SVEUČILIŠTE U SPLITU
PRIRODOSLOVNO-MATEMATIČKI FAKULTET

DOKUMENTACIJA
ALGORITMI U PRIMJENI
DEMONSTRACIJA RSA ALGORITMA

Profesorica:

Studentica:

30.6.2023.

Divna Krpan

Ema Andrea Drašković

Tablica sadržaja

1. Uvod

1.1. Odabir algoritma

Za ovaj projekt odabran je RSA (Rivest-Shamir-Adleman) algoritam razvijen 1977. godine. To je algoritam koji spada u asimetrične kriptografske algoritme. Glavna svrha RSA algoritma je osigurati sigurnu komunikaciju putem javne mreže tako da samo odabrani primatelj može dekriptirati poruku. Funkcionira na principu korištenja javnog i privatnog ključa. Javni ključ se koristi za enkripciju podataka, dok se privatni ključ koristi za dekripciju. Ključna prednost RSA algoritma je da čak i ako je javni ključ poznat svima, gotovo je nemoguće odrediti privatni ključ koji je potreban za dekripciju.
Sigurnost RSA algoritma se temelji na matematičkom problemu faktorizacije velikih brojeva. Što su brojevi veći, to je faktorizacija teža i zahtijeva ogromnu računalnu snagu. RSA algoritam je postao temelj modernih kriptografskih sustava i koristi se u raznim aplikacijama, poput sigurnog prijenosa podataka, digitalnih potpisa, SSL/TLS protokola za sigurno pregledavanje web stranica i još mnogo toga.

1.2. Opis algoritma

1.2.1. Generiranje ključeva

Metoda **GenerateKeyPair** generira par ključeva (javni i privatni). Prima duljinu bita kao ulazni parametar i vraća objekt **KeyPair**. Parametar duljine bita koristi se za generiranje dva prosta broja, p i q, pri čemu se provjerava je li najveći zajednički djelitelj od $(n - 1) * (p - 1)$ i broja e jednak 1. Metoda korisiti pomoćne algoritme sa funkcijama **FindPrime** za pronalazak primarnih brojeva i **ModularInverse** za izračun inverza modularne vrijednosti za Eulerovu funckiju $\varphi(n)$.

Metoda **FindPrime** pronalazi primarni broj zadane duljine bita. Prvo se pomoću svojstva koje vraća broj milisekundi koje su prošle od pokretanja sustava kako bi se osigurao nasumičan niz byteova. Tom se nizu najniži byte postavlja 0x0, čime se osigurava da je broj unsigned. Još se postavlja najniži bit prvog bytea (indeks 0) na 1 što osigurava da je broj neparan te najviši i drugi najviši bit predzadnjega bajta se isto postavljaju na 1 kako bi se osigurala duljina generiranih ključeva.

Slijedi **Rabin-Miller test primarnosti**, poziva se unutar **FindPrime** while petlje. To je probabilistički test koji testira primarnost velikih brojeva. Nazvan po Ronaldu Rivestu, Leonardu Adlemanu i Michaelu Milleru, koji su ga prvi puta opisali. Postavljaju se broj $d = n - 1$ i s kao broj koliko je puta d djeljiv s 2. Nakon toga se generira slučajan broj $2 < a < (n - 2)$, i to onoliko puta koliki je ulazni parametar certainty Rabin-Miller testa. Vrijednost certainty određuje točnost testa. Za svaki generirani slučajni faktor a računa se vrijednost $x = a^d mod\, n$. Ako je $x = 1$ ili $x = n - 1$ prelazi se na sljedeći slučajni faktor a jer postoji mogućnost da je broj prost. Dalje slijedi iteracija za dodatnu provjeru primarnosti, a ponavlja se $s - 1$ puta kako bi se provjerilo jeli $x = 1$ ili $x = n - 1$, gdje se pri svakoj iteraciji

računa $x = x^2 \, mod \, n$. Ako je $x = 1$ to ukazuje na složenost broja. Ako je $x = n - 1$ prekida se petlja jer trenutni broj a može biti svjedok da je broj prost. Ako su sve iteracije prošle, to znači da su sve provjere propale u dokazivanju da je broj složen i postoji visoka vjerojatnost da je broj prost. Rabin-MIller test primarnosti nije apsolutno određen algoritam, već pruža visoku vjerojatnost da će točno odrediti primarnost broja. Ponavljanje testa s više slučajno odabranih a povećava pouzdanost rezultata.

Ako broj nije primaran, u metodi **FindPrime** traži se u određenim granicama na postojećem broju tako da se inkrementira za 2 (očuvanje neparnosti). Ako ni tu nema primarnog broja generira se potpuno novi niz byteova i cijeli proces se ponavlja.

Nakon što su pronađena dva prosta broja p i q, izračuna se produkt $n = p * q$, koji se koristi kao osnova za izvođenje operacija šifriranja i dešifriranja u RSA algoritmu. On određuje veličinu i raspon mogućih vrijednosti za poruke koje se mogu šifrirati i dešifrirati. Zatim se izračuna funkcija Eulerovog totijenta $\varphi(n)$ koja predstavlja broj relativno prostih brojeva s n. Odabere se e koji je relativno prost s $\varphi(n)$ i manji od n, a u ovom algoritmu koristi se broj 65537 (hex. 0x10001). To je javni eksponent. Izračuna se broj d koji zadovoljava uvjet $e * d \equiv (mod \, \varphi(n))$. To je tajni eksponent.
Javni ključ se sastoji od para (n, e), a tajni ključ se sastoji od para (n, d).

1.2.2. Šifriranje

Metoda **Encrypt** enkriptira tekst koristeći javni ključ (n, e). Tekst se pretvara u niz bajtova m koristeći ASCII kodiranje, zatim se enkriptiraju bajtovi pomoću metode **EncryptBytes** koja bajtove dodatno popunjava da bi se spriječio gubitak podataka. Padding se dodaje tako da se na kraj dodaju dva bytea, da originalni sadržaj ostane netaknut. Zadnji byte se postavlja na 0x00 kako bi se osiguralo dovoljno prostora za enkripciju i označio kraj paddinga. Na predzadnji se postavlja marker byte 0xFF koji označava mjesto gdje padding započinje, odnosno gdje je kraj podatka. Poruka koju je potrebno šifrirati predstavlja broj m koji je manji od n. Na posljetku se računaju šifrirani byteovi c pomoću formule $c = m^e \, mod \, n$. Enkriptirani byteovi se zatim pretvaraju u Base64 string.

1.2.3. Dešifriranje

Šifrirana poruka c predstavlja broj koji je potrebno dešifrirati. Metoda **Decrypt** dekriptira enkriptirani tekst koristeći privatni ključ (n, d). Prvo se enkriptirani tekst se iz Base64 pretvara u niz byteova, a zatim se dekriptiraju byteovi pomoću metode **DecryptBytes** pomoću formule $m = c^d \, mod \, n$. S dekriptiranih byteova se uklanjaju padding byteovi. Rezultat se pretvara u tekst koristeći ASCII kodiranje, čime se dobije dešifrirana verzija šifrirane poruke c.

2. Primjena na stvarnim problemima

Kriptografija je iznimno bitna u današnjem tehnološki razvijenom svijetu. Ona pruža sigurnost podataka u vidu zaštite povjerljivosti, integriteta i autentičnosti podataka, pruža privatnost, sigurnu komunikaciju, odnosno razmjenu podataka putem interneta, elektroničke pošte, mobilnih aplikacija i drugih komunikacijskih kanala. Korištenje šifriranja i digitalnih potpisa prječava neovlašteno pristupanje, manipulaciju ili krađu informacija tijekom prijenosa. Još pruža zaštitu identiteta putem digitalnih potpisa i certifikata. Zato može sprječiti krađe identiteta, phishing napade i neovlaštene pristupe računima. Kriptografija igra ključnu ulogu u sigurnosti financijskih transakcija kod na primjer online bankarstva, digitalnih valuta i drugih električnih plaćanja. Ključna je i za zaštitu povjerljivih podataka, primjerice istraživačkih rezultata, vojnih planova, poslovnih tajni i medicinskih podataka.

RSA algoritam ima vrlo široku primjenu u stvarnom svijetu i jedan je od najčešće korištenih kriptografskih algoritama.
Neki od mnogih primjena su:

- Koristi se za sigurnu komunikaciju putem interneta, elektroničke pošte, virtualne privatne mreže (VPN) i SSL/TLS protokola koji osigurava sigurno pregledavanje web stranica.
- Koristi se za generiranje digitalnih potpisa koji omogućuju provjeru identiteta i integriteta digitalnih dokumenata. Digitalni potpisi koriste privatni ključ za potpisivanje dokumenata, a javni ključ za provjeru potpisa.
- RSA se koristi za sigurno pohranjivanje podataka na diskovima, memorijskim karticama ili drugim medijima. Podaci se šifriraju prije pohrane, a dešifriraju se samo s odgovarajućim ključem.
- Koristi se za autentikaciju korisnika na sustavima za prijavu. Javni ključ korisnika pohranjen je na poslužitelju, dok se privatni ključ korisnika koristi za potvrdu identiteta prilikom prijave.
- Koristi se u izdavanju digitalnih certifikata koji pružaju vjerodostojnost i sigurnost u elektroničkim komunikacijama. Certifikati se korsite za provjeru autentičnosti web stranica, elektroničke pošte, digitalnih potpisa i drugih digitalnih identiteta.
- RSA se koristi u tehnologiji sigurnog pokretanja računala (Secure Boot) kako bi se osiguralo da samo autorizirani i pouzdani softveri pokreću računalo.
- Koristi se za potpisivanje softverskih aplikacija kako bi se osigurala autentičnost i integritet softvera, a time i spriječile neovlaštene modifikacije ili zloupotrebe.

3. Primjena u ovom projektu

Ovaj edukativni projekt je realiziran u razor aplikaciji te implementira pojednostavljeni RSA algoritam. Na Demo stranici razor aplikacije u textbox se upisuje tekst za enkripciju. Klikom na botun pokreće se algoritam i vrijednosti varijabli se ispisuju na ekran. Svrha ovog projekta je da se na jednostavan i razumljiv način interaktivno demonstriraju i objasne najbitniji koraci algoritma koji se u pozadini izvršavaju.

Izvorni kod algoritma je preuzet od GitHub korisnika *c272*, a repozitorij se nalazi na poveznici:
https://github.com/c272/SharpRSA.git
Iz navedenog repozitorija preuzete su klase **RSA**, **Key**, **Maths** i **Utils**. Neki dijelovi koda su izbačeni, primjerice serijalizacija. Dodan je kod kojim se relevantne varijable prosljeđuju u .razor datoteku.
Što se tiče razor aplikacije, napravljene su dvije stranice. Na stranici Home sažet je opis projekta, a na stranici Demo napravljen je unos i obrada unosa, te ispis rezultata i detaljnije opisani koraci algoritma.

4. Zaključak

U ovom projektu demonstriran je RSA (Rivest-Shamir-Adleman) algoritam. To je asimetrični kriptografski algoritam koji se koristi za sigurnu komunikaciju putem javne mreže. Algoritam koristi par ključeva - javni ključ kojim se šifrira i privatni ključ kojim se dešifrira. RSA algoritam temelji se na matematičkom problemu faktorizacije velikih brojeva i postao je temelj modernih kriptografskih sustava. Ima široku primjenu u stvarnom svijetu i ima ključnu ulogu u sigurnosti podataka.

U ovoj je dokumentaciji detaljno opisan proces generiranja ključeva, šifriranja i dešifriranja, te su navedene primjene algoritma u stvarnom svijetu. U samom projektu implementiran je pojednostavljeni RSA algoritam u razor aplikaciji, koja omogućuje interaktivnu i edukativnu demonstraciju algoritma. Kroz jednostavan unos teksta za enkripciju i prikaz koraka algoritma, korisnik može bolje razumjeti i pratiti proces šifriranja i dešifriranja poruka.