UNIVERZITET CRNE GORE
ELEKTROTEHNIČKI FAKULTET

Bogdan Krivokapić

PROTOKOL ZA UPRAVLJANJE KLJUČEVIMA U PODVODNIM AKUSTIČKIM SENZORSKIM MREŽAMA BAZIRAN NA BLOCKCHAIN TEHNOLOGIJI

-MAGISTARSKI RAD-

Podgorica, 2024.

INFORMACIJE O MAGISTRANTU

Ime i prezime:	Bogdan Krivokapić
Datum i mjesto rođenja:	18.08.1990. Nikšić, Crna Gora.
Naziv završenog osnovnog studijskog programa i godina diplomiranja:	Elektronika, telekomunikacije i računari, 2013.
Naziv završenog specijalističkog studijskog programa i godina diplomiranja:	Elektronika, telekomunikacije i računari – Telekomunikacije, 2014.

INFORMACIJE O MAGISTARSKOM RADU

Naziv postdiplomskog magistarskog studijskog programa:	Telekomunikacije
Naslov rada:	Protokol za upravljanje ključevima u podvodnim akustičkim senzorskim mrežama baziran na *blockchain* tehnologiji
Fakultet:	Elektrotehnički fakultet, Univerzitet Crne Gore

OCJENA I ODBRANA MAGISTARSKOG RADA

Datum prijave magistarskog rada:	11.04.2024.
Datum sjednice Vijeća Univerzitetske jedinice na kojoj je prihvaćena tema:	20.06.2024.
Komisija za ocjenu teme i podobnosti magistranta:	Prof. dr Ana Jovanović Prof. dr Igor Radusinović Doc. dr Slavica Tomović
Mentor:	Prof. dr Igor Radusinović
Komisija za ocjenu rada:	Prof. dr Ana Jovanović Prof. dr Igor Radusinović Doc. dr Slavica Tomović
Komisija za odbranu rada:	Prof. dr Ana Jovanović Prof. dr Igor Radusinović Doc. dr Slavica Tomović
Datum odbrane:	29.10.2024.

Ime i prezime autora: Bogdan Krivokapić, Spec. Sci

ETIČKA IZJAVA

U skladu sa članom 22 Zakona o akademskom integritetu i članom 24 Pravila studiranja na postdiplomskim studijama, pod krivičnom i materijalnom odgovornošću, izjavljujem da je magistarski rad pod naslovom

"Protokol za upravljanje ključevima u podvodnim akustičkim senzorskim mrežama baziran na blockchain tehnologiji"

moje originalno djelo.

Podnosilac izjave,

Bogdan Krivokapić, Spec. Sci

Bogdan Krivokapić

U Podgorici, dana 29.10.2024. godine

Sažetak

Predmet ovog istraživanja je primjena *blockchain* tehnologije za unapređenje sigurnosti i efikasnosti komunikacije u podvodnim akustičkim senzorskim mrežama (*Underwater Acoustic Sensor Networks* - UASNs) sa klasterskom organizacijom. Osnovni cilj istraživanja je razvoj protokola koji omogućava sigurnu i brzu autentifikaciju uređaja, uz istovremeno otklanjanje potrebe za čestim reautentifikacijama, koje se javljaju zbog mobilnosti uređaja između različitih klastera. Takođe, cilj je i poboljšanje ukupne sigurnosti uzimajući u obzir jedinstvene izazove prisutne u ovakvim mrežama, poput ograničenih resursa i povećane podložnosti napadima, primjenom inovativnog pristupa u autentifikaciji i upravljanju ključevima. Predloženi protokol koristi ECQV (*Elliptic Curve Qu-Vanstone*) sertifikate i HOMQV (*Hashed One-time Menezes-Qu-Vanstone*) protokol u kombinaciji sa *blockchain* tehnologijom za zaštitu od aktivnih napada.

U radu je najprije analizirana literatura u oblasti sigurnosti podvodnih senzorskih mreža koja tretira izazove kao što su spora propagacija signala, ograničeni frekvencijski opseg, slabljenje signala i mobilnost nodova. Zatim je predložen protokol za autentifikaciju i upravljanje kriptografskim ključevima koji koristi sigurnosne prednosti *blockchain* tehnologije. Predloženi protokol podrazumijeva implementaciju *blockchain* tehnologije na resursno moćnim površinskim čvorovima mreže, i korišćenje pametnih ugovora za upravljanje sertifikatima i ključevima, automatizovanu evaluaciju povjerenja senzorskih čvorova i vođenje evidencije o njihovoj reputaciji. Na ovaj način eliminisana je potreba za centralizovanom infrastrukturom javnih ključeva i smanjeno je komunikaciono opterećenje procesa autentifikacije. Takođe, kroz integraciju ECQV i HOMQV protokola sa *blockchain* tehnologijom ostvaren je visok nivo robustnosti protiv različitih internih i eksternih napada.

U radu su prikazani rezultati simulacione analize, testiranja protokola na stvarnim podvodnim uređajima, kao i detaljna analiza sigurnosti. Predloženo rješenje je upoređeno sa *state-of-the-art* rešenjima iz literature, pri čemu je pokazano da nameće

manje računarsko i komunikaciono opterećenje, dok istovremeno pokazuje otpornost na širok dijapazon napada, što je potvrđeno AVISPA alatom. Eksperimentalna implementacija na stvarnim podvodnim i površinskim uređajima potvrdila je njegovu efikasnost.

Korišćenjem *blockchain* tehnologije, uspostavljen je siguran, skalabilan i efikasan komunikacioni okvir za misije od kritičnog značaja u UASN aplikacijama. Ovo omogućava pouzdanu i sigurnu operativnost u raznim primjenama kao što su ekološki monitoring, industrijska eksploatacija resursa i vojne operacije, naglašavajući važnost integracije *blockchain* tehnologije u podvodne komunikacione sisteme.

Ključne riječi: Autentifikacija – *Blockchain* - Podvodne akustične mreže

Abstract

The subject of this research is the application of blockchain technology to enhance security and efficiency of communication in clustered Underwater Acoustic Sensor Networks (UASNs). The main goal of the research is to develop a protocol that enables secure and fast device authentication, while simultaneously eliminating the need for frequent re-authentications that occur due to device mobility between different clusters. Additionally, the aim is to improve overall security, taking into account the unique challenges present in such networks, such as limited resources and increased vulnerability to attacks, by applying an innovative approach to authentication and key management. The proposed protocol uses ECQV (Elliptic Curve Qu-Vanstone) certificates and HOMQV (Hashed One-time Menezes-Qu-Vanstone) protocol in combination with blockchain technology to protect against active attacks.

The paper first analyzes literature in the field of underwater sensor network security that addresses challenges such as slow signal propagation, limited frequency range, signal attenuation, and node mobility. Then, a protocol for authentication and cryptographic key management is proposed that utilizes the security advantages of blockchain technology. The proposed protocol involves implementing blockchain technology on resource-powerful surface nodes of the network, and using smart contracts for certificate and key management, automated evaluation of sensor node trust, and maintaining records of their reputation. This eliminates the need for a centralized public key infrastructure and reduces the communication load of the authentication process. Also, through the integration of ECQV and HOMQV protocols with blockchain technology, a high level of robustness against various internal and external attacks has been achieved.

The paper presents the results of simulation analysis, protocol testing on real underwater devices, as well as a detailed security analysis. The proposed solution is compared with state-of-the-art solutions from the literature, demonstrating that it imposes less computational and communication load, while simultaneously sho-

wing resistance to a wide range of attacks, which is confirmed by the AVISPA tool. Experimental implementation on real underwater and surface devices confirmed its efficiency.

By using blockchain technology, a secure, scalable, and efficient communication framework has been established for mission-critical UASN applications. This enables reliable and secure operations in various applications such as environmental monitoring, industrial resource exploitation, and military operations, emphasizing the importance of integrating blockchain technology into underwater communication systems.

Keywords: Authentication – Blockchain – Underwater acoustic networks

Sadržaj

Spisak slika

Uvod

Podvodne akustične senzorske mreže (UASN) sastoje se od više prostorno distribuiranih senzorskih čvorova, koji mogu biti fiksirani ili mobilni, i djeluju kooperativno u izazovnom podvodnom okruženju, komunicirajući putem akustičnih talasa. Kako ove mreže sve više dobijaju na značaju, ne samo u naučnim istraživanjima, već i u industrijskim operacijama i inicijativama nacionalne bezbjednosti, sve jasnije se prepoznaje potreba za njihovom zaštitom. S obzirom na to da se implementiraju u okruženju koje je često nepristupačno i podložno raznim zlonamjernim napadima, osiguravanje pouzdane i sigurne komunikacije u takvim uslovima postaje ključno istraživačko pitanje [1–4].

Iako su mnoga dostupna rešenja efikasna u zaštiti zemaljskih bežičnih mreža, njihova direktna primjena u UASN mrežama je problematična iz više razloga. Ova situacija proizlazi iz specifičnih izazova karakterističnih za podvodne akustične kanale, poput veoma male brzine propagacije signala (oko 1500 m/s), ograničenog frekvencijskog opsega i značajnog slabljenja signala. Na primjer, zbog velikih kašnjenja uslijed propagacije, *wormhol*e napadi [5] mogu se relativno lako izvesti korišćenjem vanpojasne radio veze iznad površine vode [6]. Dalje, ograničen propusni opseg postavlja značajne izazove za implementaciju efikasnih sigurnosnih protokola. Standardne metode za mitigaciju takozvanih *replay* napada [5], kao što su vremensko označavanje paketa, nisu efikasne u UASN mrežama zbog ograničene veličine zaglavlja paketa i potrebe za smanjenjem komunikacijskog opterećenja u uslovima male brzine prenosa podataka. Štaviše, strategije zasnovane na vremenskim oznakama rizikuju nepredviđeni gubitak legitimnih paketa zbog značajnih kašnjenja u isporuci paketa (*Packet Delivery Delay - PDD*), koja mogu trajati do nekoliko minuta [7]. Dodatna komplikacija je nedostupnost GPS-a za sinhronizaciju mreže, što čini korišćenje vremenskih oznaka izazovnim i skupim. Takođe, vrijedi napomenuti povećanu potrošnju energije u UASN mrežama, koja je često za red veličine veća od potrošnje zemaljskih bežičnih senzorskih mreža. Ovo dodatno intezivira potrebu

za istraživanjem usmjerenim na razvoj sveobuhvatnih sigurnosnih rešenja koji balansiraju kompromis između minimizacije komunikacijskog opterećenja i potrošnje energije uz održavanje optimalnih nivoa sigurnosti.

U većim UASN mrežama obezbjeđivanje sigurne komunikacije predstavlja poseban izazov. Veće mreže obično koriste klastersku strukturu kako bi poboljšale energetsku efikasnost i smanjile kašnjenja [8]. Unutar ove strukture, glavni čvor (*Cluster Head* - CH) - obično čvor s najmanjim energetskim ograničenjima - prikuplja, obrađuje i prosljeđuje podatke svojih pridruženih senzorskih čvorova prema zemaljskoj mrežnoj infrastrukturi. Jedna od bitnih karakteristika ovih mreža je često kretanje čvorova između klastera, bilo namjerno ili nenamjerno. Ključno sigurnosno pitanje u ovom scenariju je uspostavljanje robustnog sistema za autentifikaciju uređaja. S obzirom na kretanje senzorskih čvorova, proces autentifikacije između klastera treba biti i brz i efikasan. Iako u literaturi postoji mnogo metoda autentifikacije zasnovanih na simetričnoj kriptografiji ili digitalnim potpisima, većina njih se suočava s problemima kao što su postojanje jedne tačke otkaza (*single point of failure*) ili limitirane performanse zbog centralizovane arhitekture sistema za razmjenu ključeva [9]. Ova ograničenja mogu rezultirati značajnim kašnjenjem pri reautentifikaciji kada senzorski čvorovi prelaze između mrežnih klastera.

Da bi se adresirali gore pomenuti izazovi, nedavna istraživanja zagovaraju decentralizovane pristupe autentifikaciji [9–11]. *Blockchain* tehnologija, u kojoj distribuirana mreža računarskih nodova održava uniformne podatke bez potrebe za centralizovanim sistemom, ističe se kao održivo rješenje. *Blockchain* omogućava uspostavljanje povjerenja između različitih domena, pri čemu svaki domen mora imati dedicirani računarski nod za održavanje zajedničkog globalnog registra (*ledger*) [12]. Implementacija *blockchain-a* ne samo da štiti od potencijalnih napada na centralizovanu bazu podataka, već i smanjuje komunikaciono opterećenje prilikom prelaska senzorskih čvorova između domena [13]. Motivisani ovim činjenicama, predloženi su različiti pristupi za autentifikaciju zasnovani na *blockchain-u* za mreže Interneta stvari (*Internet of Things* - IoT) [13–23], uključujući i Internet podvodnih stvari (*Internet of Underwater Things* - IoUT) [9, 10]. Ovi pristupi rješavaju autentifikaciju putem asimetrične kriptografije, uglavnom se oslanjajući na sertifikate od pouzdanih Sertifikacionih Autoriteta (žtextitCertification Authority - CA) kako bi se uspostavila provjerljiva veza između entiteta i odgovarajućeg javnog ključa. Ipak, ovi pristupi uglavnom zanemaruju problem upravljanja životnim ciklusom sertifikata, što je ključno za ukupnu sigurnosnu infrastrukturu, kako bi se osiguralo da kompro-

mitovani ili zastarjeli kredencijali ne ugroze mrežu. Pored toga, ovi pristupi često ne uzimaju u obzir resursna ograničenja senzorskih čvorova. Ovaj rad ima za cilj da popuni tu prazninu uvođenjem novog protokola za autentifikaciju i upravljanje ključevima u UASN mrežama sa klasterskom strukturom, zasnovanog na *blockchain*-u. Predloženo rešenje, pod nazivom *Blockhain Enabled Key Management Protocol* (BEKMP) [24], omogućava sigurnu i brzu autentifikaciju uređaja, uz istovremeno otklanjanje potrebe za čestim reautentifikacijama prilikom prelaska senzorskih čvorova između različitih mrežnih klastera. Sistem integriše implicitne sertifikate kako bi ponudio ne samo sigurno, već i računski i komunikaciono efikasno rešenje, pogodno za veoma resursno ograničena okruženja. Pri tome, implementacija *blockchain* tehnologije je predviđena samo na gejtvej čvorovima na površini vode koji imaju značajne resurse i funkcionišu kao glave klastera (CH),. Na ovaj način ostvaruje se robustna decentralizovane infrastruktura za upravljanje ključevima i sertifikatima. Pametni ugovori ugrađeni u *blockchain* mrežu upravljaju svim operacijama vezanim za sertifikate i ključeve i dodatno podržavaju mehanizam za procjenu pouzdanosti senzorskih čvorova, omogućavajući automatsko opozivanje sertifikata. Predloženi pristup takođe uključuje *Hashed One-time Menezes-Qu-Vanstone* (HOMQV) protokol [25] za efikasnu razmjenu ključeva u jednoj poruci, čime se ostvaruje dodatna ušteda energije kroz smanjeno komunikaciono opterećenje. Ranjivosti na *replay* napade inherentne ovom protokolu riješene su upisivanjem komunikacionih logova na *blockchain*, pružajući sigurnu, provjerljivu istoriju koja sprečava maliciozne retransmisije.

Glavni doprinosi ove teze se mogu sumirati na sledeći način:

- Predložen je efikasan sistem za autentifikaciju i upravljanje ključevima u UASN mrežama, zasnovan na *blockchain-u*. Predloženo rješenje omogućava pojednostavljenje reautentifikacije između klastera za podvodne uređaje i smanjenje rizika od internih napada u slučaju kompromitacije mreže.
- Predložena metoda autentifikacije dizajnirana je da bude računski efikasna, brza i sigurna. S obzirom na ograničene resurse podvodnih senzorskih čvorova, pretpostavlja se slojevita mrežna arhitektura, pri čemu je *blockchain* implementiran samo na gejtvej čvorovima na površini vode, koji imaju značajne resurse i djeluju kao glave klastera (CH). Kašnjenje i potrošnja energije optimizovani su korišćenjem implicitnih sertifikata i HOMQV protokola za razmjenu ključeva.

- Sprovedena je temeljna komparativna analiza predložene metode autentifikacije sa postojećim rješenjima iz literature, ispitujući pruženi nivo sigurnosti, komunikaciona i računska opterećenja, kao i energetsku efikasnost. Dodatno, sigurnost predloženog sistema je potvrđena korišćenjem AVISPA alata [26].
- Predloženi sigurnosni protokol implementiran je na autentičnim UASN uređajima, a njegove performanse u smislu kašnjenja pri autentifikaciji/reautentifikaciji procijenjene su eksperimentima na moru kao i kroz simulacije.
- Kompletan kod za implementacije protokola kao i sprovedene simulacije javno je dostupan u GitHub repozitorijumu. [1]

Struktura ove teze organizovana je na sljedeći način: Glava 1 pruža opsežan pregled literature, pokrivajući strategije mitigacije eksternih i internih napada i sigurnosna rješenja zasnovana na *blockchain-u*. Glava 2 predstavlja teorijsku analizu predloženog protokola, detaljno opisuje mrežni model, potencijalne napade kojima je mreža izlođena i operativne faze protokola, uključujući inicijalizaciju sistema, registraciju, međusobnu autentifikaciju i razmjenu ključeva, među-klastersku autentifikaciju i evaluaciju povjerenja. U Glavi 3 predstavljena je analiza sigurnosti, uključujući teorijsko ispitivanje i formalnu verifikaciju sigurnosnih mjera protokola. Glava 4 opisuje implementaciju predloženog sistema, uključujući konfiguraciju *blockchain* mreže i implementaciju ECQV i HOMQV protokola. Glava 5 predstavlja rezultate evaluacije performanse *blockchain* mreže i BEKMP protokola putem eksperimenata i simulacija. Zaključak sumira rezultate teze i daje preporuke za buduća istraživanja.

[1] `https://github.com/monusen-uom/BEKMP-Blockchain-Platform`, `https://github.com/monusen-uom/BEKMP-ECC-benchmark`, `https://github.com/monusen-uom/Desert-BEKMP`

Glava 1

Pregled literature

Podvodne akustične senzorske mreže (UASN) postaju sve važnije za razne pomorske primjene, uključujući monitoring morskog okruženja, detekciju i praćenje objekata, komercijalna istraživanja, nadzor kritične infrastrukture i mnoge druge [2, 3]. Specifičnosti podvodnog akustičnog kanala u kombinaciji s ograničenjima resursa podvodnih senzorskih uređaja, čini UASN mreže posebno ranjivim na širok spektar malicioznih napada. Ove ranjivosti su naglašene u nekoliko nedavnih studija [2, 3, 27, 28], koje ne samo da ističu jedinstvene karakteristike UASN mreža već i pružaju sveobuhvatnu analizu potencijalnih prijetnji, napada i odgovarajućih strategija borbe protiv istih. U ovoj glavi će biti dat pregled aktuelnih istraživanja, organizovan u tri ključne teme:

- Mitigacija eksternih napada
- Mitigacija internih napada
- Sigurnost na bazi *blockchain*-a

Dodatno, biće istaknuti jedinstveni doprinosi ove teze.

1.1 Mitigacija eksternih napada

U literaturi je predložen niz kriptografskih rješenja za unapređenje sigurnosti UASN mreža. U radu [29], autori su predložili korišćenje kriptografije eliptičnih krivih (*Elliptic Curve Cryptography* - ECC) na podvodnim akustičnim modemima. Koristili su digitalne signalne procesore (*Digital Signal Processing*-DSP) kako bi ubrzali kriptografske operacije. Efikasnom implementacijom ECC-a značajno su smanjili

zahtjeve za računarskim resursima. U radu [30], ispitana je upotrebljivost različitih šema digitalnog potpisa za problem autentifikacije u UASN mrežama, posebno u kontekstu potrošnje energije. Studija iz rada [27] predstavlja najučinkovitije kriptografske metode koje se mogu koristiti za unapređivanje sigurnosti UASN mreža na različitim nivoima protokol steka, od mrežnog nivoa do nivoa aplikacije. Autori su predložili sigurnosni okvir koji omogućava povjerljivost podataka, integritet, autentifikaciju i neporicanje, a izgrađen je na bazi AES (*Advanced Encryption Standard*) standarda [31] u GCM (*Galois Counter Mode*) modu i kratkim algoritmima digitalnog potpisa. Međutim, važno je napomenuti da nijedno od ovih predloženih rješenja direktno ne rješava izazov razmjene tajnog ključa.

Protokoli za distribuciju ključeva (*Key Distribution Protocols* -KDPs) za UASN mreže istraženi su u radovima [28, 32–37]. U radu [32] je pokazano da je u kontekstu veoma resursno ograničenih okruženja, kao što su UASN mreže, efikasno rješenje za izazove distribucije ključeva primjena neinteraktivnog protokola za razmjenu ključeva zasnovanog na identitetu, poput SOK-a [33]. Kriptografija zasnovana na identitetu se često koristi za dizajniranje KDP protokola za zemaljske bežične senzorske mreže [34]. Glavna ideja ovog koncepta je omogućiti uređajima da izračunaju javne ključeve drugih uređaja direktno iz njihovih identifikacionih informacija. Ovo smanjuje složenost i komunikaciono opterećenje koje unosi upravljanje sertifikatima u sistemima infrastrukture javnih ključeva (*Public Key Infrastructure* - PKI). Međutim, treća strana, poznata pod nazivom Generator privatnih ključeva (*Private Key Generator* - PKG) odgovorna je za generisanje i sigurno distribuiranje privatnih ključeva korisnicima unutar sistema. Ovo čini sistem osjetljivim na napade jer pad PKG-a ugrožava cijelu mrežu i smanjuje opštu sigurnost privatnih ključeva, s obzirom na to da PKG može dešifrovati sve komunikacije unutar sistema. Pored toga, kriptografija zasnovana na identitetu često koristi bilinearne mape, koje mogu biti osjetljive na kvazipolinomijalne napade na problem diskretnog logaritma u poljima male karakteristike [35]. U radu [36] su predložena dva protokola za prenos podataka koji koriste ključeve zasnovane na identitetu, prilagođena mobilnim UASN mrežama. Jedan od njih je zasnovan na bilinearnim mapama, a drugi na algebarskim potpisima. Ovi protokoli su dizajnirani za podvodna vozila s različitim nivoima računarskih resursa i memorije, i oba omogućavaju direktnu autentifikaciju bez potrebe za eksternom sigurnosnom infrastrukturom.

KDP protokoli koji koriste implicitne sertifikate i ECC kriptografiju razmatrani su u radovima [28, 37, 38]. Implicitni sertifikati omogućavaju potvrđivanje veze

između javnog ključa uređaja i njegovog identiteta unutar iste strukture podataka, čime se eliminiše potreba za eksplicitnim potpisom. Zbog svoje male veličine, implicitni sertifikati su praktičan izbor za razmjenu ključeva u UASN mrežama. U radu [37], ECQV (*Elliptic Curve Qu-Vanstone*) protokol [39] predložen je za distribuciju implicitnih sertifikata u UASN mrežama. Za razmjenu ključeva razmatrane su dvije opcije: FHMQV (*Fully Hashed Menezes-Qu-Vanstone*) [40] i HOMQV [25] protokoli. Kombinacije ovih protokola evaluirane su u smislu energetske efikasnosti i vremena potrebnog za kompletiranje razmjene ključeva između dva mrežna čvora. Rezultati su pokazali brojne prednosti korišćenja ovih protokola u odnosu na konvencionalni *Diffie-Hellman* (DH) protokol za razmjenu ključeva [41], koji se obično koristi sa X.509 sertifikatima. Posebno je uočeno da je kombinacija ECQV i HOMQV najefikasnija, što je u skladu sa studijom iz rada [38]. Međutim, važno je napomenuti da je HOMQV prirodno ranjiv na *replay* napade, što ove studije nisu adresirale.

Alternativni pristupi autentifikaciji i razmjeni ključeva u UASN mrežama, koji ne koriste ECC, predloženi su u radovima [42, 43]. Konkretno, rad [42] predlaže korišćenje haotičnih mapa za uspostavljanje međusobne autentifikacije i razmjenu ključeva između korisnika, pouzdanog gejtveja i senzorskih čvorova. Ovaj pristup se izdvaja svojom efikasnošću u pogledu upotrebe računarskih resursa. S druge strane, rad [43] predlaže pojednostavljenu šemu autentifikacije koja koristi sabiranje matrica umjesto množenja kako bi se smanjila računska složenost. Predloženi pristup zahtijeva pomoć bazne stanice za generisanje konfiguracije za svaki senzorski čvor korišćenjem Vandermondovih vektora [44], koji predstavljaju osnovu za generisanje tajnog ključa. Takođe, sistem koristi protokol nultog-znanja (*zero knowledge proof*) [45] kako bi se obezbjedila sigurna autentifikacija bez otkrivanja tajnog ključa, postižući znatno nižu složenost u poređenju sa standardnim metodama zasnovanim na RSA (*Rivest–Shamir–Adleman*) standardu. Oba pristupa se oslanjaju na centralizovani model povjerenja, što unosi ključnu sigurnosnu ranjivost i može dovesti do uskih grla u performansama kada su u pitanju veće mreže.

Pored rješenja za autentifikaciju i razmjenu ključeva zasnovanih na kriptografiji koja se implementiraju na višim nivoima protokol steka, predložene su mnoge metode za obezbjeđivanje sigurnosti fizičkog sloja (*Physical Layer Security* - PLS). PLS metode autentifikacije koriste prirodna svojstva podvodnih akustičnih kanala za autentifikaciju predajnika umjesto korišćenja kriptografskog tajnog ključa. Svojstva podvodnog akustičnog kanala su izuzetno teška za estimaciju, što ih čini izuzetno pogodnim za ostvarivanje autentifikacije. Uobičajeno korišćena svojstva kanala

uključuju: jačinu primljenog signala (*Received Signal Strength* - RSS) [46], spektralnu gustinu snage [47], impulsni odziv širokopojasnog kanala [48–51], frekvencijski odziv kanala [52] i nivo snage [53, 54]. Novije studije istraživale su razne metode generisanja ključeva koristeći inherentnu nepredvidljivost podvodnih akustičnih kanala. Na primjer, autentifikacija zasnovana na višestrukim karakteristikama predstavljena je u radu [52], gdje su kako karakteristike korišćeni udaljenost, lokacija i ugao dolaska signala na senzorski čvor. Impulsni odziv kanala (*Channel Impulse Response* - CIR) korišćen je za generisanje ključeva u radu [55], dok je baza podataka CIR-ova korišćena u radu [56]. Pristupi mašinskog učenja koji spajaju više karakteristika kanala radi olakšavanja autentifikacije predloženi su u radovima [57–59], dok je studija [60] koristila vrijeme dolaska paketa po ruti (*Time of Arrival* - ToA) i topologiju mreže kao zajednički izvor nasumičnosti za generisanje ključeva. Međutim, primjena PLS-a u dinamičnim, mobilnim okruženjima je ograničena zbog brzo promjenjivih uslova kanala. Takođe, efikasnost PLS-a u generisanju ključeva zavisi od recipročnosti kanala, što se ne može garantovati u svim topologijama mreže.

Iako je klasterizacija mreže dokazano efikasno rešenje za poboljšanje energetske efikasnosti, skalabilnosti i smanjenje komunikacionih kašnjenja u UASN mrežaa, aspekt sigurnosti u ovim mrežama je prilično neistražen. Jedan od pionirskih napora u ovoj oblasti je protokol za upravljanje ključevima za mobilne hijerarhijske mreže opisan u radu [61]. Ovaj protokol se fokusira na grupisanje senzorskih čvorova oko resursno moćnijih čvorova, pri čemu bazna stanica (*Base Station* - BS) upravlja identifikatorima čvorova i kriptografskim elementima, i služi kao gejtvej prema zemaljskoj mreži. Iako je protokol pokazao značajnu energetsku i memorijsku efikasnost, ipak ima određena ograničenja, posebno jer se oslanja na *a priori* učitavanje parova ključeva za komunikaciju između senzorskih čvorova i glava klastera, a i odlikuje ga odsustvo mehanizma za ažuriranje ključeva. U radu [8] je predložen sličan model za sigurnu autentifikaciju i agregaciju podataka. Istraživanje predstavljeno u radu [62] slijedi drugačiji pristup, gdje BS izdaje tikete za ponovnu autentifikaciju senzorskim čvorovima. Ovi tiketi mogu biti validirani od strane svih glava klastera u mreži koristeći zajednički grupni ključ, što značajno pojednostavljuje procedure među-klasterske autentifikacije u mobilnim scenarijima. Međutim, rad [62] uglavnom opisuje opšti koncept autentifikacije zasnovane na tiketima, pri čemu ne sadrži opis korišćenih kriptografskih algoritmima i plana za praktičnu implementaciju.

1.2 Mitigacija internih napada

Radovi analiziraniu prethodnoj sekciji efikasno obrađuju probleme zaštite tajnosti poruka, autentifikacije i integriteta. Međutim, dijele zajedničku ranjivost na interne napade. Nekoliko tipova internih napada u UASN mrežama, kao što su *wormhole* napad, *sinkhole* napad i selektivno prosljeđivanje, pojedinačno su obrađeni u radovima [63, 64], [65], i [66]. U skorijoj literatuti su predloženi modeli povjerenja zasnovani na reputaciji uređaja za zaštitu UASN mreža od širokog spektra internih napada [7, 67–69]. Ovi modeli se mogu primijeniti kada je pravilno i neočekivano ponašanje čvorova jasno definisano i samim tim moguće za detekciju. Čvorovi se ocjenjuju na osnovu njihovog prethodnog ponašanja u mreži, čime se održava ocjena reputacije. Iako ova strategija samo identifikuje napadača, a ne ograničava njegov uticaj, može se primijeniti za prevenciju mnogih napada, pružajući time sveobuhvatno rješenje za sigurnost. U radovima [67, 68] autori su primijetili da maliciozni napadi uglavnom utiču na tri aspekta rada mreže: stanje komunikacije, nepravilnosti u paketima podataka i nepravilnosti u potrošnji energije.

Nedavna literatura, uključujući sveobuhvatni pregled istraživanja dostupan u [70], ističe značaj usvajanje algoritama mašinskog učenja za detekciju anomalija u bežičnim senzorskim mrežama. Ovi pristupi koriste napredne analitičke tehnike kako bi unaprijedili detekciju i dijagnozu nepravilnosti u radu mreže i imaju potencijal za primjenu u UASN mrežama.

1.3 Sigurnost na bazi *blockchain*-a

Blockchain tehnologija je sve više prepoznata po svom potencijalu da poboljša sigurnost komunikacije u IoT okruženjima [12, 71], uključujući specifičnu oblast Interneta podvodnih stvari (IoUT) [10, 11]. Fokus nekoliko studija bio je na rješavanju *single point of failure* problema PKI infrastrukture koja se oslanja na centralizovani sertifikacioni autoritet (CA) za izdavanje, održavanje i opoziv sertifikata javnih ključeva [14, 19, 22]. Karakteristike *blockchain*-a kao što su decentralizacija, mogućnost praćenja transakcija i otpornost na manipulacije, iskorišćene su za implementaciju skalabilne i sigurne distribuirane PKI infrastrukture. U radu [19] je *blockchain* korišćen za evidentiranje ključeva uređaja u pametnim mrežama na bazi *edge computing*-a. Set pametnih ugovora korišćen je za uslovno-anonimnu autentifikaciju uređaja i podršku efikasnom ažuriranju i opozivu ključeva. U radu [22] predložen je protokol za autentifikaciju s očuvanjem privatnosti koji primjenjuje im-

plicitne sertifikate za pametna vozila. U radu [14] su korišćeni pametni ugovori na privatnoj *blockchain* platformi za generisanje, kontrolu pristupa i opoziva implicitnih sertifikata. Dok su studije navedene u radovima [14, 22] predložile sigurnosna rješenja koristeći *blockchain* i implicitne sertifikate u infrastrukturama na bazi *edge computing*-a, što je u skladu sa pristopom koji se redlože u ovoj tezi, nisu se bavile jedinstvenim izazovima koje postavljaju UASN mreže. Konkretno, ove studije su ignorisale značajna kašnjenja u propagaciji koja čine učestalu komunikaciju s *blockchain* nodovima vremenski zahtjevnom. Dodatno, ove studije se ne fokusiraju na pojednostavljivanje procesa reautentifikacije kada se uređaji kreću između *edge* servera.

Protokol za autentifikaciju s očuvanjem privatnosti zasnovan na *blockchain*-u i mehanizam za detekciju malicioznih čvorova u IoUT okruženju predloženi su u radu [10]. Slično BEKMP pristupu, autori koriste čvorove na površini vode za formiranje *blockchain* mreže za registraciju i autentifikaciju podvodnih senzorskih čvorova. Tokom faze registracije *blockchain* pohranjuje heševe kredencijala senzorskih čvorova, koji uključuju lokaciju čvora, ID čvora i ID vlasnika. Prilikom procesa autentikifacije čvor na površini vode upoređuje sačuvane heševe sa hešom genrisanim na osnovu informacija prmljenih od senzorskog čvora. Ukoliko se ustanovi podudaranje, senzorski čvor dobija pristup mreži. Dodatno, rad se bavi detekcijom i djelimičnim opozivom sumnjivih čvorova integrisanjem mehanizama za procjenu povjerenja sa težinskim faktorima (*Weighted Trust Evaluation* - WTE) [72] i mehanizma aditivnog povećanja i multiplikativnog smanjenja (*Additive Increase Multiplicative Decrease* - AIMD) [73]. Međutim, za razliku od rješenja prezentovanog u ovoj tezi, [10] pretpostavlja model UASN mreže sa statičkim senzorskim čvorovima jer bi promjena pozicije čvora zahtijevala i promjenu kredencijala. Dodatno, rad nije obradio pitanja razmjene i upravljanja ključevima, koja su ključna za održavanje sigurnosti mreže.

Problem među-domenske autentifikacije istražen je u kontekstu industrijskog IoT-a [17, 18], mreža pametnih vozila [20, 21, 23] i mreža dronova [13], gdje je *blockchain* korišćen za uspostavljanje povjerenja među različitim domenima. Kada je riječ o UASN mrežama, primjena *blockchain*-a za među-klastersku autentifikaciju istražena je samo u radu [9]. Pristup predložen u radu [9] ne koristi sertifikate, pri čemu glave klastera generišu neophodne ključeve za sve uključene čvorove. Zbog odsustva sertifikata, međusobna autentifikacija između senzorskih čvorova zahtijeva interakciju sa glavom klastera, što može značajno povećati komunikaciono opterećenje i kašnjenje.

1.4 Doprinos

Iz prethodne diskusije postaje evidentno da je predloženo mnogo sigurnosnih pristupa za UASN mreže, ali je malo istraživanja posvećeno decentralizovanoj autentifikaciji i upravljanju ključevima u UASN mrežama sa klasterskom strukturom. Imajući u vidu potrebu za pružanjem sigurnosnih servisa sa malim kašnjenjem i minimalnim komunikacionim opterećenjem, uz očuvanje adekvatnog nivoa sigurnosti, u ovoj tezi predložen je novi protokol za autentifikaciju i upravljanje ključevima zasnovan na *blockchain*-u i implicitnim sertifikatima, koji je ne samo siguran već i resursno efikasan. Predloženi protokol se oslanja na ECC kriptografske metode za zaštitu mreže od širokog spektra eksternih napada, dok se snažne sigurnosne karakteristike *blockchain*-a koriste za ublažavanje ranjivosti na *replay* napade i interne prijetnje. Dodatno, decentralizovana priroda predloženog protokola autentifikacije, podržana *blockchain*-om, omogućava nesmetano kretanje autentifikovanih senzorskih nodova između klastera bez potrebe za ponovnom autentifikacijom. Ovo povećava fleksibilnost mreže i operativnu efikasnost, čineći protokol posebno pogodnim za dinamična UASN okruženja.

Glava 2

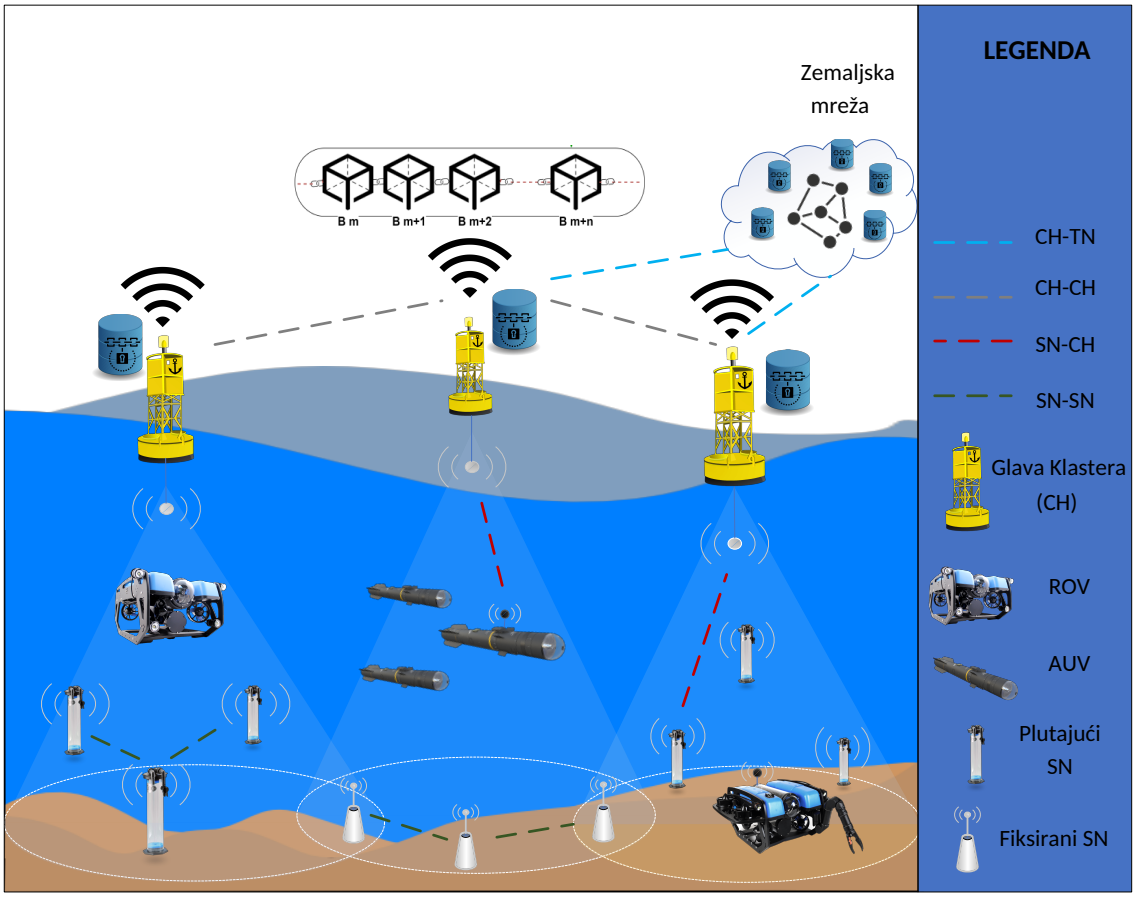
BEKMP protokol

U ovom poglavlju su najprije predstavljeni model sistema i modeli mogućih napada na mrežu, nakon čega se prelazi na detaljno objašnjenje predloženog protokola za upravljanje ključevima.

2.1 Model mreže

U fokusu je problem autentifikacije u heterogenim UASN mrežama, kao što je prikazano na Slici 2.1. Senzorski čvorovi (*Sensor Nodes* - SNs) su nasumično raspoređeni pod vodom i zajednički obavljaju zadatke nadgledanja okoline. Ovi nodovi ispoljavaju različite obrasce kretanja — neki se pasivno kreću sa morskim strujama, dok se drugi, kao što su autonomna podvodna vozila (*Autonomous Underwater Vehicles* - AUVs) ili daljinski upravljana vozila (*Remotely Operated Vehicles* - ROVs), kreću usmjereno sa ciljem. Uprkos varijacijama u energetskim resursima i računarskim kapacitetima među različitim kategorijama senzorskih čvorova, poboljšanje energetske efikasnosti je generalni cilj jer direktno utiče na trajanje operativne misije podvodnog nadgledanja.

SN čvorovi su organizovani u više klastera, pri čemu svakim klasterom upravlja Glava Klastera (CH čvor) postavljena na površini vode. CH čvorovi prikupljaju podatke od svojih pridruženih SN čvorova, procesiraju primljene informacije i proslijeđuju ih zemaljskoj mreži za dalju analizu. Takođe, one sprovode autentifikaciju uređaja i nadgledaju mobilnost senzorskih čvorova. Klasterizacija mreže donosi brojne prednosti, posebno u pogledu skalabilnosti, očuvanja energije i smanjenja kašnjenja u komunikaciji. Ograničavanjem komunikacije unutar klastera, znatno se smanjuju zahtjevi u pogledu potrebne predajne snage snezorskih čvorova, što je od



Slika 2.1: Razmatrani UASN model.

ključnog značaja za energetski ograničene UASN mreže. CH čvorovi takođe djeluju kao tačke za agregaciju podataka, filtrirajući i kompresujući informacije, čime se smanjuje količina podataka koja mora biti prenesena preko povratne veze do zemaljske mreže. Osim toga, uvođenje klastera pojednostavljuje topologiju mreže i rutiranje, pružajući posebne prednosti u mrežama sa velikim brojem aktivnih uređaja.

Dinamika komunikacije u mreži obuhvata nekoliko obrazaca: prenos od SN do CH, od SN do SN, od CH do CH, i od CH do zemaljske mreže (*Terrestrial Network* - TN). Prva dva načina komunikacije funkcionišu putem podvodnih akustičnih kanala, koje karakterišu jedinstveni izazovi uslovljeni sporom propagacijom zvuka kroz vodu, dok se poslednja dva sprovode naprednim radio komunikacionim tehnologijama u vazduhu, koje pružaju veći protok i pouzdanost prenosa podataka. U okviru ovog modela, pretpostavlja se da CH čvorovi nisu ograničene u pogledu energije, komunikacionih kapaciteta i procesorske snage. Da bi se omogućila brza i sigurna autentifikacija podvodnih SN čvorova i uspostavilo povjerenje unutar klasterizovane UASN mreže, CH čvorovi su uključeni u *blockchain* mrežu koja se proteže do zemaljske mreže. Autentifikacija se ostvaruje primjenom asimetrične kriptografije i ECQV sertifikata koje izdaje CA od povjerenja. Svi CH čvorovi održavaju identičan registar koji sadrži podatke o identitetima, sertifikate, enkriptovane ključeve i ocjene povjerenja SN čvorova. Da bi se olakšalo transparentno i automatizovano upravljanje ovim podacima, tri pametna ugovora funkcionišu unutar *blockchain* ekosistema:

- **Pametni Ugovor za Upravljanje Sertifikatima**: Ovaj pametni ugovor nadgleda izdavanje i opoziv sertifikata, osiguravajući da statusi sertifikata SN čvorova budu ažurirani na osnovu promjena u nivoima povjerenja i unaprijed definisanim rokovima trajanja.
- **Pametni Ugovor za Upravljanje Ključevima**: Odgovoran za sigurno skladištenje, ažuriranje i preuzimanje ključeva, kao i za upravljanje pravilima dijeljenja ključeva.
- **Pametni Ugovor za Upravljanje Povjerenjem**: Dizajniran za dinamičko upravljanje nivoima povjerenja SN čvorova. Omogućava CH čvorovima da se prilagode promjenama u nivoima povjerenja SN čvorova i automatski opozovu sertifikate kada je to potrebno.

Blockchain mreža u sastavu ovog sistema je privatna i ne dozvoljava pristup bez prethodno izdate eksplicitne dozvole. To znači da svaki učesnik u mreži mora biti autentifikovan i autorizovan. Proces autentifikacije sprečava neautorizovane nodo-

ve da se uključe u mrežu i pristupe podacima. Takav sistem osigurava robustan, decentralizovan pristup autentifikaciji SN čvorova upravljanju povjerenjem unutar mreže.

2.2 Model potencijalnih napada

UASN mreže izložene su raznim vrstama spoljašnjih napada, što zahtjeva robustne odbrambene strategije. Ovi napadi uključuju, ali nisu ograničeni na *Sybil* napade, manipulaciju podacima i *replay* napade. Pored toga, mehanizmi sigurnosti mreže treba da obezbijede zaštitu protiv unutrašnjih napada koji potiču od kompromitovanih SN čvorova, kao što su *sinkhole*, *wormhole*, selektivno proslijeđivanje i napadi iscrpljivanjem [1].

Pored ove klasifikacije, napadi u UASN mrežama se mogu takođe podijeliti na pasivne i aktivne. Pasivni napadi, poput prisluškivanja, usmjereni su na neovlašćeno prikupljanje informacija bez ometanja rada mreže. S druge strane, aktivni napadi, kao što su slanje lažnih podataka od strane malicioznih čvorova ili ometanje komunikacije, direktno utiču na funkcionalnost i integritet mreže. Ova dva tipa napada predstavljaju različite izazove za dizajnere UASN mreža, koji moraju osmisliti sigurnosne mehanizme prilagođene specifičnim karakteristikama podvodnog okruženja, kao što su ograničeni propusni opseg, visoko kašnjenje i ograničeni energetski resursi.

Pritom, pretpostavlja se da su samo podvodni uređaji podložni kompromitovanju, s obzirom na to da CH čvorovi posjeduju superiorne kapacitete u poređenju sa SN čvorovima, a njihova strateška postavka na površini vode omogućava redovne inspekcije. Kao rezultat toga, komunikacija između CH čvorova, kao i od CH čvorova do zemaljske mreže, smatra se visoko pouzdanom zahvaljujući primjeni naprednih sigurnosnih mehanizama i savremenih radio komunikacionih tehnologija, koje su manje podložne eksploataciji od strane protivnika.

Osim digitalnih prijetnji, UASN mreže su izložene i fizičkim napadima zbog svoje ranjivosti jer i neprijateljskom okruženju. Napadači mogu fizički kompromitovati podvodne senzorske čvorove, vršiti izmjene na hardveru ili softveru, ili ih potpuno uništiti. Ovi fizički napadi mogu ozbiljno narušiti integritet i dostupnost UASN mreže, što zahtijeva implementaciju robusnih mehanizama za detekciju upada i tehnika tolerancije grešaka radi ublažavanja njihovog uticaja.

2.2.1 *Sybil* napadi

U *Sybil* napadu maliciozni čvor oponaša više čvorova falsifikujući identitete. Ovo može ozbiljno poremetiti rad mreže, uključujući rutiranje, agregaciju podataka i raspodjelu resursa. U UASN mrežama *Sybil* napadi mogu dovesti do netačnih podataka o senzorskim očitavanjima, pogrešnog usmjeravanja saobraćaja i iscrpljivanja resursa.

Ukoliko je napad uspješno izvršen, *Sybil* čvor može početi manipulaciju ključnim funkcijama mreže. Na primjer, prilikom rutiranja podataka, lažni čvorovi mogu privući mrežni saobraćaj prema sebi, dajući napadaču kontrolu nad protokom informacija. To omogućava napadaču da izvrši selektivno prosljeđivanje, pri čemu odlučuje koje će pakete podataka proslijediti dalje, a koje će zadržati ili odbaciti. Takva manipulacija može dovesti do gubitka važnih podataka, pogrešne distribucije resursa ili čak potpune paralize određenih dijelova mreže.

Osim toga, *Sybil* napadi mogu izazvati ozbiljne probleme u procesima agregacije podataka, gdje se informacije iz različitih senzorskih čvorova objedinjuju radi donošenja odluka. Lažni čvorovi mogu unositi netačne podatke u ove procese, što može rezultirati pogrešnim procjenama ili akcijama, naročito u kritičnim aplikacijama kao što su nadzor okoline, vojne misije ili detekcija prirodnih katastrofa.

Jedan od najozbiljnijih rizika je iscrpljivanje resursa mreže. S obzirom na to da su UASN mreže energetski ograničene, povećanje broja čvorova (čak i lažnih) može dovesti do prekomjerne potrošnje energije i resursa na obradu i komunikaciju, što može ubrzati iscrpljivanje baterija stvarnih senzorskih čvorova i ugroziti dugoročnu operativnost mreže.

2.2.2 Manipulacija podacima

Manipulacija podacima uključuje neovlašćenu izmjenu podataka koji se prenose između čvorova. U kontekstu UASN mreža, ovo može rezultirati korupcijom kritičnih informacija kao što su mjerenja senzora i kontrolne poruke, što vodi do ugrožavanja procesa donošenja odluka na CH čvorovima.

Napadi manipulacije podacima mogu poprimiti različite oblike, kao što su ubacivanje lažnih podataka, modifikacija legitimnih podataka, ili kombinacija prethodnog pomenutih. Na primjer, maliciozni čvor može slati lažna senzorska očitavanja da zavara sistem u pogledu uslova okoline, potencijalno izazivajući lažne alarme ili pri-

krivajući kritične događaje. Slično tome, napadač može izmijeniti kontrolne poruke da naruši koordinaciju između čvorova, dovodeći do neefikasnog rutiranja ili gubitka podataka.

2.2.3 *Replay* napadi

Replay napadi se realizuju tako što napadač presretne i ponovo pošalje validne pakete kako bi izazvao neželjene efekte. U UASN mrežama, *replay* napadi se mogu koristiti za stvaranje zabune u mreži dupliciranjem validnih poruka, što može dovesti do problema u autentifikaciji, rasipanja resursa i pogrešnih odgovora.

Na primjer, ponovljena poruka za autentifikaciju može obmanuti prijemni čvor da prihvati zastarjele ili nevažeće kredencijale, omogućavajući napadaču neovlašćen pristup resursima mreže. Na sličan način, ponovljena kontrolna ili upravljačka poruka može izazvati nepotrebne akcije ili stvoriti nesklad u operacijama mreže, što može ugroziti njenu stabilnost.

Replay napadi predstavljaju posebno ozbiljan problem za vremenski osjetljive UASN servise, kao što su sinhronizacija vremena ili koordinacija u korišćenju komunikacionih resursa. Ako se zastarjele sinhronizacione poruke ponove, mogu poremetiti usklađivanje vremenskih rasporeda između čvorova, što može dovesti do nekoordinisanog ponašanja i kolizija. Takvi poremećaji mogu značajno smanjiti performanse i efikasnost UASN sistema, što je posebno kritično u misijama gdje su tačnost i pravovremenost isporuke podataka od presudne važnosti.

2.2.4 *Sinkhole* napadi

U *sinkhole* napadu, kompromitovani čvor predstavlja sebe kao čvor sa najkraćom rutom do CH čvora, privlačeći okolne čvorove da usmjere svoj saobraćaj kroz njega. Ovo omogućava napadaču da selektivno proslijeđuje, odbacuje ili mijenja podatke. U UASN mrežama, *sinkhole* napadi mogu poremetiti komunikacione puteve i dovesti do značajnog gubitka podataka.

Sinkhole napad eksploatiše protokole rutiranja koji favorizuju najkraće puteve do odredišta. Napadač kompromituje čvor i lažno oglašava optimalne metrike rutiranja, kao što su minimalan broj hopova ili najmanje kašnjenje, da prevari susjedne čvorove da ga odaberu kao sljedeći hop prema CH čvoru. Kao rezultat, okolni čvorovi preusmjeravaju svoj saobraćaj kroz *sinkhole* čvor, stvarajući centralizovanu tačku kontrole za napadača.

Jednom kada *sinkhole* čvor privuče značajan dio mrežnog saobraćaja, napadač može vršiti različite maliciozne aktivnosti. Na primjer, napadač može selektivno proslijediti osjetljive podatke dok odbacuje kritične poruke, ili može modifikovati sadržaj paketa prije proslijeđivanja ubacujući lažne informacije. Ovo može ozbiljno narušiti integritet i pouzdanost UASN sistema.

2.2.5 *Wormhole* napadi

Wormhole napadi uključuju stvaranje tunela između dva napadača koji sarađuju, omogućavajući im da prenose pakete bržom rutom nego što je uobičajeno. Ovo može ozbiljno poremetiti protokole rutiranja tako što čini da legitimne rute izgledaju duže. Naime, čvorovi u mreži misle da je put između ova dva zlonamerna čvora vrlo kratak (ili da su direktno povezani), iako je u stvarnosti udaljenost između njih veća. To može ometati rad mreže i narušiti njenu sigurnost.U UASN mrežama, *wormhole* napadi mogu dovesti čvorove u zabludu u vezi sa topologijom mreže.

Ovakvi napadi su posebno opasni jer se mogu izvesti čak i bez kompromitovanja pojedinačnih čvorova u mreži. Dva zlonamjerna čvora, koja sarađuju, mogu biti fizički udaljena, ali kada uspostave tunel između sebe, mogu presretati, manipulirati i preusmjeravati podatke a da ostatak mreže ne bude svjestan njihove aktivnosti. Ova vrsta napada može destabilizovati mrežu na više načina, uključujući stvaranje petlji u rutiranju, uzrokovanje nepotrebnog preusmjeravanja saobraćaja, i izazivanje gubitaka ili kašnjenja u isporuci paketa.

U kritičnim aplikacijama UASN mreža, kao što su vojne operacije ili praćenje životne sredine, *wormhole* napadi mogu imati ozbiljne posledice. Na primjer, u vojnim operacijama, ovakav napad može dovesti do pogrešnog usmjeravanja informacija, što može ugroziti misije ili dovesti do lažnih zaključaka u vezi sa pozicijama ciljeva ili prijetnji. U aplikacijama za praćenje životne sredine, *wormhole* napad može izazvati pogrešno tumačenje podataka o okruženju, što može imati dalekosežne posledice po donošenje odluka zasnovanih na tim podacima.

2.2.6 Napadi selektivnog proslijeđivanja

Napadi selektivnog proslijeđivanja se dešavaju kada maliciozni čvor selektivno odbacuje određene pakete umjesto da ih prosljeđuje sledećem čvoru. Ovo može dovesti do djelimičnog gubitka podataka i degradacije performansi mreže. U UASN mrežama, selektivno proslijeđivanje može dovesti do dodatnih retransmisija što

ugrožava operativnost usled veće potrošnje energije.

U napadu selektivnog proslijeđivanja, kompromitovani čvor se ponaša kao normalan čvor - učestvuje u protokolima rutiranja i prihvata pakete od svojih susjeda. Međutim, umjesto da pouzdano proslijedi sve primljene pakete, maliciozni čvor selektivno bira podskup paketa koje će odbaciti, zadržati ili modifikovati. Ovaj selektivni proces može biti zasnovan na različitim kriterijumima, kao što su tipovi paketa, identiteti izvora ili odredišta, ili čak nasumična verovatnoća, čineći napad težim za detekciju.

Posledice selektivnog proslijeđivanja mogu značajno varirati, od blagog gubitka podataka do ozbiljne degradacije UASN mreže. Odbacivanjem ključnih kontrolnih poruka ili kritičnih podataka ovaj napad može poremetiti koordinaciju mreže, dovesti do segmentacije mreže ili stvoriti praznine u prikupljenim senzorskim podacima. Na primjer, ako se odbace paketi sa informacijama o sinhronizaciji vremena ili koordinaciji resursa, to može izazvati kašnjenja, nedoslijednosti u podacima, ili čak potpuni gubitak komunikacije između čvorova.

Ovaj napad je posebno izazovan jer može ostati neotkriven duži vremenski period. Pošto maliciozni čvor prosljeđuje neke pakete, mreža može izgledati funkcionalno, ali sa prikrivenim problemima koji postaju očigledni tek kada je već pričinjena značajna šteta. Napad selektivnog proslijeđivanja može negativno uticati na tačnost i pouzdanost aplikacija za nadgledanje i donošenje odluka u realnom vremenu, kao što su ekološki monitoring, bezbjednosne aplikacije ili operacije pretrage i spašavanja.

2.2.7 Napadi iscrpljivanjem

Napadi iscrpljivanjem imaju za cilj da iscrpe resurse mrežnih čvorova, posebno njihovu bateriju, kroz ponovljenu obradu nepotrebnih zadataka ili poruka. S obzirom da su energetski resursi UASN mreža veoma ograničeni i teško ih je dopuniti, napadi iscrpljivanjem mogu značajno narušiti funkcionalnost mreže.

Napadi iscrpljivanjem predstavljaju ozbiljnu prijetnju za UASN mreže jer ciljaju ranjivosti u komunikacionim protokolima, mehanizmima pristupa medijumu ili procesima obrade podataka. Napadač može iskoristiti ove slabosti na različite načine, uključujući kontinuirano slanje lažnih zahtjeva ili besmislenih poruka ciljanim čvorovima, prisiljavajući ih da troše limitiranu energiju na obradu i odgovaranje na ove nelegitimne zahtjeve. Na primjer, napadač može pokrenuti lažne alarme ili inicirati nepotrebne operacije, što dovodi do preopterećenja resursa čvora beskori-

snim zadacima.

Napadi iscrpljivanjem mogu se takođe usmjeriti na mrežne resurse kao što su propusni opseg ili memorija. Neprestano emitovanje besmislenih podataka može zagušiti komunikacione kanale, ometajući prenos legitimnog saobraćaja i iscrpljujući ograničene resurse UASN mreže. Slanjem velikih ili složenih poruka napadač može preopteretiti memorijske bafere čvorova, što može izazvati odbacivanje kritičnih paketa ili čak blokadu mrežnog saobraćaja.

2.3 Predloženi protokol

Sveukupni proces autentifikacije i upravljanja ključevima u razmatranom okruženju može se podijeliti u sledećih pet faza:

- Faza inicijalizacije sistema
- Faza registracije
- Faza međusobne autentifikacije i razmjene ključeva
- Faza međuklasterkse autentifikacije
- Faza evaluacije povjerenja

Detaljan opis ovih faza je predstavljen u nastavku ove glave, sa najčešće korišćenim oznakama sažetim u Tabeli 2.1.

2.3.1 Faza inicijalizacije sistema

U ovoj fazi, CA servis generiše javne parametre za procedure autentifikacije, a zatim se svi SN čvorovi koji učestvuju u podvodnoj misiji trebaju registrovati kod odgovarajuće CH čvora kako bi dobili servis autentifikacije uz pomoć *blockchain* tehnologije. ECQV protokol, koji se koristi za generisanje implicitnih sertifikata, kao i HOMQV protokol, namijenjen za razmjenu ključeva, oslanjaju se na ECC algoritam. Stoga, tokom inicijalizacije sistema, CA bira veliki prost broj n koji definiše eliptičnu krivu $E(n)$. Tačke na ovoj krivoj formiraju aditivnu cikličnu grupu G, sa generatorom P redom n. CA bira svoj privatni ključ sk_{CA} nasumično iz multiplikativne grupe cijelih brojeva modulo n, označene kao $\mathbb{Z}_n^*$. Zatim izvodi javni ključ kao $PK_{CA} = sk_{CA} \cdot P$, gdje $(\cdot)$ predstavlja ECC multiplikativnu operaciju. CA takođe bira jednosmjernu heš funkciju (H) i funkciju derivacije ključa (KDF) koje će se

Tabela 2.1: Pregled relevantnih notacija

Oznaka	Definicija
ID_x	Identifikator entiteta x
N_x	Nonce vrijednost izabrana od strane entiteta x
P	Generator eliptične krive
n	Red eliptične krive
KDF	Funkcija za derivaciju ključeva
H	Heš funkcija
PK_x	Javni ključ entiteta x
sk_x	Privatni ključ entiteta x
R_x	Kratkotrajni javni ključ entiteta x
r_x	Kratkotrajni privatni ključ entiteta x
$Cert_{SN_x}$	Sertifikat senzorskog čvora x
T_{exp}	Vrijeme isteka sertifikata
K_{x-y}	Ključ za sesiju koji dijele x i y
ENC_k	Enkripcija ključem k
TR_e	Dokaz povjerenja u energiju
TR_d	Dokaz povjerenja u podatke
TR_c	Dokaz povjerenja u komunikaciju
E_{res}	Preostala energija senzorskog čvora
$\|\|$	Operacija nadovezivanja

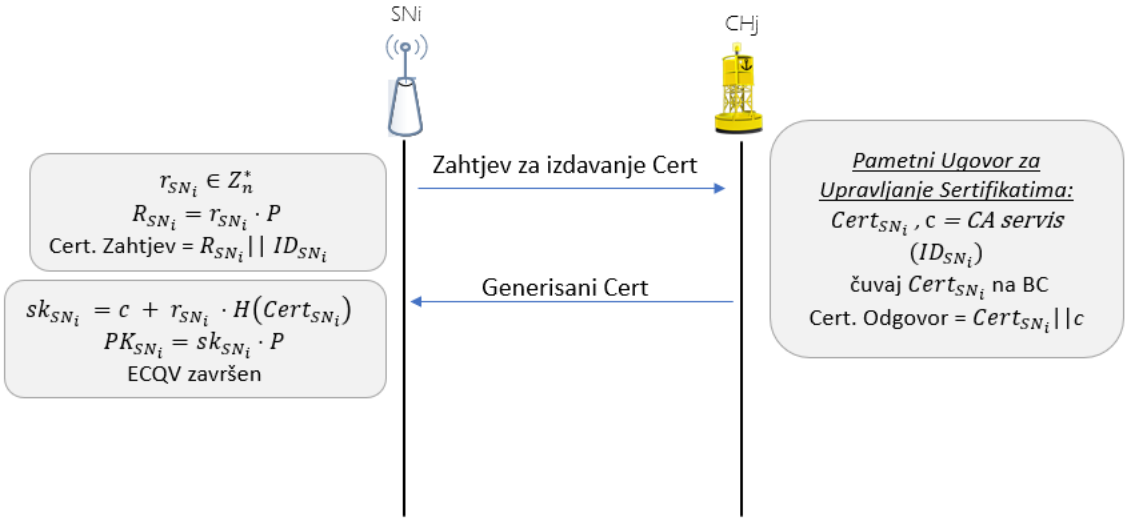
koristiti tokom faze međusobne autentifikacije i razmjene ključeva. Na kraju, CA javno objavljuje parametre $\langle E, P, n, H, KDF, PK_{CA} \rangle$.

Tokom ove faze, CH čvorovi se povezuju sa *blockchain* mrežom i identiteti SN čvorova se upisuju u *blockchain* registar. Kao dio procesa povezivanja sa *blockchain* mrežom, svaki CH čvor generiše par ključeva: privatni i javni ključ. Javni ključevi su sertifikovani od strane CA, čime se osigurava njihova autentičnost i sigurnost. Pretpostavlja se da svaki CH čvor u mreži posjeduje znanje o javnim ključevima svih drugih CH čvorova.

2.3.2 Faza registracije

Tokom ove faze, SN čvorovi se registruju preko sigurnog kanala kako bi dobili svoj par privatnog i javnog ključa. Ova faza se izvodi samo jednom i slijedi proceduru prikazanu na Slici 2.2, koja je detaljno objašnjena u nastavku:

Korak 1: Na početku, senzorski čvor i (SN_i) nasumično bira tajnu vrijednost $r_{SN_i} \in \mathbb{Z}_n^*$. Ova tajna vrijednost se koristi za računanje specifične tačke na eliptičnoj krivoj, tj. kratkotrajnog javnog ključa predstavljenog kao $R_{SN_i} = r_{SN_i} \cdot P$, gde je P generator tačaka eliptične krive. Nakon toga, SN_i prenosi izračunatu tačku eliptične



Slika 2.2: Faza registracije.

krive R_{SN_i} i svoj jedinstveni identifikator (ID_{SN_i}) do dodijeljenog CH čvora putem sigurnog komunikacionog kanala.

Korak 2: Nakon prijema zahtjeva za registraciju od SN_i, CH_j najpre poziva *Pametni Ugovor za Upravljanje Sertifikatima* na *blockchain*-u kako bi validirao identitet SN čvora i provjerio njegov status registracije. Ako primljeni identitet ne odgovara nijednom od identiteta SN čvorova u *blockchain*-u, zahtjev za registraciju se odbija. Isto se dešava ako CH čvor pronađe zapis u *blockchain*-u koji ukazuje da je SN čvor već registrovan. Ovaj korak je važan za sprečavanje *sybil* i *replay* napada. Ako zahtjev prođe proceduru validacije, pametni ugovor izdaje zahtjev za generisanje sertifikata (*Cert* na Slici 2.2) pozivajući *off-chain* CA servis.

Korak 3: Na zahtjev, CA servis konstruiše implicitni sertifikat koristeći ECQV protokol. ECQV protokol koristi složene matematičke operacije za povezivanje informacija o identitetu i javnom ključu u sertifikatu. Za razliku od tradicionalnih metoda, ECQV ne uključuje javni ključ pošiljaoca u sertifikat, već omogućava primaocu da ga izračuna koristeći sertifikat pošiljaoca i javni ključ CA, čime se smanjuje komunikaciono opterećenje. Proces generisanja sertifikata uključuje nekoliko koraka. Prvo, CA nasumično bira tajnu vrijednost $r_{CA} \in \mathbb{Z}_n^*$ i računa kratkotrajni javni ključ $R_{CA} = r_{CA} \cdot P$. Zatim generiše sertifikat

$$Cert_{SN_i} = (R_{CA} + R_{SN_i}) || ID_{SN_i} || T_{exp} \tag{2.1}$$

koji se sastoji od tačke na eliptičnoj krivoj, identifikatora čvora i vremena isteka

sertifikata T_{exp}. CA takođe generiše materijal za rekonstrukciju ključa [39]:

$$c = sk_{CA} + H(Cert_{SN_i}) \cdot r_{CA} \mod n \tag{2.2}$$

Na kraju, CA proslijeđuje sertifikat i c do CH čvora koji je poslao zahtjev za generisanje sertifikata.

Korak 4: CH prosleđuje odgovor CA servisa do SN_i. Zatim upisuje implicitni sertifikat u transakciju i dodaje je na *blockchain*. Svrha ovog koraka je eliminisanje potrebe za slanjem sertifikata tokom komunikacije između SN čvora i CH čvora.

Korak 5: Nakon što dobije odgovor od CH čvora, SN_i generiše svoje dugoročne privatne i javne ključeve u skladu sa ECQV protokolom [39], na sledeći način:

$$sk_{SN_i} = c + r_{SN_i} \cdot H(Cert_{SN_i}) \mod n \tag{2.3}$$

$$PK_{SN_i} = sk_{SN_i} \cdot P \tag{2.4}$$

Primljeni sertifikat se validira provjerom sledeće jednakost:

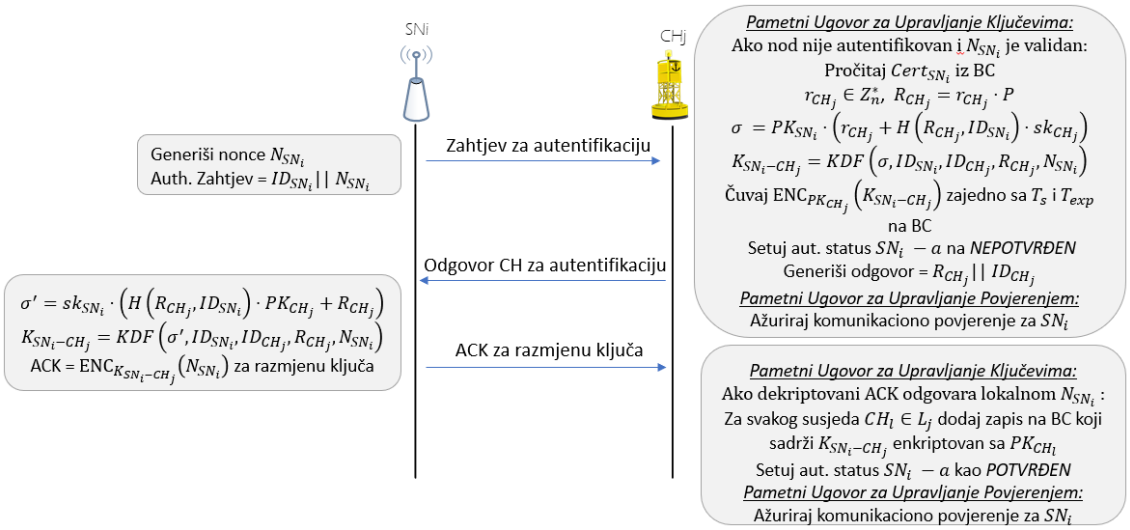
$$PK_{SN_i} = sk_{SN_i} \cdot P = PK_{\mathrm{CA}} + Cert_{SN_i} \cdot H(Cert_{SN_i}) \tag{2.5}$$

Ako su svi potrebni koraci uspješno završeni, SN_i dobija važeći sertifikat koji omogućava verifikaciju njegovog identiteta unutar mreže. Kad god CH čvor inicira ažuriranje u pametnom ugovoru u svrhe izmjene sertifikata jednog od svojih SN čvorova, ostali CH čvorovi će biti obaviješteni putem *blockchain* mreže kako bi mogli da ažuriraju svoje odgovarajuće keširane registre.

2.3.3 Faza međusobne autentifikacije i razmjene ključeva

Tokom ove faze, svaki SN čvor inicira međusobnu autentifikaciju sa svojim komunikacionim partnerom, bilo da je to CH čvor ili drugi SN čvor. Ovo se postiže primjenom HOMQV protokola, poznatog po svojoj efikasnosti i jednosmernom operativnom dizajnu. HOMQV protokol omogućava razmjenu ključa koristeći samo jednu poruku, čime se pojednostavljuje proces autentifikacije. Ova karakteristika je posebno korisna u UASN mrežama jer štedi energiju SN čvorova. Važno je napomenuti da se ova procedura autentifikacije razlikuje u zavisnosti od toga da li komunikacija uključuje SN čvor i CH čvor ili se odvija između dva SN čvora.

SN-CH Komunikacija Cjelokupan proces autentifikacije i razmjene ključeva između SN čvora i CH čvora je vizuelno predstavljen na slici 2.3 i biće detaljno objašnjen u nastavku:



Slika 2.3: SN-CH autentifikacija i razmjena ključa.

Korak 1: Da bi započeo proces, SN_i šalje zahtjev za autentifikaciju koji sadrži njegov ID i nasumično generisanu *nonce* vrijednost N_{SN_i} glavnom čvoru pripadajućeg klastera - CH_j.

Korak 2: Po prijemu zahtjeva za autentifikaciju, CH_j prvo poziva *Pametni Ugovor za Upravljanje Ključevima* na *blockchain*-u kako bi utvrdio da li je SN_i već autentifikovan i ima važeći ključ. Ako je SN_i već autentifikovan sa važećim ključem, zahtjev se odbacuje. Zahtjev se takođe odbacuje ako se utvrdi da je vrijednost nevažeća, kao u slučajevima kada je zahtjev sa istom *nonce* vrijednošću već obrađen. Ako razmjena ključeva još nije obavljena, ako nedostaje potvrda od SN_i, ili je ključ istekao, CH_j preuzima implicitni sertifikat SN_i iz svog registra i izvodi javni ključ SN_i koristeći kalkulaciju opisanu u 2.5. Zatim, CH_j generiše kratkotrajni par ključeva kao $R_{CH_j} = r_{CH_j} \cdot P$, gde je $r_{CH_j} \in \mathbb{Z}_n^*$ nasumičan pozitivan cijeli broj u opsegu $[1 \ldots n-1]$. Nakon toga, CH_j računa zajednički ključ koristeći KDF u skladu sa HOMQV protokolom [25], na sledeći način:

$$K_{SN_i-CH_j} = KDF(\sigma, ID_{SN_i}, ID_{CH_j}, R_{CH_j}, N_{SN_i}) \tag{2.6}$$

gdje je σ:

$$\sigma = PK_{SN_i} \cdot (r_{CH_j} + H(R_{CH_j}, ID_{SN_i}) \cdot sk_{CH_j}) \mod n \tag{2.7}$$

Nakon toga, CH_j kreira zapis na *blockchain-u* u obliku para ključ-vrijednost:

$$\langle ID_{CH_j}, ID_{SN_i}, N_{SN_i} \rangle \rightarrow (T_s, \mathrm{ENC}_{\mathrm{PK}_{\mathrm{CH}_j}}(K_{SN_i - CH_j}), T_{exp}, \mathrm{confirmed} = \mathrm{False})$$

Ovaj zapis pokazuje da je ključ za komunikaciju $SN_i - CH_j$ generisan na osnovu *nonce* vrijednosti N_{SN_i}, ali nije potvrđen kao važeći od strane SN_i, što je označeno poslednjim elementom zapisa. Ovdje, T_s predstavlja vremensku oznaku zapisa, $\mathrm{ENC}_{\mathrm{PK}_{\mathrm{CH}_j}}$ označava operaciju enkripcije javnim ključem CH_j, a T_{exp} predstavlja vrijeme isteka ključa. Na kraju, CH_j odgovara SN_i sa vrijednostima R_{CH_j} i ID_{CH_j}.

Važno je napomenuti da više zapisa autentifikacije sa različitim *nonce* vrijednostima može istovremeno postojati u *blockchain* registru, kao rezultat malicioznih napada. Međutim, samo SN_i može izvesti ključ iz odgovora CH_j. Kada SN_i potvrdi derivaciju ključa, specifičan zapis koji odgovara *nonce* vrijednosti odabranoj od strane SN_i se označava kao važeći, dok se ostali označavaju kao obrisani ili nevažeći. Pored toga, vremenske oznake svih zapisa povezanih sa nepotvrđenim pokušajima razmjene ključeva će biti kontinuirano pregledane, a svi zastarjeli unosi će biti označeni kao nevažeći.

Korak 3: SN_i generiše ključ koristeći jednačinu 2.6, računajući σ' na sledeći način:

$$\sigma' = sk_{SN_i} \cdot \left(H(R_{CH_j}, ID_{SN_i}) \cdot PK_{CH_j} + R_{CH_j}\right) \mod n \tag{2.8}$$

Ovaj korak osigurava da i SN_i i CH_j posjeduju identičan ključ, jer je σ jednako σ'. Da bi potvrdio prijem paketa, SN_i šalje potvrdu CH_j čvoru koja sadrži N_{SN_i} enkriptovan istim ključem.

Korak 4: Prilikom prijema potvrde od SN_i, CH_j dodaje nove zapise u *blockchain*. Svaki zapis odgovara susjednoj glavi klastera $CH_l \in L_j$, gde L_j predstavlja skup svih susjednih CH čvorova za CH_j. Format zapisa je sledeći:

$$\begin{aligned} &\langle ID_{CH_l}, ID_{SN_i}, N_{SN_i} \rangle \rightarrow \\ &(T_s, \mathrm{ENC}_{\mathrm{PK}_{\mathrm{CH}_l}}(K_{SN_i - CH_l}), T_{exp}, confirmed = True) \end{aligned} \tag{2.9}$$

Ovi zapisi imaju dvostruku svrhu: sprečavaju *replay* napade u HOMQV proto-

kolu i eliminišu potrebu za reautentifikacijom SN čvorova tokom migracija između klastera.

SN-SN Komunikacija Uspostavljanje sigurnog komunikacionog kanala između SN čvorova se obavlja bez interakcije sa *blockchain*-om. U ovom slučaju, SN čvorovi su dužni da direktno razmijenjuju svoje sertifikate. Iako *blockchain* pruža dodatni sloj sigurnosti mreži, u BEKMP protokolu je primijenjen ovaj pristup zbog značajnih kašnjenja u propagaciji koja su svojstvena UASN mrežama. Uvođenje interakcije sa *blockchain*-om za provjere statusa autentifikacije i preuzimanje sertifikata bi uvelo značajna komunikaciona kašnjenja. Stoga je odabran pristup u kojem SN čvorovi direktno razmijenjuju ECQV sertifikate i oslanjaju se isključivo na HOMQV za autentifikovanu razmjenu ključeva.

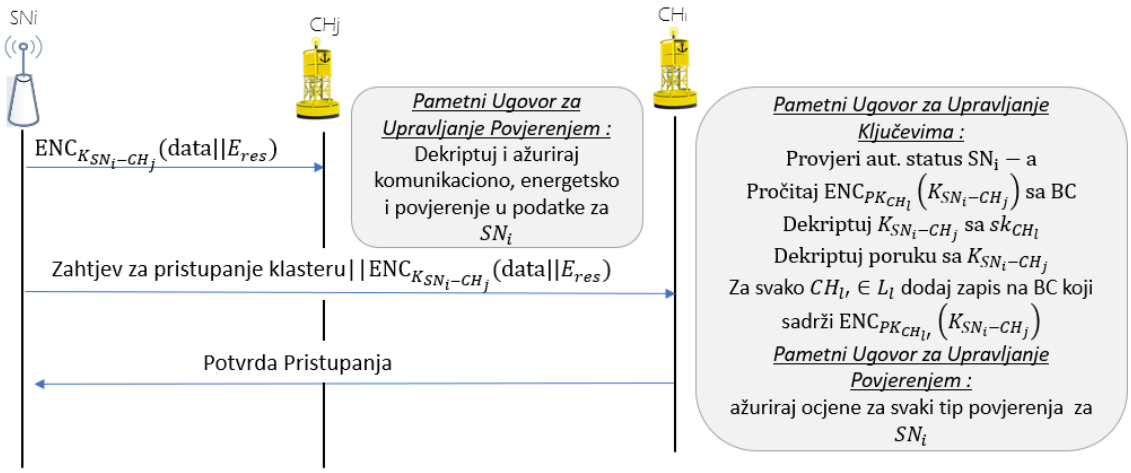
2.3.4 Faza među-klasterske autentifikacije

Tokom faze među-klasterkse autentifikacije, SN čvorovi periodično šalju svoja mjerenja i status baterije odgovarajućim CH čvorovima. Ako SN_i, bilo namjerno ili zbog vanjskih faktora kao što su morske struje, migrira u novi klaster, potrebno je da pošalje zahtjev za pridruživanje novoj glavi klastera CH_l, $l \in L_j$. Prilikom prijema zahtjeva, CH_l aktivira *Pametni Ugovor za Upravljanje Ključevima* kako bi provjerio status autentifikacije SN_i. Ovo podrazumijeva pretragu *blockchain* registra za par ključ-vrijednost, gdje ključ sadrži i ID_{SN_i} i ID_{CH_l}. Ako se takav par pronađe, CH_l zaključuje da je SN_i pouzdan i već autentifikovan, čime se eliminiše potreba za novim procesom generisanja ključa. Umjesto toga, CH_l izvlači enkriptovani ključ, $\mathrm{ENC}_{\mathrm{PK}_{\mathrm{CH}_l}}(K_{SN_i-CH_l})$, iz *blockchain* registra. Ovaj ključ se zatim dekriptuje koristeći privatni ključ CH_l čvora, sk_{CH_l}. Izuzetak je slučaj kada se ustanovi da je ključ istekao, kada se par ključ-vrijednost označava kao nevažeći. Pored navedenog, CH_l je odgovoran za ažuriranje *blockchain*-a zapisima koji potvrđuju autentifikovani status SN_i čvora u susjednim klasterima $CH_{l'}$, $l' \in L_l$. Ovi zapisi uključuju enkriptovatni ključ i sledećeg su formata:

$$\langle ID_{CH_{l'}}, ID_{SN_i}, N_{SN_i} \rangle \rightarrow (T_s, \mathrm{ENC}_{\mathrm{PK}_{\mathrm{CH}_{l'}}}(K_{SN_i-CH_j}), T_{exp}, \mathrm{True})$$

Na ovaj način, sistem eliminiše potrebu da SN_i prolazi kroz ponovnu autentifikaciju svaki put kada prelazi u različite klastere, značajno pojednostavljujući proces i osiguravajući kontinuiranu, sigurnu funckionalnost unutar mreže. Sažetak ove faze je prikazan na slici 2.4, pružajući vizuelnu reprezentaciju ključnih elemenata i procesa

koji se odigravaju.



Slika 2.4: Proces među-klasterske autentifikacije.

2.3.5 Faza evaluacije povjerenja

U cilju zaštite od unutrašnjih napada kompromitovanih SN čvorova, predloženi sistem uključuje *Pametni Ugovor za Upravljanje Povjerenjem* u okviru *blockchain*-a. Ovaj pametni ugovor dodijeljuje ocjene povjerenja svakom SN čvoru, koje ažuriraju CH čvorovi svaki put kada prime paket od SN čvora. Pametni ugovor je unaprijed konfigurisan sa specifičnim pragom za nivo povjerenja. Ako ocjena povjerenja SN čvora padne ispod ovog praga, sistem automatski pokreće proces opoziva sertifikata. Ovaj mehanizam osigurava tačno i transparentno upravljanje evaluacijama i ažuriranjima povjerenja.

Iako se UASN mreže mogu suočiti sa različitim napadima, posledice ovih napada se prvenstveno odražavaju u tri aspekta: prekid komunikacije, povećana potrošnja energije i manipulacija podacima [67,68]. Stoga, kako je predloženo u [67,68], BEKMP koristi tri tipa dokaza povjerenja za izračunavanje ocjena povjerenja SN čvorova:

- **Dokaz o energiji** TR_e - služi kao ključni pokazatelj u otkrivanju da li je SN čvor kompromitovan i uključen u napade sa velikom potrošnjom energije, poput napada iscrpljivanjem, ili u aktivnosti koje dovode do smanjene potrošnje energije, kao što su napadi selektivnog prosleđivanja. Pretpostavlja se da SN čvorovi prijavljuju nivo preostale energije u svakom poslatom paketu. Procjena povjerenja se izvršava poređenjem preostale energije čvora (E_{res}) sa unaprijed definisanim pragom, i poređenjem stope potrošnje energije (E_{cr}) sa istorijskim

prosjekom. Povjerenje opada kako se povećava odstupanje potrošnje energije SN čvora od E_{cr}. Dodatno, ako E_{res} padne ispod postavljenog praga, povjerenje u energiju se smatra nultim.

- **Dokaz tačnosti podataka** TR_d - zasnovan na pretpostavci da paketi koje prenose susjedni čvorovi u UASN mrežama obično prenose slične vrijednosti senzorskih očitavanja koje prate normalnu distribuciju [67]. CH čvor izračunava pouzdanost podataka SN čvora iz svog klastera poređenjem mjerenja senzora sa prosjekom mjerenja koje su prijavili njegovi susjedi. Veće odstupanje rezultira nižom vrijednošću povjerenja.

- **Dokaz komunikacije** TR_c - izvodi se korišćenjem okvira Subjektivne Logike, koji predstavlja povjerenje kao triplet uvjerenja, nepovjerenja i nesigurnosti [67]. CH čvor ocjenjuje povjerenje pridruženih SN čvorova na osnovu broja uspješnih i neuspješnih komunikacija sa njima. Ovaj model uzima u obzir da do gubitka paketa može doći zbog nepouzdanosti akustičnog kanala, kao i zbog potencijalno malicioznog ponašanja SN čvorova.

U okviru predloženog protokola, konačna ocjena povjerenja se računa kao suma težirana prethodno opisanih dokaza povjerenja. Ako ocjena povjerenja SN čvora padne ispod unaprijed definisane granične vrijednosti, sertifikat sačuvan u *blockchain* registru biće označen nevalidan. Nakon toga, CH čvorovi obaviještavaju sve SN čvorove u svom klasteru o opozivu sertifikata. Karatkeristike *blockchain* registra osiguravaju da su informacije o svim opozivima sigurno skladištene, pružajući dragocjen resurs za buduću analizu.

Glava 3

Analiza sigurnosti BEKMP protokola

U ovoj glavi je predstavljena analiza sigurnosnih karakteristika BEKMP protokola, nakon čega je opisana formalna verifikacija sigurnosti korišćenjem AVISPA alata [26].

3.1 Teorijska analiza sigurnosti

U predloženom rešenju, proces autentifikovane razmjene ključeva zasnovan je na ECQV i HOMQV protokolima. Ovaj pristup se u osnovi oslanja na složenost dva kriptografska problema: problem diskretnog logaritma na eliptičnoj krivoj (*Elliptic Curve Discrete Logarithm Problem* - ECDLP) i problem računanja *Diffie-Hellman* ključa (*Computational Diffie-Hellman Problem* - CDHP) [74]. U praktičnoj implementaciji predloženog sistema korištena je SECG-P192 eliptična krivu, koja nudi nivo sigurnosti od 96 bita. To implicira da bi maliciozni entitet morao izvršiti približno 2^{96} operacija da bi kompromitovao problem diskretnog logaritma [75].

HOMQV obezbjeđuje robustnu međusobnu autentifikaciju, omogućavajući da obje strane u komunikaciji mogu jednoznačno da autentifikuju jedna drugu. Ova osobina HOMQV protokola osigurava da su entiteti uključeni u razmjenu ključeva zaista oni za koje se predstavljaju, smanjujući rizik od impersonifikacije identiteta. Za svaku komunikacionu sesiju generiše se jedinstveni ključ. Ovaj ključ se izvodi iz javnih i privatnih ključeva, što povećava njegovu sigurnost protiv različitih kriptografskih napada. Protokol je dizajniran tako da osigurava da otkrivanje jednog ključa za

određenu sesiju ne ugrožava ključeve ostalih sesija ili privatne ključeve učesnika. Kompromitovanje kratkotrajnih privatnih ključeva ne ugrožava ključeve korišćene u prethodnim sesijama. Ovo je posledica činjenice da se ključevi sesije izvode na osnovu kombinacije kratkotrajnih i dugoročnih ključeva.

Iako zadržava mnoge sigurnosne osobine *Diffie-Hellman* protokola, HOMQV značajno poboljšava računsku efikasnost smanjujući broj potrebnih kriptografskih operacija. Međutim, ova efikasnost dolazi uz kompromis kompromitacije *Perfect Forward Secrecy* (PFS) osobine. PFS osigurava da povjerljivost predhodnih komunikacija ostane osigurana čak i u slučaju kasnije kompromitacije dugoročnih privatnih ključeva. Sa druge strane, dizajn HOMQV-a uključuje djelimično izvođenje ključeva iz dugoročnog privatnog ključa pošiljaoca, kršeći princip PFS-a koji zahtijeva da ključevi sesija treba da budu izvedeni iz kratkoročnih komponenti. To znači da, ako je ključ pošiljaoca kompromitovan, prethodne sesije bi potencijalno mogle biti ugrožene. Ovo ograničenje je posledica činjenice da HOMQV-a ne zahtijeva od primaoca generisanje novog kratkoročnog para ključeva za svaku sesiju, što, iako smanjuje računsku kompleksnost, povećava osjetljivost na napade. Ipak, takav dizajn predstavlja strateški balans između efikasnosti i sigurnosti, posebno koristan u okruženjima sa ograničenim resursima kao što su UASN mreže, gde je obezbjeđivanje kratkoročna povjerljivosti dovoljno za operacije poput detekcije i praćenja, podvodnog nadzora, taktičke komunikacije u mornarici, itd. Za okruženja u kojima je dugoročna sigurnost od ključnog značaja, *Full* HMQV (FHMQV) [40] nudi sigurniju alternativu. Iako je resursno intenzivniji, FHMQV se pridržava principa PFS-a, što ga čini poželjnim izborom za scenarije koji zahtijevaju obezbjeđivanje robustne dugoročne povjerljivosti.

Još jedna inherentna ranjivost osnovnog HOMQV protokola je podložnost *replay* napadima [76]. U cilju mitigacije ove ranjivosti, BEKMP protokol uključuje mehanizam *nonce*-a. U početku, pošiljalac prenosi nekriptovan *nonce* kao dio zahtjeva za autentifikaciju. U sledećoj fazi, nakon što se generiše ključ, *nonce* se enkriptuje, kao što je detaljno opisano u Glavi 2. Takođe su adresirani izazovi koje unosi visoko nepouzdani podvodni akustični kanal, koji može dovesti do scenarija u kojima odgovor na zahtjev za autentifikaciju od strane napadača bude primljen prije originalnog zahtjeva, uslijed gubitka paketa. Pored toga, napadač može pokrenuti *replay* napad i tokom drugog koraka autentifikacije, prisiljavajući primaoca da generiše novi ključ za legitimni SN čvor. Iako napadač ne može izvršiti derivaciju ovog ključa, takvi napadi mogu ometati komunikaciju sa validnim SN čvorovima. U cilju zaštitite SN-CH

komunikaciju od ovih problema, svi zahtjevi za autentifikaciju se upisuju u *blockchain* registar. Autentifikacija se smatra završenom tek nakon prijema enkriptovanog *nonce*-a od strane validnog pošiljaoca zahtjeva za autentifikaciju.

ECQV komponenta predloženog protokola autentifikacije omogućava efikasnu verifikaciju identiteta čvorova kroz generisanje implicitnih sertifikata. Dok u konvencionalnim sistemima sa PKI zasnovanim na X.509 sertifikatima čvorovi moraju dokazati poznavanje svojih privatnih ključeva Sertifikacionom Autoritetu (CA) kako bi se izbjegla neovlašćena sertifikacija javnih ključeva, ECQV sertifikati uklanjaju potrebu za takvim dokazom, jer nema unaprijed postavljenog javnog ključa prije izdavanja sertifikata. Uključenost CA u proces osigurava da čvorovi ne mogu uticati na svoj javni ključ, čime se spriječava zlonamjeran uticaj bilo kog čvora. Kao dodatna razlika u odnosu na X.509 sertifikate, implicitni sertifikati ne sadrže digitalni potpis. Iako teoretski neko može izabrati bilo koji identitet i nasumičnu vrijednost za formiranje sertifikata, bez učešća CA računanje odgovarajućeg privatnog ključa za javni ključ generisan iz sertifikata postaje praktično neizvodljivo.

Pored autentifikacije, predloženi sigurnosni protokol obezbjeđuje povjerljivost podataka kao i mogućnost trajnog praćenja istih kroz *blockchain*. Tokom cijelog procesa, osjetljive privatne informacije su enkriptovane korišćenjem razmjenjenih ključeva. Ponašanje senzorskih čvorova kao i sve aktivnosti vezane za autentifikaciju se kontinuirano prate i upisuju u privatni *blockchain* registar, koji je dostupan samo ovlašćenim korisnicima. Ako se otkriju bilo kakvi neprimjereni oblici ponašanja, *Pametni Ugovor za Upravljanje Povjerenjem* automatski opoziva sertifikat i odgovarajući javni ključ čvora. Svi zapisi u *blockchain* registru se validiraju kroz mehanizam konsenzusa, koji uključuje sve računarske nodove *blockchain* mreže. Zapisi su enkriptovani, a *blockchain* garantuje njihovu nepromjenljivost, otpornost na manipulacije i omogućava praćenje svih aktivnosti kroz vrijeme. Pored toga, decentralizovana *blockchain* infrastruktura čini servis autentifikacije koji sprovode CH čvorovi otpornim na *Denial of Service* (DOS) napade.

U Tabeli 3.1 prikazano je poređenje predloženog protokola sa nedavnim istraživanjima vezanim za autentifikaciju u IoT i IoUT okruđenjima, u smislu podržanih sigurnosnih karakteristika.

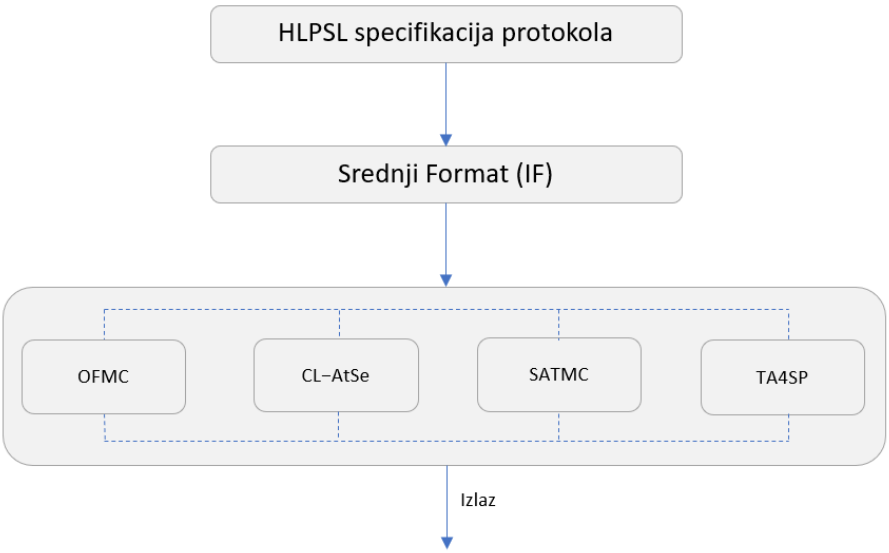
Tabela 3.1: Poređenje sigurnosnih karakteristika različitih protokola.

Karakteristike	**Protokoli**									
	[37]	[77]	[27]	[14]	[22]	[23]	[9]	[19]	[10]	BEKMP
Među-klasterksa reautentifikacija	-	-	-	-	-	✓	✓	-	-	✓
Povjerljivost	✓	✓	✓	-	✓	✓	✓	-	✓	✓
Međusobna autentifikacija	✓	✓	-	-	✓	✓	-	-	✓	✓
Lightweight dizajn	✓	✓	✓	✓	-	-	✓	-	✓	✓
Revokacija pristupa	-	-	-	✓	-	-	-	-	✓	✓
Zaštita od unutrašnjih napada	-	-	-	-	-	-	-	-	✓	✓
Zaštita od *replay* napada	✓	✓	✓	-	✓	✓	✓	-	✓	✓
Zaštita od DOS napada	-	-	-	-	✓	✓	✓	-	✓	✓

3.2 Formalna verifikacija sigurnosti

Za formalnu verifikaciju sigurnosti predloženog protokola za međusobnu autentifikaciju korišćen je poznati alat AVISPA [26]. Ovaj *open-source* alat pomaže u analizi i verifikaciji sigurnosnih protokola protiv širokog spektra etabliranih napada, primijenjujući tzv. *Dolev-Yao* (DY) model prijetnji [78]. Da bi protokol mogao biti analiziran od strane AVISPA alata, mora biti formulisan koristeći HLPSL (*High-Level Protocol Specification Language*) jezik, koji se naknadno translira u IF (*Intermediate Format*) format. IF specifikacija protokola se zatim koristi kao ulazni podatak za AVISPA *backend*-e. HLPSL strukturira svoju sintaksu oko 'uloga', koje predstavljaju učesnike u protokolu, i 'kompozicionih uloga', koje se koriste za definisanje sesija i okruženja u kojem protokol funkcioniše. Ovaj pristup omogućava jasnu i struktuiranu reprezentaciju funkcionalnosti protokola i interakcija između njegovih različitih komponenti. Arhitektura AVISPA alata je prikazana na slici 4.1, dok se u Prilogu može naći i kompletna HLPSL specifikacija protokola.

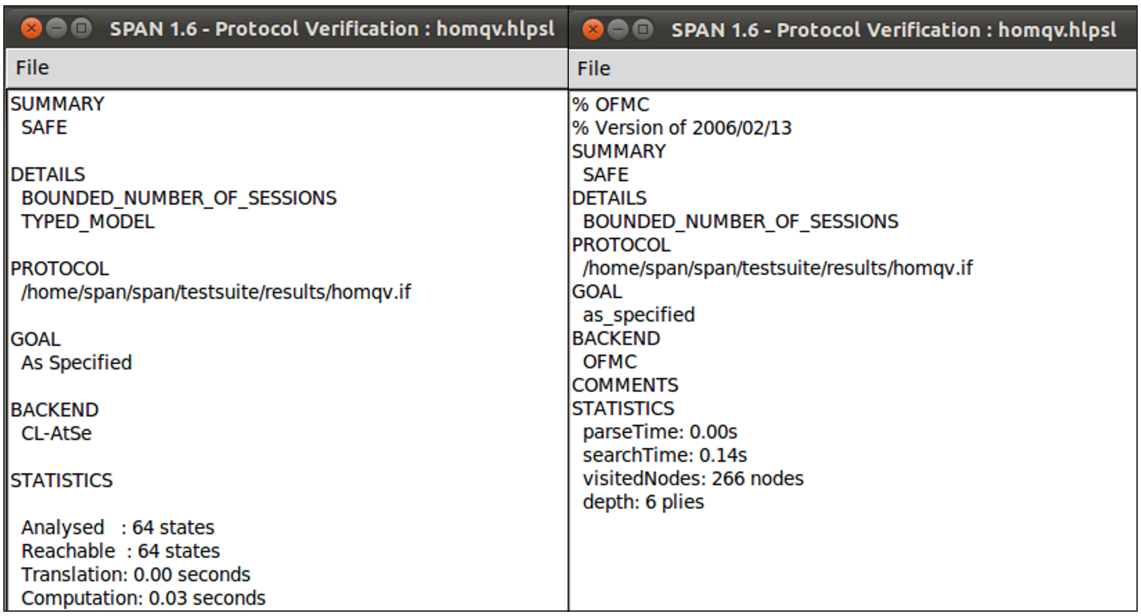
U sprovedenoj AVISPA simulaciji pretpostavljeno je da znanje napadača uključuje ID svakog čvora, javne ključeve, generatorsku tačku eliptičke krive, heš funkcije, kao i sve razmjenjene poruke. Za testiranje sigurnosti predloženog protokola definisana su dva ključna cilja. Prvi, koji se odnosi na povjerljivost, zahtijeva održavanje tajnosti razmijenjenog ključa među stranama uključenim u komunikaciju. Drugi, fokusiran na autentifikaciju, zahtijeva da strane koje primaju poruke verifikuju integritet oba *nonce*-i kratkotrajnih ključeva. AVISPA integriše četiri *backend* modula



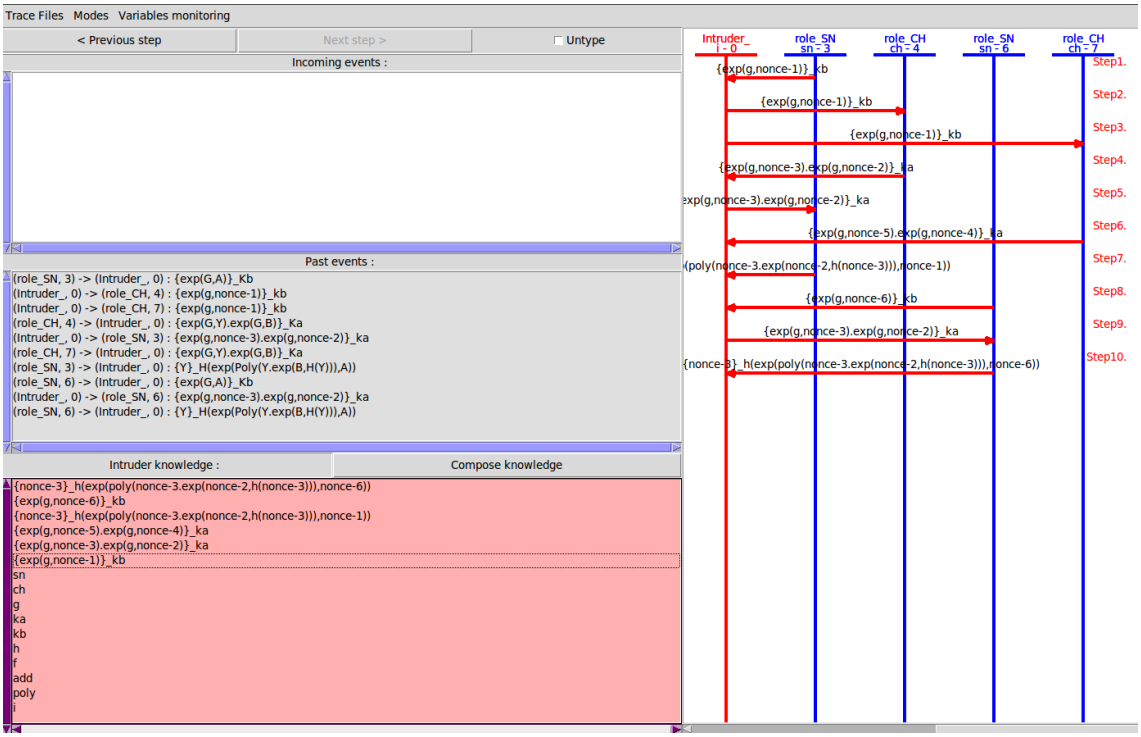
Slika 3.1: Arhitektura AVISPA alata.

koji primjenjuju različite savremene tehnike automatske analize: *"On-the-fly model-checker"*(OFMC), *Čonstraint-logic-based Attack Searcher"* (CL-AtSe), *ŠAT-based Model Checker"* (SATMC) i *"Tree Automata based on Automatic Approximations for the Analysis of Security Protocols"* (TA4SP) [26]. Ovi *backend* moduli pomažu u validaciji sigurnosti protokola u odnosu na definisane ciljeve i pružaju korisniku detaljan izvještaj u slučaju detekcije ranjivosti.

Predloženi protokol međusobne autentifikacije testiran je na OFMC i CL-AtSe *backend* modulima. Rezultati simulacije su potvrdili da je protokol siguran u oba slučaja, što se može vidjeti na slici 3.2. Slika 3.3 prikazuje simulaciju napada gdje se može vidjeti da napadač (*intruder*) ne posjeduje znanje koje može ugroziti integritet mreže.



Slika 3.2: Rezultati AVISPA simulacije.



Slika 3.3: AVISPA simulacija napada.

Glava 4

Implementacija

U ovoj glavi predstavljena je implementacija predloženog BEKMP protokola. Najpre je opisana implementacija *blockchain* mreže, a zatim su detaljno razrađene implementacije ECQV i HOMQV protokola, koji čine sastavne delove BEKMP-a.

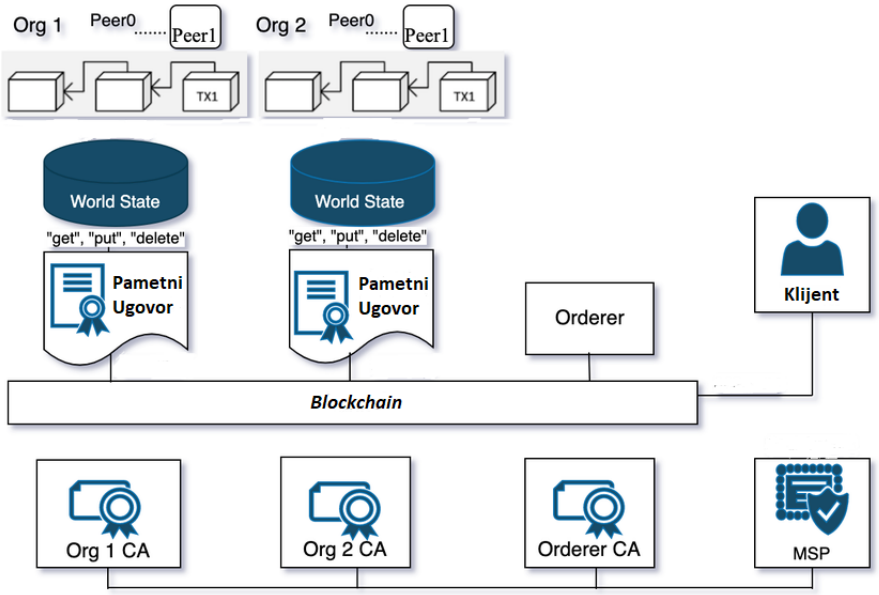
4.1 Implementacija *blockchain* mreže

U implementaciji BEKMP *blockchain* komponente korišćena je platforma *Hyperledger Fabric*. Ova platforma omogućava kreiranje privatne *blockchain* mreže, kojoj pristup imaju isključivo računarski čvorovi sa eksplicitnom dozvolom izdatom od strane pouzdanog entiteta. Generalna arhitektura koju pruža *Hyperledger Fabric* je prikazana na slici 4.1. Prednosti korišćenja ovog tipa *blockchain*-a uključuju:

- **Protok transakcija**: Zbog manje veličine mreže i efikasnijeg mehanizma konsenzusa, broj obrađenih transakcija u datom vremenskom intervalu je značajno veći. Ovo ga čini idealnim za slučajeve gde je skalabilnost ključna, kao što su UASN mreže.
- **Privatnost**: Ograničenim pristupom mreži osigurava se privatnost, što ovu platformu čini pogodnom za poverljive interakcije.

Hyperledger Fabric podržava pisanje aplikacija u programskim jezicima opšte namjene kao što su Java, Node.js i Go i prati tzv. *execute-order-validate* arhitekturu. Distribuirana aplikacija *Fabric*-a sastoji se od dvije glavne komponente:

1. ***Chaincode*** - Pametni ugovor koji implementira logiku aplikacije i izvršava se tokom faze *execute*. U sklopu istraživanja implementirana su tri specifična



Slika 4.1: Arhitektura *Hyperledger Fabric* platforme.

pametna ugovora koja pružaju podršku izvršavanju BEKMP protokola.

2. ***Endorsement* Polisa** - Statička biblioteka koja se koristi u fazi validacije da bi se odredilo koji nodovi su potrebni za odobravanje transakcija. U kontekstu BEKMP protokola, *endorsement* polisa je definisana tako da zahtjeva da većina CH čvorova validira i odobri transakciju prije nego što bude dodata na *blockchain*.

U nastavku će biti predstavljene i diskutovane ostale značajne komponente *Hyperledger Fabric* mreže i kako su integrisane za potrebe implementacije predloženog protokola.

4.1.1 *Peer* nodovi

Peer nodovi su okosnica *blockchain* mreže jer hostuju registre (*ledgers*) i pametne ugovore. Svaki čvor (*peer*) u mreži čuva svoju kopiju registra i pametnih ugovora. Iako ovo povećava količinu dupliranih podataka, distribucija ovih informacija na sve čvorove osigurava da mreža ostane otporna na kvarove ili napade na pojedinačne čvorove.

Iako *peer* nodovi teoretski mogu hostovati *blockchain* bez pametnih ugovora, u praksi je svakom nodu potreban pridruženi pametni ugovor kako bi mogao čitati, pisati i komunicirati sa *blockchain* registrom. Aplikacije se povezuju sa nodovima koristeći *Fabric Software Development Kit* (SDK) kako bi pristupile registrima i

pametnim ugovorima. Te aplikacije mogu ili pretraživati ili ažurirati *blockchain* na sledeći način:

- **Pretraživanje *blockchain*-a**: Aplikacija se povezuje sa *peer* nodom, poziva pametni ugovor i dobija odgovor. Pošto lokalna kopija registra hostovana na *peer* nodu sadrži sve potrebne informacije, on može odgovoriti odmah bez konsultovanja drugih nodova.
- **Ažuriranje *blockchain-a***: Ažuriranje uključuje postizanje konsenzusa među *peer* nodovima. Nodovi šalju digitalno potpisane predloge izmjena aplikaciji, koja ih zatim prosleđuje *Orderer* servisu. *Orderer* ih pakuje u blokove i distribuira mreži. Nodovi validiraju promjene prije nego što ih primjene na svoje lokalne registre, a aplikacija se obaviještava asinhrono. *Peer* nodovi uključeni u ovaj proces nazivaju se *Endorsers*. Oni izvršavaju predloge transakcija i validiraju transakcije.

Kao što je ranije objašnjeno, jedna od pretpostavki na kojima je baziran dizajn BEKMP protokola je da su CH čvorovi na površini vode dio privatne *blockchain* mreže. U eksperimentima su kao CH čvorovi korišćena aPad površinska vozila [79] (Slika 4.2), razvijena od strane Fakulteta za Računarstvo i Elektrotehniku Univerziteta u Zagrebu. U implementaciji BEKMP-a, svaki aPad funkcioniše kao jedan *blockchain* nod unutar umreženog *Hyperledger Fabric* sistema.

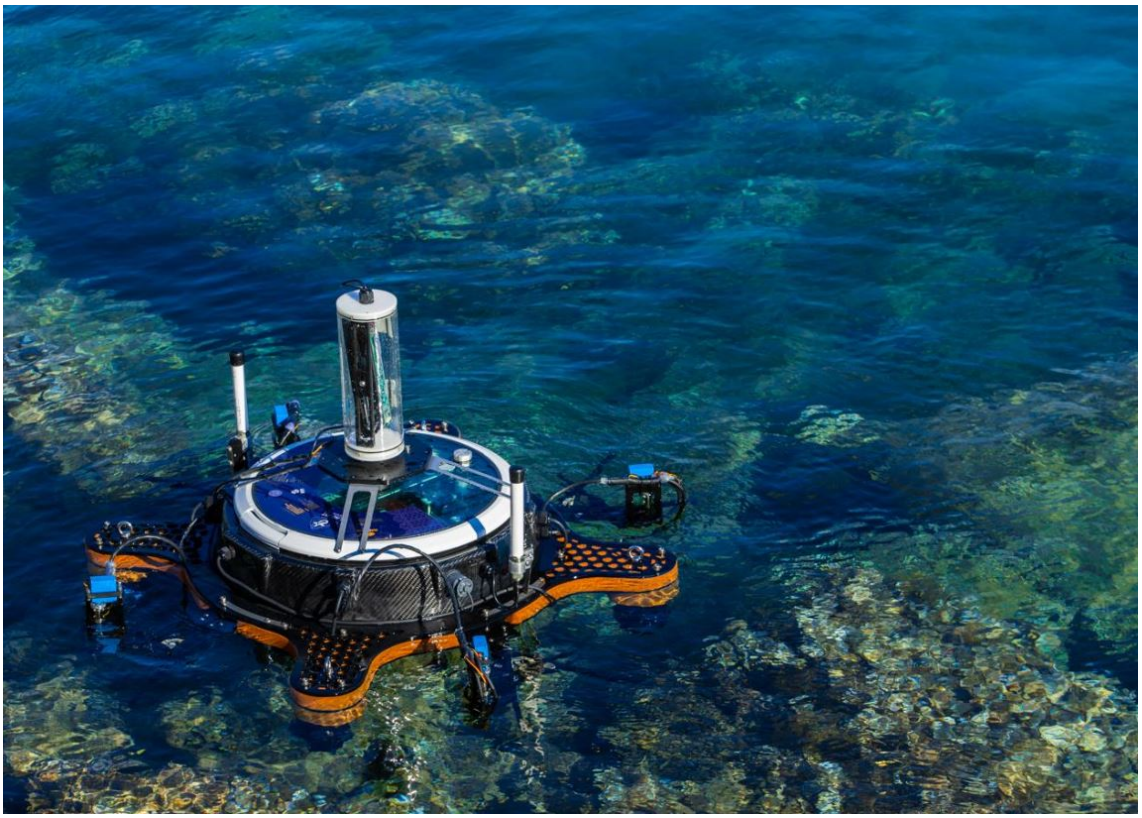
4.1.2 Klijentske aplikacije

Klijentske aplikacije iniciraju i predlažu transakcije, učestvuju u fazi izvršenja transakcije i distribuiraju transakcije *Orderer* servisu. U implementaciji BEKMP-a, funkcionalnost klijenta je integrisana u *peer* nod kao zasebni servis koja funckioniše na svakom aPad-u.

4.1.3 *Orderer* nodovi

Fabric koristi *Orderer* za upravljanje redosledom transakcija kao i formiranje novih blokova transakcija. Novi blokovi se formiraju na osnovu sledećih kriterijuma:

- Maksimalnog broja transakcija po bloku,
- Maksimalne veličine bloka,
- Unaprijed definisanih vremenskih intervala od poslednjeg formiranja bloka.



Slika 4.2: aPad vozilo korišćeno kao CH čvor u eksperimentima.

Orderer nodovi zajednički formiraju *ordering* servis, osiguravajući deterministički konsenzus, što čini validirane blokove tačnim i konačnim. Razdvajanjem procesa konsenzusa od izvršavanja pametnog ugovora poboljšane su performanse *blockchain*-a i povećana je propusnost transakcija.

U implementaciji BEKMP protokola definisan je jedan *ordering* servis i implementiran je na jednom od aPad-ova (CH čvoru). Pored obezbjeđivanja tačnog redosleda transakcija, *ordering* servis vrši i kontrolu pristupa, dozvoljavajući ili zabranjujući čitanje ili pisanje podataka na osnovu polisa postavljenih pri kreiranju mreže.

4.1.4 Registar (*Ledger*)

Sa logičke tačke gledišta, *Hyperledger Fabric* radi sa jednim glavnim registrom. Međutim, u praksi, svaki *peer* nod čuva i održava kopiju registra kroz proces konsenzusa, otuda izraz *Distributed Ledger Technology* (DLT). U registru se čuvaju trenutni i istorijski podaci aplikacije, a sastoji se od dvije glavne komponente: *world state* i *blockchain*.

- ***World State***: Ova komponenta čuva informacije o trenutnom stanju svih

podataka ili entiteta (koji se nazivaju "objekti") unutar sistema, skladišteći ih u formi parova (ključ, vrednost). To omogućava aplikacijama da direktno iz *world state*-a dobiju ažurirane informacije o ovim objektima, bez potrebe za pretraživanjem cjelokupnog *blockchain*-a. Funkcionišući kao baza podataka, *world state* nudi funkcionalnosti za čitanje, upisivanje i brisanje, kao i napredne mogućnosti upita kroz *API*. Nakon što se transakcija validira, tj. odobri sa dovoljnim brojem digitalnih potpisa, *world state* se ažurira, a aplikacija se obavještava asinhrono. Svaki (ključ, vrijednost) par takođe uključuje i broj verzije radi lakšeg praćenja promjena, osiguravajući konzistentnost tokom ažuriranja. Za implementaciju BEKMP-a, *LevelDB* je korišćen kao baza podataka za *world state*. *LevelDB* podržava upite sa kompleksnim ključevima i upite za opsege ključeva, što je ključno za potrebe predloženog protokola. Ove funkcionalnosti omogućavaju efikasno i fleksibilno upravljanje podacima.

- ***Blockchain***: Za razliku od *world state*-a, *blockchain* bilježi sve transakcije koje su ikada izvršene, prikazujući sekvencu događaja koja vodi do trenutnog stanja *world state*-a. *Blockchain* čuva transakcije u fajlu kojeg čine međusobno povezani sekvencijalni blokovi, optimizovanom za operacije dodavanja i pretrage. Svaki blok sadrži skup transakcija koje su poređane od strane *ordering* servisa. Dok *world state* sadrži samo validne transakcije, *blockchain* bilježi i validne i nevalidne. Transakcije su povezane sa blokovima putem kriptografskih heševa, osiguravajući integritet podataka i nepromjenjivost *blockchain*-a.

4.1.5 Provajder identiteta *(Membership Service Provider)*

U privatnoj *blockchain* mreži, sve operacije, poruke i interakcije moraju biti autentifikovane, obično putem digitalnih potpisa. *Membership Service Provider* (MSP) je odgovoran za izdavanje i održavanje kredencijala za sve vrste nodova, omogućavajući autorizaciju i autentifikaciju između *peer* nodova i *orderer* servisa. MSP funkcioniše na dva nivoa:

- **Lokalni MSP**: Povezan sa *peer* nodovima, *orderer* servisom i klijentskim aplikacijama, određujući njihove dozvole i prava.
- **Kanalni MSP**: Funkcioniše na nivou *blockchain-a*, definišući identitete i dozvoljene akcije u pogledu upisa i čitanja iz registra za sve nodove u *blockchain* mreži. Iako konceptualno postoji samo jedan kanalni MSP, svaki *peer* nod ima njegovu kopiju koja se sinhronizuje i ažurira kroz konsenzus.

Za implementaciju BEKMP-a, svi identiteti i lokalni MSP-ovi su kreirani *offline* i raspoređeni na svakom od aPad-ova prije startovanja UASN mreže. Kreiran je jedan *blockchain*, sa kanalnim MSP-om podešenim da svim CH čvorovima daje ista prava na upis i čitanje iz registra.

4.1.6 Životni ciklus pametnih ugovora

Životni ciklus pametnih ugovora obuhvata nekoliko koraka neophodnih za njihovu instalaciju i pokretanje unutar *blockchain* mreže. Svi klasteri u okviru UASN mreže moraju postići saglasnost o parametrima poput verzije ugovora, imena i pravila odobravanja kroz sledeće korake:

1. **Pakovanje pametnih ugovora**: Kompletan kod za pametne ugovore se pakuje u .tar datoteku.
2. **Instalacija pametnih ugovora na peer nodove**: Na svim *peer* nodovima se vrši instalacija pametnih ugovora
3. **Odobravanje definicije pametnih ugovora**: CH čvorovi zatim glasaju za odobravanje definicije pametnih ugovora, pri čemu je potrebno učešće svakog CH čvora.
4. **Pokretanje pametnih ugovora**: Nakon odobrenja, *ordering* servis kompletira podešavanje pametnih ugovora.

Ovi koraci su sprovedeni na svakom aPad-u. Pozivom funkcije *Init()* za svaki od ugovora, inicijalizovan je *blockchain* i postignut početni konsenzus.

4.1.7 Tok transakcija u *blockchain* mreži

Tok transakcija u *Hyperledger Fabric*-u uključuje nekoliko različitih faza: izvršenje (predlog), definisanje redosleda i validaciju. Ove faze zajedno osiguravaju da su transakcije pravilno obrađene i sačuvane na *blockchain-u*.

- **Faza izvršenja (Predlog)** - Započinje kada klijentska aplikacija digitalno potpisuje i šalje predlog transakcije *ordering* servisu. Predlog sadrži identitet klijentske aplikacije, sadržaj transakcije, parametre, identifikator transakcije i *nonce* za jedinstveno povezivanje transakcije sa klijentskom aplikacijom. Nakon toga, *peer* nodovi simuliraju predloženu transakciju u odnosu na stanje svog lokalnog registra. Tokom simulacije operacija se izvršava izolovano, kreirajući odobrenje koje uključuje *readset* (zavisnosti od prethodnih lokalnih stanja)

i *writeset* (promjene napravljene simulacijom). Klijentska aplikacija zatim prikuplja potrebna odobrenja od *peer* nodova na osnovu definisane politike, koja u BEKMP protokolu uključuje većinu CH čvorova. Kada se sakupi dovoljan broj odobrenja, kreira se transakcija koja se prosleđuje *ordering* servisu.

- **Faza definisanja redosleda** - U ovo fazi *ordering* servis uspostavlja redosled transakcija grupišući ih u blokove i povezujući blokove kriptografskim heševima. Klijentske aplikacije emituju transakciju zajedno sa podacima i digitalnim potpisom *ordering* servisu. Nakon toga, klijentske aplikacije prikupljaju blokove, koji sadrže listu transakcija i heš prethodnog bloka, prema rednom broju. Ova operacija se izvodi samo jednom jer su sadržaj i redosled blokova nepromjenljivi. *Ordering* servis osigurava da su blokovi pravilno formirani. *Orderer* nodovi ne izvršavaju transakcije niti održavaju stanje *blockchain*-u. Na ovaj način je razdvojen proces konsenzusa od validacije i izvršenja transakcija, čime je dodatno ojačan integritet sistema.

- **Faza validacije** - Finalna faza tokom koje se blokovi distribuiraju *peer* nodovima putem *gossip* protokola. *Peer* nodovi potom paralelno validiraju transakcije koristeći sistemski pametni ugovor za validaciju. Transakcije koje ne ispunjavaju politiku odobravanja označavaju se kao nevažeće. Zatim se sekvencijalno provjeravaju potencijalni konflikti između čitanja i pisanja kako bi se osiguralo da *readset* odgovara stvarnom stanju registra. Transakcije koje ne prođu ovu provjeru poništavaju se, a njihovi efekti se anuliraju. Nakon svih provjera, *blockchain* se ažurira dodavanjem novog bloka i ažuriranjem *world state*-a na osnovu *writeset*-a bloka.

Opisani tok transakcija u *Hyperledger Fabric*-u osigurava robustno, sigurno i efikasno procesiranje transakcija, omogućavajući visoku propusnost i modularnost.

4.1.8 Implementacija pametnih ugovora

Svi pametni ugovori koji učestvuju u izvšravanju predloženog protokola implementirani su u NodeJS jeziku. U ovoj sekciji su dati detalji implementacije.

Pametni ugovor za upravljanje sertifikatima

Pametni ugovor za upravljanje sertifikatima se sastoji iz funkcija za čitanje, čuvanje i opoziv sertifikata. U Prilogu su date implementacije ovih funkcija dok su ovdje opisane funckionalnosti koje pružaju.

- ***storeCertificate***: Ova funkcija omogućava čuvanje sertifikata povezanih sa određenim SN čvorom. Funkcija prima identifikator SN čvora (`deviceid`) i sertifikat (`certificate`) i kreira model koji sadrži `deviceid`, `certificate` i ID transakcije (`txId`). Potom kreira kompozitni ključ koristeći `deviceid` i konstantu `'crt'`, i ovaj ključ koristi za skladištenje modela u *blockchain-u*. Kompozitni ključ obezbjeđuje jedinstvenost za svaki nod i sertifikat. Funkcija koristi *putState* metod kako bi sačuvala podatke u *blockchain*. U slučaju greške, funkcija izbacuje odgovarajuću poruku o grešci koja sadrži ID transakcije kako bi olakšala dijagnostiku.

- ***getCertificate***: Ova funkcija služi za čitanje sertifikata za SN čvor. Iz konteksta (`ctx`) izvlači listu argumenata, od kojih je drugi argument identifikator noda (`deviceid`). Koristeći ovaj identifikator, kreira kompozitni ključ na isti način kao i *storeCertificate* funkcija. Ovaj ključ se koristi za dobijanje sertifikata iz *blockchain-a* putem metode *getState*. Ako sertifikat postoji, on se vraća u JSON formatu. U suprotnom, funkcija vraća informaciju da sertifikat nije pronađen.

- ***revokeNodeIfBelowThreshold***: Ova funkcija služi za opoziv sertifikata SN čvora ukoliko njegova kompozitna vrijednost povjerenja padne ispod zadatog praga. Funkcija prima kontekst (`ctx`), identifikator noda (`nodeId`), tri težinska koeficijenta (`w1`, `w2`, `w3`) i prag kompozitnog povjerenja (`composite Threshold`). Težinski koeficijenti i prag kompozitnog povjerenja se konvertuju u numeričke vrijednosti i provjerava se njihova validnost. Funkcija osigurava da suma težinskih koeficijenata iznosi 1, čime se obezbeđuje pravilna raspodjela težine među komponentama povjerenja. Zatim, funkcija dobija vrijednosti povjerenja iz *blockchain-a* koristeći ključeve kao što su `commTrust_$nodeId` za komunikaciono povjerenje (`Tc`), `packetTrust_$nodeId` za povjerenje u podatke (`Td`) i `energyTrust_$nodeId` za energetsko povjerenje (`Te`). Ako vrijednosti poverenja nisu pronađene, inicijalizuju se podrazumijevane vrijednosti. Funkcija računa kompozitnu vrijednost povjerenja kao:

$$\texttt{compositeTrust} = \texttt{w1} * \texttt{Tc} + \texttt{w2} * \texttt{Td} + \texttt{w3} * \texttt{Te} \tag{4.1}$$

  Ako je ova vrijednost ispod zadatog praga, SN čvor se dodaje na listu opozvanih čvorova koja se čuva na *blockchain*. Funkcija vraća poruku koja ukazuje na opoziv čvora, zajedno sa vrijednostima povjerenja. Ako SN čvora ostaje

validan, vraća se poruka koja potvrđuje njegovu validnost.

Pametni ugovor za upravljanje ključevima

Pametni ugovor za upravljanje ključevima omogućava čuvanje, čitanje i ažuriranje kriptografskih ključeva koji se koriste za komunikaciju između SN i CH čvorova u mreži. Funkcije unutar ovog ugovora osiguravaju bezbjednu razmjenu ključeva i verifikaciju kroz *blockchain*.

- ***getKey***: Ova funkcija omogućava dobijanje ključa za komunikaciju između određenog senzorskog čvora i glave klastera. Funkcija prima kontekst (`ctx`) i koristi argumente kako bi izvukla identifikatore (`clusterhead` i `deviceid`) i nasumičnu nonce vrijednost (`nonce`). Potom kreira kompozitni ključ koristeći ove vrijednosti i konstantu `'key'`, i koristi ga za dobijanje podataka iz *blockchain-a* putem *getState* metoda. Ako podaci nisu pronađeni, funkcija vraća informaciju da ključ nije pronađen; u suprotnom, vraća podatke o ključu. Ova funkcija se koristi kada je potrebno provjeriti postojanje i validnost ključa između senzorskog čvora i glave klastera tokom procesa autentifikacije.

- ***storeKey***: Ova funkcija omogućava čuvanje novog ključa za određeni senzorski čvor i glavu klastera. Funkcija prima kontekst (`ctx`), identifikatore noda (`deviceid`), glave klastera (`clusterhead`), ključ (`key`), nonce vrijednost (`nonce`), vremensku oznaku (`ts`), i vrijeme isteka (`texp`). Kreira model koji sadrži sve ove vrijednosti, uz dodatni atribut `confirmed` postavljen na `false` i ID transakcije (`txId`). Kompozitni ključ se kreira na isti način kao u *getKey* funkciji, i koristi se za čuvanje modela u *blockchain*-u putem *putState* metoda. Ova funkcija se koristi kada glava klastera generiše novi ključ za senzorski čvor kao dio procesa inicijalne autentifikacije.

- ***updateKey***: Ova funkcija omogućava ažuriranje postojećeg ključa u *blockchain*-u. Prima kontekst (`ctx`), identifikator senzorskog čvora (`deviceid`), identifikator glave klastera (`clusterhead`), ključ (`key`), *nonce* vrijednost (`nonce`), vremensku oznaku (`ts`), vrijeme isteka ključa (`texp`), i atribut `confirmed`. Funkcija kreira kompozitni ključ kao i prethodne funkcije, koristi *getState* metodu za čitanje trenutnog stanja ključa. Kreira model sa ažuriranim vrijednostima, sa `confirmed` atributom postavljenim na `true`, i čuva ovaj model na *blockchain* koristeći *putState* metod. Ova se funkcija koristi kada senzorski čvor potvrdi prijem i validnost ključa, kako bi se status ključa na

blockchain-u ažurirao na 'potvrđen' i označio kao spreman za upotrebu.

Pametni ugovor za upravljanje povjerenjem

Ovaj pametni omogućava upravljanje povjerenjem senzorskih čvorova u mreži na osnovu nekoliko metrika: energetski nivo, komunikacioni uspjeh i povjerenje u podatke. Funkcije unutar ovog ugovora omogućavaju skladištenje i ažuriranje ovih metrika i proračun konačne ocjene povjerenja na osnovu njih.

- ***storeMeasurement***: Ova funkcija omogućava skladištenje mjerenja i reportiranih vrijednosti nivo baterije za određeni senzorski čvor. Funkcija prima kontekst transakcije (`ctx`), identifikator čvora (`nodeId`), izmjerenu vrijednost (`measurement`), nivo baterije čvora (`energyLevel`) i vremensku oznaku mjerenja (`timestamp`). Funkcija validira ulazne podatke a zatim preuzima prethodno skladištena mjerenja i energetske nivoe iz *blockchain*-a. Dodaje nova mjerenja i energetske nivoe u postojeće liste i zadržava samo poslednjih 50 vrijednosti. Nakon toga, funkcija izračunava srednju vrijednost ($\bar{x}$) i standardnu devijaciju (σ) mjerenja. Srednja vrijednost se računa kao:

$$\bar{x} = \frac{1}{n} \sum_{i=1}^{n} x_i \tag{4.2}$$

  gde je n broj mjerenja, a x_i vrijednost pojedinačnog mjerenja. Standardna devijacija se računa kao:

$$\sigma = \sqrt{\frac{1}{n} \sum_{i=1}^{n} (x_i - \bar{x})^2} \tag{4.3}$$

  Na kraju, ažurirani podaci se skladište u *blockchain*. Ova funkcija se koristi za praćenje performansi senzorskog čvora tokom vremena i osigurava da podaci o njegovim mjerenjima i energetskom nivou budu ažurirani i tačni.

- ***updateEnergyTrust***: Ova funkcija omogućava ažuriranje povjerenja u energetski nivo čvora. Funkcija prima kontekst transakcije (`ctx`), identifikator čvora (`nodeId`) i prag ispod kojeg se energetski nivo smatra nepouzdanim (`threshold`). Funkcija preuzima postojeće energetske nivoe i njihove vremenske oznake iz *blockchain*-a. Ukoliko nema dovoljno podataka za izračunavanje stope potrošnje energije, funkcija izbacuje grešku. Vremenski prozor (`timeWindow`)

između najnovijih i prethodnih energetskih očitavanja se računa kao:

$$\text{timeWindow} = \frac{\text{mostRecentTimestamp} - \text{beforeTimestamp}}{1000} \tag{4.4}$$

a zatim se stopa potrošnje energije (r_e) računa prema formuli:

$$r_e = \frac{|\text{energyLevel}_n - \text{energyLevel}_{n-1}|}{\text{timeWindow}} \tag{4.5}$$

Funkcija takođe računa prosječnu stopu potrošnje energije (r_N) na osnovo ranijih podataka kao:

$$r_N = \frac{1}{n-2} \sum_{i=1}^{n-2} \frac{|\text{energyLevel}_{i+1} - \text{energyLevel}_i|}{\text{time}_{i+1} - \text{time}_i} \tag{4.6}$$

Na osnovu ovih vrijednosti izračunava se povjerenje u energiju (T_e):

$$T_e = \begin{cases} 0, & \text{ako je energyLevel}_n < \text{threshold} \\ 1 - |r_e - r_N|, & \text{u suprotnom.} \end{cases} \tag{4.7}$$

Povjerenje u energiju (T_e) se zatim skladišti u *blockchain*. Ova funkcija služi pravovremenu detekciju anomalija u energetskim nivoima senzorskih čvorova, koje mogu biti izazvane različitim napadima na UASN mrežu.

- ***updateCommunicationTrust***: Ova funkcija omogućava ažuriranje povjerenja u komunikaciju sa senzorskim čvorom na osnovu uspjeha ili neuspjeha komunikacionih pokušaja. Funkcija prima kontekst transakcije (`ctx`), identifikator čvora (`nodeId`) i informaciju o tome da li je komunikacioni pokušaj bio uspješan ili ne (`isSuccess`). Zatim preuzima podatke o prethodnim komunikacijma sa čvorom, i ažurira broj uspješnih (s) i neuspješnih (f) pokušaja. Na osnovu ovih podataka računa se komunikaciono povjerenje (T_c) kao:

$$T_c = \frac{s}{s+f+1} + \frac{1}{2(s+f+1)} \tag{4.8}$$

Povjerenje u komunikaciju (T_c) se zatim skladišti u *blockchain*.

- ***updatePacketTrust***: Ova funkcija omogućava ažuriranje povjerenja u pakete koje šalje senzorski čvor na osnovu statističke analize. Funkcija prima kontekst transakcije (`ctx`), identifikator čvora (`nodeId`) i vrijednost njegovog paketa koja predstavlja izvršeno mjerenje (`packetValue`). Zatim preuzima statističke podatke o očekivanoj srednjoj vrijednosti (μ) i standardnoj devijaciji (σ) mjerenja za dati čvor iz *blockchain*-a. Ocjena povjerenja u paket (T_d) iz-

vodi se na sledeći način:

$$T_d = 2\,(0.5 - 0.5 \cdot \mathrm{erf}(z)) \tag{4.9}$$

gde je erf funkcija greške, a z paramatar dobijen po formuli:

$$z = \frac{|\mathrm{packetValue} - \mu|}{\sigma\sqrt{2}.} \tag{4.10}$$

Funkcija takođe preuzima i listu postojećih ocjena povjerenja u pakete i ažurira je sa novim vrijednostima, zadržavajući samo najnovijih 50 zapisa. Na kraju, ažurirani podaci se skladište u *blockchain*. Na ovaj način se osigurava brza detekcija anomalija u vrednostima paketa.

Kompletan kod za opisane funkcije može se naći u Prilogu, osim funkcija Pametnog ugovora za upravljanje povjerenjem koje su zbog obimnosti izostavljene ali su dostupne u Github repozitorijumu [1].

4.2 Implementacija ECQV i HOMQV protokola

U cilju efikasne implementacije svih potrebnih operacija na eliptičnoj krivoj korišćena je *lightweight* microECC biblioteka [80], specijalno dizajnirana za uređaje sa ograničenim resursima. Korišćena je SECG-P192 eliptična kriva, koja postavlja kompromis između efikasnosti i sigurnosti. Ova kriva podrazumijeva robustan nivo sigurnosti od 96 bita, privatne ključeve dužine 24 bajta i javne ključeve dužine 48 bajta. Radi postizanja većeg nivoa sigurnosti izvršena je integracija SHA-256 heš funkcije sa HMAC-SHA256 funkcijom za derivaciju ključeva. U cilju pojednostavljenja komunikacije u mreži, 1 bajt poruke je korišćen za identifikaciju tipa poruke, 3 bajta za jedinstvenu identifikaciju svakog čvora a 4 bajta za prenos *nonce* vrijednosti. Kroz kompletnu implementaciju poštovani su sledeći dizajnerski principi i najbolje prakse:

- **Kriptografske operacije:** Kritične kriptografske operacije su implementirane bez uslovnog grananja, koristeći kod sa konstantnim vremenom izvršenja kako bi se smanjio rizik od vremenskih bočnih kanala [81].
- **Upravljanje memorijom:** Alokacija varijabli je vršena putem statičkih metoda za alokaciju memorije, izbjegavajući upotrebu tehnika dinamičke alokacije memorije.

[1] `https://github.com/monusen-uom/BEKMP-Blockchain-Platform`

- **Prenos podataka:** Tačke eliptične krive su prenošene u kompresovanom formatu kako bi se smanjilo opterećenje mreže, u skladu sa standardima navedenim u SEC 1 [82]. Ovaj pristup rezultira kompaktnom dužinom sertifikata od samo 32 bajta, pri čemu se 25 bajtova koristi za tačku eliptične krive, 3 bajta za identifikaciju, i 4 bajta za T_{exp}. Ova strateška optimizacija osigurava da svi podaci koji se prenose ostanu unutar maksimlanog limita za veličinu korisnog dijela paketa od 48 bajta (bez *Base64* enkodiranja) koji podržavaju NMV3 akustični modemi [83].

U nastavku će biti dato više detalja o implementaciji najvažnijih segmenata ECQV i HOMQV protokola dok se kompletan kod može pronaći u GitHub repozitorijumu [2].

4.2.1 Kreiranje zahtjeva za sertifikat i generisanje ključeva i sertifikata

U ovom dijelu opisane su funkcije za kreiranje zahtjeva za izdavanje sertifikata (*csr_request*), generisanje sertifikata (*crt_generation*), generisanje ključa sesije na CH čvoru (*homqv_generate_key*) i računanje ključa sesije na SN čvoru (*homqv_regenerate_key*). Kompletan kod ovih funkcija dat je u Prilogu teze.

- ***csr_request***: Ova funkcija generiše par javnog i privatnog ključa i kreira zahtjev za sertifikat (CSR) koji sadrži ID čvora i kompresovani javni ključ. Funkcija kao argument prima strukturu CSR (`csr`) u koju ce biti upisan generisani zahtjev za sertifikat, privatni ključ (`private`), identifikator (`id`) i parametre krive (`curve`). Funkcija prvo inicijalizuje nizove za javni ključ i kompresovani javni ključ. Zatim generiše par ključeva koristeći *uECC_make_key* funkcije, kompresuje javni ključ pomoću *uECC_compress* funkcije, i kopira ID i kompresovani ključ u CSR strukturu.

- ***crt_generation***: Ova funkcija generiše sertifikat koristeći CSR, privatni i javni ključ Sertifikacionog Autoriteta (CA) i parametre eliptične krive. Funkcija prima kao argumente strukture koje će sadržati rezultujuće podatke: parametar za rekonstrukciju sertifikata (`r`) i sam sertifikat (`crt`). Pored njih, funkciji se prosleđuju svi potrebni ulazni parametri: CSR (`csr`), privatni ključ CA (`private_ca`), javni ključ CA (`public_ca`) i parametri eliptične krive (`curve`). Funkcija inicijalizuje promjenljive za javni ključ, nasumični privatni

[2] `https://github.com/monusen-uom/BEKMP-ECC-benchmark`

ključ, izvedeni javni ključ i druge potrebne nizove. Javni ključ se dekompresuje pomoću *uECC_decompress* funkcije i validira pomoću *uECC_valid_public_key* funkcije. Ukoliko je ključ nevažeći, prekida se dalje izvršavanje. Funkcija zatim generiše nasumični par ključeva, generiše javni ključ kao tačku na eliptičnoj krivoj, kompresuje ovu tačku i računa heš vrijednost. Nakon toga, funkcija množi tačku sa nasumično generisanom vrijednošću i sabira dobijeni rezultat sa javnim ključem CA, osiguravajući da rezultat nije nula. Na kraju, računa se konačni potpis množenjem heš vrednosti sa nasumičnim privatnim ključem i sabiranjem sa privatnim ključem od CA. Rezultati se na kraju kopiraju u izlazne nizove.

- ***homqv_generate_key***: Ova funkcija generiše ključ koristeći HOMQV protokol na osnovu sertifikata (CRT), privatnog i javnog ključa Sertifikacionog Autoriteta (CA), identifikatora klastera i parametara eliptične krive. Funkciji se kao argumenti proslijeđuju struktura u koju se na kraju upisuje generisani ključ (`key`), kao i struktura `Y` u koju se upisuje vrijednost koja će biti poslata senzorskom čvoru kao odgovor i koju će čvor koristiti da regeneriše isti ključ. Zatim se kao argumenti proslijeđuju i sertifikat (`crt`), privatni ključ CA (`private_ca`), javni ključ CA (`public_ca`), identifikator klastera (`id_cluster`), i parametri krive (`curve`). Kao prvi korak funkcija poziva *crt_pk_extract* da izvuče javni ključ čvora iz njegovog implicitnog sertifikata (`crt`). Funkcija inicijalizuje nizove i promjenljive za privatni ključ, identifikatore i potrebne bafere. Zatim se generiše nasumični par ključeva (`Y` i `y`) pozivom `uECC_make_key` funkcije. Kopira se identifikator iz sertifikata (`crt`) i kombinacija (`Y + id`) se hešira kako bi se dobila vrijednost `_e`. Zatim se `_e` množi sa privatnim ključem CA i dodaje se privatni ključ (`y`) kako bi se dobila vrijednost `_yeb`. Tačka na krivoj se množi sa vrijednošću `_yeb` da bi se dobila tačka `_sigma`. Tačka `_sigma` se konvertuje nazad u bajt format i kombinuje sa identifikatorima (`id` i `id_cluster`) i javnim ključem (`Y`). Ova kombinacija se hešira kako bi se dobio konačni ključ.

- ***homqv_regenerate_key***: Ova funkcija omogućava senzorskom čvoru da rekreira isti zajednički ključ koji je generisan funkcijom *homqv_generate_key* koju je izvršila glava klastera. Funkcija kao argument prima strukturu u koju se na kraju upisuje regenerisani ključ (`key`) kao i javni ključ (`Y`) koji je dobijen od strane glave klastera, javni ključ CA (`public_ca`), privatni ključ čvora (`private`), identifikator klastera (`id_cluster`), identifikator čvora (`id`) i

parametre krive (`curve`). Zatim se vrijednosti `public_ca`, `Y` i `private` kopiraju u interne strukture. Identifikator čvora (`id`) i javni ključ (`Y`) se kombinuju i heširaju kako bi se dobila vrijednost `_e`. Nakon toga, vrši se množenje tačke na eliptičnoj krivoj sa vrijednošću `_e` i javnim ključem CA (`public_ca`), da bi se dobila tačka `_ePu`. Ova tačka se sabira sa javnim ključem (`Y`) kako bi se dobila tačka `_ePuY`. Privatni ključ čvora (`private`) se zatim množi sa tačkom `_ePuY` kako bi se dobila tačka `_sigma`. Tačka `_sigma` se konvertuje nazad u bajt format i kombinuje sa identifikatorima (`id` i `id_cluster`) i javnim ključem (`Y`). Ova kombinacija se ponovo heširа kako bi se dobio konačni ključ. Funkcija osigurava sigurno rekreiranje zajedničkog ključa primijenjujući HOMQV protokol.

Navedene funkcije omogućavaju sigurno generisanje i verifikaciju sertifikata, kao i razmjenu ključeva koristeći eliptične krive (ECC), čime se obezbjeđuje pouzdana autentifikacija i integritet podataka u mreži.

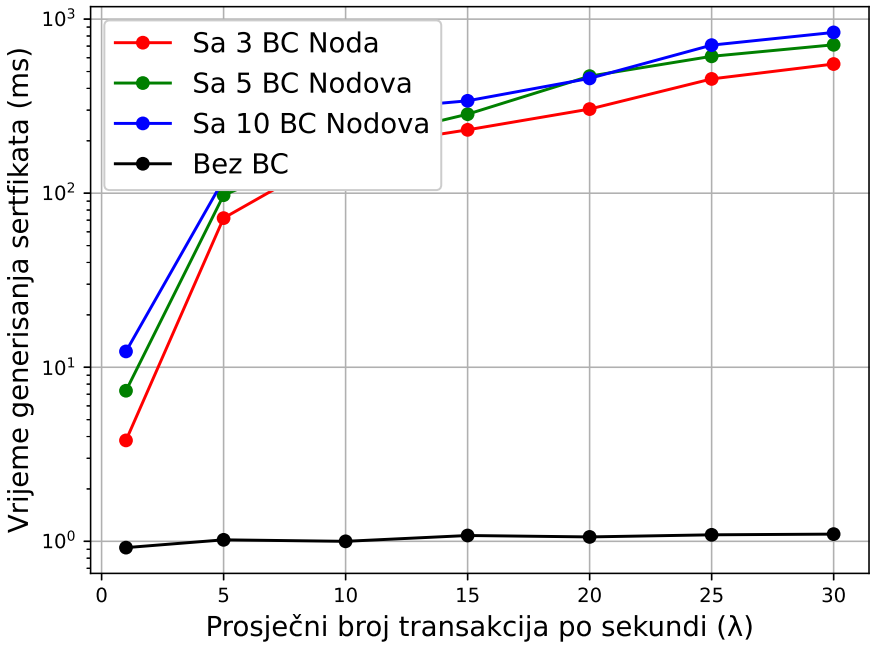
Glava 5

Analiza i evaluacija performansi

U ovoj glavi biće predstavljena detaljna analiza efikasnosti predloženog BEKMP protokola za autentifikaciju i upravljanje ključevima, kroz ispitivanje računarskih performansi, komunikacionog opterećenja i energetske efikasnosti. Inicijalno će biti prikazani rezultati analize performansi *blockchain*-a, s posebnim fokusom na vrijeme odziva pametnih ugovora tokom različitih operacija. Nakon toga, opisano je eksperimentalno ispitivanje protokola na stvarnim UASN uređajima, uz diskusiju o dobijenim rezultatima. Na kraju poglavlja, prikazana je implementacija protokola u simulatoru *DESERT Underwater* [84], gdje su analizirane složenije mrežne konfiguracije. Tokom ovih evaluacija, efikasnost BEKMP-a upoređena je sa srodnim protokolom referenciranim u [23], čime je pružena komparativna analiza njegove operativne efikasnosti.

5.1 Analiza performansi *blockchain* mreže

Simulacije su sprovedene na x86_64 GNU/Linux sistemu opremljenom sa jednim virtuelnim CPU-om (1.7 GHz) i 512 MB RAM-a. Koristeći Docker Engine, implementirana je *blockchain* mrežu koja se sastoji od $N_b \in \{3, 5, 10\}$ *Peer* nodova. U skladu sa arhitekturom *Hyperledger Fabric*-a, implementirana *blockchain* mreža se sastojala od dvije vrste *blockchain* entiteta: *Endorser* i *Orderer* nodova. Svaki CH čvor u implementiranoj mrežnoj konfiguraciji je djelovao kao *Endorser*, odgovoran za hostovanje kopije *blockchain* registra, izvršavanje pametnih ugovora i validaciju predloga transakcija. Ulogu *Orderer*-a dodijeljena je jednom CH čvoru, zaduženom za kompajliranje svih odobrenih transakcija u nove, sekvencijalno poređane blokove, čime se osigurava integritet mreže i doslijedno procesiranje transakcija.

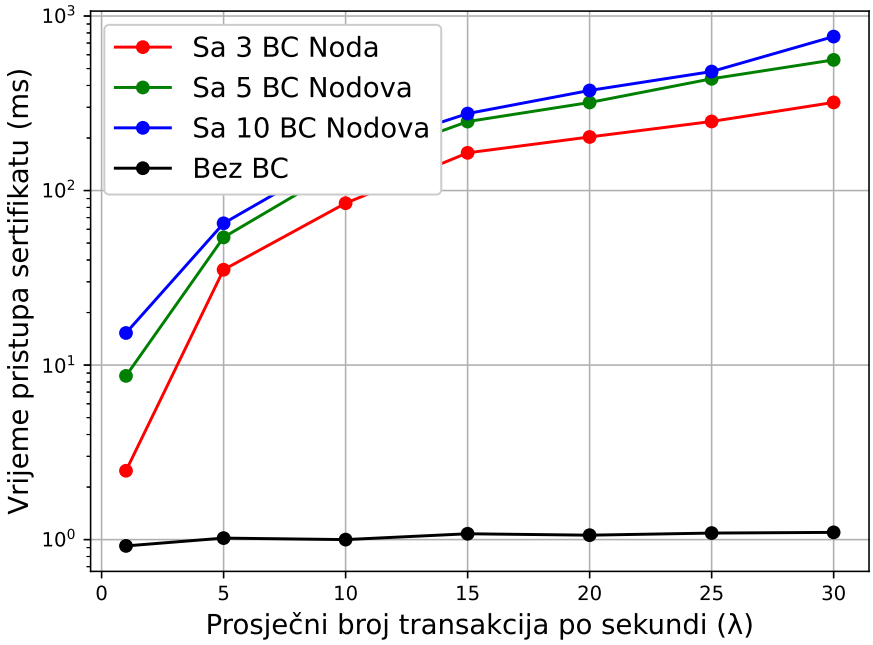


Slika 5.1: Prosječno vrijeme generisanja sertifikata.

U cilju procjene efikasnosti pametnih ugovora, podvodni SN čvorovi su simulirani sa više-nitnim softverom. U ovim simulacijama, broj niti, koje predstavljaju SN čvorove koji šalju zahtjeve *blockchain* mreži, varirao je prema Poasonovoj raspodjeli. Srednja vrijednost ovog Poasonovog procesa (λ) sistematski je mijenjana od 1 do 30 transakcija po sekundi, što je omogućilo analizu performansi pametnih ugovora pod različitim opterećenjima. Proces evaluacije je uključivao detaljno ispitivanje ključnih funkcija pametnih ugovora: generisanje sertifikata, generisanje ključeva, pristup ključevima i ažuriranje nivoa povjerenja. Rezultati ovih evaluacija su prikazani su i diskutovani u nastavku.

Generisanje sertifikata: Izmjereno je vrijeme koje protekne od trenutka kada CH čvor primi zahtjev za registraciju do trenutka kada se generiše implicitni sertifikat. Ovaj proces uključuje validaciju ID-a SN čvora na *blockchain-u* i izvršavanje CA servisa. Rezultati u pogledu prosječnog vremena generisanja sertifikata su prikazani na Slici 5.1, poredeći scenarije sa i bez integracije *blockchain*-a (BC). Kao što je očekivano, interakcije sa *blockchain*-om dovode do povećanja vremena generisanja sertifikata. Međutim, ovo dodatno vrijeme obrade nije kritično, s obzirom na sigurnosne prednosti koje donosi *blockchain* i čenjenicu da se ova operacija ne izvršava često.

Pristup sertifikatu: Pristup sertifikatima je ključna funkcija jer se koristi tokom svake razmjene ključeva. Kao što je prikazano na Slici 5.2, prosječno vrijeme

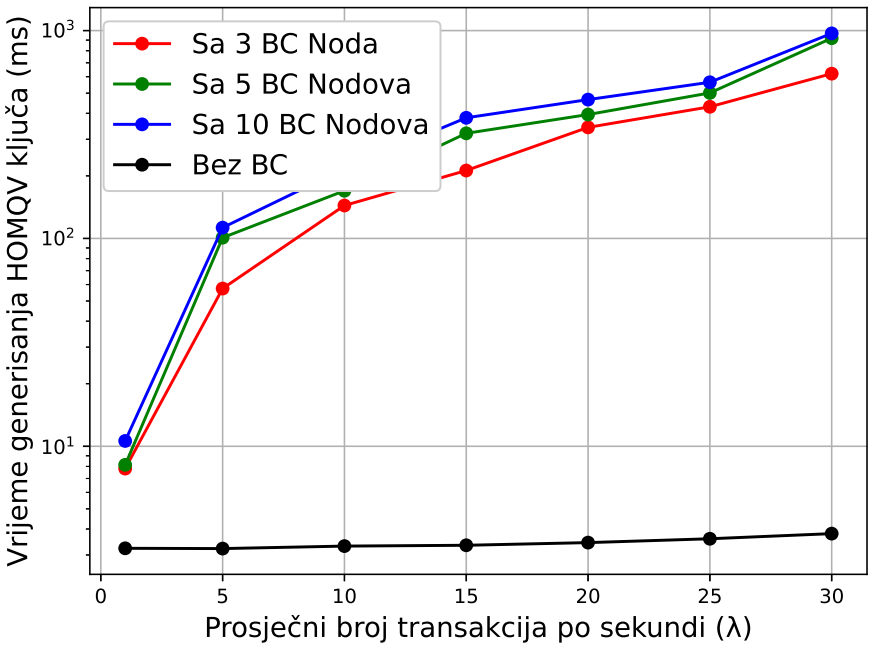


Slika 5.2: Prosječno vrijeme pristupa sertifikatu.

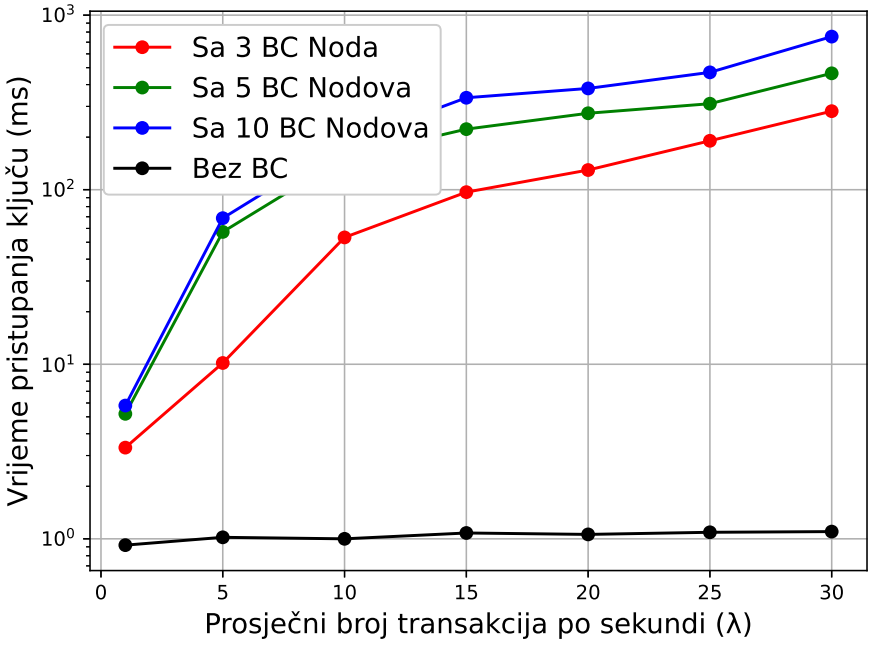
preuzimanja sertifikata je relativno malo. Korišćenje implicitnih sertifikata ne samo da minimizira zahtjeve za skladištenjem, već i značajno poboljšava brzinu pristupa.

Kreiranje ključa i pristup ključu: Proces generisanja ključeva koristi HOMQV protokol. Slika 5.3 prikazuje prosječno vrijeme potrebno za generisanje ključeva. U scenariju sa *blockchain*-om, prikazani rezultati se odnose na nekoliko koraka obrade koji slijede nakon prijema zahtjeva za generisanje ključa. Ovi koraci uključuju pristup *blockchain*-u radi verifikacije sertifikata, derivaciju ključa i skladištenje enkriptovanog ključa na *blockchain*-u kako bi se omogućila reautentifikacija između klastera. Korak enkripcije je značajan faktor koji doprinosi očiglednom povećanju vremena obrade. Rezultati u pogledu prosječnog trajanja preuzimanja enkriptovanog ključa dati su na Slici 5.4.

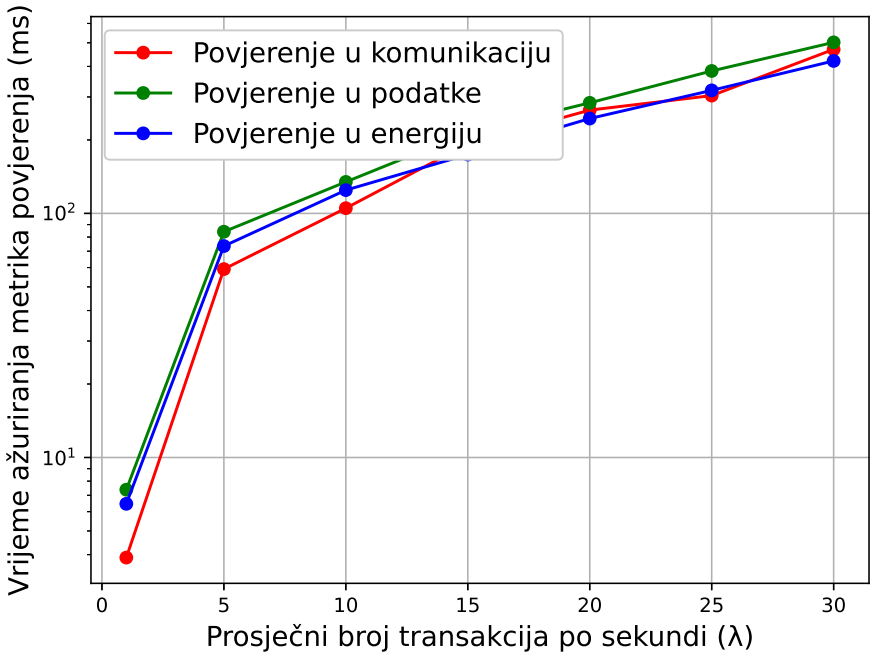
Upravljanje povjerenjem: Ocjene povjerenja SN čvorova se kontinuirano ažuriraju i izračunavaju pri prijemu paketa ili detekciji gubitka paketa. Informacije o posljednjih 50 dokaza o energiji, komunikaciji i podacima se skladište u enkriptovanom formatu na *blockchain*-u. U slučajevima kada se prijetnja identifikuje na osnovu ukupnog nivoa povjerenja SN čvora, sertifikat noda se automatski opoziva. Vrijeme potrebno za pokretanje operacija ažuriranja povjerenja putem funkcija pametnih ugovora prikazano je na Slici 5.5. Generalno, sve operacije ažuriranja povjerenja pokazuju sličnu efikasnost, pri čemu je obrada povjerenja u podatke najviše vremenski zahtjevna. Rezultati u pogledu vremena opoziva sertifikata prikazani su na Slici 5.6.



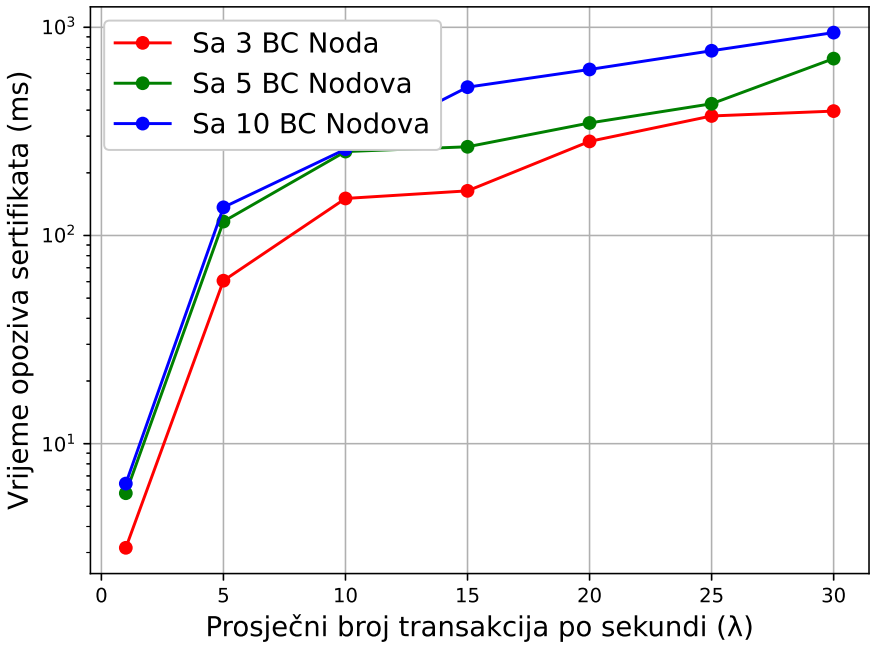
Slika 5.3: Prosječno vrijeme generisanja ključa.



Slika 5.4: Prosječno vrijeme preuzimanja ključa.



Slika 5.5: Performanse *blockchain* mreže prilikom ažuriranja povjerenja.



Slika 5.6: Vrijeme opoziva sertifikata na *blockchain* mreži.

Proces opoziva uključuje preuzimanje sva tri dokaza povjerenja sa *blockchain*-a i zatim izračunavanja težinske sume kako bi se odredio ukupni nivo povjerenja, koji se koristi kao osnova za opoziv sertifikata.

5.2 Analiza performansi UASN mreže

U ovoj sekciji sprovedena je analiza performansi BEKMP protokola u UASN okruženjima, koristeći kombinaciju simulacija i eksperimentalnih metoda. U početnoj fazi, eksperimenti su izvedeni na stvarnim uređajima kako bi se prikupili ključni podaci, čime je unaprijeđen realizam kasnijih simulacija.

5.2.1 Eksperimentalni rezultati

U cilju dodatne validacije predloženog protokola u praktičnom kontekstu, posebno njegove vremenske efikasnosti u podvodnom okruženju, sproveden je niz eksperimenata sa manjom UASN mrežom. Eksperimenti su realizovani tokom radionice *Breaking the Surface* 2023 [85] u Kumboru, Crna Gora. Mrežna konfiguracija koja je korišćena u eksperimentima uključivala je jedan Arduino Mega 2560 mikrokontrolerski uređaj, koji je funkcionisao kao podvodni SN čvor, i tri aPad površinska vozila [79], koja su služila kao CH čvorovi u mreži (Slika 5.7). Svaki uređaj u ovoj konfiguraciji bio je opremljen NMV3 podvodnim akustičnim modemom [83] za potrebe podvodne komunikaciju. Pored toga, aPad vozila su bila opremljena *WiFi* antenama za komunikaciju preko vazduha.

Tokom eksperimenata mjereno je ukupno kašnjenje koje unosi proces autentifikacije primjenom BEKMP protokola. Za potrebe komparativne analize, referentni protokol iz rada [23] implementiran je na identičnom hardveru i testiran pod istim uslovima. Mjereno je vrijeme izvršavanja svih ključnih faza analiziranih protokola na podvodnom SN uređaju. Dobijeni rezultati su prezentovani u Tabelama 5.1 i 5.2. Pored toga, mjerena je računarska efikasnost u pogledu vremena izvršavanja različtiih operacija na CH čvorovima, a rezultati su prikazani u Tabelama 5.3 i 5.4.

Važno je napomenuti da podvodni akustični modemi korišćeni u eksperimentima podržavaju maksimalnu veličinu *payload*-a paketa od 64 bajta. Zbog ovog ograničenja, proces izdavanja sertifikata u predloženom protokolu bio je podijeljen na dva dijela:

- slanje ECQV sertifikata, i

Slika 5.7: Eksperiment u Jadranskom moru sa aPad-ovima kao glavama klastera.

Tabela 5.1: Vremenska efikasnost BEKMP operacija u UASN mreži.

BEKMP operacije	**Trajanje (ms)**
Generisanje zahtjeva za sertifikat	1136
Dobijenje sertifikata	6746
Ekstrakcija javnog ključa	1077
Validacija sertifikata	2155
Preuzimanje materijala za rekonstrukciju ključa (c) i izračunavanje dugoročnog ključa	6460
Potpuna autentifikacija	10924

Tabela 5.2: Vremenska efikanost autentifikacionog protokola [23] u UASN mreži.

[23] **Operacije**	**Trajanje (ms)**
Dobijanje tajnog ključa	11302
Dobijanje javnog ključa	6750
Preuzimanja parametra M	6440
Potpuna autentifikacija	23649

Tabela 5.3: Vremenska efikasnost BEKMP *blockchain* (BC) operacija.

BEKMP BC operacija	**Trajanje (ms)**
Verifikacija identiteta i statusa SN čvora	7.2
Vrijeme kreiranja sertifikata	1.04
Vrijeme skladištenja sertifikata	1.95
Vrijeme generisanja ključa	3.44
Vrijeme skladištenja ključa	5.92
Vrijeme preuzimanja ključa	1.55

Tabela 5.4: Vremenska efikasnost BC operacija u protokolu [23].

BC operacija u [23]	**Trajanje (ms)**
Validacija uređaja	7.32
Registracija uređaja	3.09
Vrijeme skladištenja L	1.94

- prenos materijala za rekonstrukciju ključa c.

Slično tome, proces registracije za protokol predložen u [23] takođe je zahtijevao prenos više paketa uslijed ograničenja modema. U Tabelama 5.1 i 5.2, trajanje *potpune autentifikacije* je definisano kao ukupno vrijeme potrebno da čvor kompletira svoj proces autentifikacije, počevši od trenutka kada pošalje zahtjev za autentifikaciju nadređenom CH čvoru. Jasno je da predloženo rešenje postiže bržu autentifikaciju koristeći jednosmjernu razmjenu ključeva, za razliku od dvosmjerne razmjene korišćene u [23]. Kao što se može vidjeti iz Tabela 5.3 i 5.4, računarski zahtjevi na CH čvorovima imali su minimalan uticaj na ukupno trajanje autentifikacije, što se može pripisati relativno visokim procesorskim kapacitetima aPad uređaja.

Potrebno je istaći da su eksperimenti bili otežani smetnjama iz okruženja, koje su povremeno uticale na efikasnost komunikacije između SN i CH uređaja. Šum je povremeno uzrokovao gubitke paketa i dodatna kašnjenja. Međutim, u cilju obezbjeđivanja fer poređenja, vremenska efikasnost autentifikacionih protokola je evaluirana samo na osnovu mjerenja u toku sesija koje nisu pretrpljele nikakve gubitke u fazi autentifikacije.

Kada je u pitanju računarska efikasnost, BEKMP protokol je upoređen sa referentnim protokolom iz rada [23] razmatrajući broj izvršavanja kriptografske heš funkcije (N_h) i skalarnih multiplikacija zasnovanih na ECC-u (N_{ecc}) tokom faza registracije i autentifikacije. BEKMP protokol koristi ECQV protokol tokom registracije, što podrazumjeva generisanje kratkotrajnog para ključeva, sertifikata i validaciju ovog sertifikata. Ovaj proces uključuje tri skalarne multiplikacije i računanje tri heša.

Vrijeme potrebno za izvršavanje operacija XOR i konkatenacije može zanemariti zbog njihove veoma male računske kompleksnosti. Proces registracije opisan u [23] uključuje jednostavniji mehanizam, gdje centralni autoritet generiše par ključeva (privatni i javni) za sve SN čvorove. Iako je BEKMP-ov ECQV protokol složeniji, posebno u pogledu generisanja i validacije sertifikata, on nudi značajne prednosti za komunikaciju između SN čvorova. Nakon što SN čvorovi registruju i pohrane svoje ECQV sertifikate, mogu sigurno komunicirati jedni s drugima direktnom razmjenom sertifikata, što značajno pojednostavljuje rad mreže i smanjuje komunikaciono opterećenje. U fazi autentifikacije, računska kompleksnost referentnog protokola [23] je kvantifikovana kao $5 \times N_h + 6 \times N_{ecc}$. Nasuprot tome, računska kompleksnost BEKMP-a iznosi $2 \times N_h + 4 \times N_{ecc}$.

U Tabeli 5.5 je upoređeno komunikaciono opterećenje analiziranih protokola. BEKMP protokol zahtijeva ukupno 140 bajta za registraciju, dok referentni protokol [23] zahtijeva 164 bajta za isti proces. Koristeći HOMQV, BEKMP protokol smanjuje komunikaciono opterećenje SN-CH autentifikacije na samo 64 bajta. Međutim, u scenarijima međusobne autentifikacije između dva SN čvora, komunikacioniono opterećenje je veće (104 bajta), prvenstveno zbog prenosa sertifikata. Nasuprot tome, protokol iz [23] uključuje obimniju razmjenu poruka, što rezultira ukupnim komunikacionim opterećenjem od 204 bajta za kompletiranje autentifikacije, bez obzira da li se ona sprovodi između SN čvorova ili između SN i CH čvora. Važno je napomenuti da ukupno komunikaciono opterećenje značajno zavisi od ograničenja veličine paketa podvodnih akustičnih modema, jer ovo ograničenje direktno utiče na ukupan broj razmijenjenih paketa.

Značajna prednost predloženog sigurnosnog protokola je njegova efikasnost u među-klasterskoj autentifikaciji, gdje ne stvara dodatno komunikaciono i računarsko opterećenje tokom reautentifikacije prilikom prelaska SN čvora iz jednog klastera u drugi. Ova efikasnost proizlazi iz činjenice da CH čvorovi dobijaju zajednički ključ direktno iz *blockchain* mreže, za razliku od pristupa iz rada [23], koji zahtijeva dvije dodatne heš operacije, praćene prenosom dvije dodatne poruke, što stvara ukupno 40 bajta komunikacionog opterećenja tokom mobilnosti između klastera.

Analiza jasno pokazuje da predloženi protokol značajno smanjuje kašnjenje u autentifikaciji, što je rezultat smanjenog komunikacionog opterećenja i veće računarske efikasnosti.

Tabela 5.5: Poređenje komunikacionih opterećenja.

Operacija	**BEKMP (u bajtima)**	[23] **(u bajtima)**
Zahtjev za registraciju	48	36
Odgovor za registraciju	92	128
SN-CH autentifikacija	64	204
SN-SN autentifikacija	104	204

5.2.2 Simulacioni rezultati

Za potrebe procjene efikasnosti predloženog BEKMP protokola i poređenja njegove efikasnosti sa protokolom referenciran u [23] u većim UASN-ovima, korišćen je *DESERT Underwater* simulator [84]. Razmotrane su tri UASN konfiguracije sa 6, 12 i 25 SN-ova raspoređenih unutar volumetrijske podvodne oblasti dimenzija 2 km x 2 km x 600 m. Tri CH-a su strateški postavljena na fiksnim koordinatama: (1000, 333, 0), (666, 1666, 0) i (1333, 1666, 0), osiguravajući sveobuhvatnu pokrivenost. Ukupno trajanje simulacija podešeno je na 28 sati.

U simulacijama su SN i CH čvorovi bili konfigurisani sa akustičnim modemima koji rade na frekvenciji od 24 kHz i širini opsega od 5 kHz. Simulirani modemi podržavaju brzinu prenosa od 640 b/s i predajnu snagu od 160 dB re uPa, sa potrošnjom energije pri prijemu od 0.62 W. Ova konfiguracija odražava specifikaciju NMV3 akustičnih modema koji su korišćeni u eksperimentima, pružajući realističnu osnovu za procjenu performansi protokola u mrežama sa većim brojem čvorova.

Mobilnost SN čvorova je simulirana korišćenjem ugrađenog modula *uwdriftposition DESERT Underwater* simulatora [86]. Ovaj modul dodjeljuje svakom SN čvoru deterministički vektor brzine, koji se dalje modifikuje vremenski promjenljivim vektorima "šuma"kako bi se tačno predstavio uticaj morskih struja na kretanje SN čvora. Modul kontinuirano ažurira lokaciju čvora na osnovu smjera i brzine struje. Svako ažuriranje uključuje slučajni šum kako bi se imitirao talasasti pokret tipičan za objekte koji plutaju u vodi. Uvođenjem ovih dinamičkih elemenata u topologiju mreže, modul efikasno simulira uticaj morskih struja i podržava realistične scenarije mobilnosti između klastera.

Performanse protokola su evaluirane pod pretpostavkom da svi SN čvorovi uspostavljaju siguran kanal sa CH čvorovima. Podaci koje prenosi svaki SN čvor generisani su prema Poasonovoj raspodjeli srednje vrijednosti 1 paket po minuti. Kada ima paket za slanje, SN čvor šalje taj paket CH čvoru čim procijeni da je kanal slobodan, koristeći /*Carrier Sense Multiple Access* (CSMA) MAC protokol kako

bi se minimizirao rizik od kolizija. Pretpostavljeno je da se komunikacija između SN čvorova i CH čvorova uvijek odvija preko direktnih veza, tj. bez intervencije relejnih čvorova i protokola rutiranja. Veličina paketa koji se prenose mijenjana je u različitim fazama SN-CH komunikacije u skladu sa specifičnostima analiziranih protokola. U simulacijama su uzeta u obzir i kašnjenja uslijed obrade na SN i CH čvorovima, koristići rezultate mjerenja iz eksperimenata na moru koji su prethodno opisani.

Svi zahtevi za autentifikaciju prenošeni su u vidu *broadcast* paketa, tako da ih može obraditi više CH čvorova ako se SN nalazi unutar preklapajućih oblasti pokrivanja više klastera. S obzirom da CH čvorovi formiraju *blockchain* mrežu, odluke o autentifikaciji se donose putem konsenzusa. Na zahtjev obično odgovara samo CH najbliži SN čvoru, osim u slučajevima kada dođe do gubitka paketa ili kolizije. Ako više CH čvorova primi zahtjev, oni postižu konsenzus o tome koji će izvršiti autentifikaciju i odgovoriti. Ovaj konsenzus se uspostavlja brzom radio komunikacijom preko vazduha. Nakon autentifikacije, SN čvor može prenositi podatke u enkriptovanom obliku koristeći ključ razmijenjen sa odgovarajućim CH čvorom. Ovaj ključ ima rok važenje od 10 minuta. Po isteku, SN čvor se podstiče da pokrene novu proceduru autentifikacije.

U *DESERT Underwater* simulatoru dva modula su dostupna za simulaciju aplikacionog sloja: *uwcbr* i *uwvbr* moduli. Oba modula generišu mrežni saobraćaj slanjem paketa sa *dummy* sadržajem, ali to rade koristeći različite mehanizme. Modul *uwcbr* je iskorišćen za implementaciju podvodnih senzorskih čvorova i simulira saobraćaj sa konstantnom brzinom prenosa podataka. Ovaj modul podržava dva načina generisanja saobraćaja: slanje paketa kroz mrežu u konstantnim vremenskim intervalima i korišćenje Poasonovog procesa sa zadatom srednjom vrednošću. U sprovdenim simulacijama, generisanje paketa se, kao što je već navedeno, zasnivalo na Poasonovom procesu. Na transportnom nivou protokola korišćen je *uwudp* modul, dok je za mrežni nivo korišćen *uwflooding* modul kako bi se obezbijedilo *broadcast* slanje paketa za autentifikaciju, sa poljem *Time to Live* (TTL) setovanim na 1.

Za potrebe simulacije glava klastera (CH čvorova) razvijen je poseban aplikacioni modul - *uwsink*. U cilju efikasne simulacije *blockchain* komponente unutar *DESERT Underwater* simulatora, korišćena je *Redis* platforma [87], odabrana zbog svoje sposobnosti da podrži atomske operacije na kompleksnim tipovima podataka. Prilikom obrade zahteva za autentifikaciju, simulirani CH čvorovi upisuju podatke u Redis, koji funkcioniše kao decentralizovani registar. Na ovaj način uspešno se simulira de-

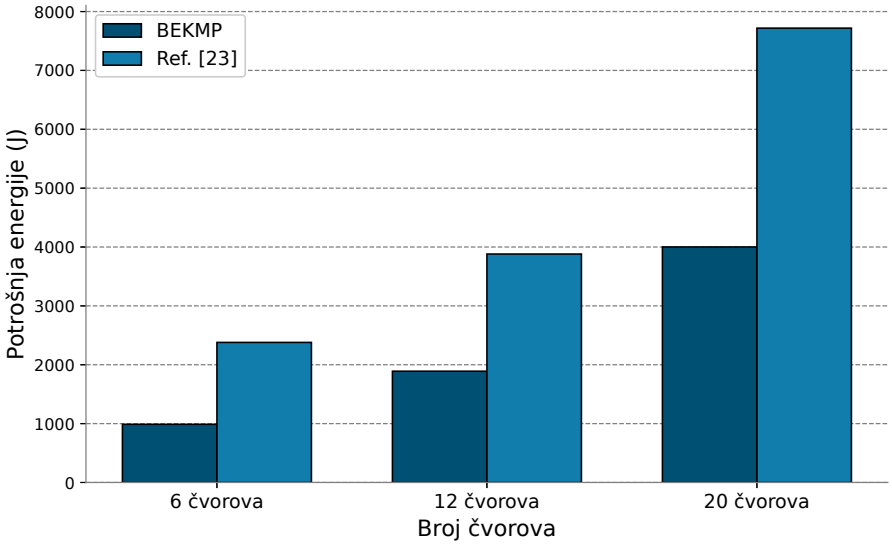
centralizovana i konsenzusna priroda *blockchain* tehnologije, osiguravajući efikasno i precizno rukovanje podacima o autentifikaciji bez potrebe za spoljnim mehanizmima zaključavanja, čime se održava integritet *multi-threaded* DESERT okruženja. U prilogu je data implementacija segmenta *uwsink* modula koja prikazuje kako je *Redis* integrisan za simulaciju *blockchain* konsenzusa za potrebe autentifikacije i među-klasterske reautentifikacije. Kompletan kod se može pronaći u GitHub repozitorijumu [1].

U simulacijama je mjerena ukupna potrošnja energije SN čvorova tokom aktivnosti prenosa i prijema podataka povezanih sa procesima autentifikacije i međuklasterkse reautentifikacije. Pored toga, mjereno je kašnjenje koje protokoli unose, uzimajući u obzir vrijeme potrebno da se SN čvor autentifikuje/reautentifikuje CH čvorovima.

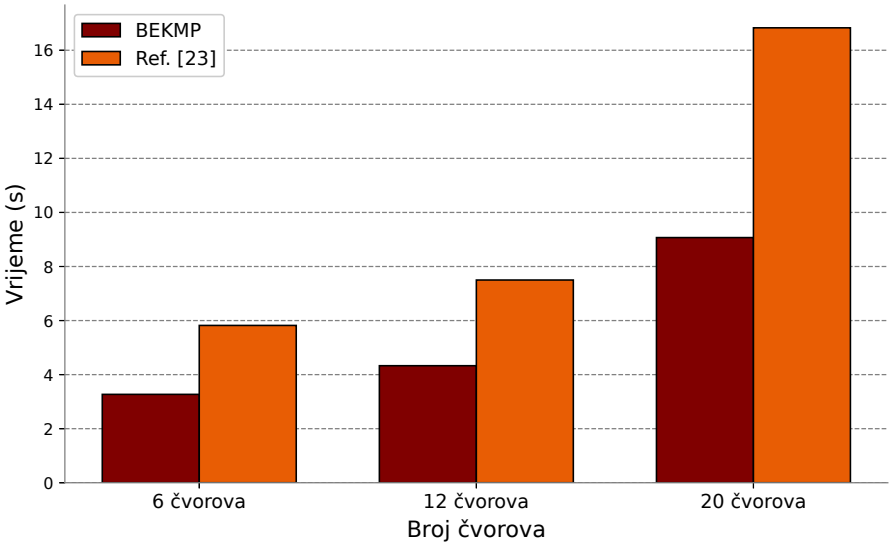
Slika 5.8 ilustruje prosječnu potrošnju energije za BEKMP protokol u poređenju sa protokolom predloženim u [23]. Kao što je i očekivano, potrošnja energije raste sa veličinom mreže zbog većeg broja uređaja uključenih u komunikaciju i veće vjerovatnoće kolizije paketa. Rezultati jasno pokazuju da je BEKMP protokol značajno energetski efikasniji, sa ukupnom potrošnjom energije koja je najmanje 92,8% niža od potrošnje protokola referenciranog u [23]. Ovo značajno smanjenje se pripisuje korišćenju manjih paketa i manjem broju transmisija paketa u BEKMP protokolu. Pored toga, za razliku od protokola [23], koji koristi pojednostavljeni proces reautentifikacije tokom kretanja između klastera, BEKMP protokol eliminiše potrebu za ponovnom autentifikacijom u potpunosti. Ako SN čvor pređe iz područja pokrivanja jednog klastera u drugi, CH drugog klastera može odmah dekriptovati njegove podatke jer svi CH čvorovi održavaju kritične sigurnosne informacije u zajedničkom decentralizovanom registru.

Slika 5.9 prikazuje prosječno vrijeme potrebno SN čvoru da uspješno uspostavi siguran kanal sa CH čvorom svog klastera, uključujući kašnjenja u propagaciji, prenosu i obradi za sve razmjenjene pakete. Važno je napomenuti da je vrijeme obrade zanemarljivo u poređenju sa ostalim kašnjenjima, s obzirom na brzinu zvuka pod vodom od približno 1500 m/s i malu brzinu prenosa akustičnih modema. Rezultati jasno pokazuju da BEKMP protokol, sa manjim komunikacionim opterećenjem, uspostavlja sigurne kanale značajno brže od protokola predloženog u [23]. Konkretno, svim simuliranim topologijama, BEKMP protokol je u prosjeku barem 73,2% brži od protokola iz [23].

[1] `https://github.com/monusen-uom/Desert-BEKMP`



Slika 5.8: Potrošnja energije SN čvora u UASN mrežama različite veličine.



Slika 5.9: Prosječno vrijeme uspostavljanja sigurnog kanala između SN i CH čvora u UASN mrežama različite veličine.

Zaključak

U ovom radu predložen je protokol za upravljanje ključevima zasnovan na *blockchain* tehnologiji koji je dizajniran da obezbijedi sigurnu i brzu autentifikaciju u podvodnim akustičnim senzorskim mrežama sa klasterskom strukturom. Predloženi protokol koristi inherentne sigurnosne karakteristike *blockchain* tehnologije kako bi eliminisao potrebu za ponovnom autentifikacijom senzorskih čvorova prilikom promjene klastera, pri tome štiteći mrežu od brojin spoljašnjih i unutrašnjih prijetnji. Jenda od ključnih specifičnosti ovog protokola je upotreba jednostavnih implicitnih sertifikata i HOMQV protokola za autentifikovanu razmjenu ključeva, što ga čini posebno pogodnim za uređaje sa ograničenim resursima. U radu je sprovedena opsežna validacija performansi protokola kroz simulacije i empirijske eksperimente, koja demonstrira njegovu robusntost, efikasnost i skalabilnost.

Fokus budućih istraživanja biće na proširenju eksperimenata kako bi se obuhvatile veće mrežne konfiguracije. S tim u vezi, planirano je korišćenje heterogenog 'jata' (*swarm*) robota [79], koje uključuje aPad površinska vozila i mobilne podvodne senzorske čvorove, kako bi se sprovela sveobuhvatnija eksperimentalna analiza. Ova eksperimentalna postavka mogla bi biti upotpunjena uključivanjem AUV-ova ili simulacijom njihovog prisustva korišćenjem dostupnijih ROV-ova. Pored toga, interesantan pravac za buduća istraživanja je integracija naprednih tehnika vještačke inteligencije u cilju efikasnije detekcije malicioznih napada. Još jedna obećavajuća oblast za buduća istraživanja uključuje prilagođavanje predloženog rešenja za UASN scenarije koji zahtijevaju dugoročnu zaštitu tajnosti podataka. S obzirom na ograničenja HOMQV protokola u ispunjavanju ovih zahtjeva, korisno je procijeniti izvodljivost alternativnih sigurnosnih metoda koje mogu ponuditi neophodan nivo zaštite za uređaje sa ograničenim resursima pod vodom. Ključno je procijeniti kako ove sigurnosne mjere utiču na ukupnu efikasnost autentifikacije, posebno fokusirajući se na njihov uticaj na povećanje kašnjenja i potrošnje energije.

Bibliografija

[1] G. Han, J. Jiang, N. Sun, and L. Shu, "Secure communication for underwater acoustic sensor networks," *IEEE Communications Magazine*, vol. 53, no. 8, pp. 54–60, 2015.

[2] C. Lal, R. Petroccia, K. Pelekanakis, M. Conti, and J. Alves, "Toward the development of secure underwater acoustic networks," *IEEE Journal of Oceanic Engineering*, vol. 42, no. 4, pp. 1075–1087, 2017.

[3] D. R. K. Mary, E. Ko, S.-G. Kim, S.-H. Yum, S. Y. Shin, and S.-H. Park, "A systematic review on recent trends, challenges, privacy and security issues of underwater internet of things," *Sensors (Basel, Switzerland)*, vol. 21, 2021.

[4] C. Lal, R. Petroccia, M. Conti, and J. Alves, "Secure underwater acoustic networks: Current and future research directions," in *2016 IEEE Third Underwater Communications and Networking Conference (UComms)*, pp. 1–5, 2016.

[5] S. Benzarti, B. Triki, and O. Korbaa, "A survey on attacks in internet of things based networks," in *2017 International Conference on Engineering MIS (ICEMIS)*, pp. 1–7, 2017.

[6] M. C. Domingo, "Securing underwater wireless communication networks," *IEEE Wireless Communications*, vol. 18, no. 1, pp. 22–28, 2011.

[7] A. Signori, F. Campagnaro, I. Nissen, and M. Zorzi, "Channel-based trust model for security in underwater acoustic networks," *IEEE Internet of Things Journal*, vol. 9, no. 20, pp. 20479–20491, 2022.

[8] N. Goyal, M. Dave, and A. K. Verma, "Sapda: Secure authentication with protected data aggregation scheme for improving qos in scalable and survivable uwsns," *Wireless Personal Communications*, vol. 113, pp. 1–15, Jul 2020.

[9] A. Yazdinejad, R. M. Parizi, G. Srivastava, A. Dehghantanha, and K.-K. R. Choo, "Energy efficient decentralized authentication in internet of underwater things using blockchain," in *2019 IEEE Globecom Workshops (GC Wkshps)*, pp. 1–6, 2019.

[10] S. Abbas, H. Nasir, A. Almogren, A. Altameem, and N. Javaid, "Blockchain based privacy preserving authentication and malicious node detection in internet of underwater things (iout) networks," *IEEE Access*, vol. 10, pp. 113945–113955, 2022.

[11] S. Bhattacharya, N. Victor, R. Chengoden, M. Ramalingam, G. C. Selvi, P. K. R. Maddikunta, P. K. Donta, S. Dustdar, R. H. Jhaveri, and T. R. Gadekallu, "Blockchain for internet of underwater things: State-of-the-art, applications, challenges, and future directions," *Sustainability*, vol. 14, no. 23, 2022.

[12] L. D. Xu, Y. Lu, and L. Li, "Embedding blockchain technology into iot for security: A survey," *IEEE Internet of Things Journal*, vol. 8, no. 13, pp. 10452–10473, 2021.

[13] A. Yazdinejad, R. M. Parizi, A. Dehghantanha, H. Karimipour, G. Srivastava, and M. Aledhari, "Enabling drones in the internet of things with decentralized blockchain-based security," *IEEE Internet of Things Journal*, vol. 8, no. 8, pp. 6406–6415, 2021.

[14] T. Hewa, A. Bracken, M. Ylianttila, and M. Liyanage, "Blockchain-based automated certificate revocation for 5g iot," in *ICC 2020 - 2020 IEEE International Conference on Communications (ICC)*, pp. 1–7, 2020.

[15] X. Hao, W. Ren, Y. Fei, T. Zhu, and K.-K. R. Choo, "A blockchain-based cross-domain and autonomous access control scheme for internet of things," *IEEE Transactions on Services Computing*, vol. 16, no. 2, pp. 773–786, 2023.

[16] M. T. Hammi, P. Bellot, and A. Serhrouchni, "Bctrust: A decentralized authentication blockchain-based mechanism," in *2018 IEEE Wireless Communications and Networking Conference (WCNC)*, pp. 1–6, 2018.

[17] M. Shen, H. Liu, L. Zhu, K. Xu, H. Yu, X. Du, and M. Guizani, "Blockchain-assisted secure device authentication for cross-domain industrial iot," *IEEE Journal on Selected Areas in Communications*, vol. 38, no. 5, pp. 942–954, 2020.

[18] J. Cui, N. Liu, Q. Zhang, D. He, C. Gu, and H. Zhong, "Efficient and anonymous cross-domain authentication for iiot based on blockchain," *IEEE Transactions on Network Science and Engineering*, vol. 10, no. 2, pp. 899–910, 2023.

[19] J. Wang, L. Wu, K.-K. R. Choo, and D. He, "Blockchain-based anonymous authentication with key management for smart grid edge computing infrastructure," *IEEE Transactions on Industrial Informatics*, vol. 16, no. 3, pp. 1984–1992, 2020.

[20] Z. Lu, Q. Wang, G. Qu, H. Zhang, and Z. Liu, "A blockchain-based privacy-preserving authentication scheme for vanets," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 27, no. 12, pp. 2792–2801, 2019.

[21] Y. Yao, X. Chang, J. Mi?i?, V. B. Mi?i?, and L. Li, "Bla: Blockchain-assisted lightweight anonymous authentication for distributed vehicular fog services," *IEEE Internet of Things Journal*, vol. 6, no. 2, pp. 3775–3784, 2019.

[22] J. Yang, J. Liu, H. Song, J. Liu, and X. Lei, "Blockchain-based conditional privacy-preserving authentication protocol with implicit certificates for vehicular edge computing," in *2022 7th International Conference on Cloud Computing and Big Data Analytics (ICCCBDA)*, pp. 210–216, 2022.

[23] K. Kaur, S. Garg, G. Kaddoum, F. Gagnon, and S. H. Ahmed, "Blockchain-based lightweight authentication mechanism for vehicular fog infrastructure," in *2019 IEEE International Conference on Communications Workshops (ICC Workshops)*, pp. 1–6, 2019.

[24] S. Tomović, B. Krivokapić, ula Nađ, and I. Radusinović, "Bekmp: A blockchain-enabled key management protocol for underwater acoustic sensor networks," *IEEE Access*, vol. 12, pp. 74108–74125, 2024.

[25] S. Halevi and H. Krawczyk, "One-pass hmqv and asymmetric key-wrapping," in *Proceedings of the 14th International Conference on Practice and Theory in Public Key Cryptography Conference on Public Key Cryptography*, PKC'11, (Berlin, Heidelberg), p. 317?334, Springer-Verlag, 2011.

[26] L. Vigano, "Automated security protocol analysis with the avispa tool," *Electronic Notes in Theoretical Computer Science*, vol. 155, pp. 61–86, 2006. Proceedings of the 21st Annual Conference on Mathematical Foundations of Programming Semantics (MFPS XXI).

[27] G. Ateniese, A. Capossele, P. Gjanci, C. Petrioli, and D. Spaccini, "Secfun: Security framework for underwater acoustic sensor networks," in *OCEANS 2015 - Genova*, pp. 1–9, 2015.

[28] A. Capossele, C. Petrioli, G. Saturni, D. Spaccini, and D. Venturi, "Securing underwater communications: Key agreement based on fully hashed mqv," in *Proceedings of the 12th International Conference on Underwater Networks & Systems*, WUWNet '17, (New York, NY, USA), Association for Computing Machinery, 2017.

[29] M. Y. Malik, "Efficient implementation of elliptic curve cryptography using low-power digital signal processor," in *2010 The 12th International Conference on Advanced Communication Technology (ICACT)*, vol. 2, pp. 1464–1468, 2010.

[30] E. Souza, H. C. Wong, I. Cunha, I. Cunha, L. F. M. Vieira, and L. B. Oliveira, "End-to-end authentication in under-water sensor networks," in *2013 IEEE Symposium on Computers and Communications (ISCC)*, pp. 000299–000304, 2013.

[31] S. Heron, "Encryption: Advanced encryption standard (aes)," *Netw. Secur.*, vol. 2009, p. 8?12, dec 2009.

[32] D. Galindo, R. Roman, and J. Lopez, "A killer application for pairings: Authenticated key establishment in underwater wireless sensor networks," in *Proceedings of the 7th International Conference on Cryptology and Network Security*, CANS '08, (Berlin, Heidelberg), p. 120?132, Springer-Verlag, 2008.

[33] M. Kasahara, "Cryptosystems based on pairing," in *The 2000 Symposium on Cryptography and Information Security*, pp. SCIS2000–C20, 2000.

[34] D. Anand, V. Khemchandani, and R. K. Sharma, "Identity-based cryptography techniques and applications (a review)," in *2013 5th International Conference and Computational Intelligence and Communication Networks*, pp. 343–348, 2013.

[35] R. Barbulescu, P. Gaudry, A. Joux, and E. Thome, "A quasi-polynomial algorithm for discrete logarithm in finite fields of small characteristic." Cryptology ePrint Archive, Paper 2013/400, 2013.

[36] C. Wan, V. V. Phoha, Y. Tang, and A. Hu, "Non-interactive identity-based underwater data transmission with anonymity and zero knowledge," *IEEE Trans-*

actions on Vehicular Technology, vol. 67, no. 2, pp. 1726–1739, 2018.

[37] C. Petrioli, G. Saturni, and D. Spaccini, "Feasibility study for authenticated key exchange protocols on underwater acoustic sensor networks," in *Proceedings of the 14th International Conference on Underwater Networks & Systems*, WUWNet '19, (New York, NY, USA), Association for Computing Machinery, 2020.

[38] B. Krivokapic, S. Tomovic, and I. Radusinović, "Authenticated key exchange in underwater acoustic sensor networks based on implicit certificates: Performance analysis," *2023 27th International Conference on Information Technology (IT)*, pp. 1–4, 2023.

[39] Standards for Efficient Cryptography Group (SECG), "SEC 4: Elliptic Curve Qu-Vanstone Implicit Certificate Scheme (ECQV)." `https://www.secg.org/`, 2013. Accessed: 14.12.2023.

[40] A. P. Sarr, P. Elbaz-Vincent, and J.-C. Bajard, "A secure and efficient authenticated diffie-hellman protocol." Cryptology ePrint Archive, Paper 2009/408, 2009. https://eprint.iacr.org/2009/408.

[41] W. Diffie and M. E. Hellman, "New directions in cryptography," *IEEE Transactions on Information Theory*, vol. 22, no. 6, pp. 644–654, 1976.

[42] S. Zhang, X. Du, and X. Liu, "A secure remote mutual authentication scheme based on chaotic map for underwater acoustic networks," *IEEE Access*, vol. 8, pp. 48285–48298, 2020.

[43] C. Yuan, W. Chen, Y. Zhu, D. Li, and J. Tan, "A low computational complexity authentication scheme in underwater wireless sensor network," in *2015 11th International Conference on Mobile Ad-hoc and Sensor Networks (MSN)*, pp. 116–123, 2015.

[44] G. H. Golub and C. F. Van Loan, *Matrix Computations*. Johns Hopkins University Press, 4th ed., 2013.

[45] S. Goldwasser, S. Micali, and C. Rackoff, "The knowledge complexity of interactive proof systems," *SIAM Journal on Computing*, vol. 18, no. 1, pp. 186–208, 1989.

[46] Q. Wang, "A novel physical layer assisted authentication scheme for mobile wireless sensor networks," *Sensors*, vol. 17, no. 2, 2017.

[47] J. K. Tugnait, "Wireless user authentication via comparison of power spectral densities," *IEEE Journal on Selected Areas in Communications*, vol. 31, no. 9, pp. 1791–1802, 2013.

[48] L. Xiao, L. Greenstein, N. Mandayam, and W. Trappe, "A physical-layer technique to enhance authentication for mobile terminals," in *2008 IEEE International Conference on Communications*, pp. 1520–1524, 2008.

[49] L. Xiao, A. Reznik, W. Trappe, C. Ye, Y. Shah, L. Greenstein, and N. Mandayam, "Phy-authentication protocol for spoofing detection in wireless networks," in *2010 IEEE Global Telecommunications Conference GLOBECOM 2010*, pp. 1–6, 2010.

[50] L. Xiao, L. Greenstein, N. Mandayam, and W. Trappe, "Fingerprints in the ether: Using the physical layer for wireless authentication," in *2007 IEEE International Conference on Communications*, pp. 4646–4651, 2007.

[51] L. Xiao, L. J. Greenstein, N. B. Mandayam, and W. Trappe, "Channel-based spoofing detection in frequency-selective rayleigh channels," *IEEE Transactions on Wireless Communications*, vol. 8, no. 12, pp. 5948–5956, 2009.

[52] F. He, H. Man, D. Kivanc, and B. McNair, "Epson: Enhanced physical security in ofdm networks," in *2009 IEEE International Conference on Communications*, pp. 1–5, 2009.

[53] D. B. Faria and D. R. Cheriton, "Detecting identity-based attacks in wireless networks using signalprints," in *Proceedings of the 5th ACM Workshop on Wireless Security*, WiSe '06, (New York, NY, USA), p. 43?52, Association for Computing Machinery, 2006.

[54] Y. Chen, W. Trappe, and R. P. Martin, "Detecting and localizing wireless spoofing attacks," in *2007 4th Annual IEEE Communications Society Conference on Sensor, Mesh and Ad Hoc Communications and Networks*, pp. 193–202, 2007.

[55] K. Pelekanakis, C. M. G. Gussen, R. Petroccia, and J. Alves, "Robust channel parameters for crypto key generation in underwater acoustic systems," in *OCEANS 2019 MTS/IEEE SEATTLE*, pp. 1–7, 2019.

[56] R. Zhao, M. Khalid, O. A. Dobre, and X. Wang, "Physical layer node authentication in underwater acoustic sensor networks using time-reversal," *IEEE Sensors Journal*, vol. 22, no. 4, pp. 3796–3809, 2022.

[57] R. Diamant, P. Casari, and S. Tomasin, "Cooperative authentication in underwater acoustic sensor networks," *IEEE Transactions on Wireless Communications*, vol. 18, no. 2, pp. 954–968, 2019.

[58] F. Ardizzon, R. Diamant, P. Casari, and S. Tomasin, "Machine learning-based distributed authentication of uwan nodes with limited shared information," in *2022 Sixth Underwater Communications and Networking Conference (UComms)*, pp. 1–5, 2022.

[59] P. Casari, F. Ardizzon, and S. Tomasin, "Physical layer authentication in underwater acoustic networks with mobile devices," in *Proceedings of the 16th International Conference on Underwater Networks & Systems*, WUWNet '22, (New York, NY, USA), Association for Computing Machinery, 2022.

[60] R. Diamant, S. Tomasin, F. Ardizzon, D. Eccher, and P. Casari, "Secret key generation from route propagation delays for underwater acoustic networks," *IEEE Transactions on Information Forensics and Security*, vol. 18, pp. 3318–3333, 2023.

[61] S. Verma and Prachi, "A cluster based key management scheme for underwater wireless sensor networks," *International Journal of Computer Network and Information Security (IJCNIS)*, vol. 7, no. 9, pp. 54–63, 2015.

[62] C.-W. Yun, J.-H. Lee, O. Yi, and S.-H. Park, "Ticket-based authentication protocol for underwater wireless sensor network," in *2016 Eighth International Conference on Ubiquitous and Future Networks (ICUFN)*, pp. 215–217, 2016.

[63] W. Wang, J. Kong, B. Bhargava, and M. Gerla, "Visualisation of wormholes in underwater sensor networks: A distributed approach," *Int. J. Secur. Netw.*, vol. 3, p. 10?23, dec 2008.

[64] R. Zhang and Y. Zhang, "Wormhole-resilient secure neighbor discovery in underwater acoustic networks," in *2010 Proceedings IEEE INFOCOM*, pp. 1–9, 2010.

[65] A. Capossele, G. De Cicco, and C. Petrioli, "R-carp: A reputation based channel aware routing protocol for underwater acoustic sensor networks," in *Proceedings of the 10th International Conference on Underwater Networks & Systems*, WUWNet '15, (New York, NY, USA), Association for Computing Machinery, 2015.

[66] S. S. N. Krishnan, "Defending selective forwarding attacks in underwater acoustic networks applying trust model," in *2018 Second International Conference on Electronics, Communication and Aerospace Technology (ICECA)*, pp. 1567–1571, 2018.

[67] G. Han, Y. He, J. Jiang, N. Wang, M. Guizani, and J. A. Ansere, "A synergetic trust model based on svm in underwater acoustic sensor networks," *IEEE Transactions on Vehicular Technology*, vol. 68, no. 11, pp. 11239–11247, 2019.

[68] Y. He, G. Han, J. Jiang, H. Wang, and M. Martinez-Garcia, "A trust update mechanism based on reinforcement learning in underwater acoustic sensor networks," *IEEE Transactions on Mobile Computing*, vol. 21, no. 3, pp. 811–821, 2022.

[69] G. Han, Y. He, J. Jiang, H. Wang, Y. Peng, and K. Fan, "Fault-tolerant trust model for hybrid attack mode in underwater acoustic sensor networks," *IEEE Network*, vol. 34, no. 5, pp. 330–336, 2020.

[70] D. Praveen Kumar, T. Amgoth, and C. S. R. Annavarapu, "Machine learning algorithms for wireless sensor networks: A survey," *Information Fusion*, vol. 49, pp. 1–25, 2019.

[71] T. M. Hewa, A. Kalla, A. Nag, M. E. Ylianttila, and M. Liyanage, "Blockchain for 5g and iot: Opportunities and challenges," in *2020 IEEE Eighth International Conference on Communications and Networking (ComNet)*, pp. 1–8, 2020.

[72] H. Hu, Y. Chen, W.-S. Ku, Z. Su, and C.-H. J. Chen, "Weighted trust evaluation-based malicious node detection for wireless sensor networks," *Int. J. Inf. Comput. Secur.*, vol. 3, p. 132?149, oct 2009.

[73] Y. Yang and S. Lam, "General aimd congestion control," in *Proceedings 2000 International Conference on Network Protocols*, pp. 187–198, 2000.

[74] D. Hankerson, S. A. Vanstone, and A. Menezes, "Guide to elliptic curve cryptography," in *Springer Professional Computing*, 2004.

[75] Standards for Efficient Cryptography Group (SECG), "Recommended elliptic curve domain parameters," 2000.

[76] J. Weng, J. Weng, J.-N. Liu, and Y. Zhang, "Secure software-defined networking based on blockchain," *ArXiv*, vol. abs/1906.04342, 2019.

[77] S. Sciancalepore, A. Capossele, G. Piro, G. Boggia, and G. Bianchi, "Key management protocol with implicit certificates for iot systems," in *Proceedings of the 2015 Workshop on IoT Challenges in Mobile and Industrial Systems*, IoT-Sys '15, (New York, NY, USA), p. 37?42, Association for Computing Machinery, 2015.

[78] D. Dolev and A. C. Yao, "On the security of public key protocols," *IEEE Transactions on Information Theory*, vol. 29, no. 2, pp. 198–208, 1983.

[79] I. Loncar, A. Babic, B. Arbanas, G. Vasiljevic, T. Petrovic, S. Bogdan, and N. Miskovic, "A heterogeneous robotic swarm for long-term monitoring of marine environments," *Applied Sciences*, vol. 9, no. 7, 2019.

[80] M. Varchola, T. Guneysu, and O. Mischke, "Microecc: A lightweight reconfigurable elliptic curve crypto-processor," in *2011 International Conference on Reconfigurable Computing and FPGAs*, pp. 204–210, 2011.

[81] E. Brier and M. Joye, "Weierstraß elliptic curves and side-channel attacks," in *Public Key Cryptography* (D. Naccache and P. Paillier, eds.), (Berlin, Heidelberg), pp. 335–345, Springer Berlin Heidelberg, 2002.

[82] Certicom Research, "Standards for Efficient Cryptography 1." Online, 2009. [Accessed: 17.12.2023.].

[83] B. Sherlock, N. Morozs, J. Neasham, and P. Mitchell, "Ultra-low-cost and ultra-low-power, miniature acoustic modems using multipath tolerant spread-spectrum techniques," *Electronics*, vol. 11, no. 9, 2022.

[84] F. Campagnaro, R. Francescon, F. Guerra, F. Favaro, P. Casari, R. Diamant, and M. Zorzi, "The desert underwater framework v2: Improved capabilities and extension tools," in *2016 IEEE Third Underwater Communications and Networking Conference (UComms)*, pp. 1–5, 2016.

[85] F. Ferreira, Z. Vukic, N. Miskovic, and I. Kvasic, "Breaking the surface - lessons learned from over a decade of interdisciplinary workshops," in *OCEANS 2021: San Diego*, pp. 1–4, 2021.

[86] R. Masiero, S. Azad, F. Favaro, M. Petrani, G. Toso, F. Guerra, P. Casari, and M. Zorzi, "Desert underwater: An ns-miracle-based framework to design, simulate, emulate and realize test-beds for underwater network protocols," in *2012 Oceans - Yeosu*, pp. 1–10, 2012.

[87] R. Labs, "Redis." `https://redis.io/`, 2023. Accessed: 2023-04-30.

Prilog

Listing 5.1: AVISPA HLPSL kod za razmjenu ključa između SN čvora i CH čvora primjenom HOMQV protokola.

```
%%CH slucajno bira y â R Zq, racuna Y = gˆy i salje na SN. CH takoÄe
    izvodi kljuc H(Ï , IDSN, IDCH, Y ) gdje je Ï  = Aˆ(y+eb) i e = H(Y,
     IDSN). Nakon prijema Y i IDCH, SN provjerava da li su Y i javni
    kljuc CH u G0 (ako nisu prekida se dalje izvrsavanje) i zatim
    racuna kljuc H(Ï , IDCH, IDSN, Y ) gdje je Ï  = (YBˆe)ˆa

role role_SN(SN:agent, CH:agent, G:text, Ka, Kb: public_key, H, F, Add,
     Poly:hash_func, SND, RCV:channel(dy))
played_by SN
def=
  local
    State:nat, A:text, SharedKey:text, B,Y:text, Sigma: text
  init
    State := 0
  transition
    1. State=0 /\ RCV(start) =|> State':=1 /\ A':=new() /\
                    SND({exp(G,A')}_Kb)

    2. State=1 /\ RCV({exp(G,Y').exp(G, B')}_Ka) =|> State':=2 /\
                    SharedKey':=H(exp(Poly(Y'.exp(B, H(Y'))),A)) /\ SND
                        ({Y'}_SharedKey') /\ secret(SharedKey',sec_1,{
                        SN,CH}) /\ witness(SN,CH, auth_1,Y')
end role

role role_CH(SN:agent, CH:agent, G:text, Ka, Kb: public_key, H, F, Add,
     Poly:hash_func, SND, RCV:channel(dy))
played_by CH
def=
  local
    State:nat, B,Y:text, SharedKey:text, A:text, Sigma: text
  init
```

```
    State := 0
  transition
    1. State=0 /\ RCV({exp(G,A')}_Kb) =|> State':=2 /\ Y':=new() /\
                    B':=new() /\ SharedKey':= H(exp(exp(G,A'),Add(Y'.
                        Poly(H(exp(G,Y').B'))))) /\
                    SND({exp(G,Y').exp(G, B')}_Ka) /\ secret(SharedKey'
                        ,sec_1,{SN,CH})
    2. State = 2 /\ RCV({Y}_SharedKey) =|> State':= 5 /\ request(CH,SN,
        auth_1,Y)

end role

role session_DH(SN:agent, CH:agent, G:text, Ka,Kb:public_key, H, F, Add
    , Poly:hash_func)
def=
  local
    SND_SN, RCV_SN, SND_CH, RCV_CH:channel(dy)
  composition
    role_SN(SN, CH, G, Ka, Kb, H, F, Add, Poly, SND_SN, RCV_SN) /\
    role_CH(SN, CH, G, Ka, Kb, H, F, Add, Poly, SND_CH, RCV_CH)
end role

role environment()
def=
  const
    sn:agent, ch:agent, g:text, sec_1, auth_1:protocol_id, ka,kb:
        public_key, h, f, add, poly: hash_func

  intruder_knowledge = {sn,ch,g,ka,kb,h,f,add,poly}

  composition
    session_DH(sn, ch, g, ka, kb, h, f, add, poly) /\ session_DH(sn, ch
        , g, ka, kb, h, f, add, poly)
end role

goal
  secrecy_of sec_1
  authentication_on auth_1
end goal

environment()
```

Listing 5.2: Funkcija za preuzimanje ključa u pametnom ugovoru za upravljanje ključevima.

```
async getKey(ctx) {
    const args = ctx.stub.getArgs();
    let clusterhead = args[1];
    let deviceid = args[2];
    let nonce = args[3];
    let indexName = 'deviceid~clusterhead~nonce~key';
    let keyIndex = await ctx.stub.createCompositeKey(indexName, [
       deviceid, clusterhead, nonce, 'key']);
    let assetAsBuffer = await ctx.stub.getState(keyIndex);

    if (!assetAsBuffer || assetAsBuffer.length === 0) {
        return {
            found: false
        };
    }
    let res = JSON.parse(assetAsBuffer);
    return {
        found: true,
        data: res
    };
}
```

Listing 5.3: Funkcija za skladištenje ključa u pametnom ugovoru za upravljanje ključevima.

```
async storeKey(ctx, deviceid, clusterhead, key, nonce, ts, texp) {
    let model = {
        deviceid: deviceid,
        clusterhead: clusterhead,
        key: key,
        ts: ts,
        texp: texp,
        confirmed: false,
        txId: this.TxId
    };

    try {
        let indexName = 'deviceid~clusterhead~nonce~key';
        let keyIndex = await ctx.stub.createCompositeKey(indexName, [
           deviceid, clusterhead, nonce, 'key']);
        await ctx.stub.putState(keyIndex, Buffer.from(JSON.stringify(
```

```
            model)));
        return {
            key: deviceid + '~' + clusterhead + '~' + nonce + '~key'
        };
    } catch (e) {
        throw new Error(`The tx ${this.TxId} can not be stored: ${e}`);
    }
}
```

Listing 5.4: Funkcija za ažuriranje ključa u pametnom ugovoru za upravljanje ključevima.

```
async updateKey(ctx, deviceid, clusterhead, key, nonce, ts, texp,
    confirmed) {
    let indexName = 'deviceid~clusterhead~nonce~key';
    let keyIndex = await ctx.stub.createCompositeKey(indexName, [
        deviceid, clusterhead, nonce, 'key']);
    let readKey = await ctx.stub.getState(keyIndex);
    const parsedKeyFromBC = JSON.parse(readKey);

    let model = {
        deviceid: parsedKeyFromBC.deviceid,
        clusterhead: parsedKeyFromBC.clusterhead,
        key: parsedKeyFromBC.key,
        ts: parsedKeyFromBC.ts,
        texp: parsedKeyFromBC.texp,
        confirmed: true,
        txId: this.TxId
    };

    try {
        await ctx.stub.putState(keyIndex, Buffer.from(JSON.stringify(
            model)));
        return {
            key: deviceid + '~' + clusterhead + '~' + nonce + '~key'
        };
    } catch (e) {
        throw new Error(`The tx ${this.TxId} can not be stored: ${e}`);
    }
}
```

Listing 5.5: Funkcija za skladištenje sertifikata u pametnom ugovoru za upravljanje sertifikatima.

```
async storeCertificate(ctx, deviceid, certificate) {
    let model = {
        deviceid: deviceid,
        certificate: certificate,
        txId: this.TxId
    }

    try {
        let indexName = 'deviceid~crt';
        let crtIndex = await ctx.stub.createCompositeKey(indexName, [
           deviceid, 'crt']);
        await ctx.stub.putState(crtIndex, Buffer.from(JSON.stringify(
           model)));
        return {
            key: deviceid + '~crt'
        };
    } catch (e) {
        throw new Error(`The tx ${this.TxId} can not be stored: ${e}`);
    }
}
```

Listing 5.6: Funkcija za preuzimanje sertifikata u pametnom ugovoru za upravljanje sertifikatima.

```
async getCertificate(ctx) {
    const args = ctx.stub.getArgs();
    let deviceid = args[1];
    let indexName = 'deviceid~crt';
    let crtIndex = await ctx.stub.createCompositeKey(indexName, [
       deviceid, 'crt']);
    let assetAsBuffer = await ctx.stub.getState(crtIndex);

    if (!assetAsBuffer || assetAsBuffer.length === 0) {
        return {
            found: false
        }
    }
    let res = JSON.parse(assetAsBuffer);

    return {
        found: true,
        data: res
    }
}
```

Listing 5.7: Funkcija za opoziv sertifikata u pametnom ugovoru za upravljanje povjerenjem.

```
async revokeNodeIfBelowThreshold(ctx, nodeId, w1, w2, w3,
    compositeThreshold) {
    w1 = Number(w1);
    w2 = Number(w2);
    w3 = Number(w3);
    compositeThreshold = Number(compositeThreshold);

    if (isNaN(w1) || isNaN(w2) || isNaN(w3) || isNaN(compositeThreshold
        )) {
        throw new Error('Invalid input: All trust values, weights, and
            the threshold must be numbers');
    }

    if (Math.abs(w1 + w2 + w3 - 1) > 1e-9) {
        throw new Error('Weights must sum to 1');
    }

    let assetAsBuffer = await ctx.stub.getState(`commTrust_${nodeId}`);

    if (!assetAsBuffer || assetAsBuffer.length === 0) {
        assetAsBuffer = '{"success": 0, "fail": 0}';
    }
    const existingCommHistory = JSON.parse(assetAsBuffer);
    let Tc = Number(existingCommHistory.Tc || 0)

    let packetBuffer = await ctx.stub.getState(`packetTrust_${nodeId}`)
        ;

    if (!packetBuffer || packetBuffer.length === 0) {
        packetBuffer = '[]';
    }
    const existingPacketTrust = JSON.parse(packetBuffer);
    let Td = existingPacketTrust.length > 0 ? existingPacketTrust[
        existingPacketTrust.length - 1] : 1;

    let energyBuffer = await ctx.stub.getState(`energyTrust_${nodeId}`)
        ;

    if (!energyBuffer || energyBuffer.length === 0) {
```

```
        energyBuffer = '[]';
    }
    const existingEnergyTrust = JSON.parse(energyBuffer);
    let Te = existingEnergyTrust.length > 0 ? existingEnergyTrust : 1;

    if (Tc < 0 || Tc > 1 || Td < 0 || Td > 1 || Te < 0 || Te > 1) {
        throw new Error('Invalid trust values: Trust values must be
           between 0 and 1');
    }

    const compositeTrust = w1 * Tc + w2 * Td + w3 * Te;

    if (compositeTrust < compositeThreshold) {
        let revokedListJSON = await ctx.stub.getState('revokedList');
        let revokedList = [];
        if (revokedListJSON && revokedListJSON.length > 0) {
            revokedList = JSON.parse(revokedListJSON.toString());
        }
        revokedList.push(nodeId);
        await ctx.stub.putState('revokedList', Buffer.from(JSON.
           stringify(revokedList)));
        return {
            'res': `Node ${nodeId} has been revoked.`,
            'Tc': Tc,
            'Td': Td,
            'Te': Te
        };
    }
    return {
        'res': `Node ${nodeId} is still valid.`,
        'Tc': Tc,
        'Td': Td,
        'Te': Te
    };
}
```

Listing 5.8: Kreiranje zahtjeva za izdavanje sertifikata (CSR) i Generisanje sertifikata (CRT) koristeci ECC.

```
int csr_request(struct csr_t *csr,
                uint8_t *private,
                const uint8_t id[ID_SIZE],
                const uECC_Curve curve) {
```

```
    uint8_t public[PUBLIC_SIZE] = {0x5B, ...};
    uint8_t compressed[PT_COMPRESSED_SIZE] = {0};

    if (!uECC_make_key(public, private, curve)) {
        return 0;
    }

    uECC_compress(public, compressed, curve);

    memcpy(csr->id, id, ID_SIZE);
    memcpy(csr->point, compressed, PT_COMPRESSED_SIZE);

    return 1;
}

int crt_generation(uint8_t *r,
                   uint8_t *crt,
                   const struct csr_t *csr,
                   const uint8_t *private_ca,
                   const uint8_t *public_ca,
                   const uECC_Curve curve) {

    wordcount_t num_words = curve->num_words;
    wordcount_t num_bytes = curve->num_bytes;

    uint8_t public[PUBLIC_SIZE] = {0};
    uint8_t k[PRIVATE_SIZE]     = {0};
    uint8_t K[PUBLIC_SIZE]      = {0};
    uint8_t PU[PUBLIC_SIZE]     = {0};
    uint8_t compressed[PT_COMPRESSED_SIZE] = {0};
    uint8_t hash[32];

    uECC_word_t _e[uECC_MAX_WORDS];
    uECC_word_t _ek[uECC_MAX_WORDS];
    uECC_word_t _Pu[uECC_MAX_WORDS * 2];
    uECC_word_t _ePu[uECC_MAX_WORDS * 2];
    uECC_word_t _ePuQca[uECC_MAX_WORDS * 2];

    #if uECC_VLI_NATIVE_LITTLE_ENDIAN
    uECC_word_t *_public     = (uECC_word_t *)public;
    uECC_word_t *_public_ca  = (uECC_word_t *)public_ca;
    uECC_word_t *_private_ca = (uECC_word_t *)private_ca;
    uECC_word_t *_K          = (uECC_word_t *)K;
```

```
    uECC_word_t *_k          = (uECC_word_t *)k;
    uECC_word_t *_r          = (uECC_word_t *)r;
    #else
    uECC_word_t _public[uECC_MAX_WORDS * 2];
    uECC_word_t _public_ca[uECC_MAX_WORDS * 2];
    uECC_word_t _private_ca[uECC_MAX_WORDS];
    uECC_word_t _K[uECC_MAX_WORDS * 2];
    uECC_word_t _k[uECC_MAX_WORDS];
    uECC_word_t _r[uECC_MAX_WORDS];
    #endif

    uECC_decompress(csr->point, public, curve);

    if (!uECC_valid_public_key(public, curve)) {
        return 0;
    }

    #if uECC_VLI_NATIVE_LITTLE_ENDIAN
    bcopy((uint8_t *) _public,     public,     num_bytes*2);
    bcopy((uint8_t *) _public_ca,  public_ca,  num_bytes*2);
    bcopy((uint8_t *) _private_ca, private_ca, num_bytes);
    #else
    uECC_vli_bytesToNative(_public, public, num_bytes);
    uECC_vli_bytesToNative(_public + num_words, public + num_bytes,
       num_bytes);
    uECC_vli_bytesToNative(_public_ca, public_ca, num_bytes);
    uECC_vli_bytesToNative(_public_ca + num_words, public_ca +
       num_bytes, num_bytes);
    uECC_vli_bytesToNative(_private_ca, private_ca, BITS_TO_BYTES(curve
       ->num_n_bits));
    #endif

    do {
        if (!uECC_make_key(K, k, curve)) {
            return 0;
        }

        #if uECC_VLI_NATIVE_LITTLE_ENDIAN
        bcopy((uint8_t *) _K,  K, num_bytes*2);
        bcopy((uint8_t *) _k,  k, num_bytes);
        #else
        uECC_vli_bytesToNative(_K, K, num_bytes);
        uECC_vli_bytesToNative(_K + num_words, K + num_bytes, num_bytes
```

```
            );
        uECC_vli_bytesToNative(_k, k, BITS_TO_BYTES(curve->num_n_bits))
            ;
        #endif

        _addPoint(_Pu, _public, _K, curve);

        #if uECC_VLI_NATIVE_LITTLE_ENDIAN
        bcopy((uint8_t *) PU, (uint8_t *) _Pu, num_bytes);
        #else
        uECC_vli_nativeToBytes(PU, num_bytes, _Pu);
        uECC_vli_nativeToBytes(PU + num_bytes, num_bytes, _Pu +
            num_words);
        #endif

        uECC_compress(PU, compressed, curve);

        memcpy(crt, compressed, PT_COMPRESSED_SIZE);
        memcpy(crt+PT_COMPRESSED_SIZE, csr->id, ID_SIZE);

        custom_hash(hash, crt, CRT_SIZE);
        bits2int(_e, hash, sizeof(hash), curve);

        _mulPoint(_ePu, _e, _Pu, curve);

        _addPoint(_ePuQca, _ePu, _public_ca, curve);
    }
    while (EccPoint_isZero(_ePuQca, curve));

    uECC_vli_modMult(_ek, _e, _k, curve->n, num_words);
    uECC_vli_modAdd(_r, _ek, _private_ca, curve->n, num_words);

    #if uECC_VLI_NATIVE_LITTLE_ENDIAN
    bcopy((uint8_t *) r, (uint8_t *) _r, num_bytes);
    #else
    uECC_vli_nativeToBytes(r, BITS_TO_BYTES(curve->num_n_bits), _r);
    #endif

    return 1;
}
```

Listing 5.9: Generisanje ključa koristeći HOMQV protokol.

```
int homqv_generate_key(uint8_t *key,
```

```
                    uint8_t *Y,
                    uint8_t *extracted,
                    const uint8_t *crt,
                    const uint8_t *private_ca,
                    const uint8_t *public_ca,
                    const uint8_t *id_cluster,
                    int x509,
                    const uECC_Curve curve) {

if(!x509) {
    crt_pk_extract(extracted, crt, public_ca, curve);
}

wordcount_t num_words = curve->num_words;
wordcount_t num_bytes = curve->num_bytes;

uint8_t y[PRIVATE_SIZE]                                 = {0};
uint8_t id[ID_SIZE]                                     = {0};
uint8_t Yid[PUBLIC_SIZE + ID_SIZE]                      = {0};
uint8_t sigma[PUBLIC_SIZE]                              = {0};
uint8_t sigmaId1Id2Y[PUBLIC_SIZE + ID_SIZE + ID_SIZE + PUBLIC_SIZE]
    = {0};

uint8_t hash[32];

uECC_word_t _e[uECC_MAX_WORDS]          = { 0 };
uECC_word_t _eb[uECC_MAX_WORDS]         = { 0 };
uECC_word_t _yeb[uECC_MAX_WORDS]         = { 0 };
uECC_word_t _sigma[uECC_MAX_WORDS * 2]    = { 0 };

#if uECC_VLI_NATIVE_LITTLE_ENDIAN
uECC_word_t *_extracted  = (uECC_word_t *)extracted;
uECC_word_t *_public_ca  = (uECC_word_t *)public_ca;
uECC_word_t *_private_ca = (uECC_word_t *)private_ca;
uECC_word_t *_Y          = (uECC_word_t *)Y;
uECC_word_t *_y          = (uECC_word_t *)y;
#else
uECC_word_t _extracted[uECC_MAX_WORDS * 2];
uECC_word_t _public_ca[uECC_MAX_WORDS * 2];
uECC_word_t _private_ca[uECC_MAX_WORDS];
uECC_word_t _Y[uECC_MAX_WORDS * 2];
uECC_word_t _y[uECC_MAX_WORDS];
#endif
```

```
#if uECC_VLI_NATIVE_LITTLE_ENDIAN
bcopy((uint8_t *) _extracted,     extracted,     num_bytes*2);
bcopy((uint8_t *) _public_ca,  public_ca,  num_bytes*2);
bcopy((uint8_t *) _private_ca, private_ca, num_bytes);
#else
uECC_vli_bytesToNative(_extracted, extracted, num_bytes);
uECC_vli_bytesToNative(_extracted + num_words, extracted +
    num_bytes, num_bytes);
uECC_vli_bytesToNative(_public_ca, public_ca, num_bytes);
uECC_vli_bytesToNative(_public_ca + num_words, public_ca +
    num_bytes, num_bytes);
uECC_vli_bytesToNative(_private_ca, private_ca, BITS_TO_BYTES(curve
    ->num_n_bits));
#endif

if (!uECC_make_key(Y, y, curve)) {
   return 0;
}

#if uECC_VLI_NATIVE_LITTLE_ENDIAN
bcopy((uint8_t *) _Y,  Y, num_bytes*2);
bcopy((uint8_t *) _y,  y, num_bytes);
#else
uECC_vli_bytesToNative(_Y, Y, num_bytes);
uECC_vli_bytesToNative(_Y + num_words, Y + num_bytes, num_bytes);
uECC_vli_bytesToNative(_y, y, BITS_TO_BYTES(curve->num_n_bits));
#endif

memcpy(id, crt+PT_COMPRESSED_SIZE, ID_SIZE);
memcpy(Yid, Y, PUBLIC_SIZE);
memcpy(Yid+PUBLIC_SIZE, id, ID_SIZE);

custom_hash(hash, Yid, PUBLIC_SIZE + ID_SIZE);
bits2int(_e, hash, sizeof(hash), curve);

uECC_vli_modMult(_eb, _e, _private_ca, curve->n, num_words);
uECC_vli_modAdd(_yeb, _eb, _y, curve->n, num_words);

_mulPoint(_sigma, _yeb, _extracted, curve);

#if uECC_VLI_NATIVE_LITTLE_ENDIAN
bcopy((uint8_t *) sigma, (uint8_t *) _sigma, num_bytes*2);
```

```
    #else
    uECC_vli_nativeToBytes(sigma, num_bytes, _sigma);
    uECC_vli_nativeToBytes(sigma + num_bytes, num_bytes, _sigma +
       num_words);
    #endif

    memcpy(sigmaId1Id2Y, sigma, PUBLIC_SIZE);
    memcpy(sigmaId1Id2Y+PUBLIC_SIZE, id, ID_SIZE);
    memcpy(sigmaId1Id2Y+PUBLIC_SIZE+ID_SIZE, id_cluster, ID_SIZE);
    memcpy(sigmaId1Id2Y+PUBLIC_SIZE+ID_SIZE+ID_SIZE, Y, PUBLIC_SIZE);

    custom_hash(key, sigmaId1Id2Y, PUBLIC_SIZE + ID_SIZE + ID_SIZE);

    return 1;
}
```

Listing 5.10: HOMQV Rekreiranje ključa.

```
int homqv_regenerate_key(uint8_t *key,
                         const uint8_t *Y,
                         const uint8_t *public_ca,
                         const uint8_t *private,
                         const uint8_t *id_cluster,
                         const uint8_t *id,
                         const uECC_Curve curve) {

    wordcount_t num_words = curve->num_words;
    wordcount_t num_bytes = curve->num_bytes;

    uint8_t sigma[PUBLIC_SIZE]
                                                = {0};
    uint8_t hash[32];
    uint8_t Yid[PUBLIC_SIZE + ID_SIZE]
                                          = {0};
    uint8_t sigmaId1Id2Y[PUBLIC_SIZE + ID_SIZE + ID_SIZE + PUBLIC_SIZE]
        = {0};

    uECC_word_t _e[uECC_MAX_WORDS]          = { 0 };
    uECC_word_t _ePu[uECC_MAX_WORDS * 2]    = { 0 };
    uECC_word_t _ePuY[uECC_MAX_WORDS * 2]   = { 0 };
    uECC_word_t _sigma[uECC_MAX_WORDS * 2]  = { 0 };

    #if uECC_VLI_NATIVE_LITTLE_ENDIAN
    uECC_word_t *_public_ca  = (uECC_word_t *)public_ca;
```

```
uECC_word_t *_Y          = (uECC_word_t *)Y;
uECC_word_t *_private    = (uECC_word_t *)private;
#else
uECC_word_t _public_ca[uECC_MAX_WORDS * 2];
uECC_word_t _Y[uECC_MAX_WORDS * 2];
uECC_word_t _private[uECC_MAX_WORDS];
#endif

#if uECC_VLI_NATIVE_LITTLE_ENDIAN
bcopy((uint8_t *) _public_ca,  public_ca,  num_bytes*2);
bcopy((uint8_t *) _Y,  Y,  num_bytes*2);
bcopy((uint8_t *) _private, private, num_bytes);
#else
uECC_vli_bytesToNative(_public_ca, public_ca, num_bytes);
uECC_vli_bytesToNative(_public_ca + num_words, public_ca +
    num_bytes, num_bytes);
uECC_vli_bytesToNative(_Y, Y, num_bytes);
uECC_vli_bytesToNative(_Y + num_words, Y + num_bytes, num_bytes);
uECC_vli_bytesToNative(_private, private, BITS_TO_BYTES(curve->
    num_n_bits));
#endif

memcpy(Yid, Y, PUBLIC_SIZE);
memcpy(Yid + PUBLIC_SIZE, id, ID_SIZE);

custom_hash(hash, Yid, PUBLIC_SIZE + ID_SIZE);
bits2int(_e, hash, sizeof(hash), curve);

_mulPoint(_ePu, _e, _public_ca, curve);

_addPoint(_ePuY, _ePu, _Y, curve);

_mulPoint(_sigma, _private, _ePuY, curve);

#if uECC_VLI_NATIVE_LITTLE_ENDIAN
bcopy((uint8_t *) sigma, (uint8_t *) _sigma, num_bytes*2);
#else
uECC_vli_nativeToBytes(sigma, num_bytes, _sigma);
uECC_vli_nativeToBytes(sigma + num_bytes, num_bytes, _sigma +
    num_words);
#endif

memcpy(sigmaId1Id2Y, sigma, PUBLIC_SIZE);
```

```
    memcpy(sigmaId1Id2Y + PUBLIC_SIZE, id, ID_SIZE);
    memcpy(sigmaId1Id2Y + PUBLIC_SIZE + ID_SIZE, id_cluster, ID_SIZE);
    memcpy(sigmaId1Id2Y + PUBLIC_SIZE + ID_SIZE + ID_SIZE, Y,
        PUBLIC_SIZE);

    custom_hash(key, sigmaId1Id2Y, PUBLIC_SIZE + ID_SIZE + ID_SIZE);

    return 1;
}
```

Listing 5.11: UWSINK implementacija *blockchain* konsenzusa za autentifikaciju i reautentifikaciju u Desert simulacionom okruženju.

```
std::string packet_identifier = std::to_string(sourceAddress) + ":" +
    std::to_string(sequenceNumber);
std::string firstReceiverKey = "packet:first_receiver:" +
    packet_identifier;

auto firstReceiverReply = (redisReply*)redisCommand(c, "SETNX %s %d",
    firstReceiverKey.c_str(), Sinkid);
bool isFirstReceiver = firstReceiverReply && firstReceiverReply->type
    == REDIS_REPLY_INTEGER && firstReceiverReply->integer == 1;
if (firstReceiverReply) freeReplyObject(firstReceiverReply);
std::string authKey = "node:auth_status:" + std::to_string(
    sourceAddress);
std::string reAuthKey = "node:re_auth_status:" + std::to_string(
    sourceAddress) + ":" + std::to_string(Sinkid);

if (isFirstReceiver) {
    if (tracefile_enabler_) {
        printReceivedPacket(p);
    }
    if (receivedTrafficType == 5) {
        authenticationRequests++;
        double authExpirationTime = Scheduler::instance().clock() +
            keyExpiry;
        redisCommand(c, "HMSET %s expiration %f sinkId %d", authKey.
            c_str(), authExpirationTime, Sinkid);
        sendPktKey(0.0, authResponseSize, 5, static_cast<nsaddr_t>(
            sourceAddress));
    } else if (receivedTrafficType == 10 || receivedTrafficType == 8) {
        double currentTime = Scheduler::instance().clock();
        auto authReply = (redisReply*)redisCommand(c, "HGETALL %s",
            authKey.c_str());
```

```
if (authReply && authReply->type == REDIS_REPLY_ARRAY) {
    if (authReply->elements != 0) {
        double authExpirationTime = 0;
        int registeredSinkId = -1;
        for (size_t i = 0; i < authReply->elements; i += 2) {
            std::string field = authReply->element[i]->str;
            if (field == "expiration") {
                authExpirationTime = atof(authReply->element[i
                    + 1]->str);
            } else if (field == "sinkId") {
                registeredSinkId = atoi(authReply->element[i +
                    1]->str);
            }
        }
        if (Sinkid != registeredSinkId) {
            crossClusterConnections++;
        }
        if (currentTime > authExpirationTime ||
           authExpirationTime == 0) {
            sendPktKey(0.0, 10, 6, static_cast<nsaddr_t>(
               sourceAddress));
        } else {
            if (Sinkid != registeredSinkId) {
                if (receivedTrafficType == 8) {
                    double reAuthExpirationTime = Scheduler::
                        instance().clock() + reAuthExpiry;
                    redisCommand(c, "HMSET %s expiration %f
                        sinkId %d", reAuthKey.c_str(),
                        reAuthExpirationTime, Sinkid);
                    sendPktKey(0.0, 8, 8, static_cast<nsaddr_t
                        >(sourceAddress));
                } else {
                    auto reAuthReply = (redisReply*)
                        redisCommand(c, "HGETALL %s", reAuthKey
                        .c_str());
                    if (reAuthReply && reAuthReply->type ==
                        REDIS_REPLY_ARRAY) {
                        if (reAuthReply->elements != 0) {
                            double reAuthExpirationTime = 0;
                            for (size_t i = 0; i < reAuthReply
                                ->elements; i += 2) {
                                std::string field = reAuthReply
                                    ->element[i]->str;
```

```
                                        if (field == "expiration") {
                                            reAuthExpirationTime = atof
                                                (reAuthReply->element[i
                                                 + 1]->str);
                                        }
                                    }
                                    if (currentTime >
                                        reAuthExpirationTime ||
                                        reAuthExpirationTime == 0) {
                                        sendPktKey(0.0, 10, 7,
                                            static_cast<nsaddr_t>(
                                            sourceAddress));
                                    }
                                } else {
                                    sendPktKey(0.0, 10, 7, static_cast<
                                        nsaddr_t>(sourceAddress));
                                }
                            }
                            if (reAuthReply) freeReplyObject(
                                reAuthReply);
                        }
                    }
                }
            } else {
                sendPktKey(0.0, 10, 6, static_cast<nsaddr_t>(
                    sourceAddress));
            }
        }
        if (authReply) freeReplyObject(authReply);
    }
}
```