Elektrotehnički fakultet u Beogradu
Katedra za računarsku tehniku i informatiku

Sistem za naplatu potrošnje električne energije uz očuvanje privatnosti podataka korišćenjem funkcionalne enkripcije

diplomski rad

Mentor

prof. dr Pavle Vuletić

Student

Pavle Đurić
2019/153

Beograd, 2023.

0 Sadržaj

1 Uvod

Razvoj i široka dostupnost digitalnih tehnologija, od cloud računarstva do Interneta stvari (IoT), i oslanjanje ljudi na ovakve tehnologije u svim sferama njihovih života, eksponencijalno su povećali količinu i vrste podataka koji se prikupljaju, čuvaju i analiziraju. Uz ovu ekspanziju, došlo je i do rasta količine i ozbiljnosti sajber kriminala, jer se ovakvi podaci mogu iskoristiti u najrazličitije svrhe. Zbog toga, postavljaju se brojna pitanja o mogućnostima zaštite, kako privatnosti pojedinaca i biznis tajni, tako i informacija važnih za nacionalnu sigurnost. Zbog toga, razvoj tehnologije ide u pravcu nalaženja dovoljno efikasnih metoda za analizu podataka, uz potpuno očuvanje njihove privatnosti. Tradicionalne kriptografske metode, iako suštinske za funkcionisanje svakog računarskog sistema, često se pokazuju veoma nepogodnim za ovakve svrhe.

U ovom radu izdvajamo funkcionalnu enkripciju kao nov koncept u oblasti kriptografije, koji dozvoljava sigurnu obradu podataka na suštinski potpuno drugačiji način od tradicionalne kriptografije. Pogodnosti ove tehnike ćemo demonstrirati implementacijom sistema za naplatu električne energije, koja bi bez funkcionalne enkripcije bila neprihvatljiva sa stanovišta sigurnosti.

U poglavlju 2 biće reči o konceptu Funkcionalne Enkripcije, njenom značaju, primeni i ključnim principima funkcionisanja. Dodatno, biće objašnjene i neke od efikasnijih šema funkcionalne enkripcije. Poglavlje 3 je rezervisano za motivaciju za izradom ovakvog projekta, kao i za detalje dizajna posmatranog sistema. Četvrto poglavlje sadrži tehničke detalje vezane za realizovani sistem. U njemu će biti priložena i analiza perforansi ovakvog sistema.

2 Funkcionalna enkripcija

U ovom poglavlju biće pojašnjen pojam funkcionalne enkripcije, neke od šema FE kao i posebne osobine nekih šema koje ih čine primenljivijim u praksi.

2.1 Uvod

Za klasičnu asimetričnu enkripciju važi pravilo 'sve ili ništa' - uz pomoć odgovarajućeg ključa, primalac šifrovane poruke može dešifrovati isključivo ceo originalni sadržaj, a pomoću bilo kog drugog, ne može da sazna ništa o originalnim podacima. Dakle, jedan i samo jedan ključ može otkriti šifrovani sadržaj. Ova osobina asimetrične enkripcije nije pogodna u slučajevima kada je potrebno otkriti samo deo originalnih podataka ili vrednost neke agregatne funkcije (suma, prosek, varijansa, ...) nad ovim podacima.

Funkcionalna enkripcija (Functional Encryption - FE) je generalizacija asimetrične enkripcije koja omogućava izračunavanje određenih funkcija nad šifrovanim podacima, otkrivajući samo željene rezultate, ali ne i osnovni šifrovani sadržaj. Dozvoljava generisanje više različitih kriptografskih ključeva, samim tim i izračunavanje više različitih funkcija nad istim šifrovanim podacima. Zasniva se na složenim matematičkim i kriptografskim konceptima.

Kao oblast u kriptografiji, FE je svoju ekspanziju doživela u poslednjih petnaest godina, što je čini znatno mlađom od asimetrične kriptografije. Dostupni naučni radovi su dosta više fokusirani na teorijski aspekt ovog koncepta, sa ciljem da dokažu korektnost, principe funkcionisanja ali i zanimljive i revolucionarne mogućnosti ostvarive korišćenjem FE. Problem ovih rešenja je bio to što zbog previše složenih, kompleksnih i neefikasnih konstrukcija nisu primenljivi u praksi. Ipak, postoje i radovi koji razmatraju kako konstruisati dovoljno efikasne šeme FE primenljive na realne probleme koristeći trenutno raspoloživ hardver.

2.2 Definicija

Funkcionalnost F definisana nad (K, X) je preslikavanje $F : K \times X \to \Sigma \cup \{\perp\}$, gde je K prostor ključeva, X prostor poruka, Σ prostor izlaza, a $\perp$ specijalan znak koji ne pripada prostoru izlaz.

Kada se Funkcionalna Enkripcija definiše za funkcionalnost F, u definiciji figuriše ključ k. Da bi umesto ključa k figurisala funkcija f_k , definišemo $\mathcal{F}$ kao familiju funkcija izvedenu iz F:

$$\mathcal{F} = \{f_k | k \in K \wedge f_k(\cdot) = F(k, \cdot)\}.$$

Šema funkcionalne enkripcije (FE) za familiju funkcija $\mathcal{F}$ je torka algoritama (Setup, KeyGen, Enc, Dec), gde je

1. Setup $(1^\lambda) \to (pp, mk)$ — na ulazu: sigurnosni parametar λ; na izlazu: javni ključ mpk i privatni ključ msk
2. KeyGen $(msk, k) \to sk_f$ — na ulazu: glavni privatni ključ msk i funkcija $f \in \mathcal{F}$; na izlazu: tajni ključ sk_f
3. Enc $(mpk, k) \to c_x$ — na ulazu: javni ključ mpk i poruka $x \in X$; na izlazu: šifrovana poruka c_x
4. Dec $(sk_f, c_x) \to y$ — na ulazu: tajni ključ sk_f i šifrovana poruka c_x; na izlazu: $y \in \Sigma \cup \{\perp\}$

uz uslov da je $y = f(x)$ sa verovatnoćom 1.

U poređenju sa šemama asimetrične kriptografije, koje se sastoje od tri algoritma (Setup, Enc, Dec), šeme funkcionalne enkripcije sadrže i četvrti - KeyGen. Ovaj algoritam, za glavni tajni ključ msk (dobijen pomoću algoritma Setup) i funkciju f, generiše tajni ključ sk_f, koji se još naziva i ključem za funkcionalnu dekripciju (Functional Decryption Key). Ovim ključem, u opštem slučaju, nije moguće dešifrovati originalnu poruku, već samo i isključivo vrednost funkcije $f(x)$. U specijalnim slučajevima, za funkciju je moguće odabrati identičko preslikavanje $i(x) = x$ i tako postići efekat klasične asimetrične enkripcije. Sistem FE se smatra sigurnim ako napadač, sa skupom ključeva $\{sk_{f_1}, ..., sk_{f_t}\}$ ne može da sazna ništa o originalnim podacima, sem onoga što otkrivaju specifične funkcije $f_1, ..., f_t$.

2.3 Asimetrične Bilinearne Grupe u FE

Asimetrične bilinearne mape su matematički koncept koji se često koristi za konstrukciju različitih šema FE. U nastavku sledi objašnjenje ovog pojma, u opsegu... shvatanje..

Neka su $\mathbb{G}_1$ i $\mathbb{G}_2$ dve ciklične grupe reda q (q je prost broj iz $\mathbb{N}$), i neka su $g_1 \in \mathbb{G}_1$ i $g_2 \in \mathbb{G}_2$ generatori grupa $\mathbb{G}_1$ i $\mathbb{G}_2$, respektivno. Neka je $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$ funkcija koja preslikava dva elementa grupa $\mathbb{G}_1$ i $\mathbb{G}_2$ u element ciljne grupe $\mathbb{G}_T$, takođe reda q. U ovom radu, podrazumevamo da su sve grupe multiplikativne, i da je 1 neutralni element za ovu operaciju. Torku $(\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, q, e)$ nazivamo **asimetričnom bilinearnom grupom** ako važi:

1. operacije u grupama $\mathbb{G}_1$, $\mathbb{G}_2$ i $\mathbb{G}_T$, kao i preslikavanje e, su efikasno izračunljive,
2. $e(g_1, g_2) \neq 1$,
3.preslikavanje e je bilinearno: $(\forall x, y \in \mathbb{Z}_q)\ e(g_1^x, g_2^y) = e(g_1, g_2)^{xy}$.

Preslikavanje e često nazivamo **uparivanjem** (eng. *pairing*), jer uparuje dva elementa iz grupa $\mathbb{G}_1$ i $\mathbb{G}_2$ u element grupe $\mathbb{G}_T$.

U FE, elementi ovih grupa se često smeštaju u vektore. Neka je $\mathbb{G}$ grupa reda q (q je prost broj iz $\mathbb{N}$). Za bilo koji element $g \in \mathbb{G}$ i vektor $\mathbf{v} = (v_1, ..., v_n) \in \mathbb{Z}_q^n$ za $n \in \mathbb{N}$, oznakom $g^{\mathbf{v}}$ označavamo vektor $(g^{v_1}, ..., g^{v_n})$. Takođe, za bilo koji skalar $k \in \mathbb{Z}_q$ i vektore $\mathbf{v}, \mathbf{w} \in \mathbb{Z}_q^n$ važi:

$$(g^{\mathbf{v}})^k = g^{(k\mathbf{v})} \text{ i } g^{\mathbf{v}} \cdot g^{\mathbf{w}} = g^{\mathbf{v}+\mathbf{w}}$$

. Operacija uparivanja se može proširiti na vektore na sledeći način:

$$e(g_1^{\mathbf{v}}, g_2^{\mathbf{w}}) = \prod_{i=1}^{n} e(g_1^{v_i}, g_2^{w_i}) = e(g_1, g_2)^{\langle v,w \rangle}$$

2.4 Šeme funkcionalne enkripcije

Kao što je već napomenuto u poglavlju 2.1, dosadašnja istraživanja u oblasti FE su fokusirana na dokaze o postojanju i korektnosti FE. Postojanje mnogih šema je teorijski dokazano, međutim samo mali broj njih je dovoljno efikasan da bi bio primenljiv u praksi. Za sada, efikasnost konkretne šeme FE zavisi od konkretnog tipa funkcije koju ona realizuje. Opštenamenska

šema FE (ona koja bi mogla da radi sa bilo kojom izračunljivom funkcijom) je još uvek predmet mnogih istraživanja, ali do ovakvog pronalaska još niko nije došao.

Razmotrićemo dve podvrste FE u odnosu na tip funkcija koje realizuju. Iako je podela FE veoma složena, i drugačije definisana u različitim izvorima, mnogi od njih se slažu da su ove podvrste najčešće korišćene, upravo zbog svoje efikasne implementacije.

2.4.1 Predikatna enkripcija

Predikatna enkripcija je jedna vrsta FE, veoma korišćena u praksi, sa brojnim, specijalizovanim podvrstama.

Pod predikatom se podrazumeva funkcija koja na ulazu prima niz atributa, a vraća Bulovu vrednost. Svaki ključ za funkcionalnu dekripciju odgovara nekom predikatu, dok šifrovani tekst, pored same poruke, u sebi sadrži i niz atributa. Tokom dekripcije, atributi se stavljaju na ulaz predikata, a ako je izlaz predikata vrednost true, moguće je dešifrovati celu šifrovanu poruku.

2.4.2 Enkripcija unutrašnjeg proizvoda (Inner-Product Encryption)

Enkripcija unutrašnjeg proizvoda (Inner-Product Encryption - IPE) je podvrsta FE gde šifrovani tekst $c_{\mathbf{x}}$ odgovara vektoru $\mathbf{x} = (x_1, .., x_n) \in (\mathbb{Z}_q)^n$, a ključ za funkcionalnu dekripciju $sk_{\mathbf{y}}$ odgovara vektoru $\mathbf{y} = (y_1, .., y_n) \in (\mathbb{Z}_q)^n$. Na osnovu šifrovanog teksta i ključa za dekripciju, algoritmom dešifrovanja izračunava se unutrašnji proizvod $\langle \mathbf{x}, \mathbf{y} \rangle = \sum_{i=1}^{n} y_i x_i \in \mathbb{Z}_q$, tačnije

$$F(\mathbf{y}, \mathbf{x}) = \sum_{i=1}^{n} y_i x_i \text{ mod q.}$$

Prva šema IPE [ABCP15] je objavljena 2015. godine.

Glavna primena ove šeme je izračunavanje težinske sume za statističke potrebe. Na primer, pretpostavimo da se ocene jednog učenika za sve predmete čuvaju u vektoru y, koji je enkriptovan javnim ključem škole. Fakultet hoće da ponudi stipendiju onim učenicima koji su odlični iz matematike i fizike, a vrlo dobri iz ostalih predmeta. Srednja škola želi da omogući fakultetu da pronađe učenike za ovu stipendiju, ali ne želi da otkrije ocene svojih učenika, zbog zaštite privatnosti. Sa IPE, srednja škola može generisati ključ za funkcionalnu dekripciju koji odgovara vektoru x koji dostavlja fakultet. Recimo da x=(10, 8, 8, 5, 5) predstavlja težine za odgovarajuće predmete, i da se za ovakav vektor generiše sk_x. Fakultet sprovodi algoritam dekripcije pokoću ključa sk_x da bi dobio težinsku sumu ocena svakog studenta, i

ništa više. Na primer, ako jedan učenik ima ocene y=(5, 5, 4, 2, 2), dekripcijom se dobija $\langle x, y \rangle = 10 * 5 + 8 * 5 + 8 * 4 + 5 * 2 + 5 * 2 = 132$, tako da fakultet saznaje samo traženu sumu.

2.5 Skrivanje funkcije

Pretpostavimo sledeći scenario: Bolnica skladišti informacije o pacijentima na eksternom klaud serveru. Da bi obezbedila tajnost podataka, a u isto vreme mogla da obavlja potrebne analize i upite nad podacima, odlučuje se za korišćenje FE. Recimo da bolnica želi da nađe listu svih pacijenata koji se leče od određene hronične bolesti. Da bi ovo postigla, bolnica mora da izda poseban ključ za funkcionalnu dekripciju i pošalje ga klaud serveru. Međutim, sa dobijanjem ključa za dekripciju, klaud server neminovno saznaje koji upit bolnica želi da izvrši, tačnije, funkciju kojoj odgovara dati ključ za dekripciju. Nakon sprovođenja potrebnih izračunavanja, ako bi klaud server detektovao ime neke poznate osobe u dobijenoj listi pacijenata sa traženom bolešću, mogao bi ovakvu informaciju da otkrije medijima radi finansijske dobiti. Ovo je neprihvatljiv scneario sa stanovišta privatnosti.

U slučajevima kada je funkcija deo poslovne tajne, sadrži biznis logiku ili sadrži poverljive informacije, pored zaštite originalnih poruka, nephodna je i dodatna vrsta zaštite - skrivanje funkcije. FE sa skrivanjem funkcije (Function-Hiding FE) podrazumeva da se na osnovu ključa za funkcionalnu dekripciju ne može saznati koja funkcija odgovara datom ključu.

2.5.1 FHIPE

Enkripcija unutrašnjeg proizvoda sa skrivanjem funkcije (FHIPE [KLM+18]) je IPE šema FE koja ujedno nudi i skrivanje funkcije.

Za pozitivan ceo broj n i skup $S \subset Z_q$ sa brojem elemenata poly(n):

1. Setup $(1^\lambda, S) \to (pp, mk)$ na ulazu: sigurnosni parametar λ
   1. generisanje bilinearne grupe (G_1, G_2, G_T, q, e) na slučajan način
   2. izdvajanje generatora grupa $g_1 \in G_1$ i $g_2 \in G_2$
   2. generisanje regularne kvadratne matrice $\mathbf{B}_{n\times n}$ na slučajan način
   3. računanje matrice $\mathbf{B}^* = \det(\mathbf{B}) \cdot (\mathbf{B}^{-1})^\top$

   na izlazu: javni ključ $mpk = (G_1, G_2, G_T, q, e, S)$ i privatni ključ $msk = (mpk, g_1, g_2, \mathbf{B}, \mathbf{B}^*)$

2. KeyGen $(msk, y) \to sk_y$ na ulazu: glavni privatni ključ msk i vektor $\mathbf{y} \in Z_q^n$
   1. odabir slučajnog broja $\alpha \xleftarrow{rand} Z_q$

   na izlazu: tajni ključ $sk_y = (K_1, K_2) = (g_1^{\alpha\cdot\det(\mathbf{B})}, g_1^{\alpha\cdot\mathbf{y}\cdot\mathbf{B}})$

3. Enc $(mpk, x) \to c_x$ na ulazu: javni ključ mpk i vektor $x \in Z_q^n$
   1. odabir slučajnog broja $\beta \xleftarrow{rand} Z_q$

   na izlazu: šifrovana poruka $c = (C_1, C_2) = (g_2^\beta, g_2^{\beta\cdot\mathbf{x}\cdot\mathbf{B}^*})$

4. Dec $(mpk, sk_y, c_x) \to z$ na ulazu: javni ključ mpk, tajni ključ $sk_y = (K_1, K_2)$ i šifrovana poruka $c_x = (C_1, C_2)$

   izračunavanje $D_1 = e(K_1, C_1)$ i $D_2 = e(K_2, C_2)$

   na izlazu: $z \in S$ takav da je $(D_1)^z = D_2$

Korektnost. Za svaki $sk_y = (K_1, K_2)$ i za svaki $c_x = (C_1, C_2)$ važi:

$$D_1 = e(K_1, C_1) = e(g_1^{\alpha\cdot\det(\mathbf{B})}, g_2^\beta) = e(g_1, g_2)^{\alpha\beta\cdot\det(\mathbf{B})} \text{ i}$$

$$D_2 = e(K_2, C_2) = e(g_1^{\alpha\cdot\mathbf{y}\cdot\mathbf{B}}, g_2^{\beta\cdot\mathbf{x}\cdot\mathbf{B}^*}) = e(g_1, g_2)^{\alpha\beta\cdot\mathbf{y}\mathbf{B}(\mathbf{B}^*)^\top\mathbf{x}^\top} = e(g_1, g_2)^{\alpha\beta\cdot\det(\mathbf{B})\langle x,y\rangle}$$

gde zadnja jednakost važi akko $\mathbf{B}\cdot\mathbf{B}^\top = \det(\mathbf{B})\cdot\mathbf{I}$, gde je $\mathbf{I}$ jedinična matrica. Stoga, ako je $\langle x, y\rangle \in S$, algoritam za enkripciju konkretno računa $\langle x, y\rangle$.

2.6 Šeme FE sa više ulaza

FE sa više ulaza (eng. Multi-input functional encryption - MIFE), prvi put predstavljena u radu Goldwasser et al. [GGG+14], je varijanta FE kod koje se koriste funkcije sa više ulaza $f(\mathbf{x}_0, ..., \mathbf{x}_n)$, za $n \in N$. Prethodno opisane šeme FE se mogu posmatrati kao specijalan slučaj MIFE, za $n = 1$.

MIFE šeme imaju tačno n slotova za enkripciju - u svaki slot se smešta po jedna šifrovana poruka. Poruke se šifruju potpuno nezavisno jedne od drugih, što omogućava popunjavanje slotova u proizvoljnim vremenskim trenucima, proizvoljnim redosledom. U procesu dekripcije se simultano dešifruje svih n slotova, gde i-ti slot koristi kao i-ti argument funkcije $f(\mathbf{x}_0, ..., \mathbf{x}_n)$. Kao rezultat dekripcije se i dalje dobija jedinstvena vrednost.

MIFE šeme su veoma korisne u slučajevima gde podaci postaju dostupni u različitim vremenskim trenucima, ili ih dostavljaju različiti entiteti, pa ih je veoma pogodno šifrovati odvojeno. U praksi, ovakva šema je veoma primenljiva kada postoji više korisnika od kojih svaki dostavlja deo podataka. Tipičan primer je obrada rezultata glasanja, gde svaki od n glasača šifruje svoj glas i šalje ga serveru na obradu.

2.6.1 FHMultiIPE

Šema za Enkripciju unutrašnjeg proizvoda, sa skrivanjem funkcije sa više ulaza (Function-Hiding Multi-Input Inner-Product Encryption - FHMultiIPE [DOT18]) je još opštija, ali i kompleksnija šema od prethodno razmatrane FHIPE. Složena konstrukcija ove šeme neće biti objašnjena u ovom radu. Zbog veoma efikasnih primitiva na kojima je zasnovana, i pogodnosti za njeno korišćenje u praksi, razmotrićemo princip funkcionisanja ove šeme, koji je dovoljan za shvatanje ...rada

U ovoj šemi postoji sigurnosni parametar $k \in N$, koji direktno utiče na nivo sigurnosti šeme. Originalna poruka (vektor) x_i od n elemenata se posle šifrovanja sastoji iz $2n+2k+1$ elemenata, a označavamo ga c_{x_i}. Kako ova šema koristi funkciju sa više ulaza, potrebno je enkriptovati m ovakvih vektora, svaki za po jedan ulaz funkcije f. Svih m enkriptovanih vektora zajedno čine c_x. Ključ za funkcionalnu dekripciju ima dimenzije $m \times 2n + 2k + 1$, kao i c_x. Analogno šifrovanju vektora x, izvođenje ključa za funkcionalnu dekripciju polazi od matrice y dimenzija $m \times n$. Zbog činjenice da se ova šema zasniva na asimetričnim bilinearnim grupama, tokom

dekripcije se svaki element iz c_x uparuje sa svakim elementom sk_f. Nakon uparivanja, sledi sumiranje svih rezultata, pa operacija nalaženja diskretnog logaritma, a kao rezultat, dobija se unutrašnji proizvod matrica x i y.

3 Pregled sistema

3.1 Razvoj merenja električne energije

Prvi električni merači, često mehaničke prirode, koristili su jednotarifni sistem naplate - potrošači su plaćali ujednačenu cenu električne energije bez obzira na vreme tokom dana kada su je trošili. Ovaj jednostavan sistem dobro je funkcionisao u početnim fazama razvoja električne mreže. Međutim, kako su zahtevi za snabdevanjem rasli, pojavila se potreba za upravljanjem vršnog opterećenja. Uvedeni su merači sa dvostrukom tarifom, koji su omogućavali različitu naplatu tokom vršnih i vanvršnih sati, time podstičući korisnike da ostvaruju veću potrošnju kada je potražnja manja. Koncept je dalje sazreo sa pojavom merača s tri tarife, koji uvode dodatan, prelazni period, sa srednjom tarifom. Ovakve tradicionalne metode merenja i naplaćivanja električne energije, posebno za stambene potrošače, podrazumevale su instalaciju brojila koji su ručno čitani, najčešće na mesečnom nivou.

Sa razvojem pametnih merača i njihovim ulaskom u široku upotrebu, došlo je do značajne promene u načinu naplate električne energije. Ovi uređaji su omogućili još granularniju naplatu - podaci o potrošnji se skupljaju u manjim vremenskim intervalima, a moguće je definisati različitu cenu struje za svaki od tih intervala. Formula za naplatu u ovakvom sistemu tarifiranja se može predstaviti kao

$$\sum_{n-1}^{i=0} r_i u_i,$$

gde je n broj perioda merenja potrošnje tokom jednog perioda naplate, r_i cena električne energije za period merenja i, a u_i količina utrošene električne energije u periodu i. Ova formula se može prestaviti i u obliku unutrašnjeg proizvoda vektora:

$$\langle \mathbf{r}, \mathbf{u} \rangle, \tag{3.1}$$

gde su $\mathbf{r}$ i $\mathbf{u}$ n-dimenzioni vektori, $\mathbf{r} = (r_0, ..., r_{n-1})$ i $\mathbf{u} = (u_0, ..., u_{n-1})$.

U Sjedinjenim Američkim Državama, merači koji se koriste u opisanu svrhu skupljaju detalje o potrošnji veoma malim intervalima, često i na svakih pet minuta, a te podatke zatim šalju kompaniji za snabdevanje električnom energijom. Analizom prikupljenih podataka može doći do ličnih informacija o korisnicima, kao što su prisustvo ili odsustvo ljudi u domaćinstvu, raspored spavanja i vremena obroka[15]. U ekstremnim slučajevima, kada je period merenja dovoljno mali, čak je moguće i otkriti koji se televizijski kanal trenutno gleda u tom domaćinstvu.[16] Zbog toga, ovakav novi model merenja potrošnje doveo je i do problema zaštite privatnosti potrošača.

3.2 Uloga FE

Potreba za zaštitom podataka dobijenih merenjem potrošnje struje, kao i mogućnost da se cena potrošnje dobije koristeći formulu 3.1, podstiče grupu autora Im, Kwon, Jeon i Lee da primene enkripciju unutrašnjeg proizvoda u svrhu rešavanja ovog problema [IKJL19]. U navedenom radu, autori koriste FHIPE šemu izloženu u poglavlju 2.5.1. Predloženim rešenjem postižu zadovoljavajuću zaštitu podataka uz potpuno očuvanje granularnosti i bez narušavanja kvaliteta rezultata merenja. Za obradu rezultata merenja za period od mesec dana, njihov sistem enkriptuje izmerene vrednosti i dekriptuje rezultat za nešto više od 0,5s, što dokazuje da je FE odgovarajuć kriptografski koncept za rešavanje izloženog problema.

3.3 Pregled sistema

Sistem koji će biti opisan u ovom radu ima za cilj da obezbedi mogućnost izračunavanja svote koju bi korisnik trebalo da plati snabdevaču električnom energijom, uz rešavanje prethodno navedenog problema privatnosti podataka. Sistem će koristiti FHIPE šemu, opisanu u 2.5.1, a već korišćenu u [IKJL19], ako i FHMultiIPE šemu, opisanu u 2.6.1.

U sistemu postoje četiri uloge: klijent, snabdevač električnom energijom, senzor i autoritet od poverenja. Klijent, kao spoljni akter, pristupa sistemu preko REST API-ja, dok su ostale tri uloge realizovane kao tri različita podsistema namenjena da se izvršavaju kao web servisi.

Klijent pretplaćivanjem kod snabdevača pristaje da plati naknadu za potrošenu struju, ali želi da od snabdevača sakrije konkretne izmerene vrednosti.

Senzor je pametni merač koji klijent ugrađuje umesto strujomera, koji na konfigurabilan period može meriti potrošnju električne energije, i slati podatke o potrošnji snabdevaču.

Snabdevač električnom energijom je kompanija koja želi da sakuplja podatke sa senzora svojih klijenata, i tako odredi cenu za potrošenu električnu energiju.

3.4 Pouzdanost sistema i uloga autoriteta od poverenja

Pretpostavljamo da je senzor zaštićen od neovlašćenog pristupa, da precizno meri i pouzdano izveštava o potrošnji električne energije. Takođe, pretpostavljamo i da je snabdevač 'pouzdan ali znatiželjan' - korektno sprovodi protokol izračunavanja, ali može probati da dođe do informacija o potrošnji klijenta. Radi kontrole snabdevača, u protokol je uveden dodatni entitet - autoritet od poverenja.

Autoritet od poverenja je regulatorno telo koje nadgleda ceo proces, i time sprečava snabdevača da pokuša da dođe do informacija o potrošnji klijenta koje nije predviđeno da sazna. Zaduženja autoriteta su:
(1) Instanciranje same šeme funkcionalne enkripcije na osnovu parametara dostavljenih od strane snabdevača.
(2) Siguran transport parametara za enkripciju do senzora, radi otklanjanja mogućnosti da snabdevač podmetne parametre kako bi neovlašćeno analizirao potrošnju struje.
(3) Provera cena struje u odnosu na izabranu tarifu.
(4) Generisanje ključa za funkcionalnu dekripciju koji je potreban snabdevaču radi napate.

Autoritet od poverenja nikada ne postupa drugačije od propisanog protokola.

3.5 Osnovni pojmovi

U svrhu boljeg razumevanja dizajna sistema, u nastavku će biti data objašnjenja osnovnih termina koji će se u koristiti u nastavku. Objašnjenja koja slede su uprošćena, a pojedini elementi su izostavljeni, jer će ova tema biti detaljnije obrađena u odeljku 4.2.

Period merenja (eng. *sampling time*) je vreme koje protekne između dva očitavanja

vrednosti potrošene struje.

Cena struje (eng. *rate*) je svota novca koju klijent plaća po jedinici utrošene električne energije, i definiše se za svaki period merenja posebno.

Tarifa (eng. *tariff*) je specifikacija pomoću koje se određuju konkretne cene struje, ali i parametri rada senzora. Klijent od snabdevača dobija informacije o raspoloživim tarifama, ali ne i konkretne cene struje. Takođe, pomoću tarife autoritet proverava da li su cene struje koje je snabdevač odredio odgovarajuće.

Pretplata podrazumeva period u toku kog snabdevač isporučuje struju i prikuplja podatke o potrošnji sa senzora klijenta. Pre početka pretplate, klijent bira tarifu koju želi, kao i vreme početka i dužinu pretplate.

Paket (eng. *batch*) označava enkriptovani vektor rezultata merenja, koji se šalje od servera ka snabdevaču u jednom navratu. **Veličina paketa sa rezultatima merenja** (eng. *batch size*) je broj koji označava koliko rezultata merenja se smešta u jedan paket.

Parametri rada senzora specificiraju konfiguraciju senzora koja je primenjena u toku pretplate klijenta na svakom od senzora klijenta. Ovi konfiguracija uključuje period merenja, početak pretplate, kao i veličina paketa sa rezultatima merenja.

3.6 Pregled protokola naplate električne energije

Protokol naplate električne energije možemo podeliti u četiri faze: fazu registracije, pretplate, merenja i naplate.

Faza registracije. Klijent kreira novi nalog na serveru snabdevača, a zatim svoje senzore registruje na isti server. Ovim se omogućava da se merenja na svom sensorima agregiraju, i da se na kraju isteka pretplate odredi jedinstvena cena za potrošnju očitanu na svim senzorima.

Faza pretplate. Snabdevač najpre dostavlja detalje svih raspoloživih tarifa klijentu. Klijent bira tarifu koju želi, vreme početka i dužinu pretplate. Na osnovu ovih detalja, snabdevač definiše konkretne cene struje, a autoritet, na zahtev snabdevača, generiše ključeve za enkripciju i funkcionalnu dekripciju. Senzor se konfiguriše odgovarajućim parametrima rada.

Faza merenja. Kada dođe vreme početka pretplate, senzori počinju da mere potrošnju, a rezultate merenja šifruju i šalju na server snabdevača u paketima veličine definisane konfiguracijom.

Faza naplate. Snabdevač, na osnovu šifrovanih merenja i ključa za dekripciju, određuje cenu potrošene struje. Klijent može da proveri ovaj iznos kontaktiranjem snabdevača.

3.7 Funkcionalnosti van opsega teme projekta

Ovaj rad ima za cilj da demonstrira prednosti korišćenja šema FE u svrhu zaštite podataka merenja. U skladu sa tim, određeni delovi sistema su uprošćeni. Po potrebi, svaki od delova bi se mogao zameniti, uz potpuno očuvanje funkcionalnosti ostatka sistema.

1. Očitavanje vrednosti na senzoru bi zahtevalo poznavanje konkretne hardverske implementacije senzora. U ovoj implementaciji, pretpostavljen je interfejs za o;itavanje vrednosti, a iza konkretne implementacije je generator pseudoslučajnih brojeva.

2. Generisanje cena struje bi u praksi zahtevalo uzimanje u obzir više faktora, kao što su cena po kojoj je struja proizvedena, doba dana, godišnje doba i slično. U izloženom sistemu će se ove cene generisati na pseudoslučajan način, u opsegu koji će biti zadat tarifom.

3. U praksi bi se odobravanje cena struje na osnovu tarife moglo realizovati modelom mašinskog učenja, ali u ovom sistemu će postojati poseban API kojim bi ovlašćena lica mogla da ručno odobre, odnosno odbiju ove cene.

4. Iako FE omogućava da se podaci o potrošnji zaštite od snabdevača, u implementaciji sistema se svi podaci prenose HTTP protokolom, što ih čini nedovoljno zaštićenim. Za produkcijsko okruženje, neophodno bi bilo korišćenje HTTPS protokola.

4 Realizacija sistema za naplatu električne energije

U ovoj glavi će biti opisani tehnički detalji realizovanog sistema za naplatu električne energije.

4.1 Programski jezik i korišćene biblioteke

Sistem je realizovan u programskom jeziku Go verzije 1.17. Za izradu sistema, iskorišćene su biblioteke GoFE, Gin i logrus.

GoFE [2] je kriptografska biblioteka pisana u programskom jeziku Go koja nudi efikasne implementacije šema FE, među kojima su šeme za Enkripciju unutrašnjeg proizvoda koje će biti korišćene u ovom sistemu.

Gin [1] je HTTP web softverski okvir (eng. *framework*) pisan u programskom jeziku Go. U opisanom sistemu, ovaj softverski okvir će biti korišćen za implementaicju REST API-ja za servise unutar sistema.

Logrus [3] je biblioteka koja omogućava finu kontrolu evidentiranja poruka o statusu sistema (eng. *logging*). Upravo u ovu svrhu će biti korišćena u opisanom sistemu.

4.2 Najvažnije strukture podataka

U odeljku 3.5 su ukratko opisani osnovni pojmovi koji se koriste u sistemu. U nastavku slede detaljniji opisi struktura.

Tarifa

Struktura Tariff predstavlja tarifu koju snabdevač nudi klijentima. Njena polja su:

id	jedinstveni identifikator tarife,
Description	tekstualni opis tarife, namenjen klijentu,
SamplingPeriod	period očitavanja potrošnje, izražen u milisekundama,
BatchSize	broj rezultata merenja koji se zajedno šifruje i šalje serveru,
MaxSampleValue	maksimalna dozvoljena potrošnja struje po periodu merenja, ili najveća količina električne energije koju snabdevač može isporučiti u ovom periodu
MaxRateValue	maksimalna cena struje koju snabdevač može odrediti.

Parametri merenja

Parametri merenja, odnosno struktura SamplingParams, čuva sve parametare potrebne za rad senzora. Sastoji se iz sledećih polja:

Start	vreme početka pretplate, kada se senzor resetuje, a počinje isporuka struje do klijenta,
BatchCnt	ukupan broj paketa (eng. *batch*), odnosno vektora sa šifrovanim merenjima, koje svaki senzor šalje serveru,
SamplingPeriod	kao u strukturi Tariff,
BatchSize	kao u strukturi Tariff,
MaxSampleValue	kao u strukturi Tariff.

Paket

Paket, ili Batch, je struktura podataka koju koristi senzor, u kojoj se čuvaju rezultati merenja i metrika vezana za enkripciju. Ova struktura ....

Pretplata

Na svakom od podsistema postoji posebna struktura u kojoj se čuvaju metrike i podaci o pretplati relevantni za dati podsistem. U sva tri podsistema, ova struktura je nazvana Task.

Na serveru, ova struktura čuva podatke o odabranoj tarifi, cene struje za datu pretplatu, šifrovana merenja primljena od senzora i ključ za funkcionalnu dekripciju.

Na senzoru, ova struktura čuva parametre konfiguracije senzora, pakete i parametre za enkripciju paketa.

Na autoritetu, u ovoj strukturi sačuvani su identifikatori senzora koji mere potrošnju, kao i parametri za enkripciju potrebni senzorima, i parametri za dekripciju potrebni serveru snabdevača.

Pretplate se na podsistemima kreiraju na osnovu struktura ServerTaskRequest, SensorTaskRequest, odnosno AuthorityTaskRequest koje se prosleđuju kroz tela HTTP POST zahteva.

4.3 Funkcionalna Enkripcija u posmatranom sistemu

U ovom odeljku biće obrazložen način korišćenja postojećih implementacija šema FE, kao i način na koji je bibliotečka implemenacija nadograđena radi postizanja boljih performansi.

4.3.1 Korišćene šeme FE

U implementaciji projekta korišćene su dve šeme za Enkripciju unutrašnjeg proizvoda, uz skrivanje funkcije - FHIPE i FHMultiIPE, opisane u odeljcima 2.5.1 i 2.6.1, respektivno. Implementacije ovih šema se nalaze u paketu innerprod/fullysec biblioteke GoFE.

U ovom sistemu, FHIPE se koristi samo u slučaju kada klijent koristi tačno jedan senzor, i kada se svi rezultati merenja šalju odjednom na server, tačnije, u jednom paketu. U svim preostalim slučajevima, koristi se šema FHMultiIPE.

4.3.2 Paralelizacija FHMultiIPE

O principu rada ove šeme je bilo reči u glavi 2.6.1.

U bibliotečkoj implementaciji ove šeme, proces dekripcije otpočinje tek nakon što se prikupe svi vektori c_{x_i}. Na početku dekripcije najpre treba izvršiti odgovarajuće operacije uparivanja, zatim sumiranje rezultata i na kraju nalaženje diskretnog logaritma. Pošto je uparivanje dva elementa u c_x i sk_f nezavisno od svih ostalih elemenata ovih struktura, uočavamo da je uparivanje elemenata u redu i moguće otpočeti odmah kada c_{x_i} i i-ti red sk_f postanu dostupni. Zbog toga, implementaciju ove šeme nadograđujemo tako da podrži paralelno procesiranje.

U paket fullysec u biblioteci GoFE dodata je struktura FHMultiIPEParallelDecryption, koja čuva tekuću kumulativnu sumu uparenih elemenata iz već obrađenih redova.

Kada vektor c_{x_i} postane raspoloživ, server ga prosleđuje u metodu ParallelDecryption koja izvršava operaciju uparivanja svih elemenata prosleđenog vektora sa odgovarajućim elementima ključa za dekripciju, sumira rezultate uparivanja, i dodaje ih već spomenutoj kumulativnoj sumi.

Algorithm 1 ParallelDecryption Algorithm

Require: $slotIdx \in \mathbb{Z}, c_{x_i} \in VectorG_1, key \in MatrixG_2$
Ensure: Returns an error, if it occurs
  if *idx* out of bounds **then**
    return *error*
  end if
  if batch at index *idx* already encrypted **then**
    return *error*
  end if
  $localSum \leftarrow 0$
  for $idx \leftarrow 0$ **to** $len(c_{x_i})$ **do**
    $paired \leftarrow \text{Pair}(c_{x_i}[idx], key[slotIdx][idx])$
    $localSum \leftarrow \text{Add}(paired, localSum)$
  end for
  $globalSum \leftarrow \text{Add}(globalSum, localSum)$
  return nil

Kada se obrade svi c_{x_i} za $0 \leq i < m$, server poziva funkciju GetResult da bi dohvatio konačan rezultat dekripcije, tačnije vrednost unutrašnjeg proizvoda.

Ova modifikacija ni na koji način ne može da uspori rad postojećeg algoritma. Količinu postojećeg posla je nepromenjena, ali umesto da se ceo posao uradi tek u dekripciji, što produžava vreme čekanja na rezultat, delovi posla se obavljaju pre iniciranja dekripcije.

Algorithm 2 GetResult Algorithm

Require: $searchForNegativeResult \in \{\text{true, false}\}, pubKey \in G_T$
Ensure: Returns decrypted value or an error
  if there are unprocessed batches **then**
    return *error*
  end if
  $maxSumValue \leftarrow \text{Mul}(MaxSampleValue, BatchSize, BatchCnt, MaxRateValue)$
  if $searchForNegativeResult$ **then**
    $minSumValue \leftarrow -maxSumValue$
  else
    $minSumValue \leftarrow 0$
  end if
  find discrete logarithm $z \in (minSumValue, maxSumValue)$ of $pubKey^z = globalSum$
  return z

4.3.3 Ostvarivanje tajnosti pomoću FE

Enkripcija izmerenih vrednosti

Senzor najpre smešta rezultate merenja u vektor x_i. Kada se u ovaj vektor doda tačno $BatchSize$ rezultata, vektor se šifruje, i nastaje paket (eng. *batch*) u oznaci c_{x_i}. Svaki paket je predstavlja jedan red strukture c_x, a i ozna;ava broj reda paketa u ovoj strukturi.

vaki senzor će poslati $BatchCnt$ ovakvih paketa, što znači da će server primiti ukupno $BatchCnt \cdot SensorCnt$ paketa, ukoliko je $SensorCnt$ broj senzora.

Generisanje šeme FE

Odluka o tome koja će se šema koristiti, kao i instanciranje odgovarajuće šeme FE se vrši od strane autoriteta od poverenja.

Šema FE se bira u odnosu na broj očekivanih paketa. Ukoliko se očekuje samo jedan paket (a to je u slučaju kada radi samo jedan senzor, i kada on sva merenja šifruje odjednom), koristi se šema FHIPE, a u suprotnom FHMultiIPE. Strukture x, c_x, y i sk_y su vektori ako se radi sa FHIPE, odnosno matrice sa m redova ako se radi o FHMultiIPE. Radi uniformnog referisanja na ove strukture u oba slučaja, kroz ovaj rad će nabrojane strukture biti smatrane matricom sa jednim redom u slučaju FHIPE šeme.

U nastavku slede parametri potrebni za kreiranje šeme FE, kao i vrednosti koje server prosleđuje autoritetu na osnovu kojih se ovi parametri računaju:

n	dužina vektora - odgovara vrednosti $BatchSize$,
m	broj vektora - odgovara vrednosti $BatchCnt \cdot SensorCnt$,
$boundX$	najveća dozvoljena vrednost elementa u matrici x, odnosno maksimalna moguća potrošnja - odgovara vrednosti $MaxSampleValue$,
$boundY$	najveća dozvoljena vrednost elementa u matrici y, odnosno cene struje - odgovara vrednosti $MaxRateValue$.

Za šemu FHMultiIPE potreban je i sigurnosni parametar $SecKey$, koji će u sistemu biti definisan kao konstanta.

Nakon što autoritet dobije od servera potrebne parametre, generiše odgovarajuću šemu FE, a zatim i javni (mpk) i glavni privatni ključ (msk).

Generisanje ključa za funkcionalnu dekripciju

Za generisanje ključa za funkcionalnu dekripciju, neophodne su cene struje za svaki period merenja tokom trajanja pretplate, kao i glavni tajni ključ msk. Ovaj ključ generiše autoritet, koji ujedno i čuva msk.

Server dostavlja autoritetu tačno $BatchSize \cdot BatchCnt$ cena struje, po jednu za svaki period merenja tokom trajanja pretplate. Cena struje koja važi za određeni period merenja se koristi za merenja sa svih senzora u tom periodu. Zbog toga, autoritet ovu sekencu cena smešta po vrstama u matricu y, ali tako da je ponovi tačno $SensorCnt$ puta, za svaki senzor po jednom.

Dekripcija unutrašnjeg proizvoda

U slučaju FHIPE šeme, enkripcijom merenja dobija se vektor c_x, koji se šalje na server, gde se odmah otpočinje sa dekripcijom rezultata.

U slučaju FHMultiIPE šeme, enkripcijom merenja dobija se c_{x_i}. Ovaj vektor se šalje na server, i najpre smešta u tačno određen slot slot za enkripciju pri FHMultiIPE šemi. Zbog opisane podrške za paralelizacijom dekripcije, funkcijom *ParallelDecrypt* se ovaj vektor odmah pretprocesira. Tek nakon što se prime c_{x_i} za $0 \leq i < n$, odnosno popune svi slotovi u c_x, može se otpočeti računanje konačnog rezultata.

Napominjemo da cene struje i izmerene vrednosti moraju biti celi brojevi, jer su vektori $x, y \in \mathbb{Z}_q^n$, odnosno vektori

4.4 Skaliranje decimalnih vrednosti

Šeme FE koje će biti korišćene rade isključivo sa vektorima celih brojeva. Ukoliko su cene struje ili očitana merenja decimalni brojevi, moraju se na odgovarajući način skalirati pre upisivanja u vektore.

Nakon određene maksimalne cene struje r, određuje se i broj značajnih decimala cene dr. Tada je $MaxRateValue = r * 10^{dr}$.

Analogno, određujemo maksimalnu dozvoljenu potrošnju tokom perioda s i broj značajnih cifara ds. Tada je $MaxSampleValue = s * 10^{ds}$.

Po dekripciji rezultata, dobijenu vrednost je potrebno podeliti sa $10dr + ds$.

4.5 Protokol komunikacije

U ovom odeljku će biti detaljnije objašnjeno kako se odvijaju četiri faze protokola sigurne naplate električne energije. Procedure vezane za FE su već detaljno opisane, tako da će u ovom odeljku biti samo referisane.

Posebne obrade, kao što su generisanje ključeva, šifrovanje i dešifrovanje, kao i odobravanje cena struje će biti detaljnije obrađene u narednim odeljcima, ali će biti referisane na dijagramima u okviru ovog odeljka.

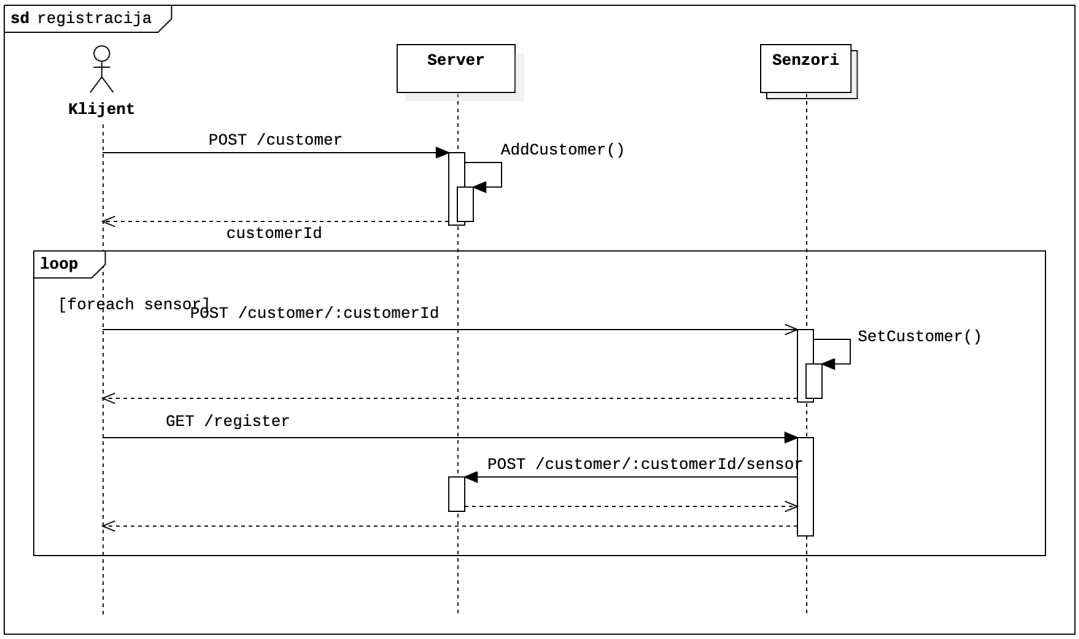
Faza registracije

Dijagram 4.1 prikazuje fazu registracije klijenta na sistem.

Klijent najpre kreira svoj nalog na serveru, i dobija svoj jedinstveni identifikator, kojim parametrizuje sve svoje senzore. Nakon toga, klijent šalje zahtev svakom senzoru da se registruje na server snabdevača, zajedno sa IP adresom tog servera. Tokom ove registracije, senzor dostavlja identifikator klijenta, kao i svoj jedinstveni identifikator, na osnovu kojih ga server prepoznaje. Nakon što klijent doda željene senzore, spreman je za fazu pretplate.

Faza pretplate

Dijagram 4.2 prikazuje fazu pretplate klijenta na sistem.



Dijagram 4.1: faza registracije.

Klijent od servera najpre zahteva pregled raspoloživih tarifa. Zatim, bira tarifu, vreme početka i dužinu pretplate, i ove detalje šalje serveru.

Server ih obrađuje, i generiše parametre koji su potrebni autoritetu za instanciranje šeme FE. Zatim ih prosleđuje autoritetu, i čeka na potvrdu da su parametri generisani.

Nakon pristigle potvrde, server paralelno šalje detalje o pretplati svim senzorima, i započinje pripremu za izvođenje ključa za funkcionalnu dekripciju.

Nakon što svaki senzor dobije detalje o pretplati, dohvata parmetre za enkripciju od autoriteta, i počinje da meri potrošnju. Zbog prethodno sačekane potvrde autorietea o generisanju šeme FE, ovi parametri su sigurno već dostupni.

Prvi korak u dobijanju ključa za dekripciju se sastoji iz generisanja konkretnih cena struje za svaki period merenja. Ove cene se šalju autoritetu, i čeka se na odobrenje ovih cena i generisanje ključa. Ukoliko autoritet ne odobri cene, server će generisati nove cene struje.

Faza merenja

Dijagram 4.3 prikazuje fazu merenja potrošnje struje. Kada dođe vreme početka pretplate, (što je ujedno i vreme početka merenja potrošnje), sensori se resetuju kako bi poništili prethodno izmerenu potrošnju. Zatim, na regularne vremenske intervale (na $SamplingPeriod$ sekundi),

očitavaju potrošnju električne energije. Posle *BatchSize* merenja, rezultati se šifruju, a šifra šalje na server snabdevača. Ovaj postupak se ponavlja *BatchCnt* puta, tačnije za svo vreme trajanja pretplate.

Po prijenu šifrovanih merenja, u slučaju FHMultiIPE šeme, server snabdevača obavlja parcijalnu obradu nad primljenim podacima, opisanu funkcijom *ParallelDecrypt*.

Faza naplate

Dijagram 4.4 prikazuje fazu naplate za potrošenu struju. Kada se obave sva merenja, senzori pošalju šifrovane rezultate, i server snabdevača ih parcijalno obradi, rezultat se dekriptuje koristeći ključ za funkcionalnu dekripciju dobijen od autoriteta koji odgovara cenama struje za odabranu tarifu. Nakon što je dobijen rezultat, klijent može zatražiti od snabdevača uvid u rezultat, nakon čega se očekuje plaćanje.

4.6 Peroformanse sistema

U ovom odeljku biće razmatrane performanse sistema, i to kroz poređenje vremena izvršavanja za po dve pretplate.

4.6.1 Tratki test za FHIPE i FHMultiIPE

Poređenje ove dve šeme izvršeno je tako da se tokom obe pretplate potrošnja izmeri tačno 60 puta. Tokom korišćenja FHIPE šeme, senzor šalje jedan vektor sa 60 rezultata merenja, dok kod FHMultiIPE, senzor šalje šest vektora sa po deset rezultata.

U nastavku sledi pregled log fajlova.

Listing 4.1: Log autoriteta, šema FHIPE

```
[2023-08-31 01:05:28] INFO - Task params: {
  "Id": "f63ad78c-42fa-4537-8604-98866c6206b2",
  "SensorIds": [
    "922eeb3e-62d1-4ae3-b34e-746a2b30655e"
  ],
  "batchSize": 60,
  "batchCnt": 1,
  "maxRateValue": 200000,
  "maxSampleValue": 300,
  "enableEncryption": true
}
```

```
[2023-08-31 01:05:28] INFO - using SingleFE
[2023-08-31 01:05:28] INFO [fe param generator] - generating FE Schema
[2023-08-31 01:05:28] INFO [fe param generator] - setting FE Schema Params
[2023-08-31 01:05:28] INFO [fe param generator] - generating FE Master Key
[2023-08-31 01:05:28] INFO [fe param generator] - elapsed: 138705804 ns
[2023-08-31 01:05:28] INFO [fe param generator] - FE Master key generation succeeded
[2023-08-31 01:05:38] INFO [fe param generator] - deriving decryption key
[2023-08-31 01:05:38] INFO [fe param generator] - sensor no 0 fetched encryption
   params
[2023-08-31 01:05:38] INFO [fe param generator] - elapsed: 75131729 ns
[2023-08-31 01:05:38] INFO [fe param generator] - decryption key derived successfully
[2023-08-31 01:05:43] INFO [fe param generator] - server fetched decryption params
```

Listing 4.2: Log autoriteta, šema FHMultiIPE

```
[2023-08-31 01:08:08] INFO - Task params: {
  "Id": "3090f61e-ef7a-4794-8df9-c0623c2aea01",
  "SensorIds": [
    "922eeb3e-62d1-4ae3-b34e-746a2b30655e"
  ],
  "batchSize": 10,
  "batchCnt": 6,
  "maxRateValue": 200000,
  "maxSampleValue": 300,
  "enableEncryption": true
}
[2023-08-31 01:08:08] INFO - using MultiFE
[2023-08-31 01:08:08] INFO [fe param generator] - generating FE Schema
[2023-08-31 01:08:08] INFO [fe param generator] - setting FE Schema Params
[2023-08-31 01:08:08] INFO [fe param generator] - generating FE Master Key
[2023-08-31 01:08:08] INFO [fe param generator] - elapsed: 81042002 ns
[2023-08-31 01:08:08] INFO [fe param generator] - FE Master key generation succeeded
[2023-08-31 01:08:18] INFO [fe param generator] - deriving decryption key
[2023-08-31 01:08:18] INFO [fe param generator] - sensor no 0 fetched encryption
   params
[2023-08-31 01:08:18] INFO [fe param generator] - elapsed: 80689932 ns
[2023-08-31 01:08:18] INFO [fe param generator] - decryption key derived successfully
[2023-08-31 01:08:23] INFO [fe param generator] - server fetched decryption params
```

Iz datih logova se vidi da je vreme generisanja FE šeme i *mpk* za 50% kraće za FHMultiIPE šemu, nego za FHIPE, dok je vreme izvođenja ključa za dekripciju približno. Ovo može imati značajan uticaj na merenje potrošnje. Naime, ako je zadati početak pretplate relativno blizu, FHIPE šema se pokazuje znatno rizičnijom jer.. kašnjenje

Log senzora zbog opširnosti neće biti izlistan. Enkripcija paketa kod FHIPE šeme traje

oko 38ms. Enkripcija paketa FHMultiIPE šeme traje 5,7 ms, ali zbog slanja 6 paketa, ukupno vreme enkripcije je 34,2ms, što znači da ova šema manje opterećuje senzor, koji svakako ima dosta ograničene raspoložive resurse.

Listing 4.3: Log servera, šema FHIPE

```
[2023-08-31 01:05:28] INFO - Task params: {
  "customerId": "15cd33d1-6a27-466c-9077-715a282b8433",
  "start": 1693440343,
  "duration": 60,
  "tariffId": "701e677e-afab-4f53-86ce-89934d1ceb35",
  "enableEncryption": true
}
[2023-08-31 01:05:28] INFO - setting sensors for task f63ad78c-42fa-4537-8604-98866
    c6206b2
[2023-08-31 01:05:28] INFO - submitting task to authority
[2023-08-31 01:05:38] INFO - fe params ready
[2023-08-31 01:05:38] INFO - submitting task to sensor 922eeb3e-62d1-4ae3-b34e-746
    a2b30655e
[2023-08-31 01:05:38] DEBUG - generated rates: [247 256 24 116 77 134 91 39 80 25 249
     26 215 233 261 61 143 152 59 282 156 192 238 9 136 58 155 189 227 154 34 267 28
    281 194 276 116 27 31 180 289 263 156 57 21 111 157 271 261 69 224 213 278 26 95
    34 130 218 96 108]
[2023-08-31 01:05:38] INFO - sending rates
[2023-08-31 01:05:38] INFO - rates sent successfully
[2023-08-31 01:05:38] INFO - task submitted to sensor 922eeb3e-62d1-4ae3-b34e-746
    a2b30655e
[2023-08-31 01:05:43] INFO - fe decryption params ready
[2023-08-31 01:05:43] INFO - fetching fe decryption params
[2023-08-31 01:05:43] INFO - fe decryption params fetched
[2023-08-31 01:06:43] INFO - adding cipher
[2023-08-31 01:06:43] INFO [fe decryptor] - cipher no 0: decryption time: 171444400
    ns
[2023-08-31 01:06:43] DEBUG - result: 1291538
```

Listing 4.4: Log servera, šema FHMultiIPE

```
[2023-08-31 01:08:08] INFO - Task params: {
  "customerId": "15cd33d1-6a27-466c-9077-715a282b8433",
  "start": 1693440503,
  "duration": 60,
  "tariffId": "df0a7725-1658-48b8-b068-ce0d0ae51d25",
  "enableEncryption": true
}
[2023-08-31 01:08:08] INFO - setting sensors for task 3090f61e-ef7a-4794-8df9-
    c0623c2aea01
```

```
[2023-08-31 01:08:08] INFO - submitting task to authority
[2023-08-31 01:08:18] INFO - fe params ready
[2023-08-31 01:08:18] DEBUG - generated rates: [247 256 24 116 77 134 91 39 80 25 249
    26 215 233 261 61 143 152 59 282 156 192 238 9 136 58 155 189 227 154 34 267 28
   281 194 276 116 27 31 180 289 263 156 57 21 111 157 271 261 69 224 213 278 26 95
   34 130 218 96 108]
[2023-08-31 01:08:18] INFO - sending rates
[2023-08-31 01:08:18] INFO - submitting task to sensor 922eeb3e-62d1-4ae3-b34e-746
   a2b30655e
[2023-08-31 01:08:18] INFO - rates sent successfully
[2023-08-31 01:08:18] INFO - task submitted to sensor 922eeb3e-62d1-4ae3-b34e-746
   a2b30655e
[2023-08-31 01:08:23] INFO - fe decryption params ready
[2023-08-31 01:08:23] INFO - fetching fe decryption params
[2023-08-31 01:08:23] INFO - fe decryption params fetched
[2023-08-31 01:08:33] INFO - adding cipher
[2023-08-31 01:08:33] INFO [fe decryptor] - cipher no 0: partial processing time:
   45305073 ns
[2023-08-31 01:08:43] INFO - adding cipher
[2023-08-31 01:08:43] INFO [fe decryptor] - cipher no 1: partial processing time:
   43356718 ns
[2023-08-31 01:08:53] INFO - adding cipher
[2023-08-31 01:08:53] INFO [fe decryptor] - cipher no 2: partial processing time:
   44106683 ns
[2023-08-31 01:09:03] INFO - adding cipher
[2023-08-31 01:09:03] INFO [fe decryptor] - cipher no 3: partial processing time:
   43777499 ns
[2023-08-31 01:09:13] INFO - adding cipher
[2023-08-31 01:09:13] INFO [fe decryptor] - cipher no 4: partial processing time:
   43782425 ns
[2023-08-31 01:09:23] INFO - adding cipher
[2023-08-31 01:09:23] INFO [fe decryptor] - cipher no 5: partial processing time:
   43790277 ns
[2023-08-31 01:09:23] INFO [fe decryptor] - decryption time: 62843671 ns
[2023-08-31 01:09:23] INFO [fe decryptor] - total decryption time: 346015740 ns
[2023-08-31 01:09:23] DEBUG - result: 1291538
```

Iz priloženih logova servera, evidentno je da dekripcija sa FHMultiIPE traje 346ms, što je oko duplo više nego 171ms, koliko traje dekripcija za FHIPE. Ipak, ako se porede vremena dostupnosti rezultata od trenutka slanja zadnjeg paketa, za FHMultiIPE, ovo vreme je 1005ms, što je dosta brže u odnosu na 171ms, koje odgovara šemi FHIPE.

Ovaj primer dokazuje da, iako ceo proces duže traje ako se koristi FHMultiIPE šema sa modifikacijom opisanom u 4.3.2, benefiti se vide na sva tri podsistema.

4.6.2 Dugi test za FHIPE i FHMultiIPE

Dugi test podrazumeva očitavanje 2880 merenja sa senzora. FHMultiIPE šemom se rezultati šalju u 360 paketa po 8 vrednosti. Ovaj test simulira prikupljanje podataka na mesečnom nivou, kada bi se potrošnja merila na 15 minuta. Tokom korišćenja FHIPE šeme, generisanje mpk i msk je trajalo preko jednog sata, te je test prekinut, zbog ogromne neefikasnosti.. Ključevi za FHMultiIPE su generisani za 2728ms, a ključ za dekripciju za 3486ms. Parcijalna obrada svakog paketa je trajla oko 35ms, a traženje diskretnog logaritma nešto ispod 9s.

Ovakvi rezultati ukazuju da bi se korišćenje FHMultiIPE na mesečnom nivou drastično bolje od FHIPE.

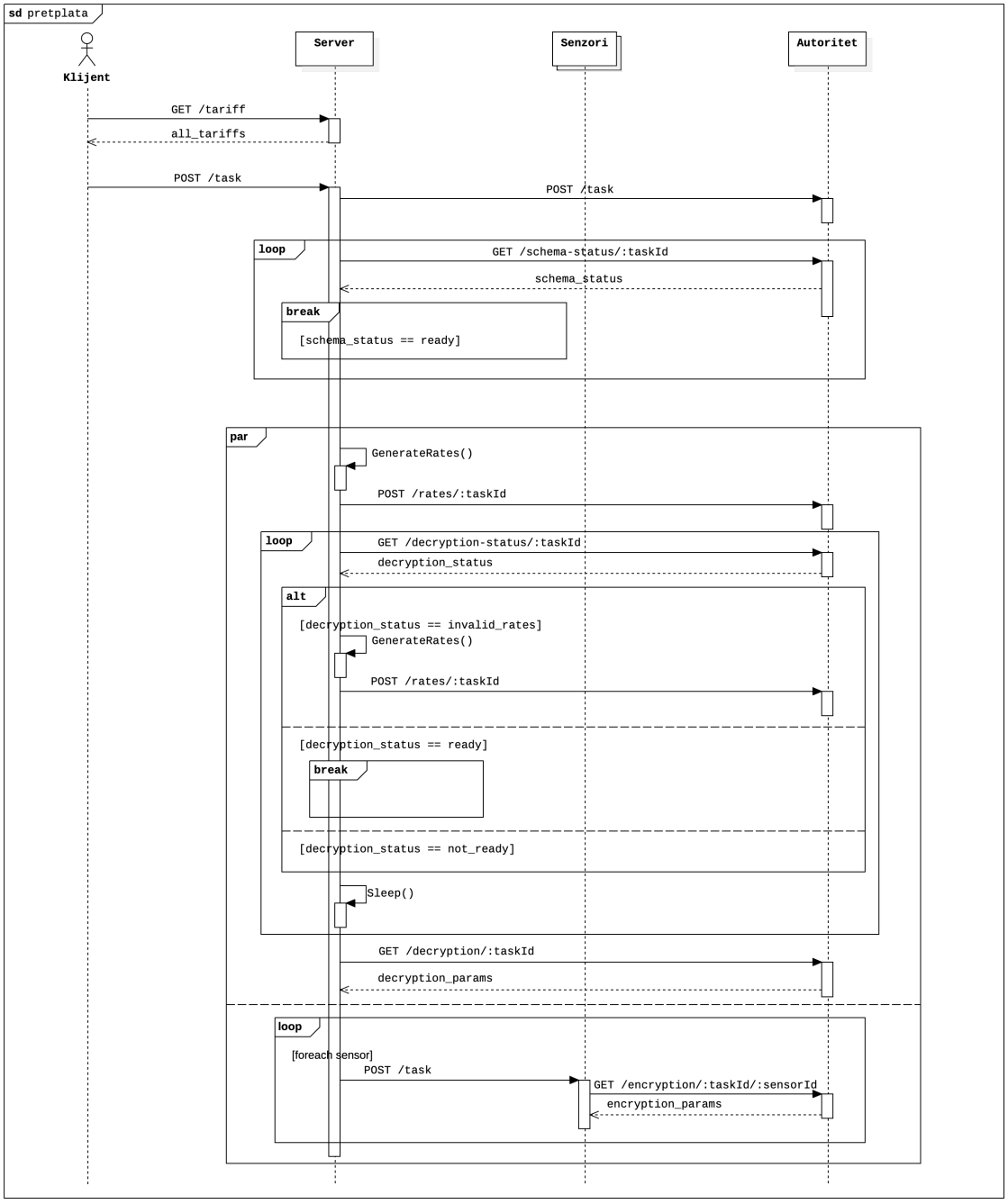
4.6.3 Peroformanse bez enkripcije

Za dve prethodne pretplate, izvedena je simulacija bez enkripcije. U oba slučaja, izračunavanje konačne cene je trajalo 21 µs. S obzirom da je ukupno trajanje simulacije za FHIPE trajalo 348ms, a FHMultiIPPE 433ms, korišćenje šema FE usporava izrašunavanje svote koju klijent plaća 15500, odnosno 20500 puta.

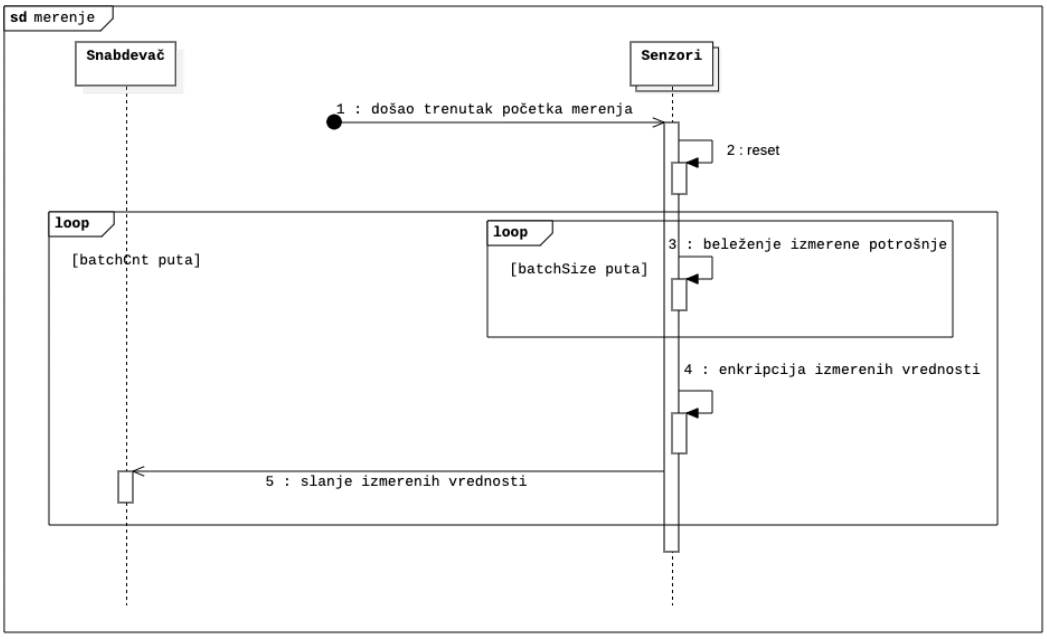
4.7 Zaključak

U prvom poglavlju, obrađena je funkcionalna enkripcija, sa posebnim naglaskom na enkripciju unutrašnjeg proizvoda, uz osvrt na skrivanje funkcije i funkcije sa više ulaza. Takođe, obrađene su i šeme FHIPE i FHMultiIPE. Drugo poglavlje je sadržalo pregled sistema za naplatu, objašnjenje uloga u sistemu kao i osnovnog scenarija pretplate. U trećem poglavlju je detaljnije razrađena implementacija ovog sistema. Četvrto poglavlje sadrži metrike ovakvog sistema za nekoliko različitih pretplata.

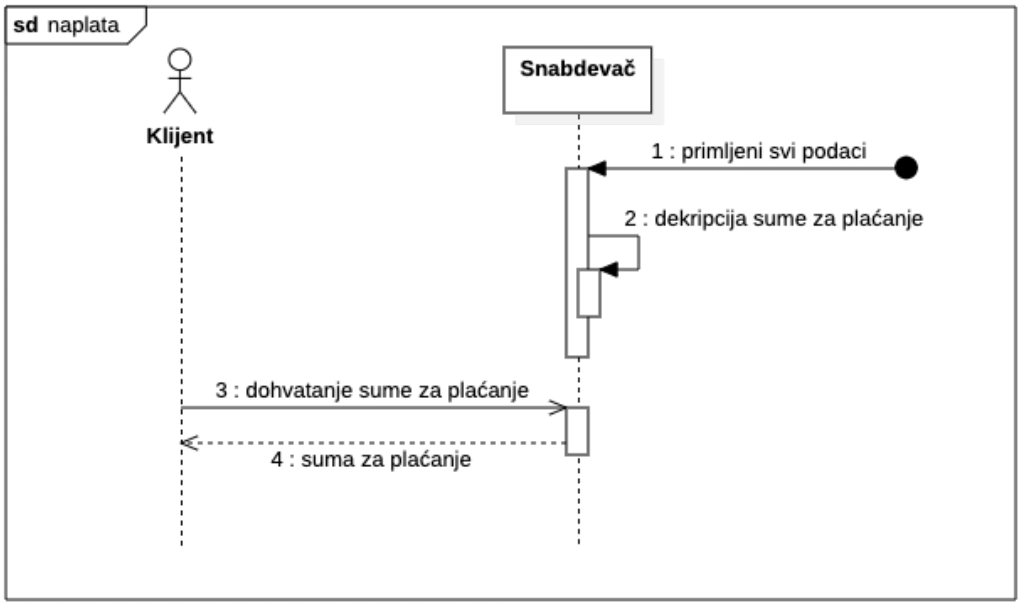
Izmerene performanse potvrđuju da je funkcionalna enkripcija još uvek nezrela za korišćenje u sistemima velikih razmera, zbog pogoršanja performansi do 20000 puta u odnosu na sistem bez enkripcije. Iako je pojava ovakvog koncepta revolucionarna u oblasti kriptografije, evidentno je da naučna zajednica još uvek nije došla do dovoljno dobrog rešenja za brojne izazove u osiguravanju privatnosti podataka.



Dijagram 4.2: faza pretplate.



Dijagram 4.3: faza merenja.



Dijagram 4.4: fazu naplate.

4 Reference

[1] Gin-Gonic. *Biblioteka Gin*. URL: https://github.com/gin-gonic/gin.

[IKJL19] Seong-Yun Jeon Mun-Kyu Lee Jong-Hyuk Im Hee-Yong Kwon. "Privacy-Preserving Electricity Billing System Using Functional Encryption". In: ().

[ABCP15] Angelo De Caro-David Pointcheval Michel Abdalla Florian Bourse. "Simple Functional Encryption Schemes for Inner Products". In: (2015). URL: https://eprint.iacr.org/2015/017.pdf.

[DOT18] Junichi Tomida Pratish Datta Tatsuaki Okamoto. "Full-Hiding (Unbounded) Multi-Input Inner Product Functional Encryption from the k-Linear Assumption". In: (2018). URL: https://eprint.iacr.org/2018/061.pdf.

[2] FENTEC Project. *GoFE Library*. 2019-2022. URL: https://github.com/fentec-project/gofe.

[KLM+18] Avradip Mandal Hart Montgomery Arnab Roy-David J. Wu Sam Kim Kevin Lewi. "Function-Hiding Inner Product Encryption is Practical". In: ().

[3] sirupsen@github. *Biblioteka Logrus*. URL: https://github.com/gin-gonic/gin.