne dvaju sudionika u komunikaciji.



Slika 5.1. Napadač dolazi do X i Y vrijednosti [17]



Slika 5.2. X i Y vrijednosti su beskorisne napadaču [17]

Osnovni DH protokol pruža sigurnost od pasivnih napadača. Ako je napadač u mogućnosti presresti poruke koje se šalju, veoma mu je lako glumiti da je Bob i zavarati Alice da ima privatan razgovor s Bobom. Upravo se zbog toga koristi AKE (*Authenticated Key-Exchange*) protokol koji, povrh korištenja DH protokola, pruža autentifikaciju objema stranama te garantira privatnost ključa. Autentifikacija DH protokola često se provodi korištenjem javnih ključeva dviju strana. Ciljevi su autentifikacije da obje strane ne budu zavarane od trećih strana te da se očuva identitet sudionika u komunikaciji. Za dodavanje autentifikacije DH protokolu, OTR koristi javne ključeve i digitalne potpise. Svaka strana u komunikaciji ima par javnih/tajnih ključeva (ska i vka) za stvaranje sheme digitalnog

potpisa. U shemi potpisa, tajni je ključ ska korišten za kreiranje valjanog potpisa od strane vlasnika ključa tako da niti jedna druga strana ne može stvoriti valjan potpis. U isto vrijeme, svatko, koristeći javni ključ vka, može potvrditi valjanost potpisa. OTR pruža jednostavan i hijerarhijski pristup distribucije javnih ključeva gdje svaka strana sprema javni ključ strane s kojom komunicira. Kod ulaska u komunikaciju, korisnici moraju potvrditi valjanost javnog ključa koristeći CA certifikat. OTR faza razmjene ključeva i autentifikacije zahtijeva da svaka strana potpiše svoju DH vrijednost. Javni je ključ poslan preko prve poruke:

$$A \rightarrow B: Sign_{sk_a}(g^x), vk_a$$

$$A \leftarrow B: Sign_{sk_b}(g^y), vk_b$$

Ako je javni ključ (vka) već memoriran unutar B i povezan je s identitetom A, to strani B osigurava da je g^x došao od A i obrnuto. Nakon verifikacije potpisa, obje strane izračunavaju zajedničku vrijednost tajne i brišu DH eksponente. Glavni je cilj ovog protokola garantiranje autentifikacije, PFS (*Perfect Forward Secrecy*) i onemogućavanje krađe podataka od trećih strana. Nakon što je prvi sesijski ključ generiran, OTR započinje s procedurom obnavljanja ključa. Za svaku poruku, svaka strana generira DH eksponent koji će se koristiti za kreiranje novog DH ključa. Autentifikacija se izvodi korištenjem ključa iz prethodne poruke. Novi generirani ključ služi za enkripciju i autentifikaciju, dok se prethodni briše. Poruke razmijenjene tijekom OTR sesije kriptirane su i autentificirane. Poruka je prvo kriptirana korištenjem AES u CTR modu, a dobiveni rezultat je autentificiran korištenjem HMAC (sa *hash* funkcijom SHA-1). HMAC je specijalni tip MAC-a (*message authentication code*) koji pruža autentifikaciju porukama koristeći dijeljenju tajnu umjesto digitalnog potpisa u asimetričnoj kriptografiji. Također, postoji još jedan mehanizam kojeg OTR koristi u svrhu poricanja, a riječ je o mehanizmu osvježavanja ključa koji, postavljanjem novog MAC ključa, otkriva prethodni ključ, odnosno, čini ga javnim. Na taj način, nakon što se ključ otkrije, autorska prava povezana, u ovom slučaju, s Alice i Bobom se gube te svatko može stvoriti novu poruku, povezanu sa starim MAC ključem.

Sigurnost OTR mehanizma ima nekoliko mana, a jedna od njih koja se ističe je AKE (*Authenticated Key Exchange*) protokol. Jedan od ciljeva AKE protokola je garantirati da je identitet sudionika u razgovoru poznat svim sudionicima razgovora te da je ključ poznat

samo njima. Ako se protokol uspješno provede (sudionici u razgovoru kreiraju zajednički ključ) tada obje strane razgovora moraju kreirati isti ključ koji mora biti povezan s točnim identitetima. Alice i Bob moraju posjedovati isti ključ K te imati saznanja s kime je taj ključ podijeljen. Razvojem DH protokola, dokazana je njegova ranjivost na MITM napade. Tako je Eve u mogućnosti umiješati se u razgovor između Alice i Boba, na način da Bob misli da komunicira s Alice, dok Alice misli da komunicira s Bobom, a zapravo oboje komuniciraju s Eve. U tom slučaju, zaštita ove komunikacije neće biti kreirana između Boba i Alice, već između Boba i Eve. Ovdje Eve ima ulogu MITM-a te vodi dva razgovora istovremeno. DH vrijednosti poslane od strane Alice prema Bobu su prenesene od strane Eve prema Bobu, ali pod imenom Eve (sve što Eve treba napraviti je dodati DH vrijednost svom privatnom ključu). U navedenom primjeru vidimo kako ovo može biti korišteno od strane Eve da prevari Boba ili Alice. Simbol $E[B]$ predstavlja Eve u ovom razgovoru.

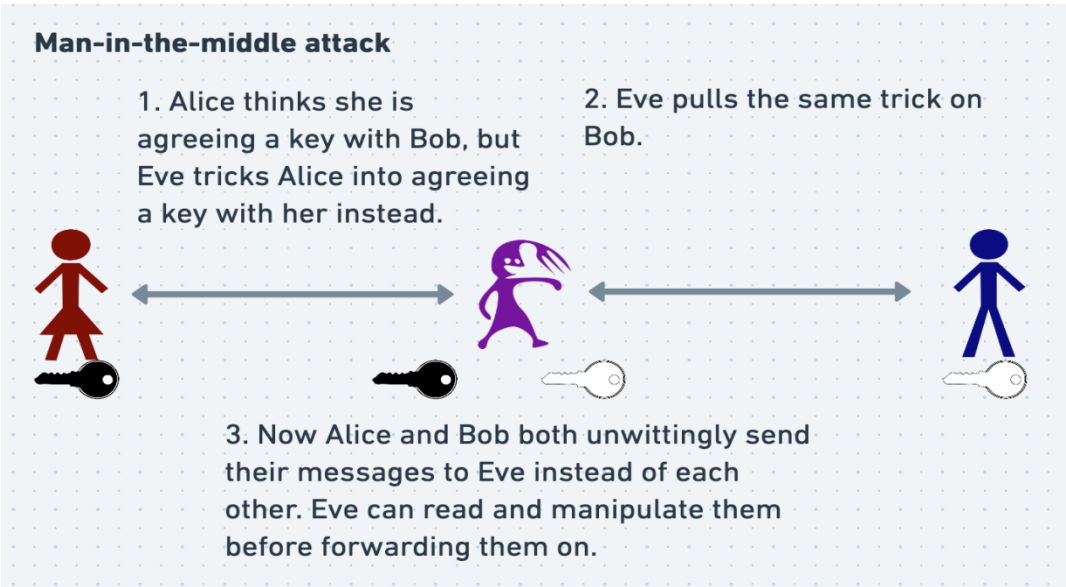
$$A \rightarrow E[B] \ : \ g^X, Sign_{sk_A}(g^x), vk_A$$

$$E \rightarrow B : g^x, Sign_{sk_E}(g^x), vk_E$$

$$E \leftarrow B \ : g^y, Sign_{sk_B}(g^y), vk_B$$

$$A \leftarrow E[B] : g^y, Sign_{sk_B}(g^y), vk_B$$

Kreirani ključ g^{xy} jednak je za obje sesije, ali Alice ovaj ključ povezuje s Bobom, dok Bob povezuje ključ s Alice. Iako ne zna ključ, Eve je u mogućnosti proslijediti poruke između dvije sesije i tako prevariti sudionike razgovora.



Slika 5.3. MITM napad kod OTR protokola [17]

Jedan od jednostavnijih načina kako bi se izbjegao ovaj napad je dodavanje identiteta željenog primatelja u potpis. Na ovaj način postaje nemoguće proslijediti potpisane poruke u sesiji gdje su uključeni različiti identiteti. Izgled potpisanog DH dogovora je sljedeći:

$$A \rightarrow B : g^x, Sign_{sk_A}(g^x, B), vk_A$$

$$A \leftarrow B : g^y, Sign_{sk_B}(g^y, A), vk_B$$

Nažalost, koristeći navedenu soluciju, protokol gubi svojstvo poricanja zbog kojeg je i kreiran. Uistinu, u navedenoj soluciji svaki sudionik ostavlja dokaz da je komunicirao s drugim sudionikom.

Poricanje je vrlo važno svojstvo OTR protokola, no upareno s mogućnošću autentifikacije postaje kontradiktorno. Alice želi biti uvjerena da priča s Bobom, dok Bob ne želi da Alice (ili bilo koji drugi sudionik) uspije dokazati da je Bob poslao poruku ili da je uopće razgovarao s Alice. Poricanje može biti definirano na više razina. Naprimjer, Bob želi priznati da je pričao s Alice sve dok može osporiti sadržaj razgovora. Najviša razina poricanja je kada Bob ne želi priznati da je pričao s Alice. Ovaj dio se definira provođenjem zahtjeva simulacije. Protokol je osporiv kao cijeli ako postoji algoritam (zvan simulacija) koji kreira prijepise, nevezane za originalne, bez znanja tajnih ključeva uključenih sudionika. Ako takav simulator postoji, svaki razgovor može biti osporen jer je produkt simulacije, a ne pravog razgovora. Prijepis Aliceinog razgovora nedovoljan je da se dokaže da je riječ o razgovoru između Alice i Boba.

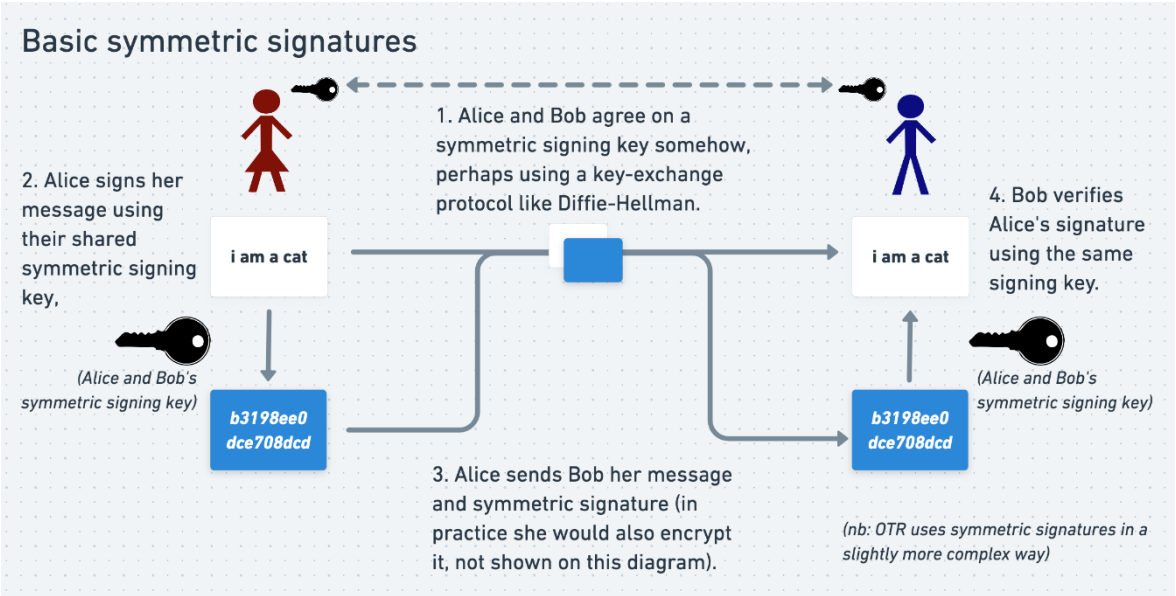
Alice i Bob su dogovorili zajedničku tajnu, simetrični ključ sesije. Sada je potrebno specificirati simetrični enkripcijski algoritam s kojim je, uz pomoć ključa, moguće kriptirati i dekriptirati poruke. OTR koristi već postojeće algoritme.

OTR zahtijeva fleksibilnu šifru koja, kada je dekriptirana, može dati više rezultata. Napadač može upravljati navedenom šifrom te može promijeniti tekst unutar poruke na način da primatelj poruke, nakon dekriptiranja iste, ne posumnja u legitimnost poruke. Kako bi iskoristio fleksibilnu šifru koju koristi OTR, napadač mora pogoditi otvoreni tekst u koji se šifrirani tekst dekriptira. Alternativni tekst koji napadač želi podmetnuti mora biti iste duljine kao i izvorni. Ako napadač pogodi, može generirati novi šifrirani tekst bez

znanja ključeva za kriptiranje i dekriptiranje originalnog šifriranog teksta. OTR fleksibilna šifra se naziva tok šifra u AES modu brojača.

Alice i Bob su dogovorili zajednički ključ, verificirali jedan drugom identitet i imaju ključ za dekriptiranje. Nisu koristili svoje privatne ključeve za potpis, što znači da nema ništa što bi ih moglo povezati s već poslanim porukama. Isto tako, Alice i Bob nemaju način detekcije je li poruka promijenjena u prijenosu. Svojstvo fleksibilnosti nije poželjno te je detekcija promjene poruke u prijenosu izrazito važna kod OTR protokola.

Da bi autentificirao sadržaj poruke, pošiljatelj mora potpisati poruku, dok primatelj mora provjeriti validnost potpisa. Alice i Bob ne žele potpisati poruke direktno koristeći svoje privatne ključeve jer u tom slučaju ne bi mogli poricati poruke. Za dokaz integriteta poruke, OTR koristi drugi tip kriptografskog potpisa. Nadalje, za kriptiranje poruka uz svojstvo unaprijed tajnosti, OTR koristi simetričnu enkripciju. Isto tako, za autentifikaciju razmijenjenih poruka, koristi algoritam simetričnog potpisa, čuvajući pri tom svojstvo poricanja. Algoritam asimetričnog potpisa koristi jedan ključ za kreiranje potpisa, drugi za verifikaciju, dok simetrični koristi jedan ključ i za kreiranje i za verifikaciju potpisa. OTR algoritam simetričnog potpisa naziva se HMAC (*Hash-Based Message Authentication Code*). Za generiranje HMAC potpisa, potpisnik unosi poruku i simetrični dijeljeni ključ u HMAC algoritam. Izlazna vrijednost algoritma je HMAC potpis, nakon čega pošiljatelj šalje poruku primatelju skupa s HMAC potpisom.



Slika 5.4. Generiranje HMAC potpisa i razmjena između sudionika razgovora [17]

Da bi verificirao HMAC potpis, primatelj poruke mora ponoviti isti proces kao i pošiljatelj. Unosi dijeljeni ključ i poruku u HMAC algoritam te dobiva HMAC potpis. Ako dobiveni potpis odgovara potpisu prenesenom u poruci, znači da se poruka nije mijenjala u prijenosu (pretpostavljajući da dijeljeni ključ nije otkriven). U navedenom primjeru vidimo da HMAC potpis pruža autentifikaciju, ali svojstvo poricanja nije očuvano.

Da bi sudionici koristili HMAC potpis za autentifikaciju, pošiljatelj i primatelj moraju dogovoriti zajednički tajni ključ s kojim mogu koristiti HMAC algoritam. OTR protokol koristi *hash* dijeljenog zajedničkog tajnog ključa za kriptiranje kao dijeljeni tajni ključ za potpisivanje. Kao što je već spomenuto, *hash* funkcija je funkcija koja proizvodi nasumični izlaz za svaki ulaz. Imajući izlaz *hash* funkcije, nemoguće je dobiti ulaz koji je generirao navedeni izlaz, dok je imajući ulaz vrlo lako dobiti izlaz *hash* funkcije. U OTR protokolu Alice i Bob koriste *hash* funkciju sa specifičnim sigurnosnim svojstvima za generiranje zajedničkog ključa za potpisivanje iz zajedničkog ključa za kriptiranje.

Slanje poruke započinje kada Alice kriptira poruku koristeći tok šifru u AES modu brojača sa simetričnim ključem za kriptiranje, koji su Alice i Bob zajednički dogovorili koristeći DH razmjenu ključeva. Alice računa *hash* ključ za kriptiranje kako bi dobila simetrični ključ za potpis, nakon čega šifrira kriptiranu poruku i potpisuje je koristeći HMAC algoritam s ključem za potpis. Dobivenu poruku i potpis šalje Bobu.

Kada primi Aliceinu poruku, Bob provodi isti proces, samo u obrnutom smjeru. Dekriptira poruku koristeći zajednički dijeljeni ključ za dekriptiranje. Bob računa *hash* ključ za kriptiranje kako bi dobio simetrični ključ za potpis. Isto kao i Alice, Bob šifrira kriptiranu poruku i izračunava potpis koristeći HMAC algoritam te dijeljeni ključ za potpis. Uspoređivanjem potpisa koji je dobio, Bob verificira potpis s potpisom koji je dobio od Alice. Alice i Bob sada mogu razmijeniti tajne, autentične poruke koje se mogu poreći.

Nakon razgovora, Alice može objaviti svoj i Bobov dijeljeni ključ za potpise, no bez obzira na činjenicu da je ključ javno dostupan, Bob je već verificirao Aliceine poruke, te je nebitno ima li Eve pristup objavljenom ključu. Za usporedbu, mnogo nesigurnije bi bilo kada bi Alice objavila svoj i Bobov zajednički dijeljeni ključ za kriptiranje jer, kada bi pohranila njihov podataka s navedenim ključem, Eve bi mogla dekriptirati podatke. Nakon što je Bob primio, dekriptirao i verificirao poruku, a Alice objavila dijeljeni ključ za potpise, sudionici brišu enkripcijski ključ sesije. Ovaj je korak neophodan kako bi se

očuvala unaprijed tajnost i kako bi se onemogućilo napadača da dođe do jasnog prometa. Alice i Bob ponavljaju isti proces dogovarajući novi zajednički ključ za kriptiranje/dekriptiranje.

6. ZAKLJUČAK

Obradom teme sigurnosnih mehanizama u aplikacijama za komunikaciju, zaključujemo da su razgovori sudionika zaštićeni implementacijom više sigurnosnih mehanizama. Cilj je omogućiti svim korisnicima integritet, autentifikaciju, tajnost i poricanje. Kriptografija se kao znanstvena disciplina razvijala godinama te je najveću raširenost postigla početkom 1990-ih godina pojavom razmjene poruka na mreži. Danas, najkorišteniji uređaj je mobilni, kojeg koristimo za pohranu datoteka, razmjenu poruka, organizaciju aktivnosti te plaćanje putem aplikacija mobilnog bankarstva. Najvažnija je stavka siguran prijenos informacija od korisnika do drugog korisnika ili servera. E2E enkripcija nam pruža najveću razinu sigurnosti koristeći Diffie Hellman razmjenu ključeva i digitalni potpis.

U E2E enkripciji korisnik je siguran da su njegovi podaci sigurni prilikom pohrane ili razmjene s drugim korisnicima. Također, podaci u prijenosu otvorenom mrežom ostaju sigurni i nepromijenjeni od trećih strana. WhatsApp aplikacija je najpoznatiji primjer korištenja E2E metode te je istu implementirala 2016. godine. Osim sigurne komunikacije porukama, WhatsApp pruža pohranu u oblak (*cloud*), VoIP (telefoniranje preko interneta), videopoziv i WhatsApp *bussines* opciju. Naravno, navedena sigurnost je „kratkotrajna“ jer, ukoliko sustav stagnira i ne implementiraju se novi načini kriptiranja i zaštite naših podataka, razina sigurnosti pada proporcionalno s porastom snage računala i pojavom novih metoda. Iz tog je razloga potrebno konstantno implementirati nove sigurnosne protokole i unaprjeđivati stare kako bi razina sigurnosti aplikacija za komunikaciju bila u skladu sa zahtjevima korisnika.

LITERATURA

[1] *Encryption overview, Technical whitepaper*, WHATSAPP, objavljeno 5. travnja 2016, https://www.whatsapp.com/security/WhatsApp-Security-Whitepaper.pdf [28.12.2022]

[2] Krassovsky, S., Cadden, G. *How WhatsApp is enabling end-to-end encrypted backups*, ENGINEERING.FB, objavljeno 10.rujna 2021, https://engineering.fb.com/2021/09/10/security/whatsapp-e2ee-backups/ [29.12.2022]

[3] Lutkevich. B., Bacon, M., *end-to-end encryption (E2EE)*, TECHTARGET, objavljeno lipanj 2021, https://www.techtarget.com/searchsecurity/definition/end-to-end-encryption-E2EE [03.01.2023]

[4] *What is end-to-end encryption?*, IBM, https://www.ibm.com/topics/end-to-end-encryption [03.01.2023]

[5] *Off-the-record Messaging*, WIKIPEDIA, objavljeno 10.12.2022, https://en.wikipedia.org/wiki/Off-the-Record_Messaging [05.01.2023]

[6] Di Raimondo, M., Gennaro, R., Krawczyk, H. *Secure Off-the-Record Messaging*, DIRAIMONDO.DMI.UNICT.IT, Alexandria, Virginia USA, objavljeno 07.11.2005, https://diraimondo.dmi.unict.it/wp-content/uploads/papers/otr.pdf [06.01.2023]

[7] Panghal, A. W*hatsApp End to End Encryption, How does it work?*, MEDIUM, objavljeno 06.listopada 2018, https://medium.com/@panghalamit/whatsapp-s-end-to-end-encryption-how-does-it-work-80020977caa0 [08.01.2023]

[8] Parashar, S., Shekhar, R., *WhatsApp and the End-to-End Encryption Saga: Analyzing the Tussle Between Government Guidelines and Right to Privacy*, JURIST, objavljeno 25.lipnja 2021, https://www.jurist.org/commentary/2021/06/parashar-shekar-whatsapp-encryption-privacy/ [10.01.2023]

[9] *Hashing Algorithm*, NETWORK ENCYCLOPEDIA, https://networkencyclopedia.com/hashing-algorithm/ [10.01.2023]

[10] *Block cipher mode of operation*, WIKIPEDIA, objavljeno 08.siječnja 2023, https://en.wikipedia.org/wiki/Block_cipher_mode_of_operation [11.01.2023]

[11] Kutlačić, D. Infrastruktura javnog ključa – PKI, OTVORENI SUSTAVI I SIGURNOST, objavljeno 12.siječnja 2012 https://security.foi.hr/wiki/index.php/Infrastruktura_javnog_klju%C4%8Da_-_PKI.html#Kriptografija [13.01.2023]

[12] Gillis, A.S. *Diffie-Hellman key exchange (exponential key exchange)*, TECHTARGET, objavljeno listopad 2022, https://www.techtarget.com/searchsecurity/definition/Diffie-Hellman-key-exchange [15.01.2023]

[13] *Man in the Middle Attack: A Havoc to Network Security*, THREATCOP, https://threatcop.com/blog/man-in-the-middle-attack/ [17.01.2023]

[14] *What Is A Backdoor Attack?*, WALLARM, https://www.wallarm.com/what/what-is-a-backdoor-attack [17.01.2023]

[15] SSL.com Support Team, *What Is a Certificate Authority (CA)?*, TSSL, objavljeno 06.prosinca 2021 https://www.ssl.com/faqs/what-is-a-certificate-authority/ [22.01.2023]

[16] *What are digital signatures*, DOCUSIGN, https://www.docusign.com/how-it-works/electronic-signature/digital-signature/digital-signature-faq [24.01.2023]

[17] Heaton, R. *Off-The-Record Messaging part 3: how OTR works*, ROBERTHEATON, objavljeno 15.veljače 2022, https://robertheaton.com/otr3 [25.01.2023.]

POPIS SLIKA