UNIVERZITET SINGIDUNUM
BEOGRAD
DEPARTMAN ZA POSLEDIPLOMSKE STUDIJE

DOKTORSKA DISERTACIJA

POVEĆANJE BEZBEDNOSTI I PRIVATNOSTI INTEGRISANJEM SIGURNOG BLOKČEJN INTERFEJSA U ARHITEKTURU INTERNETA STVARI

MENTOR:

Prof. dr Marko Šarac

STUDENT: Nikola Pavlović

BROJ INDEKSA: 2019460019

Beograd, 2022.god.

Sadržaj

Listing slika

Listing tabela

Sažetak

Internet stvari i blokčejn se smatraju za dve glavne tehnologije današnjice. Smanjenje kašnjena i povezanost sistema je dovelo do veće fleksibilnosti pri korišćenju aplikacija koje se nalaze na udaljenim uređajima. Najveći problem interneta stvari je to da nemaju dovoljno računarskih resusrsa, nedovoljna količina memorije i slabi procesori koji su optimizovani da troše malo energiije sprečavaju korišćenje robusnih algoritama za šifrovanje. Internet stvari se suočava sa mnogim izazovima, kao što su slaba interoperabilnost, bezbednosne ranjivosti, privatnost i nedostatak standarda. U ovom radu daje se predlog korišćenja softverskog interfejsa kao arhitekture sigurnosnog prolaza za pametne uređaje. Sigurnosni interfejs omogućava korišenje jačih kriptografskih algoritama za udaljenje servise pamethih uređaja. Ovaj pristup pobojšava bezbednost podataka koji šalju pametni uređaji korišćenjem kompitablnih algoritama za šifrovanje podataka pre nego što se proslede na udaljenje servise. Pored interfejsa u ovom radu se koristi i blokčejn tehnologiija. Razlog korišćenja blokčejn tehnolgije je da se u mrežu povezanih pametnih uređaja ubaci decentralizacija i autentifikacija. Samom integracijom blokčejna dobija se toliko potrebna anonimnost i fleksibilnost koju trenutni internet stvari nema. Blokčejn štiti interfejs svojim tehnologijama kojima se izbacuje jedna tačka kontrole, beleže se sve transakcije, proverava njihova validnost i samim tim se pruža poverenje među uređajima u jednoj mreži. Rezultat ovog rada je razvoj interfejsa koji daje podršku pametnim uređajima da koriste bilo koji kriptografski algoritam, daje mogućnost mapiranja IP adresa (engl. Internet Protocol address) i na taj način spreče pristup neautorizovani pristup mreži. Pored razvoja interfejsa, blokčejn tehnologija će biti uključena u kompitablnom režimu tako da je efikasna za rad na pametnim uređajima imajući u obzir limitacije sa resursima.

Abstract

The Internet of Things and blockchain are considered to be the two main technologies of today. Reducing delays and system connectivity has led to greater flexibility in using applications located on remote devices. The biggest problem with the Internet of Things is that they do not have enough computing resources, insufficient memory, and weak processors that are optimized to consume little energy prevent the use of robust encryption algorithms. The Internet of Things faces many challenges, such as poor interoperability, security vulnerabilities, privacy, and a lack of standards. This paper proposes the use of a software interface as a security gateway architecture for smart devices. The security interface allows the use of stronger cryptographic algorithms to remotely service smart devices. This approach enhances the security of data sent by smart devices by using compatible data encryption algorithms before being forwarded to remote services. In addition to the interface, blockchain technology is used in this paper. The reason for using blockchain technology is to include decentralization and authentication in the network of connected smart devices. The integration of the

blockchain itself gives the much-needed anonymity and flexibility that the current Internet of Things does not have. Blockchain protects the interface with its single point control technologies, records all transactions, validates them, and thus provides trust between devices on a single network. The result of this work is the development of an interface that supports smart devices to use any cryptographic algorithm, provides the ability to map IP addresses (Internet Protocol addresses), and thus prevents unauthorized access to the network. In addition to the development of the interface, blockchain technology will be included in the compatible mode so that it is efficient to work on smart devices, taking into account resource limitations.

1. UVOD

1.1. Motivacija i opšta razmatranja

Skoro svaki uređaj koji ima mogućnost da se poveže na internet može biti pretvoren u IoT uređaj. IoT uređaji može biti povezan na internet žičnom ili bežičnom mrežom. 5G mreža je postala veoma aktuelna kao potencijalno rešenje za umreživanje velikog broja pametnih uređaja. Veoma važno za pametne uređaje je da imaju mogućnost da prikupe neke informacije u fizičkom svetu oko sebe i da ih pošalju dalje u digitalnom obliku.

U jednom IoT sistemu mogu da se nalaze sledeći uređaji:

- Senzori za praćenje promena nastalih u fizičkom okruženju.
- Aktuatori koji na osnovu primećenih promena izvršavaju fizičku aktivnost.
- Mikrokontroleri sa uređajem.
- Mikroračunari.

Karakteristike IoT uređaja su sledeće:

- Samostalno, odnosno dinamički se prilagođavaju promenama u okruženju.
- Podešavaju/konfigurišu se bez učešća korisnika.
- Međusobna komunikacija između pametnih uređaja je predefinisana putem standardizovanih interoperabilnih komunikacionih protokola.
- Svaki pametni uređaj je jedinstven, ima svoju jedinstvenu IP adresu ili jedinstveni indentifikator resursa.
- Svaki pametni uređaj ima odnosno mora da ima pristup internetu da bi funkcionisao i bio vidljiv ostalim uređajima, aplikacijama i korisnicima.

Iako Internet stvari može rešiti širok spektar problema, bezbednosni izazovi i dalje postoje. Zabrinutost se odnosi na bezbednost i privatnost. Bezbednosni izazovi dolaze u obliku dizajna, nedostatka standarda i propisa. Zabrinutost u pogledu privatnosti najvidljivija je u prikupljanju i upotrebi podataka i lošoj implementaciji privatnosti prema dizajnu. Mnogi proizvođači pametnih uređaja prikupljaju korisničke aktivnosti radi poboljšanja usluge. Ovo je, međutim, velika problem za privatnost zbog nemogućnosti da se od toga odustane. Ostali uređaji interneta stvari nemaju dovoljno informacija o održivosti i nadogradnji, što znači da postoji velika verovatnoća da bi mogao imati eksploataciju i da neće biti načina da se to reši. Jedan od načina za rešavanje izazova u pogledu privatnosti i bezbednosti je sinteza blokčejn tehnologije sa internetom stvari. Blokčejn tehnologija uklanja servere koji su centar IoT infrastrukture. To znači

da IoT podaci teku kroz čvorove blokova i svaka transakcija ima odgovarajuću autentifikaciju. Srž blokčejn tehnologije su blokovi koji se generišu transakcijama učesnika. Detalji svake blok transakcije se validiraju ako se održavaju na ispravan način. Ovo ne dozvoljava neovlašćeno menjanje podataka unutar bloka. Zbog protoka podataka unutar arhitekture interneta stvari, blokčejn je odlično rešenje. Blokčejn omogućava uređajima interneta stvari da održavaju trenutni protok podataka i poboljšaju bezbednost i privatnost tako što uzastopno proveravaju svaku transakciju odnosno mrežni zahtev. Standardni protok podataka za uređaje interneta stvari je sledeći. Senzori šalju podatke putem interneta ka centralnom serveru. Centralni server radi sa podacima i obezbeđuje uređaj interneta stvari sa određenim skupom komandi šta treba da uradi ili obrađenom informacijom koja je dostupna korisniku.

Ako ovom protoku podataka dodamo blokčejn, centralni server bi bio uklonjen i zamenjen distribuiranim blokčejnom.

Uvođenje blokčejn tehnologije u infrastrukturu interneta stvari ima sledeće prednosti:

- Ukidanje jedinstvenog kontrolnog organa.
- Ugrađeno poverenje između uređaja interneta stvari.
- Zapisi o istorijskim radnjama koje su izvršili uređaji interneta stvari.
- Svi podaci koje deli ovi uređaji su privatni i dostupni samo unutar mreže.
- Pouzdanost ovih uređaja je povećana zbog toga što se nalaze u zatvorenoj/privatnoj decentralizovanoj mreži.
- Omogućavanje drugim, mnogo jačim, kriptografskim algoritmima da budu implementirani u ovoj mreži.

Blokčejn nije savršeno rešenje koje bi rešilo sve probleme arhitekture interneta stvari. Jedan od najvećih problema je ograničenje skladištenja podataka, skalabilnost i vreme obrade podataka potrebno za uređaje sa ograničenom procesorskom moći. Ograničenje skladištenja je direktno povezano sa načinom na koji blokčejn funkcioniše, zahteva jedinstvenu bazu podataka sa spiskom svih uzastopnih transakcija. Dodavanje više uređaja interneta stvari u jednu decentralizovanu mrežu će stvoriti problem skalabilnosti, što znači da je više prostora za skladištenje direktno povezano sa više uređaja interneta stvari.. Vreme obrade je povezano sa dopuštanjem drugim mnogo jačim kriptografskim algoritmima da zaštite tj. obradi podatke. Više uređaja znači više radnji i više obrade podataka.

U ovom radu se razmatra problem bezbednosti i privatnosti podataka koji razmenjuju pametni uređaji.

U radu se razmatra više međusobno povezanih problema:

- Analiza napada na uređaje internet stvari i mrežu u kojoj se oni nalaze.

- Analiza nedostataka u trenutnoj mrežnoj komunikaciji između uređaja interneta stvari.
- Analiza napada na blokčejn mrežu.
- Predlog rešenja za odbranu od napada kao i njenu prevenciju.
- Predlog rešenja za integraciju interneta stvari sa blokčejn tehnologijom.

Motivacija za ovaj rad potiče od uočenih problema koji postoje u svim uređajima interneta stvari. Hab za uređaje interneta stvari koje obezbeđuje proizvođač (ako postoji) nudi malo ili nimalo bezbednosnih funkcija. Ova čvorišta uglavnom integrišu različite uređaje interneta stvari istog brenda. Drugi habovi interneta stvari se uglavnom koriste za posmatranje uređaja u pametnoj kući i prikazivanje podataka koje pružaju na računaru. Rešenje je jednostavan interfejs pogodan za bilo koji uređaj interneta stvari i mrežnu infrastrukturu. Korišćenje blokčejna kao dodatnog sloja sprečava druge napadače da pristupe pametnim uređajima. Glavna karakteristika ovog rada je da podržava bilo koji algoritam šifrovanja koji koriste udaljeni servisi u oblaku kao i fleksibilnost predloženog rešenja da bude primenjiv na svakom uređaju interneta stvari bez obzira na njegov način primene i uloge u samoj mreži.

1.2. Predmet istraživanja

Sajber-napadi nisu novina za internet stvari jer je duboko uključen u aktivnosti i društva, tako da postaje neophodno shvatiti ozbiljnost napada na uređaje interneta stvari. Primena bezbednosti u internetu stvari je prepoznata kao jedna od najvećih prepreka.

Kevin Ešton je uveo pojam internet stvari 1998. godine. On je naveo da je internet stvari mreža računara koja zna sve o ljudima i da su specijalizovani da koriste podatke koje prikupljaju bez ikakve pomoći čoveka. Uređaji su međusobno povezani. Stoga se internet stvari može odnositi na međusobnu povezanost između često korišćenih elektronskih uređaja. Mogućnost interneta stvari da ponudi različite usluge je razlog zašto je jedna od najbrže rastućih tehnologija.

Nekoliko grupa i lideri industrije su predložili standardizaciju sistema i protokola interneta stvari, ali nažalost nisu došli do konkretnog rešenja. Kao odgovor na rastuću potražnju za povezanim uređajima i uslugama širom sveta, internet stvari je stvorio preveliku potražnju za bezbednošću. Da bi internet stvari ispunio svoj potencijal, potrebna mu je zaštita od potencijalnih napadača. Različiti napadi, kao i nekoliko pretnji, svakodnevno se povećavaju i po broju i po složenosti. Internet stvari treba da garantuje bezbednost i privatnost obrađenih podataka kako bi korisnicima pružio korisne rezultate.

Da bi se obezbedila robusnost i pouzdanost na nivou usluge i da bi se mogla podržati bezbednost, stvara se značajna potreba za takvim bezbedonosnim sistemima. Problemi bezbednosti i privatnosti su sve veća briga kupaca u njihovom prelasku na internet stvari. Primena uređaja interneta stvari u kući i na radnom mestu uvodi nova bezbednosna

pitanja.

Šenonov koncept savršenih šifarskih sistema je polazna tačka istraživanja. Ono što se proučava je sledeće:

- Od kakvih napada sistem treba da obezbedi zaštitu?
- Šta je vidljivo napadačima sistema?
- Šta je neophodno da korisnici sistema vide?
- Da li je moguće da pametni uređaj bude bez direktnog pristupa internetu ?

Opšta hipoteza od koje se krenulo u istraživanje u disertaciji je: „Blokčejn i distribuirana infrastruktura omogućavaju da u komunikaciji između pametnih uređaja ili udaljenih servisa bude primenjen moderni kriptografski algoritam i da se omogući laka distribucija kriptoloških ključeva"

Posebna hipoteza koja proizilazi iz opšte je: „Kombinovanjem više kriptografskih algoritama zajedno sa sistemima validacije i autentifikacije koji se koriste u blokčejn infrastrukturi moguće je pobojšati bezbednost komunikacije između pametnih uređaja i servisa bez znatnog uticaja na performanse sistema"

Pojedinačne hipoteze koje su korišćene u disertaciji su:

- Postojeće metode napada na pametne uređaje je moguće sprečiti ili bar smanjiti mogućnosti njihovog izvođenja i efekte samih napada.
- Procena nivoa sigurnosti koji treba postići.
- Broj uređaja koji će koristiti sistem.
- Kompromis između troškova razvoja i performansi sistema.
- Razmena ključeva i primena naprednih kriptografskih algoritama zavisi od udaljenih servisa i podrške proizovođača pametnog uređaja.
- Limitacija u količini memorije pri čuvanju podataka u blokčejn bazu podataka kao i primena kompresije.
- Komunikacija između pametnih uređaja se validira i upisuje u blokčejn bazu kao transakcije i time se ima čitav uvid u samu aktivnost infrastrukture.

1.3. Dopronosi disertacije

Rad je utemeljen na dostignućima naučnih stvaralaca kao i na sopstvenom radu na ovoj tezi.

U ovoj disertaciji predstavljeni su analizirani brojni rezultati istraživanja sprovedenih u poslednjih par godina objavljenih u prestižnim naučnim časopisa. Ovi rezultati su upotrebljeni kao polazna osnova za istraživanje koje je sprovedeno i izneto u ovoj disertaciji. Očekuje se da

će analizirani rezultati više istraživanja, kao i istraživanje ove disertacije poslužiti kao polazna osnova za dalja istraživanja i produbljivanje ove teme.

Tema doktorske disertacije je aktuelna, a doprinos predloženog istraživanja je u uočavanju značaja na polju kriptografske bezbednosti.

Disertacija sadrži i predlog radnog okvira sistema zasnovanog na sintezi modernih kriptografskih algoritama sa blokčejn tehnologijom. Cilj jednog ovakvog sistema je olakšana integracija modernih tehnika za bezbednost i privatnost sa uređajima koji imaju malu količinu računarskih resursa i nisu u mogućnosti samostalno da izvrše date algoritme.

Stručni doprinosi:

- Pregled svetskih iskustava u oblasti zaštite, sigurne komunikacije i privatnosti podataka.
- Pregled tehnologija koje se koriste za projektovanje i implementaciju sistema za zaštitu.
- Razvoj softvera u Node.js programskom jeziku sa potpunom implementacijom projektovanog rešenja.

Naučni doprinosi:

- Pregled i analiza istraživanja i dostignuća iz integracije blokčejn tehnologije sa internetom stvari.
- Predlog sopstvenog rešenja koji integriše blokčejn sa internetom stvari i nudi primenu modernih kriptografskih algoritama.
- Predlog za dalji razvoj.

1.4. Struktura disertacije

Proces naučnog istraživanja je sproveden u nekoliko koraka:

- Uvod
- Pregled u oblasti istraživanja
- Teorijske osnove istraživanja
- Integracija blokčejna sa internetom stvari
- Zaključak sa sumarnim rezultatima i predlog za budući rad

U nastavku teze, drugom delu, razmatrano je opšte stanje u oblasti istraživanja. Uočeni su nedostaci u bezbednosti i privatnosti koji postoje u trenutnim protokolima koji koriste uređaji internet stvari. Uvidom u naučne radove uočeni su nedostaci koji trenutno postoje u internetu stvari kao i prednosti blokčejn-a pri sintezi sa internetom stvari.

U trećem delu rada obrazložene su teorijske osnove istraživanja. U nastavku ovog dela urađena je analiza trenutnih sistema bezbednosti. Analizirana je blokčejn tehnologija i njen uticaj na internet stvari. Date su procene unapređenja privatnosti kao i dodavanja kriptografksih algoritama u postojeću mrežnu komunikaciju između pametnih uređaja.

U četvrtom delu rada dat je predlog predloženog rešenja za integraciju interneta stvari i blokčejn tehnologije. Napravljena je generička šema predloženog rešenja. Prikazan je izbor eksperimentalnog okruženja i ciljevi koji treba da budu zadovoljeni predloženim rešenjem. Na kraju ovog dela urađena je evaluacija čitavog sistema i prikazani su rezultati.

U petom delu rada, poslednjem, dat je zaključak. Prikazani su rezultati i doprinosi rada, sumirani svi ciljevi postavljeni na početku istraživanja kao i predlog daljeg istraživanja.

2. Pregled u oblasti istraživanja

Trenutno stanje u bezbednosti infrastrukture interneta stvari je tako da postoje različiti protokoli koju nude neki vid bezbednosti, ali da bi se mogli koristiti neophodno je postojanje hub-a. Problem kod njih je što različiti proizvođači ne nude integraciju sa različitim uređajima i da bi se obezbedila jedna mreža neophodno je postojanje više različitih hub-ova koji neće garantovati da je moguća ponovo integracija svih uređaja i hub-ova međusobno u jednu jedinstvenu mrežu.

Mnogi uređaji interneta stvari ne koriste nikakve naprednije protokole koji bi obezbedili osnovni nivo privatnosti i bezbednosti podataka koji se šalju na internet.

Autori u nastavku rada kao i njihova istraživanja su analizira i korišćena kao osnova za dalji rad ove disertacije. Autori su uradili analizu modernih problema vezanih za bezbednost i privatnost protokola interneta stvari. Pregledali su koji su najčešći napadi, zašto se dešavaju i kako mogu da se postave bezbednosne mere koje bi ih sprečile. Pored pregleda problema, bezbednosnih mera i privatnosti, autori su dali predloge za nove protokole koji bi bili efikasniji i nudili privatnost pa čak i bezbednost sa novim inovativnim kriptografskim algoritmima.

Interakcija između uređaja interneta stvari, mrežnih protokola kao i blokčejna predstavlja svoj komplementarnih tehnologija. Autori i njihova istraživanja koji su služili za osnovu ove disertacije naveli su zašto je blokčejn potreban internetu stvari i šta se postiže sa uvodom blokčejna u arhitekturu interneta stvari, kao na primer decentralizacija mreže, dodavanje novih sistema za validaciju podataka, kao i omogućavanje komunikacije između različitih uređaja interneta stvari.

Pored predloga, potrebno je odgovoriti na tehnološke izazove. Da bi se uradila sinteza blokčejna sa internetom stvari potrebno je da se čitava blokčejn struktura prilagodi arhitekturi interneta stvari. Sa optimizacijom blokčejna i primenom na arhitekturu interneta stvari treba da se obrati pažnja na pouzdanost, tačnost, sigurnost podataka, limitacije u hardveru koji se koristi za blokčejn i validaciju i zaštitu privatnosti.

Fokus disertacije je na unapređenju bezbednosti i privatnosti u komunikaciji između uređaja interneta stvari kao i njihove komunikacije sa servisima u oblaku. Nezaštićene računarske mreže, računarske mreže koje koriste stare protokole koje nude slabu ili nikakvu bezbednost, bežične računarske mreže kojima svako ima pristup su sve mreže i protokoli koji su meta napadača.

Autentifikacija i validacija podataka su od velike važnosti u interakciji između uređaja interneta stvari. Korišćenje tradicionalne autentifikacije sa primenom korisničkog imena i lozinke ne bi bilo primenjivo u ovim mrežama. Sa povezivanjem blokčejn tehnologije i tehnikama validacije i autentifikacije koje blokčejn tehnologije koriste, ovaj problem je moguće rešiti i postignuti ne samo visok nivo poverljivosti već i sistem koji bi omogućavao olako da se dodaju članovi u mrežu kao i uklone. Da bi ovo bilo primenjivo mora da postoji jedna osoba od poverenja koja bi imala pristup sistemu, administrator sistema.

U nastavku ovog poglavlja dat je pregled na osnovu radova u kojima su urađene bezbedonosne analize, pregled napada na uređaje interneta stvari kao i predloženi jedinstveni protokoli koji bi obezbedili bezbednost kao i privatnost podataka. Autori su uradili pregled postojećih rešenja kao i hub-ova i koje funkcionalnosti oni donose i pregled njihove efektivnosti u povećanju privatnosti i bezbednosti podataka sa kojima rade uređaji interneta stvari. Pored toga dati su predlozi zašto je neophodno da se integriše blokčejn tehnologija u arhitekturu interneta stvari i koje beneficije kao i izazove to donosi.

N. Kumar i P. Mallick [26] istraživali su sa kakvim se izazovima suočava trenutna infrastruktura interneta stvari. U radu su se autori bavili izazovima privatnosti i bezbednosti. Oni su identifikovali najveće probleme sa trenutnom infrastrukturom i pružili pregled svih njih. Uz obezbeđen pregled, autori su takođe naveli zašto je blokčejn potreban internetu stvari. Neki od sektora u kojima se blokčejn i internet stvari mogu spojiti i pružiti dobre prednosti su poljoprivreda, poslovanje, distribucija, energitka, finansije, zdravstvo, transport i logistika i pametni grad. Autori su takođe dali spisak prednosti kao što su podaci o neovlašćenom pristupu, uklanjanje jedinstvenog kontrolnog autoriteta, evidentiraju podatke o starim transakcijama na pametnim uređajima i druge.

M. Miraz [27] je u svom radu dao pregled blokčejna i njegovog rada sa osnovama, kao i osnovama interneta stvari. Nakon što je dao osnove interneta stvari i blokčejna, autor je predložio spajanje ove dve tehnologije i predstavio prednosti spajanja, poboljšanu bezbednost, nepromenljivost podataka, proverljivost i pristup pametnim ugovorima iz blokčejn. Korist koju internet stvari pruža blokčejnu je aktivno učešće u procesu konsenzusa.

V. Hasija i dr. [28] pružio je pregled istorijskih promena infrastrukture interneta stvari, od zasnovane na oblaku do sadašnje infrastrukture sa mnogo servera sa jednom tačkom otkaza i budućim zasnovanim na blokčejnu. U istraživanju su autori predstavili sigurnosne pretnje na različitim nivoima mreže. Neki od napada se raspravljaju i nude rešenja za njih na različitim mrežnim slojevima. Očekuje se da će istraživanja poslužiti kao izvor za buduća istraživanja i rešiti izazove u pogledu sigurnosti i privatnosti koji postoje u trenutnoj infrastrukturi interneta stvari.

Napad	**Meta**	**Slabost**	**Tehnika**
Napad uskraćivanja usluge (DoS)	Uređaji interneta stvari koji su povezani putem interneta.	Smanjenje kapaciteta mreže. Onemogućite mrežu.	Status omogućavanja IP-a doprinosi skupu. Iskorišćen je distribuirani napad i automatski zatvara sistem.
Wormholes	Lokacija paketa.	Problem u proveri informacija o rutiranju.	Snimite pakete na jednoj lokaciji i tunelirajte ih na drugu lokaciju.
Lažne, izmenjene ili ponovo reprodukovane informacije o rutiranju	Informacije o rutiranju. Mogu se otkriti uređaji interneta stvari.	Visoko kašnjenje od kraja do kraja. Izvori rutiranja mogu biti prošireni ili skraćeni.	Prvo, napadač samo sluša. Napada samo kada predajnik prestane da šalje signal, a zatim šalje nepouzdan signal.
Sybil	Integritet bezbednosti podataka i korišćenje resursa.	Pokrenite pretnju za protokol geografskog rutiranja. Skupa mreža.	Propagirajte malver na veb lokaciju. Napadač maskira normalnog korisnika.

Tabela 1. Najčešći napadi na internet i tehnike.

Sloj	**Napad**	**Metode/Strategije napada**
Fizički	Ometanje	Stvara radio smetnje na uređajima interneta stvari.

	Petljanje	Stvara kompromitovane čvorove.
Veze podataka	Sudari	Istovremeno prenesite dva čvora na istoj frekvenciji.
	Iscrpljenost	Ponavljajućim sudarom, čvorovima.
	Nepravednost	Korišćenje gornjih napada sloja veze.
Mreža	Lažno izmenjene ili ponovo reprodukovane informacije o rutiranju.	Kreira petlje za rutiranje, produžava ili skraćuje izvorne rute, privlači ili odbija mrežu od odabranih čvorova.
	Selektivno prosleđivanje	Biraju se koje informacije koje ste prikupili će biti prosleđene.
	Sinkhole	Nadgledanje, redundantnost, autentifikacija.
	Sybil	Jedan čvor se duplira da bi bio na više lokacija.
	Wormholes	Selektivno tuneliranje ili ponovno prenošenje informacija na uređaje.
	HELLO flood	Korišćenje HELLO paketa za pokretanje napada na sistem interneta stvari.
	Lažiranje potvrde	Menjanje glave paketa da se lažira potvrda u

		sloju mreže.
Transportni	Poplave	Ponavljajte zahtev za novu vezu dok sistem interneta stvari ne dostigne maksimalni nivo.
	De-sinhronizacija	Prekid postojeće veze.
Aplikacije	Napadi na pouzdanost klonskog napada: Iskrivljenje sata, selektivno prosleđivanje poruka, izobličenje agregacije podataka	Napadači se obično maskiraju kao normalni učesnici u sistemu interneta stvari. Napadači takođe mogu da izaberu poruku koju nameravaju da pošaljuu u sistemu i pokrenuti sopstvene zlonamerne aktivnosti.

Tabela 2. Napadi na uređaje interneta stvari po sloju mreže.

H.F Atlam i saradnici [31] u svom radu su se bavili problemima sa klijent/server modelom koji koristi trenutnu infrastrukturu interneta stvari. Od mnogih pitanja koja postoje, najvažniji su skalabilnost i bezbednost. U radu autori su dali pregled integracije blokčejna sa internetom stvari. Spajanje ove dve tehnologije donelo bi mnoge prednosti i izazove koje bi trebalo rešiti. Izazovi sa kojima se suočava spajanje su skalabilnost, usklađenost sa zakonima, procesorska snaga i vreme, skladištenje.

U radu [8] autori nastoje da doprinesu boljem razumevanju pretnji. Autori su objasnili zašto su uređaji interneta stvari izuzetno vredni za napadače. Većina uređaja radi bez ljudske interakcije tako da napadačima je lako da dobiju fizički pristup njima. Ovi uređaji takođe komuniciraju korišćenjem bežičnih mreža što napadaču olakšava da dođe do poverljivih informacija. Većina uređaja ne može da podrži tj. koristi složene bezbednosne algoritme zbog svojih hardverskog ograničenja. U ovom radu autori su dali pregled najvažnijih bezbednosnih problema interneta stvari sa fokusom na izazove oko uređaja i usluga. Autori su zaključili da je važno definisati standard koji će se baviti nedostacima postojećih bezbednosnih mehanizama interneta stvari.

U radu autori [130] su dali sveobuhvatan pregled napada na svaki sloj mrežnog protokola. Da bi se poboljšala bezbednost, uređaji bi trebalo da ugrade nove mrežne protokole kao što su IPv6. Većina signala koji se prenose između pametnih uređaja može biti ugrožena ometanjem talasa. Veoma popularan napad na pametne uređaje je Relay Attack. Relay Attack se može izvršiti lažiranjem, promenom ili ponovnim reprodukcijom informacija o identitetu koje daje uređaj. Drugi napad koji se može izvršiti na ovom sloju je Timing Attack. Timing Attack se izvodi analizom potrebnog vremena za izvršenje šifrovanja. Rezultat ovog napada je da napadač dobija pristup ključu za šifrovanje. Moguće je da napadači dobiju fizički pristup čvoru i snime sve

informacije i podatke. Ovo se zove napad Node Capture. Na mrežnom sloju, najpopularniji napadi su napad uskraćivanja usluge (DoS) i napad čoveka u sredini. Zbog nedostatka standarda ili bezbednosnih politika interneta stvari veliki broj uređaja je osetljiv na napade na sloj aplikacije. Različiti softver i aplikacije imaju različitu ili nikakvu bezbednosnu implementaciju. Autori su zaključili da uređaji treba da koriste novije mrežne standarde. Postoji potreba za novim identifikacionim, bežičnim, softverskim i hardverskim tehnologijama za rešavanje izazova u internetu stvari.

U radu [9] je dat pregled pitanja privatnosti usmerenih na pametne kućne uređaje. Da bi korisnicima pružili vredne rezultate, pametni uređaji moraju da garantuju da su obrađeni podaci bezbedni. Većina pametnih kućnih sistema kao što je AllJoyn, koristi asimetričnu kriptografiju za obavljanje autentifikacije tokom rada sistema. Ovaj uređaj koristi Wi-Fi ruter da bude centralni čvor sistema. Što znači da je celokupna bezbednost sistema pametne kuće zasnovana na bezbednosti Wi-Fi rutera. Najpopularniji pametni uređaji kao što su Sense monitor za spavanje, Nest Cam Indoor sigurnosna kamera, WeMo prekidač i Amazon Echo otkrivaju osetljivu interakciju korisnika u svojim mrežnim paketima. To znači da postoji potreba za boljom enkripcijom kako bi se zaštitila privatnost korisnika. Autor je takođe predložio protokol u kome ljudi mogu da kontrolišu otkrivanje svojih ličnih podataka. Autori su zaključili da bi po principu privatnosti korisnici trebalo da budu u mogućnosti da zadrže kontrolu nad svojim podacima i odustanu od prikupljanja informacija bez negativnih posledica. Bezbednost i privatnost na nivou uređaja su od ključne važnosti za poboljšanje ukupne bezbednosti pametnih uređaja.

U radu [10] autori su izvršili procenu bezbednosti pametnih kućnih uređaja. Oni su smatrali da je eksploatacija čvorišta ekvivalentna iskorišćavanju svih povezanih uređaja. Čvorišta povezuju uređaje niske energije sa IP mrežama i imaju unapred uspostavljen odnos poverenja. Mnoga čvorišta su podložna napadima čoveka u sredini jer podržavaju starije verzije TLS/SSL protokola kako bi održali kompatibilnost sa svim uređajima. Mnoga čvorišta pokreću interni veb server sa TLS-om na portu 443 i SSL-om na portu 22. Sertifikat koji se koristi za TLS je samopotpisan. Ova čvorišta koriste slabe algoritme za šifrovanje kao što je RC4 i nebezbedni protokol kao što je SSLv3. Autori vrše revizije na Insteon habu, Wink 2, Sonos zvučnicima, nVidia Shield-u, Google Home-u, Samsung SmartTV-u i Samsung SmartThings-u i svi ovi uređaji su imali isti problem sa isteklim samopotpisanim sertifikatom i upotrebom slabog šifrovanja. Uređaji koji pokreću UPnP uopšte nemaju autentifikaciju i dozvoljavaju bilo kome na lokalnoj mreži da kontroliše uređaj. Primer ovih uređaja je Mi-CasaVerda VeraLite, Wink 2, Sonos, Bose SoundTouch 10, Samsung SmartTV, Logitech Harmony i Roku. Autori su predložili sledeće. Za proizvođače da implementiraju bolja bezbednosna rešenja. Krajnji korisnici treba da promene podrazumevanu konfiguraciju, omoguće šifrovanje i onemoguće daljinski pristup čvorištima.

U radu [11] autori su predložili jedan algoritam šifrovanja za multimedijalne IoT uređaje. Autori su predložili zaštitu od pasivnih napada (modifikovanje ili brisanje paketa). Autori se u ovom radu bave problemom obezbeđenja multimedijalnih sistema u IoT-u (MIoT), koji se sastoje od kamera i mikrofona. U poređenju sa drugim kriptografskim algoritmima, predložena šifra koristi samo jednu dinamičku funkciju zavisnu od ključa. Funkcija Round se zasniva na jednostavnim operacijama i postiže potrebne kriptografske performanse. Pored jednostavnih operacija, algoritam za šifrovanje se zasniva na pristupu dinamičkog ključa i dinamičkom izboru podmatrica. Ključne promene su promenljive i na pseudo-slučajan način za svaku novu sesiju. Dinamička priroda predložene šifre obezbeđuje visoku otpornost na bilo koju vrstu napada. Izbor dinamičkih podmatrica je uveden da bi se nasumično izvršio redosled podmatrica

za šifrovanje i dešifrovanje. Ovo čini predloženi pristup šiframa robusnijim u poređenju sa postojećim. Autori su sproveli nekoliko bezbednosnih eksperimenata. Eksperimenti su zasnovani na bezbednosnim merama kao što su statistička analiza, vizuelna degradacija i osetljivost. Pokazalo se da je šema koju su autori pružili efikasna i sigurna. Zasnovan je na dinamičkoj strukturi za razliku od standardnih tehnika.

2.1. Nedostaci u bezbednosti interneta stvari

Bezbednost interneta stvari je od velike važnost za normalan i nesmetan rad uređaja koji iz dana u dan su sve više u primeni i raspostranjeni oko nas. Pored postojećih pretnji, bezbednošću dodatno se nalaze i loše prakse korisnika i organizacija koji nemaju dovoljno resursa ili znanja da zaštite svoje ekosisteme interneta stvari.

Najčešći bezbedonosni problemi su sledeći:

- Ranjivost
- Zlonameran/štetan softver poznat kao i malver
- DoS napad
- Krađa informacija
- Loša konfiguracija uređaja

Ranjivost je problem koji muči korisnike i organizacije. Nedostatak resursa i procesorske snage uređaja interneta stvari je razlog zašto ne postoje ili su slabe bezbedonosne polise postavljene na njih. Veliki razlog nedostatka bezbednosti je da sami proizvođači prave veliki broj uređaja i nisu zainteresovani da potroše vreme i novac na postavljanje bezbedonosnih polisa. Osim ranjivosti na samim uređajima, nalazi se i na nivou aplikacije i povezana je sa softverom koji koristi sam uređaj interneta stvari. Problem na nivou aplikacije je što napadači često uspevaju da čak i na novijim uređajima izvrše stare napada zbog prethodno pomenutog problema, da proizvođači ne rade na tome da uređaje koji proizvode i aplikacije koji koriste na njemu nije redovno ažuriran i ne krpe se bezbedonosni propusti.

Iako dosta uređaja interneta stvari ima ograničene resurse i procesorsku moć, ponovo mogu da budu zaraženi zlonamernim softverom. U poslednjih nekoliko godina ovo je jedan od najpopunarnijih napda na uređaje. Najprofitabilniji zlonameran softver je botnet. Pored ovoga prisutan je ransomver, ucenjivački softver, koji ograničava pristup uređaju i traži otkupninu od žrtve. Neki malo jači uređaji se koriste kao mašine za rudarenje kriptovaluta.

Napadači koriste u velikom broju slučajeva zaražene uređaje kao deo botnet mreže. Zbog velikog broja uređaja koji se nalaze, najzasutpljeniji napad je napad distribuiranog uskraćivanja usluge - DDoS. Uređaji interneta stvari mogu međusobno da se zaraze unutar jedne mreže i tako botnet može samostalno bez uticaja napadača da se širi i ima

veliki broj učesnika spreman za napad.

Uređaji unutar jedne mreže pri svakoj komunikaciji preko interneta izlažu informacije koje mogu da budu vidljive napadaču. Zbog nedostatka jakih bezbedonosnih protokola i polisa informacije se obično nalaze u otvorenom obliku i mogu da se snimaju čime se narušava privatnost samog korisnika.

Loše konfiguracije uređaja interneta stvari, slabe lozinke i slabe ili nikakve bezbedonosne polise su neki od razloga zašto su jedna od omiljenih meta napada. Korisnici nemaju dovoljno znanja da podese uređaje ili proizvođači ne dostavljaju uređaje sa prekonfigurisanim softverom koji bi olakšao korisniku da lakše i sigurnije postavi uređaj u postojeću mrežu.

Internet stvari je povezan sa virtuelnim kao i sa fizičkim sistemima. To znači da bilo koji napad na njega može da ima nepredvidive efekte koji za sobom mogu da povuku fizičke posledice. Mesta na kome treba da se obrati najveća pažnja je industrija i zdravstvo gde bilo koja vrsta napada na internet stvari može da izloži osetljive informacije o pacijentima ili čak da ugrozi samo zdravlje i bezbednost. U pametnim kućama napadači mogu da iskoriste same sisteme za nadzor protiv korisnika i dobiju informacije o tome kada korisnik ovog sistema nije kod kuće.

2.2. Unapređenje bezbednosti interneta stvari

Specificne strategije i alati su neophodni da se konfiguriše mreža interneta stvari. Neke od najboljih praksi za smanjenje bezbedonosnih rizika i pretnji su:

- Prisustvo administratora u mreži interneta stvari.
- Redovno proveravanje da li postoje zakrpe i ažuriranja za uređaje unutar mreže.
- Korišćenje jakih i jedinstvenih lozinki.
- Povećanje bezbednosti Wi-Fi mreže.
- Monitorisanje/nadgledanje mreže i uređaja.
- Segmentacija mreže.
- Postavljanje jačih i novijih protokola za IoT uređaje.

Pristustvo administratora u mreži je od velike važnosti jer može smanjiti mogućnost bezbedonosnih problema. Oni su zaduženi za bezbednost uređaja i njihovu pravilnu konfiguraciju. Prisustvo administratora je od velike važnosti u mrežama kao što su industrija i zdravstvo.

Redovno proveravanje i dodavanje zakrpi kao i ažuriranje uređaja unutar mreže je od velike važnosti. Napadi na uređaje interneta stvari mogu da na ovaj način budu sprečeni.

Primena jakih i jedinstvenih lozinki na svaki uređaj interneta stvari podrazumeva da

administrator mreže koristi neki od alata kao što su menadžeri lozinki i time generiše lozinke koje ne mogu da budu probijene u realnom vremenu.

Konfigurisanje Wi-Fi mreže može da spreči mnoge napade. Aktiviranje zaštitnog zida, isključivanje WPS protokola i aktiviranje WPA2 bezbedonosnog protokola i korišćenje jakih lozinki na Wi-Fi pristupnoj tački su osnova za jaku i sigurnu mrežu.

Ukoliko postoji administrator mreže, korišćenje alata za monitorisanje mreže može da spreči i ima uvid na to ukoliko se napad desio ili je deo mreže kompromitovan.

Korisnici mogu da minimizuju rizik napada tako što će da stvore nezavisne mreže samo za IoT uređaje, a druge za sebe i goste. Segmentacija mreže pomaže u sprečavanju širenja napada i izolovanju potencijanih problematičnih uređaja koji se ne mogu odmah ukloniti iz mreže.

Za komunikaciju IoT uređaji ne koriste samo internet protokol, već ogroman skup različitih mrežnih protkola, kao što su Bluetooth i Near Field Communication (NFC), do manje poznatih nRF24, nRFkk, 443MHz, LoRA, LoRaVAN kao i optička, infracrvena komunikacija. Pri konfigurisanju ovih protkola korisnici i administratori moraju da razumeju ove protokole da bi smanjili rizik i sprečili pretnje.

2.3. Nedostaci u bezbednosti blokčejna

Dok blokčejn tehnologija se smatra jednim od najbezbednijih sistema, ima glavnu knjigu i decentralizovana je postoje mnogi bezbedonosni problemi koji mogu ugroziti čak i osnovnu prirodu tehnologije.

Najvažniji napadi na blokčejn su:

- 51% napad
- Sybil napad
- Napad dvostruke potrošnje
- Napad usmeravanja
- Napad na privatni ključ
- Napad sebičnog rudarenja
- Ranjivi pametni kontakti

Rudari igraju ključnu ulogu u validaciji transakcija na blokčejnu i na taj način pomažu da se on dalje razvija. U slučaju da se dva bloka rudare u isto vreme u mreži ostaje onaj koji dobije većinsko odobrenje a drugi postaje zastareo. Ukoliko napadači uspeju da dobije kontrolu nad 51% ili više moći rudarenja, rezultati mogu biti katastrofalni. Napadači tada mogu da iskoriste svoj položaj većine da otkažu transakcije i izvrše lažne

transakcije. Tada čak mogu biti u mogućnosti da ponovo napišu neke blokove, ali ponovno pisanje celog blokčejna (iako je teoretski moguće) bilo bi praktično nemoguće. Blokčejn bezbednosni problemi kao što je napad od 51% su verovatnije da će se desiti u ranim fazama lanca. U vreme kada je vrlo malo rudara prisutno na mreži, bilo bi moguće dobiti 51% rudarske snage.

U Sibil napadu, napadač stvara više lažnih čvorova na mreži. Koristeći te čvorove, napadač može postići konsenzus većine i ometati transakcije u lancu. Dakle, veliki napad na Sibil nije ništa drugo do napad od 51%. Za bezbednosna pitanja blokčejna kao što su napadi na Sibil, mnogi blokčejnovi koriste dokaz o radu i dokaz algoritama udela. Dok ovi algoritmi ne sprečavaju u potpunosti takve napade, već jednostavno onemogućavaju napadaču da ih izvede.

Napad dvostruke potrošnje je vrsta napada gde se ista transakcija pošalje na dva različita novčanika. Generalno, postoje mehanizmi ugrađeni u blokčejn za sprečavanje takvih vrsta napada. Na primer, ako se novčić pošalje u prvom bloku, transakcija će biti zanemarena u narednim blokovima. Ali u slučaju da su obe transakcije dospele do dva različita bloka rudarena u isto vreme, blok sa najvećim brojem potvrda sa čvorova na mreži će biti zadržan, a drugi će biti ignorisan nakon nekog vremena. Međutim, ovo ublažavanje bezbednosnih problema blokčejna kao što su napadi dvostruke potrošnje dolazi sa sopstvenim nedostacima. Na primer, drugi primaoc i dalje može da očekuje transakcije u svom novčaniku. Kao rezultat toga, generalno, korisnici čekaju da se iskopa još najmanje 6 blokova (kao što je predložio Satoshi Nakamoto u svojoj knjizi) pre nego što budu zaista sigurni da će dobiti transakcije.

Jedna stvar koju do sada lako možemo da vidimo je da blokčejn tehnologija zahteva robusnu mrežu da bi funkcionisala. ISP-ovi se povezuju jedni sa drugima i dele informacije o ruti preko BGP-a (Border Gatevai Protocol). Ovaj protokol je star i ima neke ranjivosti koje napadač može da iskoristi.Na primer, napadač koji kontroliše ISP može objaviti lažnu rutu i na taj način odbiti transakcije za neke čvorove ili čak podeliti blokčejn mrežu na pola. Pretpostavimo da korisnikov čvor nalazi na 100.0.0.0/16 (/16 je IP prefiks). Sada, ako napadač propagira rutu do 100.0.0.0/17 preko BGP-a, uskoro će ove informacije biti ažurirane u svim ruterima (tako funkcioniše BGP tako što deli informacije o ruti sa susedima). Kao rezultat toga, podaci namenjeni korisniku biće preusmereni na unos koji je naveo napadač. Bezbednosni problemi blokčejna u vezi sa rutiranjem mogu imati ozbiljne implikacije jer je 2014. napadač uspeo da izvrši napad na rutiranje. Ovo je omogućilo napadaču da spreči širenje blokova koje su rudareni preko mreže. Umesto toga, on/ona je koristio informacije da tvrdi da je obavljen posao njegov/njen i tako je nagrađen naknadama za rudarenje.

Kriptografija sa javnim ključem je srž blokčejn tehnologije. Zbog toga, nepravilna primena ili rukovanje kriptografijom javnog ključa može izazvati ozbiljne probleme u vezi sa sigurnošću blokčejna. Ako je potpisivanje ključa loše implementirano u blokčejnu (na primer, korišćenje istog ključa za višestruko potpisivanje umesto Merkleovog stabla), to može dozvoliti napadaču da dobije vaš privatni ključ iz javnog ključa. Imati kontrolu nad vašim privatnim ključem u osnovi znači posedovanje svih

podataka povezanih sa vama u blokčejnu.

Napad sebičnog rudarenja, takođe poznat kao napad blokiranja, opisuje zlonamerni pokušaj diskreditovanja integriteta mreže blokova. Napadi sebičnog rudarenja se dešavaju kada pojedinac u rudarskom skupu pokuša da spreči emitovanje uspešno potvrđenog bloka. Nakon što sebični rudar zadrži svoj uspešno rudaren blok iz grupe, oni nastavljaju da rudare sledeći blok, što dovodi do toga da je sebični rudar pokazao više dokaza o radu u poređenju sa drugim rudarima. Ovo omogućava sebičnom rudaru da zahteva nagradu za blok dok ostatak mreže usvaja svoja rešenja za blok.

Pametni kontakti su u osnovi ugovori napisani u kodu koji koriste blockchain za čuvanje zapisa. Glavni nedostatak ovde je što ugovori mogu da budu loše napisani u kodu i samim tim imaju grešku koje napadač može da iskoristi.

2.4. Unapređenje bezbednosti blokčejna

Neki od predloga za unapređenje bezbednosti su sledeći:

- Korišćenje dokaza o ulozi umesto dokaza o radu može pomoći u sprečavanju napada od 51%.
- Postoji veliki broj algoritama za sprečavanje Sibil napada. Jedan od njih koji se zove dokaz o radu implementiran je u većini kriptovaluta.
- Važno je da se kontrolišetu rudarske bazeni blokčejna. Potrebno je da svaki bazen koji prekorači ograničenje od 40%, preusmeri neke od svojih rudara u druge grupe.
- Upotreba bezbednih protokola za rutiranje (sa sertifikatima) može pomoći u sprečavanju napada rutiranja na blokčejn.
- Pametni kontakti moraju biti temeljno provereni na bilo kakve greške od strane stručnjaka pre implementacije.

2.5. Unapređenje bezbednosti sintezom blokčejna sa internetom stvari

Prednosti koje blokčejn može da ponudi, kao što su bezbednost, transparentnost, nepromenjivost, proverljivost kao i pametni ugovori, poseduju sposobnost ograničenja ekosistema interneta stvari ako se kombinuju zajedno sa njim. Internet stvari takođe poseduje sposobnost da koristi blokčejn tako što aktivno učestvuje u procesu konsenzusa. U blokčejnu stvari (BCoT) – fuziji blokčejna i interneta stvari tehnologija – obe mogu koristi jedna drugu na recipročan način.

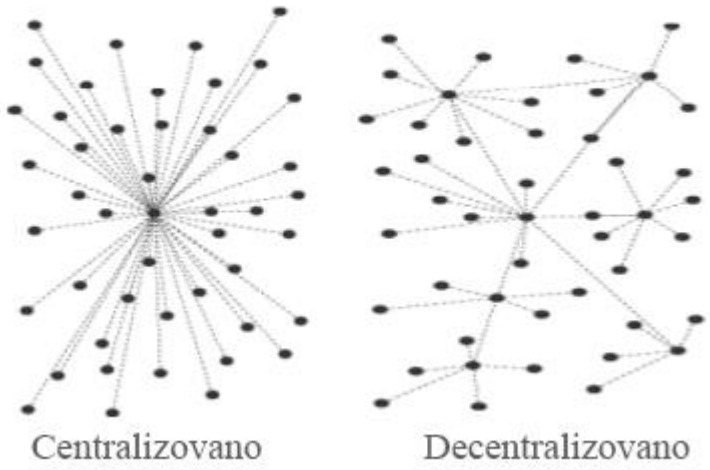
Prenosi podataka se dešavaju brzo i kontinuirano, a blokčejn omogućavaju verifikaciju podataka iz nekoliko različitih izvora. Ovo korisnicima daje više slobode i fleksibilnosti

kada je u pitanju deljenje podataka između različitih uređaja.

Postoji nekoliko prednosti sinteze blokčejna i interneta stvari, kao što su niži troškovi, brže transakcije, transparentnost pristupačnosti i poboljšana bezbednost. Ovo su neki od načina na koji se te prednosti postižu.

Primenom glavne knjige podaci se šire na gomilu računara, što čini verifikaciju boljom i napade mnogo težim da se ostvare jer podaci nisu na jednom mestu. Ovo pomaže opštoj bezbednosti, kao i sajber bezbednosti; čak i ako je čvor van mreže, informacije su i dalje zaštićene i pravilno uskladištene. Mreže i uređaji interneta stvari su potencijalno mnogo bezbedniji sa blokčejnovima zbog distribuirane arhitekture, posebno od botova.

Centralizovani sistemi imaju sve čvorove povezane na jedan glavni server. Decentralizovani sistemi koriste nekoliko čvorova koji rade nezavisno. Umesto da imaju hijerarhiju sa jednim glavnim čvorom, čvorovi su na nivou „vršnjaka“. Sa stvarima kao što su distribuirani konsenzus i šifrovanje podataka, decentralizacija pomaže u smanjenju troškova i povećava efikasnost korišćenjem interneta stvari.



Slika 1. Razlika između centralizovanog i decentralizovanog blokčejna

Blokčejn imaju nepromenljivu prirodu, što znači da se promene lako mogu otkriti jer podaci o blokčejnu obično ne mogu da se menjaju. Blokovi se prave sa vremenskim oznakama, tako da postoji hronološki redosled lanca blokova. Kada se kreira novi blok, on je povezan sa prethodnim blokom, čineći lanac „neraskidivim“. U suštini, podaci se mogu samo dodavati, a ne menjati, što čini podatke mnogo sigurnijim u svetu interneta stvari.

2.6. Primena tehnologije zasnovane na blokčejnu

Zahvaljujući svom decentralizovanom dizajnu, blokčejn uklanja zavisnost interneta stvari od centralizovanih servera i pojedinačnih tačaka kvara koje ga izlažu napadačima. Jer su sve transakcije trenutno vidljive u zajedničkoj nepromenljivoj knjizi, aplikacije koje su zasnove na blokčejnu i internetu stvari mogu omogućiti bolju transparentnost u oblastima kao što je na primer lanac snabdevanja.

Implementacija blokčejna i interneta stvari nudi rešenja koja mogu klijentima da obezbedi sigurnu i besprekornu uslugu mobilnosti koja se plaća dok ide (pay-as-you-go). Ne samo da blokčejn pruža osnovu otpornu na neovlašćene radnje koja omogućava bezbedan prenos podataka, već njegova funkcionalnost pametnog ugovora može automatizovati naplatu, čineći iskustvo bezbednijim i pouzdanijim.

Praćenje robe u celom lancu snabdevanja je dugo bio logistički izazov za trgovce, proizvođače i druge učesnike. Međutim, zahvaljujući internetu stvari i senzorima u kamionima za isporuku i pametnim kontejnerima, sada su dostupni podaci u realnom vremenu. Ovo omogućava danonoćno praćenje, kao i da učesnici budu upozoreni kada na bilo šta treba obratiti pažnju, kao što je podešavanje temperature ili održavanje kontejnera. Koristeći blokčejn, preduzeća mogu biti sigurna da sve relevantne strane u lancu snabdevanja mogu pristupiti informacijama koje su im potrebne jer su one obezbeđene u nepromenljivoj, zaštićenoj knjizi. Sve uključene strane mogu da provere istu knjigu, koja deluje kao univerzalni izvor istine, pružajući transparentnost i poboljšavajući odnose između zainteresovanih strana. Potrošači imaju koristi od veće efikasnosti i visokokvalitetnih proizvoda koji se mogu proveriti, a preduzeća ostvaruju uštede eliminišući neispravnu robu, stavke koje nedostaju.

Uređaji interneta stvari su od suštinskog značaja za rast pametnih gradova. Korišćenjem povezanih senzora, svetla i merača, podaci se mogu lako prikupiti i analizirati, obezbeđujući danonoćnu funkcionalnost. Blokčejn, još jednom, pruža savršenu osnovu za ove uređaje interneta stvari, osiguravajući sigurnost, efikasnost i validnost podataka. Blokčejn i internet stvari rešenja za praćenje kvaliteta vazduha u pametnim gradovima brzo dobijaju na snazi. Pošto blokčejn uklanja mogućnost bezbednosnih opasnosti i neefikasnosti u radu, kvalitet vazduha se može efikasno pratiti i korumpirani ili loši akteri nisu u mogućnosti da izmene bilo kakve podatke.

Uređaji interneta stvari već pomažu proizvođačima širom sveta da eliminišu neefikasnost i pobojšaju svoje rezultate. Korišćenjem pametnih senzora na svojoj opremi, preduzeća mogu da smanje skupo vreme zastoja kroz održavanje vođeno mašinama. Senzori prenose podatke o mašinama u realnom vremenu i šalju izveštaje ili poruke o greškama omogućavajući vlasnicima preduzeća da zakažu preventivno održavanje, u suštini rešavajući problem pre nego što se pojavi. Međutim, s obzirom da je napad na uređaje interneta stvari i dalje meta za napadače, ljudska intervencija i provere se i dalje široko koriste kako bi se osiguralo da se izveštaji o greškama ne menjaju ili zamenjuju, blokčejn može pomoći da se ovo eliminiše. Pored toga, sa

podacima raspoređenim na više servera, blokčejn eliminiše pretnju od napada i jednu tačku kvara centralizovanih sistema, uveliko povećavajući pouzdanost interneta stvari dok uklanja potrebu za ručnim procesima i eliminiše ljudske greške. Ovo smanjenje rizika dovodi do značajnih ušteda troškova za preduzeća koju mogu preneti na potrošače.

3. Teorijske osnove istraživanja

3.1. Sistem za otkrivanje upada

Sistem za otkrivanje upada (IDS) je jedan od načina da se obezbedi sigurnost mreže. Njegova uloga je da otkrije upad i manipulacije u mreži negativne prirode. Pored zaštitnog zida (engl. firewall) ovaj sistem će da i pored svog filtiriranja paktera detektovati bilo kakvo neželjeno ponašanje u mreži. Detekcija upadaima za cilj da otkrije entitete koji pokušavaju da naruše postojeće sigurnosne kontrole i protokole. Zaštitni zid, svojim dizajnom radi samo zaštitu i filtiranje krajnih paketa i ulaska odnosno izlaska iz lokalne mreže. Dešavanja unutar mreže su van njegovog opsega. Zato tad čitava nadležnost je pod kontrolom sistema za otkrivanje upada.

Detektovanje upada može da bude unutrašnjih i spoljašnjih. Spoljašnji napadi se mogu izvesti na sledeći način:

- Zaštitni zid poseduje neke slabosti i lako se prolazi kroz njega.
- Korišćenjem slabosti nekog bežičnog protokola.
- Mreža se ne održava redovno i postoje rupe u bezbedonosnim protokolima.

Glavni izazov kod detekcije upada je razdvajanje aktivnosti, koje su očekivane a koje abnormalne. Postoji i mogućnost da sistem za detekciju upada izveštava o lažnim pozitivnim i broj lažnih pozitivnih se smatra jednom od glavnih ocena za dobar rad sistema za detekciju upada..

Funkcije sistema za detekciju upada su sledeće:

- Posmatranje i analiza aktivnosti svih korisnika mreže.
- Proveravanje konfiguracije uređaja i procena ranjivosti.
- Evaluacija integriteta sistema.
- Uočavanje paterna kao na neki od poznatih napada.
- Analiza uzorka abnormalnih aktovnisti.

Sistem za detekciju upada ima sledeće karakteristike:

- Performanse predviđanja
- Osetljivost
- Određenost
- Tačnost

- Stepen detekcije
- Tolerancija na grešku

Pod performansama predviđanja podrazumeva se stepen merenja tačnosti predviđanja. To znači da sistem mora da uspešno detektuje napad i ne detektuje legitimne akcije kao napad. Sistem može da uspešno detektuje napad, nepostojeći napad prijavi kao da postoji, propusti napad ili legitimnu akciju detektuje kao napad. Performanse predviđanja treba da dostave informaciju o tome koliko je sistem uspešan u detektovanju upada, tj. na koliko aktivnosti u mreži je uspešno odgovorio klasifikujući je kao poželjn ili ne.

Osteljivost je broj koji predstavlja koliko je sistem detektovao u odnosu na sve aktivnosti u mreži.

Određenost je broj koji predstavlja koliko je bilo legitimnih aktivnosti u odnosu na sve u mreži.

Tačnost je broj koji prikazuje koliko je bilo ispravnih rezultata.

Stepen detekcije je jedan od najvažnijih karakteristika sistema za detekciju upada. Definiše se kao broj pravih detektovanih napada u odnosu na sve detektovane napade.

Sistem za detekciju upada mora da bude dovoljno robustan i da se lako oporavi od napada. Jedan od glavnih napada koji često ovi sistemi imaju problem da im se suprostave su napadi distribuiranog uskraćivanja usluge gde veliki broj zahteva preplavi mrežu i na taj način sistem za detekciju upada ne može da obradi količinu informacija što dovodi do njegovog pada. Tolerancija na grešku podrazumeva situaciju kao što je ova prethodno opisana, da sistem može da obradi zahteve i pronađe probleme i prijavi tj. detektuje.

Sistemi za detekciju upada se dele na sisteme za detekciju anomalija i na sisteme za detekciju zloupotrebe. Kada sistem radi na osnovu toga što otkriva napade na osnovu paterna ponašanja i poznatih aktivnosti onda govorimo o sistemu koji je klasifikovan kao sistem za detekciju zloupotreba. Kada sistem posmatra aktivnosti u mreži i detektuje neku kao nestandardnu onda se govori o sistemu koji je klasifikovan kao sistem za detekciju anomalija. Sistem za detekciju anomalija monitoriše mrežu i klasifikuje aktivnosti na normalne i neuobičajne. Nepoznata IP adresa, nepoznat port, drugačija veličina paketa od uobičajne i slično su neki od faktora koji dovode do toga da sistem primeti anomaliju u mreži.

Potreba modernih računarskih mreža da imaju sistem za detekciju upada je sve veća. Kad se govori o internetu stvari i pametnim uređajima koji nemaju sopstevene bezbednosne mehanizme, sistemi za detekciju upada postaju sve važniji deo njih. Oni su potrebni da kontrolišu mrežni saobraćaj, primećuju abnormalne aktivnosti i preduzme određene akcije u cilju zaustavljanju napada. Korišćenje veštačke inteligenciju u ovim sistemima postaje sve važnije i stvara se okruženje u kome će moći da se detektuju i najmanje abnormalne aktivnosti koje bi prouzrokovale kasnije i veće napade na mrežu.

3.2. Sistem za sprečavanje upada

Sistem za sprečavanje upada (IPS) je alat za mrežnu bezbednost i prevenciju pretnji. Ideja koja stoji iza prevencije upada je da se stvori preventivni pristup bezbednosti mreže kako bi se potencijalne pretnje mogle identifikovati i na njih brzo reagovati. Sistemi za sprečavanje upada se na taj način koriste za ispitivanje tokova mrežnog saobraćaja u cilju pronalaženja zlonamernih aktivnosti i sprečavanja eksploatacije ranjivosti.

Sistem za sprečavanje upada se koristi za identifikaciju zlonamerne aktivnosti, evidentiranje otkrivenih pretnji, prijavljivanje otkrivenih pretnji i preduzimanje preventivnih mera da spreči pretnji da napravi štetu. Sistem za sprečavanje upada se može koristiti za kontinuirano praćenje mreže u realnom vremenu.

Sistem za sprečavanje upada će raditi tako što će skenirati sav mrežni saobraćaj. Da bi to uradio, IPS će obično biti smešten odmah iza zaštitnog zida, delujući kao dodatni sloj koji će posmatrati događaje u potrazi za zlonamernim sadržajem. Na ovaj način, IPS se postavljaju u direktne komunikacione puteve između sistema i mreže, omogućavajući alatu da analizira mrežni saobraćaj.

Tri uobičajena pristupa za IPS-a za zaštitu mreža su:

- Detekcija zasnovana na potpisima u kojoj IPS koristi prethodno definisane potpise napada poznatih mrežnih pretnji da otkrije pretnje i preduzme akciju
- Detekcija zasnovana na anomalijama u kojoj IPS traži neočekivano ponašanje mreže i blokira pristup hostu ako se otkrije anomalija
- Detekcija zasnovana na polisama u kojoj IPS prvo zahteva od administratora da naprave bezbednosne polise- kada se desi događaj koji krši definisanu bezbednosnu polisu, upozorenje se šalje administratorima sistema

Ako se otkriju bilo kakve pretnje, IPS je obično sposoban da šalje upozorenja administratoru, odbaci sve zlonamerne mrežne pakete i resetuje veze rekonfigurisanjem zaštitnih zidova, prepakivanjem korisnih podataka i uklanjanjem zaraženih priloga sa servera.

Prednosti sistema za sprečavanje upada su sledeći:

- Smanjenje šansi za bezbednosne incidente
- Pružanje dinamičke zaštite od pretnji
- Automatsko obaveštavanje administratora kada se pronađe sumnjiva aktivnost
- Ublažavanje napada kao što su pretnje nultog dana, DoS napadi i DDoS napadi
- Smanjenje održavanja mreža za IT osoblje
- Dozvoljavanje ili odbijanje određenog dolaznog saobraćaja ka mreži

Nedostaci sistema za sprečavanje upada su sledeći:

- Kada sistem blokira abnormalnu aktivnost na mreži pod pretpostavkom da je

zlonamerna, to može biti lažno pozitivno i dovesti do DoS-a za legitimnog korisnika

- Ako organizacija nema dovoljno propusnog opsega i mrežnog kapaciteta, IPS može usporiti sistem
- Ako na mreži postoji više IPS-ova, podaci će morati da prođu kroz svaki da bi došli do krajnjeg korisnika, što će uzrokovati gubitak performansi mreže

3.3. Blokčejn

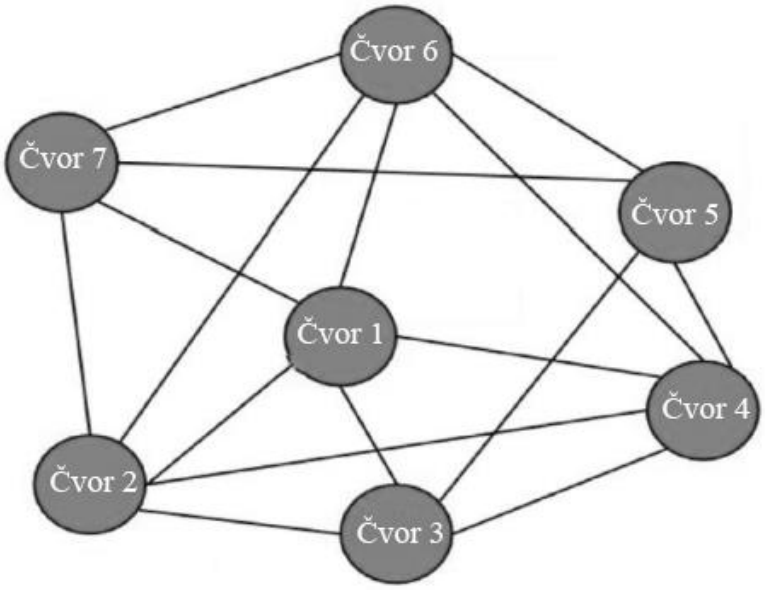
Glavna svrha blokčejna je skladištenje digitalnih informacija i omogućavanje distribucije informacija, ali da te informacije se ne mogu uređivati. Ideju o blockchain tehnologiji su prvi izneli 1991. Stuart Haber i V. Scott Stornetta u svom radu „Kako vremenski označiti digitalni dokument". Oni su želeli da implementiraju sistem u kome vremenske oznake dokumenta ne mogu da se menjaju. Međutim, projekat nije implementiran sve do 2009. godine kada je Santoshi Nakamoto pokrenuo bitkoin projekat izgrađen na blokčejnu. Nakamoto je definisao bitkoin kao sistem elektronskog plaćanja koji je izgrađen na kriptografskom mehanizmu umesto da koristi centralne organe za kontrolu transakcija, omogućavajući na taj način dve strane da direktno međusobno obavljaju transakcije bez potrebe za trećom stranom od poverenja kao što je centralna banka.

Bitkoin je bio prva verzija blokčejna koja je rešila nekoliko problema u finansijskom polju poboljšanjem finansijskih usluga. Kasnije, 2015. godine, ethereum je bio drugi najveći blok lanac koji je implementiran koji je poboljšao skladištenje i rad računarskog koda, omogućavajući pametne ugovore. Iako Stuart Haber i V. Scott Stornetta (1990) nisu spomenuli reč blockchain u njihovom radu, njihova ideja o implementaciji sistema gde vremenske oznake dokumenta ne mogu da se uređuju je ista ideja koja stoji iza blokčejna.

Decentralizovana mreža uključuje čvorove koji se ponašaju i kao klijenti i kao serveri. Ovo implicira da je svaki čvor na mreži nezavisan i da sadrži istu kopiju podataka koje imaju drugi čvorovi na mreži. Decentralizovana baza podataka je strukturirana na način koji omogućava direktan prenos vrednosti na ravnopravnoj osnovi, bez centralnog posrednika od poverenja. Dok, u centralizovanoj mreži, postoje čvorovi koji deluju kao klijenti, a drugi kao serveri. Klijent zahteva uslugu serverskom čvoru. Osnovna ideja iza funkcije decentralizacije u blokčejnu je da je većina čvorova uključena u donošenje odluka. Obrnuto, u centralizovanoj mreži, donošenje odluka donosi samo jedan akter ili mala grupa. Tradicionalno, centralna baza podataka kontroliše tok transakcija, dok u bitkoinu autoritet imaju čvorovi decentralizovane mreže.

Arhitektura blokčejn tehnologije se sastoji od više slojeva. Broj slojeva varira u zavisnosti od primene upotrebe. Svaka aplikacija blokčejna ima različite zahteve što rezultira određenim brojem slojeva. Štaviše, većina istraživača se slaže oko šest zajedničkih slojeva blokčejn tehnologije, uključujući sloj aplikacije, sloj ugovora, sloj podsticaja, sloj konsenzusa, sloj mreže i sloj podataka. Većina istraživača usvaja četiri glavna osnovna sloja blokčejna uključujući sloj aplikacije, sloj konsenzusa, sloj mreže i sloj podataka koji omogućavaju da se izoluju različite prirode bezbednosnih pretnji u blokčejnu. Analiza bezbednosnih pitanja može se sprovesti i klasifikovati na četiri glavna sloja koji čine mrežu. Sloj podataka se bavi

transakcijama i drugim podacima sadržanim u bloku, tj. vremenskom oznakom, privatnim i javnim ključevima, nonce, heš vrednošću zajedno sa transakcijama i to su glavne komponente u mreži. Da bi transakcija tekla u P2P mreži, ovde dolazi mrežni sloj sa uslugama adresiranja, rutiranja i imenovanja. Na kraju, protokol mehanizma konsenzusa koji omogućava čvorovima na P2P mreži da donose odluke i verifikuju transakcije koje treba da obradi sloj konsenzusa.



Slika 2. Izgled komunikacije između blokčejn čvorova u jednoj mreži

Sloj podataka se sastoji od transakcija koje su raspoređene u blokove u decentralizovanoj mreži. Transakcije se sastoje od različitih podataka, odnosno informacija o računu. Pored toga, blokčejn komponente koje obezbeđuju nepromenjivost distribuirane knjige uključuju heš, merkle drvo, vremensku oznaku i šifrovanje.

Mrežni sloj je takođe poznat kao P2P; ovaj sloj obezbeđuje da čvorovi u mreži mogu da komuniciraju i da se sinhronizuju jedni sa drugima kako bi zadržali važeći status distribuirane knjige. Glavna odgovornost ovog sloja je da proveri da li je transakcija koja se šalje validna.

Konsenzus sloj sadrži različite konsenzus algoritme koji se koriste u blokčejnu za donošenje odluka.

Aplikacioni sloj je gornji sloj arhitekture blokčejn tehnologije koji obezbeđuje upotrebu tehnologije od strane krajnjih korisnika.

Blokčejn tehnologije su sastavljene od različitih tehnika, odnosno algoritma, kriptografske matematike i peer-to-peer mreža. Algoritmi konsenzusa su u srž ove tehnologije jer pomažu aktivnim čvorovima na mreži da donose odluke.

Glavni elementi blokčejn tehnologije su:

- Decentralizacija: U tradicionalnom finansijskom sistemu, centralizovani entitet kao što su centralne banke je obavezan da dozvoli i nadgleda transakcije. U blokčejnu, centralizovana strana nije potrebna za obavljanje transakcija. Pouzdanost podataka u blokčejnu se održava konsenzusnim algoritmom.
- Nepromenljivost: Nije moguće uređivati ga kada je dodat u blokčejn sistem. Svaki čvor u mreži ima kopiju distribuirane knjige i pre dodavanja transakcije u blok, većina čvorova u mreži treba da proveri njenu validnost.
- Transparentnost: Pošto je sistem decentralizovan, svi učesnici imaju jednaka prava i svaki učesnik može da vidi i verifikuje transakcije što donosi veću transparentnost u sistemu.
- Anonimnost: Transakcije u blokčejn tehnologijama su anonimne. Jedina stvar koju treba znati u prenosu transakcije je blokčejn adresa osobe. Ovo osigurava robusnost u blokčejnu.
- Distribuirane knjige: Ovo je jedna od važnih karakteristika koja omogućava vlasništvo nad verifikacijom. Kada učesnik koji želi da doda blok u blokčejn, drugi će morati da ga verifikuju i odobre, čime će omogućiti pošteno učešće.
- Konsenzus: Ovo je algoritam koji omogućava čvorovima u mreži da donose odluke, omogućava čvorovima da se brzo dogovore. To je vrsta sistema glasanja gde većina pobeđuje, manjina to mora da podrži. Ovo je jedna od važnih karakteristika koja omogućava da mreža bude nepoverljiva.

3.4. Stvaranje blokova

U blokčejnu, svi važeći zapisi o transakcijama se prikupljaju zajedno i čuvaju u grupama koje se nazivaju blokovi. Blok sadrži određeni broj transakcija zajedno sa dokazom o radu i hešom prethodnog bloka.

Blok se sastoji od dve glavne komponente, zaglavlja bloka i tela koje sadrži listu transakcija.

Slika 3. Izgled jednog bloka

Glava bloka je podeljena na pet komponenti:

- Nonce broj koji se koristi samo jednom je broj koji se dodaje heširanom bloku u lancu blokova tako da ostaje teško da se izbacuje kada se ponovo hešuje.
- Podaci sadrže detaljne informacije o transakciji uključujući podatke o pošiljaocu i primaocu, iznos novca koji se šalje. Ovaj deo je takođe veoma važan za ostale čvorove za validaciju transakcije. Kada čvor započne transakciju i emituje je drugim čvorovima u mreži da bi je potvrdio, drugi čvorovi se zasnivaju na starim transakcijama.
- Koren heš merkle drveta, transakcije u bloku su organizovane u strukturu markle drveta i mogu biti agregirani u heš, a time i heš trenutnog bloka.
- Heš prethodnog bloka, blokovi su kriptografski povezani sa digitalnim otiskom prsta generisanim heš funkcijom. Blokovi su međusobno povezani; blok sadrži heš vrednost prethodnog bloka. Ovo je ključna komponenta koja omogućava vezu između blokova. Ovo je razlog zašto je blokčejn veoma teško preokrenuti..
- Vremenska oznaka u samom bloku. Ovo vreme omogućava da se utvrdi tačno vreme u kome je blok bio napravljen i potvrđen od strane drugih čvorova u sistemu blokčejnu..
- Heš bloka, svaki heš ima veličinu u bitovima i cilj teže je dobiti odgovarajući heš.

3.4.1. Kriptografske heš funkcije u blokčejnu

Heširanje u blokčejnu se odnosi na proces posedovanja ulazne stavke bilo koje dužine koja odražava izlaznu stavku fiksne dužine. Ako uzmemo primer korišćenja blokčejna u kriptovalutama, transakcije različite dužine se pokreću kroz dati algoritam heširanja i sve daju izlaz fiksne dužine. Ovo je bez obzira na dužinu ulazne transakcije. Izlaz je ono što zovemo heš. Dobar primer je Bitcoin-ov bezbedni algoritam heširanja 256 (obično skraćen na SHA-256). Heširanje korišćenjem SHA-256 uvek daje izlazni rezultat fiksne dužine, koji ima dužinu od 256 bita (izlaz je 32 bajta). Ovo je uvek slučaj bilo da je transakcija samo jedna reč ili složena transakcija sa ogromnom količinom podataka. Ovo znači da praćenje transakcije postaje lakše kada možete da se setite/pratite heš. Veličina heš-a će zavisiti od korišćene heš funkcije, ali izlaz koji koristi određeni algoritam heširanja biće određene veličine.

3.4.2. Vrste blokčejna

Postoje četiri vrste blokčejna:

- Javni blok lanci su po prirodi bez dozvole, dozvoljavaju bilo kome da se pridruži i potpuno su decentralizovani. Javni blok lanci omogućavaju svim čvorovima blokčejna da imaju jednaka prava za pristup blok lancu, kreiranje novih blokova podataka i validaciju blokova podataka.

- Privatni blok lanci, koji se takođe mogu nazvati upravljanim lancima blokova, su dozvoljeni blok lanci koje kontroliše jedna organizacija. U privatnom blokčejnu, centralna vlast određuje ko može biti čvor. Centralna vlast takođe ne daje nužno svakom čvoru jednaka prava za obavljanje funkcija. Privatni blok lanci su samo delimično decentralizovani jer je javni pristup ovim blok lancima ograničen.
- Blok lanci konzorcijuma su dozvoljeni blok lanci kojima upravlja grupa organizacija, a ne jedan entitet, kao u slučaju privatnog blockchaina. Konzorcijumski blok lanci, prema tome, uživaju u većoj decentralizaciji od privatnih blokova, što rezultira višim nivoima bezbednosti. Međutim, uspostavljanje konzorcijuma može biti naporan proces jer zahteva saradnju između brojnih organizacija, što predstavlja logističke izazove, kao i potencijalni antimonopolski rizik.

3.4.3. Proof-of-Work

Proof-of-work je algoritam koji obezbeđuje mnoge kriptovalute, uključujući Bitcoin i Ethereum. Većina digitalnih valuta ima centralni entitet ili lidera koji prati svakog korisnika i koliko novca ima. Ali ne postoji takav lider zadužen za kriptovalute kao što je Bitcoin. Tačnije, dokaz o radu rešava „problem dvostruke potrošnje“, koji je teže rešiti bez odgovornog vođe. Ako korisnici mogu duplo da potroše svoj novac čini valutu nepredvidivom i bezvrednom. Dvostruka potrošnja je problem za onlajn transakcije jer je moguće digitalne radnje veoma lako replicirati. Dokaz o radu čini udvostručenje digitalnog novca veoma, veoma teškim. Dokaz o radu je neophodan deo dodavanja novih blokova u blokčejn. Blokove oživljavaju rudari, koji sprovode proveru rada. Mreža prihvata novi blok svaki put kada rudar donese novi dokaz o radu. Pronalaženje dokaza o radu je tako teško da je jedini način da se obezbedi posao koji je rudarima potreban za osvajanje je sa skupim, specijalizovanim računarima. Rudari će zaraditi nagradu ako pogode odgovarajući proračun. Što više proračuna naprave, više nagrada će verovatno zaraditi.

3.4.4. Proof-of-Stake

Proof-of-stake je mehanizam konsenzusa kriptovalute za obradu transakcija i kreiranje novih blokova u blok lancu. Mehanizam konsenzusa je metod za proveru valjanosti unosa u distribuiranu bazu podataka i čuvanje baze podataka bezbednom. Proof-of-stake smanjuje količinu računarskog rada potrebnog za verifikaciju blokova i transakcija koje čuvaju lanac blokova, a samim tim i kriptovalutu, bezbednim. Proof-of-stake menja način na koji se blokovi verifikuje. Vlasnici nude svoje novčiće kao zalog za priliku da validiraju blokove. Vlasnici novčića sa uloženim novčićima postaju „validatori“. Validatori se zatim nasumično biraju za validaciju bloka. Ovaj sistem nasumično bira ko stigne da „rudari“ umesto da koristi mehanizam zasnovan na takmičenju kao što je dokaz o radu.

Proof-of-stake je dizajniran da smanji probleme skalabilnosti i ekološke održivosti u vezi sa protokolom proof-of-work. Dokaz o radu je konkurentski pristup verifikaciji transakcija, koji prirodno podstiče ljude da traže načine da steknu prednost, posebno zato što je u pitanju novčana vrednost.

3.4.5. Delegirani proof-of-stake

Delegirani proof-of-stake je popularna evolucija proof-of-stake koncepta, pri čemu korisnici mreže glasaju i biraju delegate za validaciju sledećeg bloka. Delegati se takođe nazivaju svedoci ili producenti blokova. Koristeći delegiranog proof-of-stake-a, može se glasati za delegate tako što će objediniti svoje tokene u skup za ulaganje i povezati ih sa određenim delegatom. Ne prenose se fizički tokeni u drugi novčanik, već umesto toga koristi se provajdera usluge zalaganja da se ulože tokeni u skup za ulaganje.

Ograničen broj delegata (većina protokola bira između 20 i 100) se bira za svaki novi blok, tako da delegati jednog bloka možda neće biti delegati sledećeg. Izabrani delegati dobijaju naknade za transakcije od validiranog bloka, a ta nagrada se zatim deli sa korisnicima koji su svoje tokene udružili u skup uspešnog delegata. Nagrade se dele na osnovu uloga svakog korisnika.

3.4.6. Pametni ugovori

Pametni ugovor je ugovor koji se samostalno izvršava, a uslovi ugovora između kupca i prodavca su direktno upisani u kodu. Kod i ugovori sadržani u njemu postoje širom distribuirane, decentralizovane blokčejn mreže. Kod kontroliše izvršenje, a transakcije se mogu pratiti i nepovratne su.

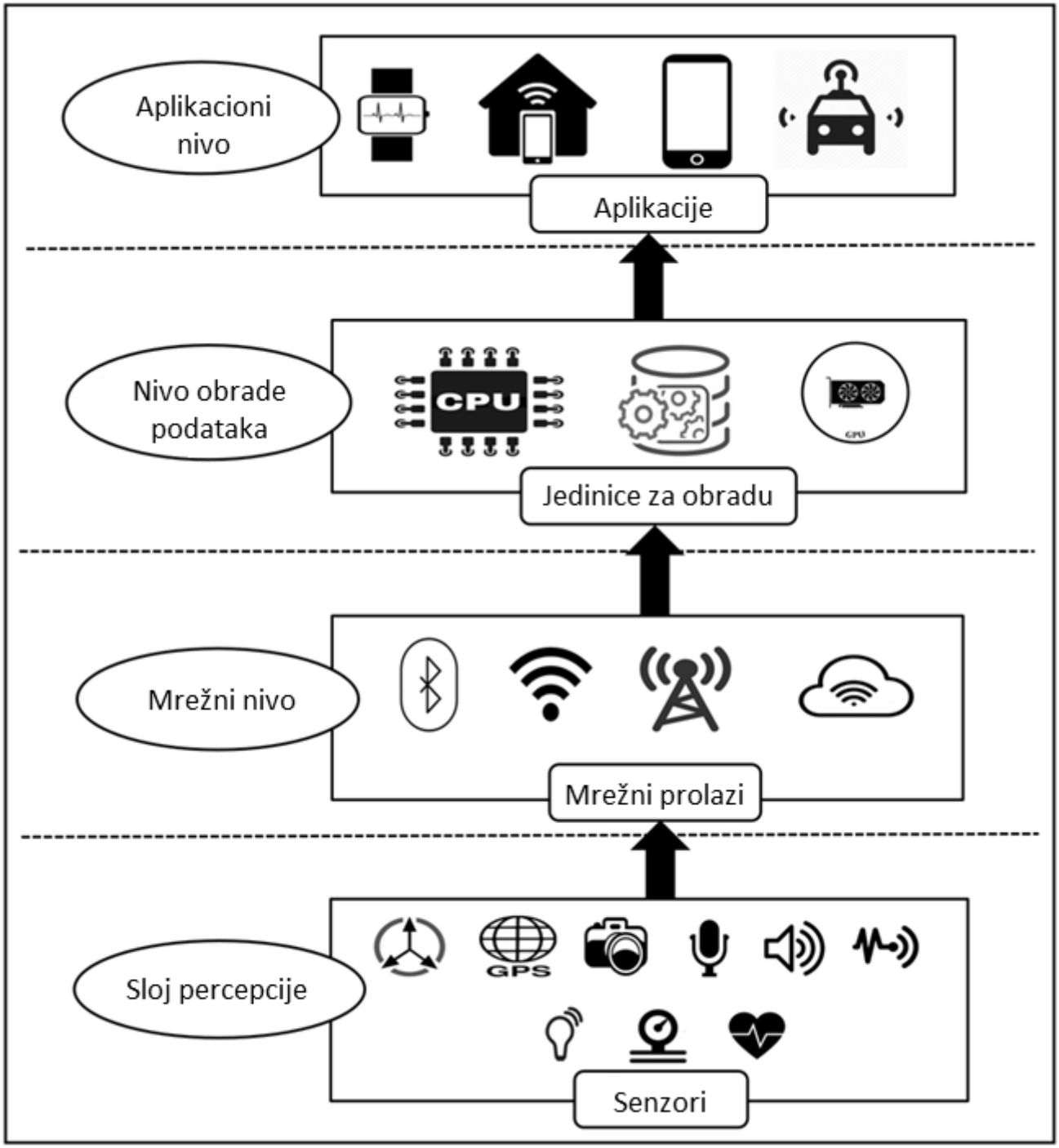
Pametni ugovori dozvoljavaju da se transakcije i sporazumi od poverenja sprovode između različitih, anonimnih strana bez potrebe za centralnim organom, pravnim sistemom ili spoljnim mehanizmom za sprovođenje.

Pametne ugovore je 1994. godine prvi predložio Nik Sabo, američki naučnik koji je izumeo virtuelnu valutu pod nazivom „Bit Gold“ 1998. godine, punih 10 godina pre pronalaska bitkoina.

Sabo je definisao pametne ugovore kao kompjuterizovane transakcijske protokole koji izvršavaju uslove ugovora. Želeo je da proširi funkcionalnost metoda elektronskih transakcija, kao što je POS (prodajno mesto), na digitalno područje.

3.5. Internet stvari

Internet stvari, ili IoT, je sistem međusobno povezanih računarskih uređaja, mehaničkih i digitalnih mašina, objekata, životinja ili ljudi koji imaju jedinstvene identifikatore (UID) i mogućnost prenosa podataka preko mreže bez potrebe za ljudima interakcija čoveka ili čoveka-računara. Stvar na internetu stvari može biti osoba sa implantom za monitor srca, životinja sa farme sa transponderom biočipa, automobil koji ima ugrađene senzore koji upozoravaju vozača kada je pritisak u gumama nizak ili bilo koja druga prirodna ili veštačka ruka objekat kome se može dodeliti adresa internet protokola (IP) i koji može da prenosi podatke preko mreže. Organizacije u različitim industrijama sve više koriste internet stvari za efikasnije poslovanje, bolje razumevanje klijenata kako bi pružile poboljšanu uslugu korisnicima, poboljšale donošenje odluka i povećale vrednost poslovanja.



Slika 4. Arhitektura interneta stvari sa komponentama

Ekosistem interneta stvari se sastoji od pametnih uređaja omogućenih za veb koji koriste ugrađene sisteme, kao što su procesori, senzori i komunikacioni hardver, za prikupljanje, slanje i delovanje na podatke koje dobijaju iz svog okruženja. Uređaji interneta stvari dele podatke senzora koje prikupljaju povezivanjem na gateway ili drugi uređaj gde se podaci ili šalju u oblak da bi se analizirali ili analizirali lokalno. Ponekad ovi uređaji komuniciraju sa drugim povezanim uređajima i deluju na osnovu informacija koje dobijaju jedan od drugog. Uređaji obavljaju većinu posla bez ljudske intervencije, iako ljudi mogu da komuniciraju sa uređajima - na primer, da ih podese, daju im uputstva ili pristupe podacima. Povezivanje, umrežavanje i komunikacioni protokoli koji se koriste sa ovim uređajima sa omogućenim vebom u velikoj

meri zavise od specifičnih aplikacija koje se primenjuju.

Internet stvari pomaže ljudima da žive i rade pametnije, kao i da steknu potpunu kontrolu nad svojim životima. Osim što nudi uređaje za automatizaciju domova, internet stvari je od suštinskog značaja za poslovanje. Internet stvari pruža preduzećima uvid u realnom vremenu kako njihovi sistemi zaista funkcionišu, pružajući uvid u sve, od performansi mašina do lanca snabdevanja i logističkih operacija.

Internet stvari omogućava kompanijama da automatizuju procese i smanje troškove rada. Takođe smanjuje otpad i poboljšava isporuku usluga, čineći jeftinijim proizvodnju i isporuku robe, kao i nudi transparentnost u transakcijama kupaca.

Kao takav, internet stvari je jedna od najvažnijih tehnologija svakodnevnog života, i nastaviće da raste kako sve više preduzeća bude shvatilo potencijal povezanih uređaja da ih održi konkurentnim.

Internet stvari nudi nekoliko prednosti organizacijama. Neke prednosti su specifične za industriju, a neke su primenljive u više industrija.

Neke od uobičajenih prednosti internet stvari omogućavaju preduzećima da:

- prate njihove ukupne poslovne procese
- poboljšati korisničko iskustvo (CKS)
- uštede vreme i novac
- poboljšati produktivnost zaposlenih
- integrišu i prilagođavaju poslovne modele
- donosi bolje poslovne odluke
- generišu više prihoda
- Internet stvari podstiče kompanije da preispitaju načine na koje pristupaju svom poslovanju i daje im alate za poboljšanje svojih poslovnih strategija

Generalno, internet stvari je najzastupljeniji u proizvodnim, transportnim i komunalnim organizacijama, koristeći senzore i druge uređaje; međutim, takođe je pronašao slučajeve upotrebe za organizacije u poljoprivredi, infrastrukturi i industriji kućne automatizacije, vodeći neke organizacije ka digitalnoj transformaciji.

Internet stvari može koristiti poljoprivrednicima u poljoprivredi tako što će im olakšati posao. Senzori mogu prikupljati podatke o padavinama, vlažnosti, temperaturi i sadržaju zemljišta, kao i drugim faktorima koji bi pomogli u automatizaciji poljoprivrednih tehnika.

Sposobnost praćenja operacija koje okružuju infrastrukturu takođe je faktor sa kojim internet stvari može pomoći. Senzori, na primer, mogu da se koriste za praćenje događaja ili promena unutar strukturalnih zgrada, mostova i druge infrastrukture. Ovo donosi prednosti sa sobom, kao što su ušteda troškova, uštađeno vreme, promene kvaliteta života i tok rada bez papira.

Preduzeće za kućnu automatizaciju može da koristi IoT za nadgledanje i manipulaciju mehaničkim i električnim sistemima u zgradi. U širem smislu, pametni gradovi mogu pomoći građanima da smanje otpad i potrošnju energije.

Internet stvari dotiče svaku industriju, uključujući preduzeća u zdravstvu, finansijama, maloprodaji i proizvodnji.

Postoji nekoliko novih standarda, uključujući sledeće:

- IPv6 preko bežičnih ličnih mreža male snage (6LoVPAN) je otvoreni standard definisan od strane Internet Engineering Task Force (IETF). Standard 6LoVPAN omogućava bilo kom radiju male snage da komunicira sa internetom, uključujući 804.15.4, Bluetooth Low Energy (BLE) i Z-Wave (za kućnu automatizaciju).
- ZigBee je bežična mreža male snage i niske brzine prenosa podataka koja se koristi uglavnom u industrijskim okruženjima. ZigBee je zasnovan na standardu Instituta za elektrotehniku i elektroniku (IEEE) 802.15.4. ZigBee Alliance je stvorila Dotdot, univerzalni jezik za IoT koji omogućava pametnim objektima da bezbedno rade na bilo kojoj mreži i da se međusobno razumeju.
- LiteOS je operativni sistem (OS) sličan Unix-u za bežične senzorske mreže. LiteOS podržava pametne telefone, nosive uređaje, aplikacije za inteligentnu proizvodnju, pametne kuće i internet vozila (IoV). LiteOS takođe služi kao platforma za razvoj pametnih uređaja.
- OneM2M je servisni sloj od mašine do mašine koji se može ugraditi u softver i hardver za povezivanje uređaja. Globalno telo za standardizaciju, OneM2M, stvoreno je da razvije standarde za višekratnu upotrebu kako bi se omogućila komunikacija IoT aplikacijama u različitim vertikalama.
- Data Distribution Service (DDS) je razvijena od strane Object Management Group (OMG) i predstavlja standard interneta stvari za skalabilnu M2M komunikaciju u realnom vremenu i visokih performansi.
- Advanced Message Queuing Protocol (AMKP) je objavljeni standard otvorenog koda za asinhrono slanje poruka putem žice. AMKP omogućava šifrovanu i interoperabilnu razmenu poruka između organizacija i aplikacija. Protokol se koristi u razmjeni poruka klijent-server i u upravljanju uređajima.
- Constrained Application Protocol (CoAP) je protokol koji je dizajnirao IETF koji određuje kako uređaji male snage, ograničeni na računare, mogu funkcionisati u Internetu stvari.
- Mreža širokog dometa (LoRaWAN) je protokol za WAN mreže dizajniran da podrži ogromne mreže, kao što su pametni gradovi, sa milionima uređaja male snage.

Okviri interneta stvari uključuju sledeće:

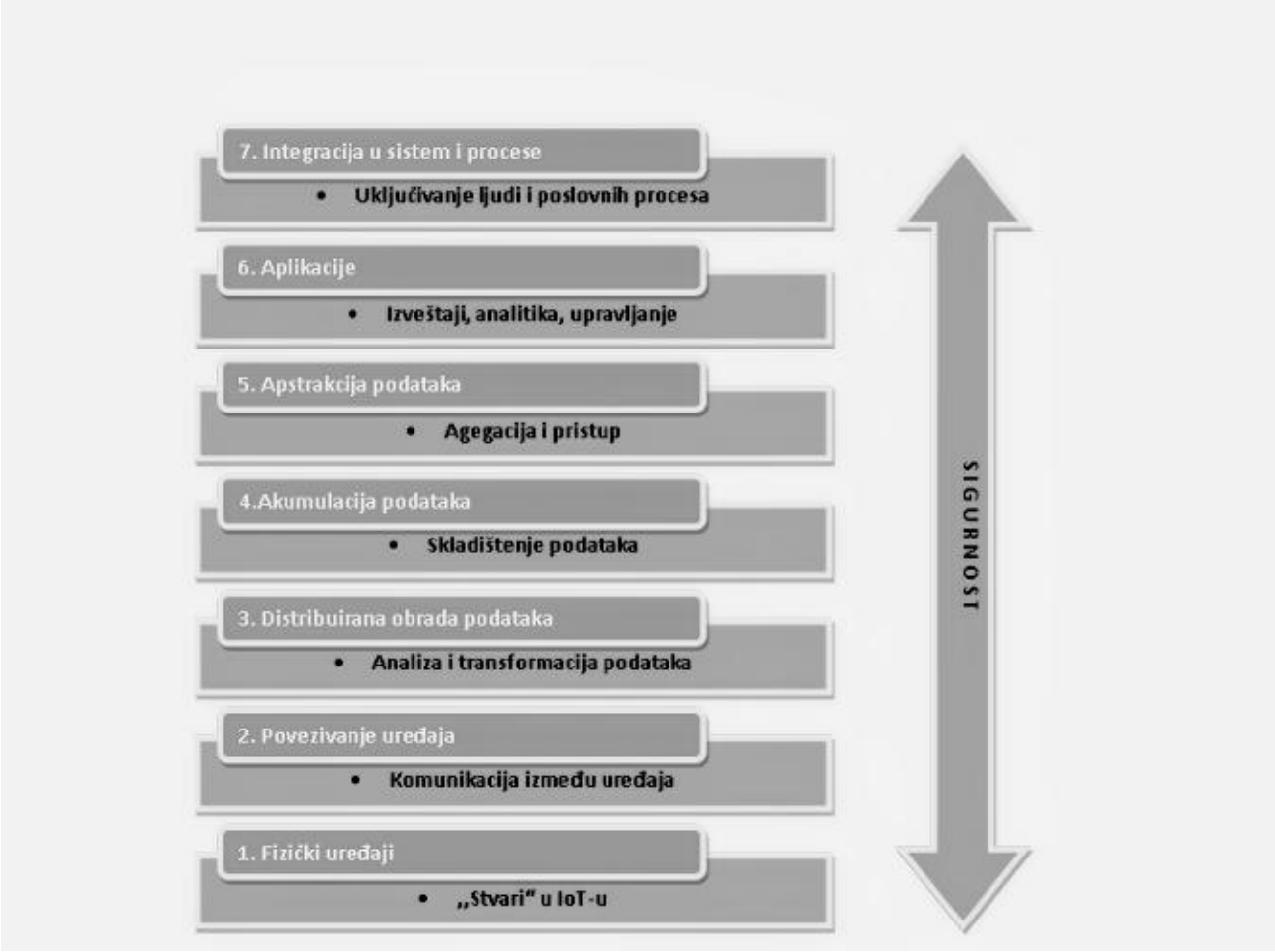
- Amazon Web Services (AWS) je platforma za računarstvo u oblaku za internet stvari koju je objavio Amazon. Ovaj okvir je dizajniran da omogući uređajima da se lako povežu i bezbedno komuniciraju sa AWS oblakom i drugim povezanim uređajima.
- Arm Mbed IoT je platforma za razvoj aplikacija za internet stvari zasnovana na Arm mikrokontrolerima. Cilj Arm Mbed IoT platforme je da obezbedi skalabilno, povezano i bezbedno okruženje za uređaje interneta stvari sa integracijom Mbed alata i usluga.
- Microsoft-ov Azure IoT Suite je platforma koja se sastoji od skupa usluga koje omogućavaju korisnicima da komuniciraju i primaju podatke sa svojih uređaja, kao i da obavljaju različite operacije nad podacima, kao što su višedimenzionalna analiza, transformacija i agregacija, i vizualizuju te operacije na način koji je pogodan za poslovanje.
- Google-ov Brillo/Weave je platforma za brzu implementaciju aplikacija interneta stvari. Platforma se sastoji od dve glavne okosnice: Brillo, operativni sistem zasnovan na Androidu za razvoj ugrađenih uređaja male potrošnje, i Weave, komunikacioni protokol orijentisan ka internetu stvari koji služi kao jezik komunikacije između uređaja i oblaka.
- Calvin je platforma interneta stvari otvorenog koda koju je objavio Ericsson dizajniran za izgradnju i upravljanje distribuiranim aplikacijama koje omogućavaju uređajima da razgovaraju jedni sa drugima. Calvin uključuje razvojni okvir za programere aplikacija, kao i runtime okruženje za rukovanje pokrenutom aplikacijom.

Internet stvari je danas zastupljen u svim sferama našeg života. Jedan od glavnih problema u međusobnoj komunikaciji između pametnih uređaja je standardizacija. Pored toga što različiti uređaji interneta stvari imaju različite tehničke specifikacije i različite oblasti primene potrebno je da postoji zajednička osnova za sve njih u domenu komunikacije i saradnje. Zajedničko za uređaje interneta stvari treba da bude referentna arhitektura. Ona je definisana u konceptu definicija na koji način može da bude razvijana. Trenutno ne postoji jedan prihvaćen model, ali zato postoji nekoliko različitih modela kao što su Ciskov, ARM-a i p2413 model.

3.5.1. Ciskov referentni model

Jedan od najviše raspostranjenih modela je Ciskov referentni model. U primeni je jos od 2014. godine. Sastoji se iz sedam nivoa. Najniži nivo se smatra fizičkim novoom. U njega ulaze elementi kao što su senzori, aktuatori, fizički elementi interneta stvari. Iznad njega se nalazi nivo veze zadužen za komunikaciju i podatke. Potom nivo zadužen za obradu podataka kao i njihovu transformaciju u neophodan format za dalje razumevanje. Skladištenje podataka se nalazi na sledećem nivou. Peti nivo je nivo zadužen za agregaciju i apstrakciju kao i pristup podacima. Pretposlednji nivo je nivo aplikacije koji je zadužen za updavljanje podacima. Poslednji nivo je sedmi i on uključuje spolje faktore kao što su drugi uređaji i ljudi da imaju

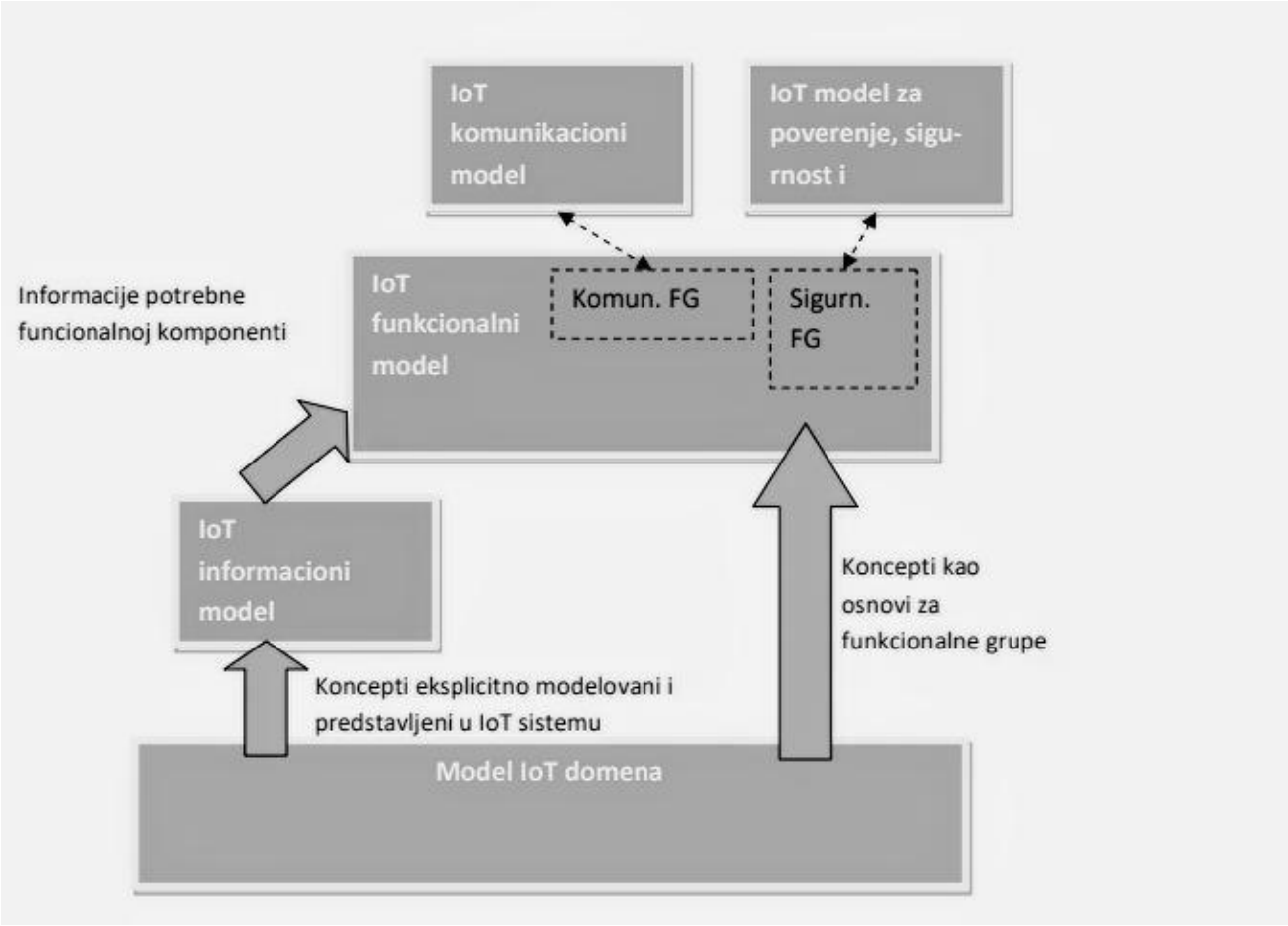
pristup podacima koje uređaj dostavlja.



Slika 5. Arhitektura Ciskovog referentnog modela

3.5.2. ARM-A referentni model

Referentni model interneta stvari ima za cilj uspostavljanje zajedničke osnove i zajedničkog jezika za arhitekture i sisteme. Sastoji se od podmodela. Osnova referentnog modela interneta stvari je model domena, koji uvodi glavne koncepte interneta stvari kao što su uređaji, usluge i virtuelni entiteti, a takođe uvodi odnose između ovih koncepata. Nivo apstrakcije IoT Domain Modela je izabran na takav način da su njegovi koncepti nezavisni od specifičnih tehnologija i slučajeva upotrebe. Ideja je da se ne očekuje da će se ovi koncepti mnogo promeniti u narednim decenijama ili duže.



Slika 6. ARM-A referentni model

Na osnovu modela IoT domena, razvijen je IoT informacioni model. On definiše strukturu informacija povezanih sa IoT-om u sistemu IoT-a na konceptualnom nivou bez razmatranja kako bi one bile predstavljene. Modeliraju se informacije koje se odnose na te koncepte modela IoT domena, koji se eksplicitno prikupljaju, čuvaju i obrađuju u IoT sistemu, npr. informacije o uređajima, IoT uslugama i virtuelnim entitetima.

Funkcionalni model IoT-a identifikuje grupe funkcionalnosti, od kojih je većina zasnovana na ključnim konceptima modela domena IoT-a. Jedan broj ovih funkcionalnih grupa se nadovezuje jedna na drugu, prateći odnose identifikovane u modelu domena interneta stvari. Grupe funkcionalnosti obezbeđuju funkcionalnosti za interakciju sa instancama ovih koncepata ili upravljanje informacijama u vezi sa konceptima, npr. informacije o virtuelnim entitetima ili opisi IoT usluga. Funkcionalnosti funkcionalnih grupa koje upravljaju informacijama koriste IoT informacioni model kao osnovu za strukturiranje svojih informacija.

Ključna funkcionalnost u bilo kom distribuiranom računarskom sistemu je komunikacija između različitih komponenti. Jedna od karakteristika IoT sistema je često heterogenost korišćenih komunikacionih tehnologija, što je često direktan odraz složenih potreba koje takvi sistemi moraju da zadovolje. IoT komunikacioni model uvodi koncepte za rukovanje složenošću komunikacije u heterogenim IoT okruženjima. Komunikacija takođe čini jednu funkcionalnu grupu u IoT funkcionalnom modelu.

3.5.3. P2413 model

P2413 je rezultat višegodišnje serije IoT standarda. P2413 je pokrenut kroz uputstva IEEE-a Industrijski strateški IoT tima sa fokusom na integraciju tržišta potrebe sa razvojem IoT tehnologije.

Internet stvari je ključni pokretač za mnoge nove i buduće „pametne" aplikacije i promene tehnologije u različitim tehnološka tržišta. Ovo se kreće od povezanog potrošača do Smart Home & Buildings, E-Health, Smart Grids, Nekt Generation Manufacturing i pametni gradovi. Stoga je predviđa da će postati jedan od najznačajnijih pokretača rast na ovim tržištima.

P2413 standard definiše arhitektonski okvir za IoT, uključujući opise različitih IoT domena, definicije IoT-a apstrakcije domena i identifikacija zajedničkih karakteristika između različitih IoT domena.

Arhitektonski okvir za IoT obezbeđuje:

- referentni model koji definiše odnose između različitih IoT-a domene (npr. transport, zdravstvena zaštita, itd.) i zajedničke elementi arhitekture
- nadovezuje se na referentni model
- definiše osnovne arhitektonske blokove i njihovu sposobnost da se integrišu u višeslojne sisteme
- bavi se kako dokumentovati i ublažiti divergenciju u arhitekturi
- plan za apstrakciju podataka i kvalitet "četvorostruko" veruju u to uključuje zaštitu, bezbednost, privatnost i bezbednost

Identifikujte zajedničke karakteristike unutar vertikala i potencijalno među određenim vertikale. Rešavanje odnosa između bezbednosnih zahteva, energetske efikasnosti tokom prenos podataka (komunikacija), zahtevi usluge, primena svesno rutiranje (uključujući bezbednosne zahteve), naspram osnovnog mrežne tehnologije.

Arhitektura je definisana ISO/IEC/IEEE 42010 Systems and software engineering - Architecture description (2011) standardom.

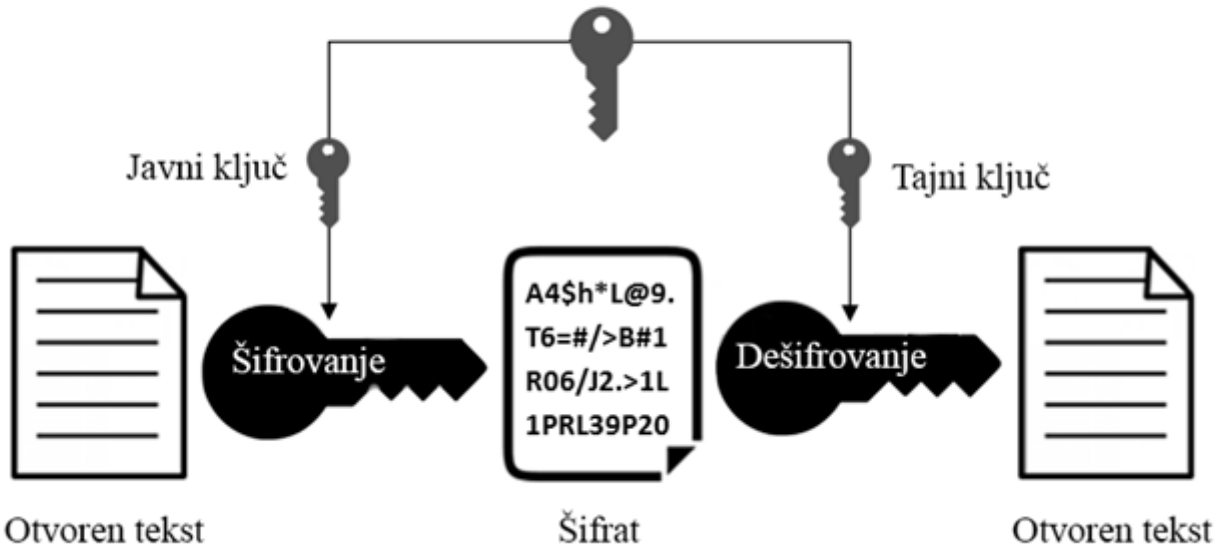
3.6. Simetrična kriptografija

Simetrično šifrovanje je široko korišćena tehnika šifrovanja podataka pri čemu se podaci šifruju i dešifruju korišćenjem jednog, tajnog simetričnog kriptografskog ključa.

Konkretno, ključ se koristi za šifrovanje otvorenog teksta - stanja pre šifrovanja ili post-dešifrovanja podataka - i dešifrovanje šifrovanog teksta - stanja podataka nakon šifrovanja ili pre dešifrovanja.

Simetrično šifrovanje je jedna od najčešće korišćenih tehnika šifrovanja, a takođe i jedna od

najstarijih, koja datira još iz vremena Rimskog carstva. Cezarova šifra, nazvana po nikom drugom nego po Juliju Cezaru, koji ju je koristio za šifrovanje svoje vojne prepiske, čuven je istorijski primer simetričnog šifrovanja u praksi.



Slika 7. Način rada simetričnih algoritama za šifrovanje

Cilj simetričnog šifrovanja je da obezbedi osetljive, tajne ili poverljive informacije. Svakodnevno se koristi u mnogim velikim industrijama, uključujući odbranu, vazduhoplovstvo, bankarstvo, zdravstvenu zaštitu i druge industrije u kojima je obezbeđivanje osetljivih podataka neke osobe, preduzeća ili organizacije od najveće važnosti.

Simetrično šifrovanje funkcioniše korišćenjem šifre toka ili blok šifre za šifrovanje i dešifrovanje podataka. Šifra konvertuje otvoreni tekst u šifrovani tekst jedan po bajt, a blok šifra konvertuje cele jedinice ili blokove otvorenog teksta koristeći unapred određenu dužinu ključa, kao što je 128, 192 ili 256 bita.

Pošiljaoci i primaoci koji koriste simetričnu enkripciju za prenos podataka jedni drugima moraju znati tajni ključ kako bi, u slučaju pošiljalaca, šifrovali podatke koje nameravaju da podele sa primaocima, a u slučaju primalaca, dešifrovali i pročitali šifrovane podatke koje pošiljaoci podelite sa njima, kao i šifrujte sve neophodne odgovore.

Popularni primeri simetričnog šifrovanja uključuju:

- Standard šifrovanja podataka (DES)
- Trostruko standardno šifrovanje podataka (trostruki DES)
- Napredni standard šifrovanja (AES)
- IDEA

- TLS/SSL protokol

AES šifrovanje, koje koristi blok šifre od 128, 192 ili 256 bita za šifrovanje i dešifrovanje podataka, jedna je od najpoznatijih i najefikasnijih tehnika simetričnog šifrovanja koja se danas koristi. Trebale bi milijarde godina da se probije, i zato se koristi za obezbeđenje osetljivih, tajnih ili poverljivih informacija u vladi, zdravstvu, bankarstvu i drugim industrijama. Bezbedniji je od DES i Triple DES.

Nacionalni institut za standarde i tehnologiju (NIST) sada smatra da je DES enkripcija stari algoritam simetričnog šifrovanja jer je dugo bila neefikasna u zaštiti osetljivih informacija od napada grubom silom. U stvari, NIST je u potpunosti povukao standard, a njegov sigurniji brat, Triple DES enkripcija, imaće istu sudbinu. Iako se i danas koristi, Triple DES enkripcija je povučena i zabranjena od strane NIST-a 2023. zbog rastućih bezbednosnih zabrinutosti.

IDEA enkripcija je razvijena kao zamena za DES 1990-ih, ali se AES na kraju smatrao sigurnijim. IDEA je sada otvoren i besplatan algoritam blok-šifriranja, tako da ga svako može koristiti, ali se generalno smatra da je zastareo i neefikasan u obezbeđivanju osetljivih i strogo poverljivih informacija danas. AES enkripcija je zlatni standard za obe svrhe.

Sigurnost transportnog sloja (TLS), kao i njegov prethodnik, Secure Sockets Laier (SSL), koristi simetrično šifrovanje. U osnovi, kada klijent pristupi serveru, generišu se jedinstveni simetrični ključevi, koji se nazivaju ključevi sesije. Ovi ključevi sesije se koriste za šifrovanje i dešifrovanje podataka koje dele klijent i server u toj specifičnoj sesiji klijent-server u tom određenom trenutku. Nova sesija klijent-server bi generisala nove, jedinstvene ključeve sesije.

TLS/SSL koristi ne samo simetrično šifrovanje, već i simetrično i asimetrično šifrovanje, kako bi se osigurala bezbednost klijent-server sesija i informacija koje se razmenjuju u njima.

Simetrična enkripcija se danas koristi jer može brzo da šifruje i dešifruje velike količine podataka i lako se primenjuje. Jednostavan je za upotrebu, a njegova AES iteracija je jedan od najbezbednijih dostupnih oblika šifrovanja podataka.

Sada, simetrično šifrovanje ima nekoliko prednosti u odnosu na asimetrično.

Neke prednosti simetrične enkripcije uključuju:

- Bezbednost: algoritmima simetričnog šifrovanja kao što je AES potrebne su milijarde godina da se razbiju korišćenjem brute-force napada.
- Brzina: simetrično šifrovanje se, zbog kraće dužine ključa i relativne jednostavnosti u poređenju sa asimetričnim šifrovanjem, mnogo brže izvršava.
- Usvajanje i prihvatanje u industriji: simetrični algoritmi za šifrovanje kao što je AES postali su standard šifrovanja podataka zbog svoje prednosti u pogledu bezbednosti i brzine, i kao takvi, uživali su decenijama usvajanja i prihvatanja u industriji.

3.6.1. AES

Popularniji i široko prihvaćeni algoritam simetričnog šifrovanja na koji se danas može sresti je Advanced Encription Standard (AES). Nalazi se najmanje šest puta brže od trostrukog DES-a.

Zamena za DES je bila potrebna jer je njegov ključ bio premali. Sa povećanjem računarske snage, smatralo se ranjivim na napad iscrpnog pretraživanja ključa. Trostruki DES je dizajniran da prevaziđe ovaj nedostatak, ali je utvrđeno da je spor.

Karakteristike AES-a su sledeće:

- Simetrična blok šifra simetričnog ključa
- 128-bitni podaci, 128/192/256-bitni ključevi
- Jači i brži od Trostrukog-DES-a
- Postoji punu specifikaciju i detalji dizajna
- Softver implementiran u skoro svim modernim programskim jezicima

AES je iterativna, a ne Feistelova šifra. Zasniva se na „mreži supstitucije-permutacije“. Sastoji se od niza povezanih operacija, od kojih neke uključuju zamenu ulaza specifičnim izlazima (zamene), a druge uključuju mešanje bitova okolo (permutacije).

Zanimljivo je da AES sve svoje proračune izvodi na bajtovima, a ne na bitovima. Dakle, AES tretira 128 bita bloka otvorenog teksta kao 16 bajtova. Ovih 16 bajtova su raspoređeni u četiri kolone i četiri reda za obradu kao matrica −

Za razliku od DES-a, broj rundi u AES-u je promenljiv i zavisi od dužine ključa. AES koristi 10 rundi za 128-bitne ključeve, 12 rundi za 192-bitne ključeve i 14 rundi za 256-bitne ključeve. Svaki od ovih krugova koristi drugačiji 128-bitni kružni ključ, koji se izračunava iz originalnog AES ključa.

Svaki krug šifrovanja se sastoji od četiri podprocesa.

Zamena bajtova, 16 ulaznih bajtova se zamenjuju traženjem fiksne tabele (S-bok) date u dizajnu. Rezultat je u matrici od četiri reda i četiri kolone.

Shiftrows, Svaki od četiri reda matrice je pomeren ulevo. Svi unosi koji „otpadnu“ ponovo se ubacuju na desnu stranu reda. Promena se vrši na sledeći način:

- Prvi red se ne pomera.
- Drugi red je pomeren za jednu (bajt) poziciju ulevo.
- Treći red je pomeren za dve pozicije ulevo.
- Četvrti red se pomera tri pozicije ulevo.

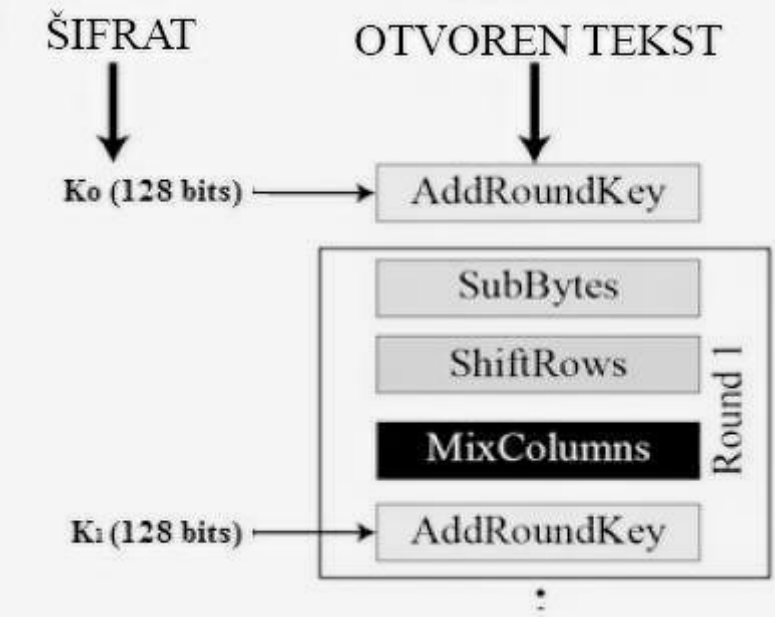
Rezultat je nova matrica koja se sastoji od istih 16 bajtova ali pomerenih jedan u odnosu na drugu.

MixColumns, svaka kolona od četiri bajta se sada transformiše pomoću posebne matematičke funkcije. Ova funkcija uzima kao ulaz četiri bajta jedne kolone i daje četiri potpuno nova bajta, koji zamenjuju originalnu kolonu. Rezultat je još jedna nova matrica koja se sastoji od 16 novih bajtova. Treba napomenuti da se ovaj korak ne izvodi u poslednjem kolu.

AddRoundKey, 16 bajtova matrice se sada smatraju 128 bitova i XOR se stavljaju na 128 bita ključa. Ako je ovo poslednja runda, onda je izlaz šifrovani tekst. U suprotnom, rezultujućih 128 bitova se tumači kao 16 bajtova i počinjemo još jedan sličan krug.

Proces dešifrovanja AES šifrovanog teksta je sličan procesu šifrovanja u obrnutom redosledu. Svaki krug se sastoji od četiri procesa koji se vode obrnutim redosledom:

- Dodajte ključ
- Mešajte kolone
- Shiftrows
- Zamena bajtova



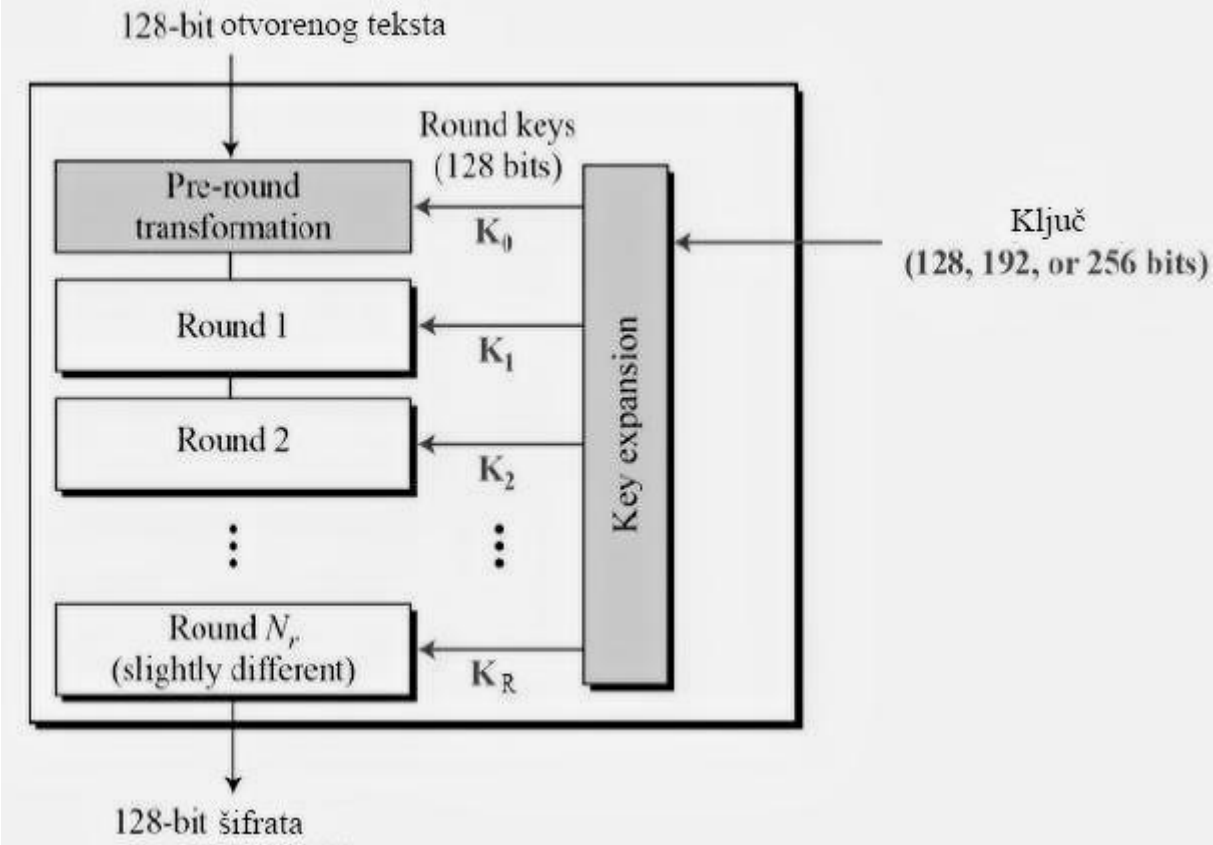
Slika 8. AES šifrovanje - Prva runda

Pošto su podprocesi u svakoj rundi obrnuti, za razliku od Fejstelove šifre, algoritmi za šifrovanje i dešifrovanje treba da se implementiraju odvojeno, iako su veoma blisko povezani.

U današnjoj kriptografiji, AES je široko prihvaćen i podržan i u hardveru i u softveru. Do danas nisu otkriveni praktični kriptoanalitički napadi na AES. Pored toga, AES ima ugrađenu

fleksibilnost dužine ključa, što omogućava određeni stepen „zaštićenosti od budućnosti“ u odnosu na napredak u mogućnosti da se izvrši iscrpna pretraga ključeva.

Međutim, kao i za DES, sigurnost AES-a je osigurana samo ako je pravilno implementirana i ako se koristi dobro upravljanje ključevima.

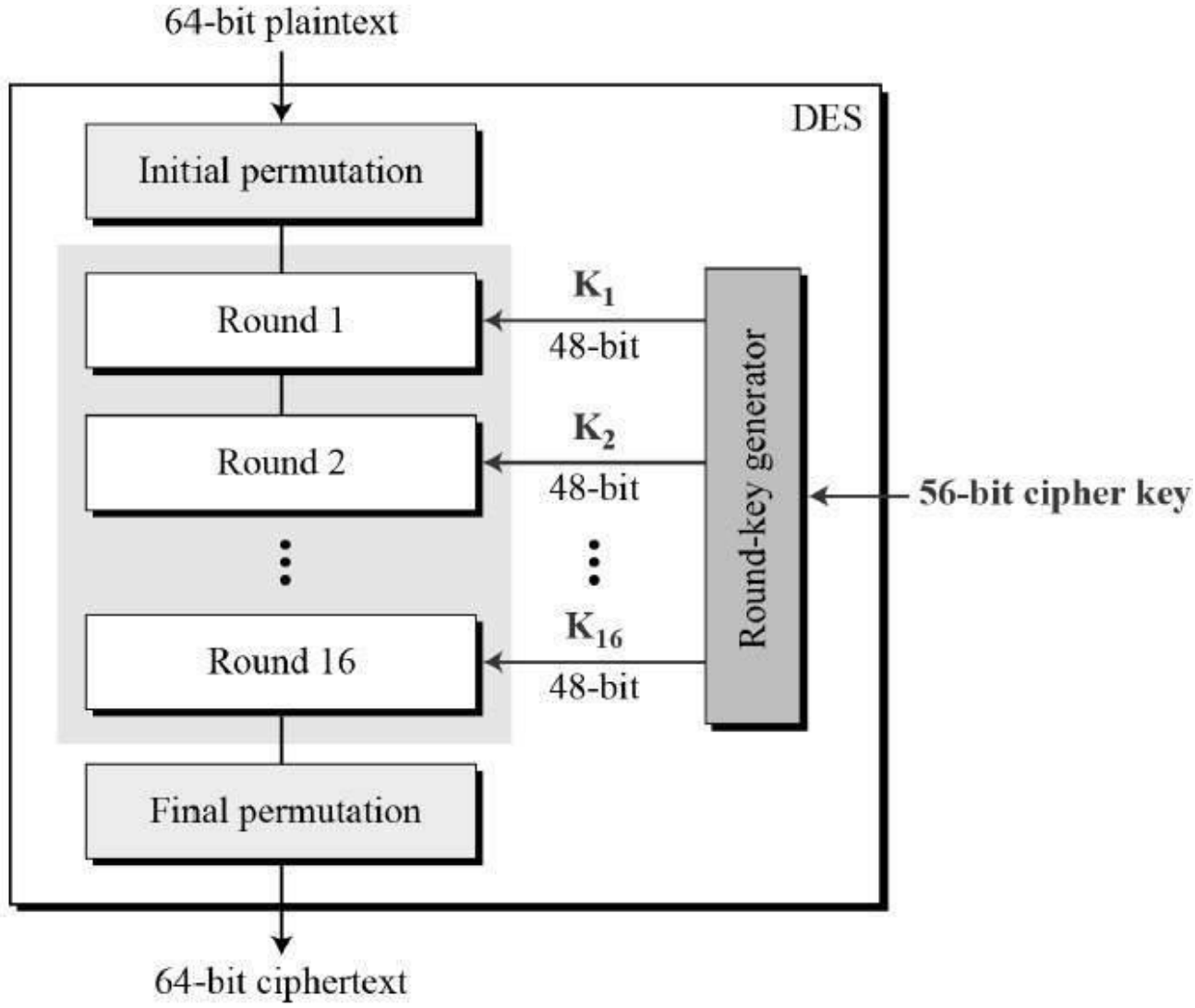


Slika 9. Struktura AES Algoritma

3.6.2. DES

Standard za šifrovanje podataka (DES) je blok šifra sa simetričnim ključem koju je objavio Nacionalni institut za standarde i tehnologiju (NIST).

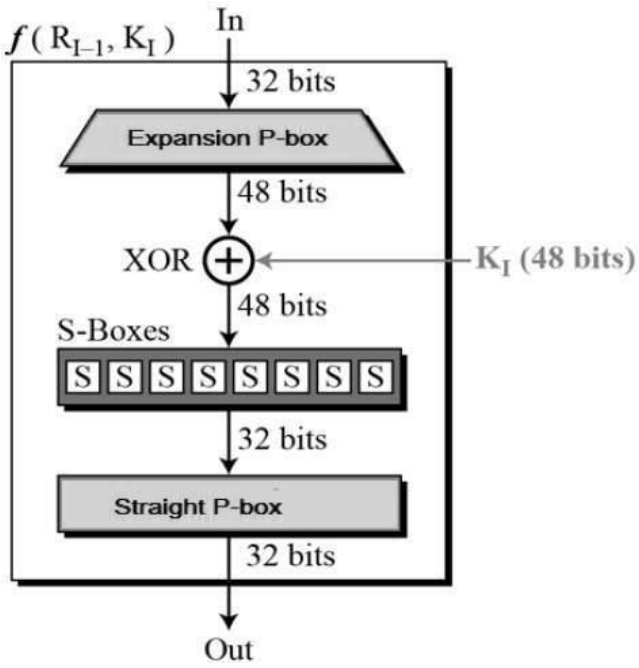
DES je implementacija Feistelove šifre. Koristi Feistelovu strukturu od 16 krugova. Veličina bloka je 64-bitna. Iako je dužina ključa 64-bitna, DES ima efektivnu dužinu ključa od 56 bita, pošto 8 od 64 bita ključa ne koristi algoritam za šifrovanje (funkcioniše samo kao bitova za proveru).



Slika 10. DES struktura

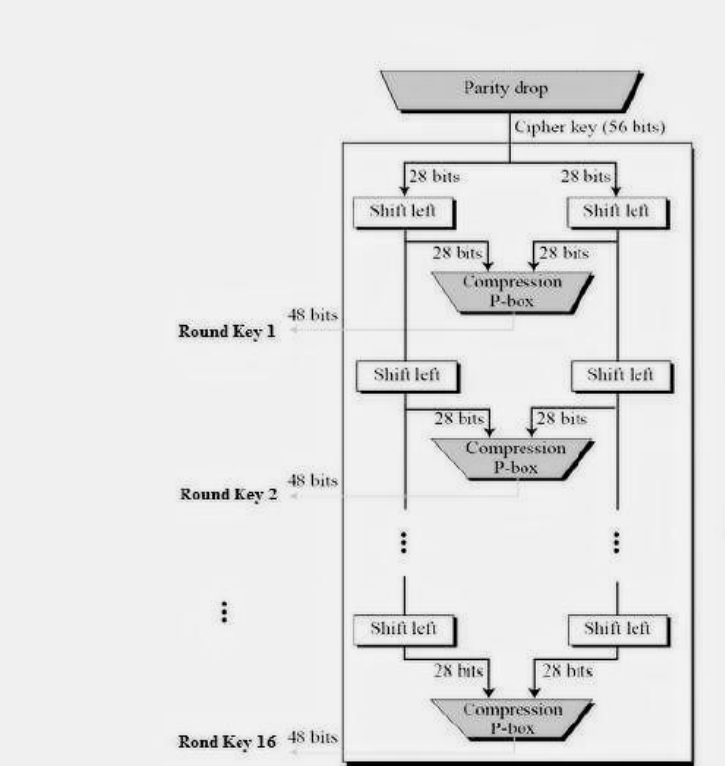
Pošto je DES zasnovan na Feistelovoj šifri, sve što je potrebno da se navede DES je:

- Okrugla funkcija
- Ključni raspored
- Svaka dodatna obrada − Početna i konačna permutacija



Slika 11. DES okrugla funkcija

Početna i konačna permutacija su ravne permutacione kutije (P-kutije) koje su inverzne jedna drugoj. Oni nemaju kriptografski značaj u DES-u. Srce ove šifre je DES funkcija. Funkcija DES primenjuje 48-bitni ključ na krajnja desna 32 bita da bi proizvela 32-bitni izlaz. Okvir za permutaciju proširenja – Pošto je desni ulaz 32-bitni, a okrugli ključ 48-bitni, prvo moramo da proširimo desni ulaz na 48 bita.



Slika 12. DES algoritam - Generisanje ključa

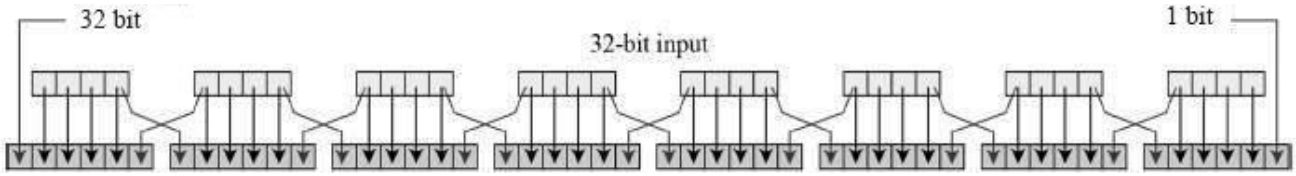
XOR − Nakon permutacije proširenja, DES radi XOR operaciju na proširenom desnom delu i okruglom ključu. Okrugli ključ se koristi samo u ovoj operaciji.

Kutije za zamenu. − S-kutije vrše pravo mešanje (zabuna). DES koristi 8 S-boksova, svaki sa 6-bitnim ulazom i 4-bitnim izlazom.
Postoji ukupno osam S-bok stolova. Izlaz svih osam S-boksova se zatim kombinuje u 32-bitnu sekciju.

16	07	20	21	29	12	28	17
01	15	23	26	05	18	31	10
02	08	24	14	32	27	03	09
19	13	30	06	22	11	04	25

Slika 13. Izgled permutaciija nakon S-kutije

Prava permutacija – 32-bitni izlaz S-kutija se zatim podvrgava pravoj permutaciji sa pravilom.



Slika 14. DES permutacije

Generator okruglog ključa kreira šesnaest 48-bitnih ključeva od 56-bitnog ključa za šifrovanje.

DES zadovoljava oba željena svojstva blok šifre. Ova dva svojstva čine šifru veoma snažnom:

- Efekat lavine – Mala promena otvorenog teksta dovodi do veoma velike promene u šifrovanom tekstu.
- Kompletnost – Svaki bit šifrovanog teksta zavisi od mnogo bitova otvorenog teksta.

Tokom poslednjih nekoliko godina, kriptoanaliza je pronašla neke slabosti u DES-u kada su izabrani ključevi slabi ključevi. Ove ključeve treba izbegavati.

DES se pokazao kao veoma dobro dizajnirana blok šifra. Nije bilo značajnih kriptoanalitičkih napada na DES osim iscrpnog pretraživanja ključeva.

3.6.3. 3DES

Brzina iscrpnih pretraga ključeva protiv DES-a nakon 1990. godine počela je da izaziva nelagodu među korisnicima DES-a. Međutim, korisnici nisu želeli da zamene DES jer je potrebno ogromno vreme i novac za promenu algoritama šifrovanja koji su široko prihvaćeni i ugrađeni u velike bezbednosne arhitekture.

Pragmatičan pristup nije bio da se DES potpuno napusti, već da se promeni način na koji se DES koristi. Ovo je dovelo do modifikovanih šema Triple DES-a (ponekad poznatih kao 3DES).

Uzgred, postoje dve varijante Triple DES poznate kao Triple DES sa 3 ključa (3TDES) i Triple DES sa 2 ključa (2TDES).

Pre upotrebe 3DES, korisnik prvo generiše i distribuira 3DES ključ K, koji se sastoji od tri različita DES ključa K1, K2 i K3. To znači da stvarni 3DES ključ ima dužinu 3×56 = 168 bita.

Proces šifrovanja-dešifrovanja je sledeći:

- Šifrujte blokove otvorenog teksta koristeći jedan DES sa ključem K1.
- Sada dešifrujte izlaz koraka 1 koristeći jedan DES sa ključem K2.
- Konačno, šifrujte izlaz iz koraka 2 koristeći jedan DES sa ključem K3.
- Izlaz iz koraka 3 je šifrovani tekst.
- Dešifrovanje šifrovanog teksta je obrnut proces. Korisnik prvo dešifruje pomoću K3, zatim šifruje pomoću K2 i na kraju dešifruje pomoću K1.

Zbog ovakvog dizajna Trostrukog DES-a kao procesa šifrovanje–dešifrovanje–šifrovanje, moguće je koristiti 3TDES (hardversku) implementaciju za pojedinačni DES postavljanjem K1, K2 i K3 na istu vrednost. Ovo obezbeđuje kompatibilnost unazad sa DES-om.

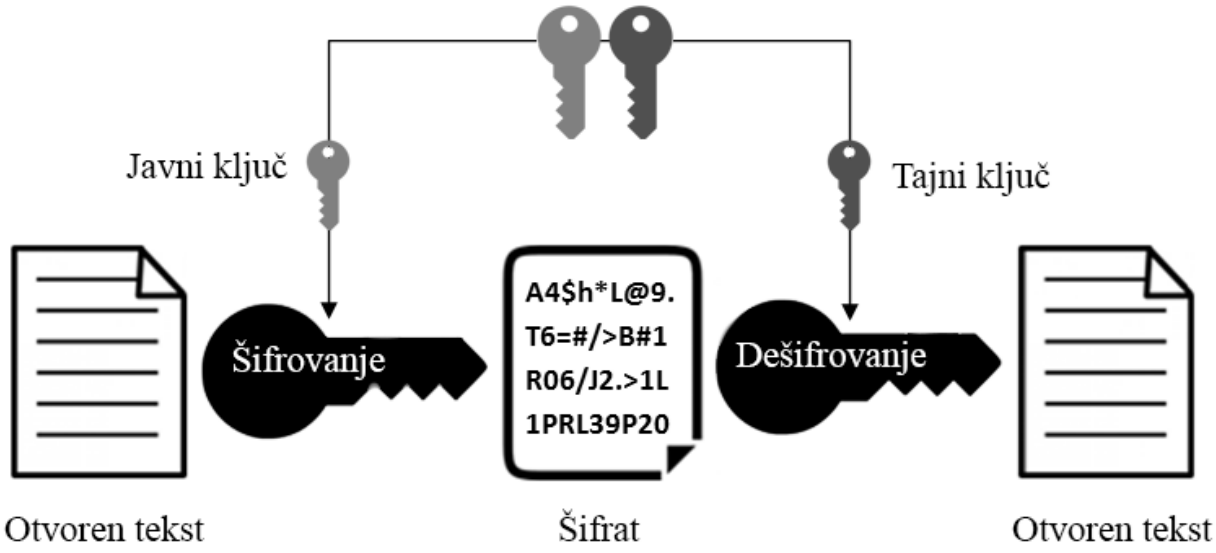
Druga varijanta Triple DES (2TDES) je identična 3TDES osim što je K3 zamenjen sa K1. Drugim rečima, korisnik šifruje blokove otvorenog teksta ključem K1, zatim dešifruje ključem K2 i na kraju ponovo šifruje sa K1. Stoga, 2TDES ima dužinu ključa od 112 bita.

Trostruki DES sistemi su znatno bezbedniji od jednog DES-a, ali je očigledno da je to mnogo sporiji proces od šifrovanja korišćenjem jednog DES-a.

3.7. Asimetrična kriptografija

Za razliku od simetrične enkripcije, koja koristi isti tajni ključ za šifrovanje i dešifrovanje osetljivih informacija, asimetrična enkripcija, poznata i kao kriptografija sa javnim ključem ili šifrovanje sa javnim ključem, koristi matematički povezane parove javnog i privatnog ključa za šifrovanje i dešifrovanje.

Kao i kod simetričnog šifrovanja, otvoreni tekst se i dalje pretvara u šifrovani tekst i obrnuto tokom šifrovanja, odnosno dešifrovanja. Glavna razlika je u tome što se dva jedinstvena para ključeva koriste za asimetrično šifrovanje podataka.



Slika 15. Šifrovanje/dešifrovanje u asimetričnoj kriptografiji

Jedan od razloga zašto se asimetrična enkripcija često smatra sigurnijom od simetrične enkripcije je taj što asimetrična enkripcija, za razliku od simetrične enkripcije, ne zahteva razmenu istog ključa za šifrovanje-dešifrovanje između dve ili više strana. Da, javni ključevi se razmenjuju, ali korisnici koji dele podatke u asimetričnom kriptosistemu imaju jedinstvene parove javnih i privatnih ključeva, a njihovi javni ključevi, pošto se koriste samo za šifrovanje, ne predstavljaju rizik od neovlašćenog dešifrovanja od strane napadača ako postanu poznati, jer napadači, pod pretpostavkom da su privatni ključevi privatni, ne znaju privatne ključeve korisnika i stoga ne mogu da dešifruju šifrovane podatke.

Asimetrično šifrovanje takođe omogućava autentifikaciju digitalnog potpisa, za razliku od simetrične enkripcije. U osnovi, ovo uključuje korišćenje privatnih ključeva za digitalno potpisivanje poruka ili datoteka, a njihovi odgovarajući javni ključevi se koriste da bi se potvrdilo da ove poruke potiču od ispravnog, verifikovanog pošiljaoca.

Primeri asimetrične enkripcije uključuju:

- Rivest Šamir Adleman (RSA)
- standard digitalnog potpisa (DSS), koji uključuje algoritam digitalnog potpisa (DSA)
- Kriptografija eliptične krive (ECC)
- Diffie-Hellman metod razmene

- TLS/SSL protokol

Objavljen 1977. godine, RSA je jedan od najstarijih primera asimetrične enkripcije. Razvili su Ron Rivest, Adi Šamir i Leonard Adleman, RSA enkripcija generiše javni ključ množenjem dva velika, nasumična prosta broja zajedno, i koristeći te iste proste brojeve, generiše privatni ključ. Odatle dolazi do standardnog asimetričnog šifrovanja: informacije se šifruju pomoću javnog ključa i dešifruju korišćenjem privatnog ključa.

DSS, koji uključuje algoritam digitalnog potpisa (DSA), savršen je primer autentifikacije asimetričnog digitalnog potpisa. Privatni ključ pošiljaoca se koristi za digitalno potpisivanje poruke ili datoteke, a primalac koristi odgovarajući javni ključ pošiljaoca da potvrdi da potpis potiče od tačnog pošiljaoca, a ne od sumnjivog ili neovlašćenog izvora.

ECC je RSA alternativa koja koristi manje veličine ključeva i matematičke eliptičke krive da izvrši asimetrično šifrovanje. Često se koristi za digitalno potpisivanje transakcija kriptovaluta; u stvari, popularna kriptovaluta Bitcoin koristi ECC – algoritam digitalnog potpisa eliptične krive (ECDSA), tačnije – da digitalno potpiše transakcije i osigura da sredstva troše samo ovlašćeni korisnici. ECC je mnogo brži od RSA u smislu generisanja ključeva i potpisa, i mnogi ga smatraju budućnošću asimetrične enkripcije, uglavnom za veb saobraćaj i kriptovalute, ali i za druge aplikacije.

Diffie-Hellman, jedno od najvećih otkrića kriptografije, je metod razmene ključeva koji dve strane koje se nikada nisu srele mogu da koriste za razmenu javnih i privatnih parova ključeva preko javnih, nesigurnih kanala komunikacije. Pre Diffie-Hellmana, dve strane koje su želele da šifruju međusobnu komunikaciju morale su fizički unapred da razmene ključeve za šifrovanje kako bi obe strane mogle da dešifruju međusobne šifrovane poruke. Diffie-Hellman je napravio tako da se ovi ključevi mogu bezbedno razmenjivati preko javnih komunikacionih kanala, gde treće strane obično izvlače osetljive informacije i ključeve za šifrovanje.

TLS/SSL koristi asimetrično šifrovanje za uspostavljanje bezbedne klijent-server sesije dok klijent i server generišu simetrične ključeve za šifrovanje. Ovo je poznato kao TLS rukovanje. Nakon što je TLS rukovanje završeno, ključevi sesije klijent-server se koriste za šifrovanje informacija koje se razmenjuju u toj sesiji.

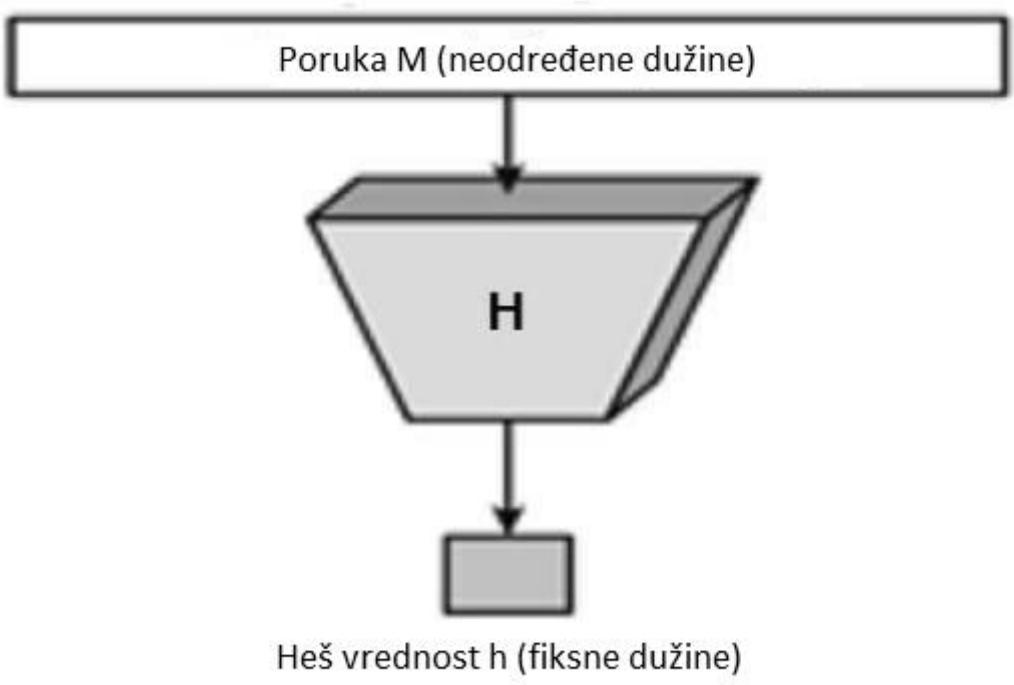
Prednosti korišćenja asimetrične enkripcije uključuju:

- Distribucija ključeva nije neophodna: obezbeđivanje kanala distribucije ključeva je dugo bila problem u kriptografiji. Asimetrično šifrovanje u potpunosti eliminiše distribuciju ključeva. Potrebni javni ključevi se razmenjuju preko servera javnih ključeva, a otkrivanje javnih ključeva u ovom trenutku nije štetno za bezbednost šifrovanih poruka, jer se ne mogu koristiti za dobijanje privatnih ključeva.
- Razmena privatnih ključeva nije neophodna: sa asimetričnom enkripcijom, privatni ključevi treba da ostanu uskladišteni na bezbednoj lokaciji i stoga privatni za entitete koji ih koriste. U osnovi, ključevi potrebni za dešifrovanje osetljivih informacija se nikada ne razmenjuju i ne bi trebalo da se razmenjuju preko potencijalno kompromitovanog kanala komunikacije, a to je veliki plus za bezbednost i integritet šifrovanih poruka.

- Provera autentičnosti digitalnog potpisa/poruke: sa asimetričnim šifrovanjem, pošiljaoci mogu da koriste svoje privatne ključeve da digitalno potpišu i verifikuju da poruka ili datoteka potiče od njih, a ne od nepouzdane treće strane.

3.8. Heš funkcije

Heš funkcije su izuzetno korisne i pojavljuju se u skoro svim aplikacijama za bezbednost informacija.



Slika 16. Rezultat jedne heš funkcije je uvek fiksne dužine

Heš funkcija je matematička funkcija koja pretvara numeričku ulaznu vrednost u drugu komprimovanu numeričku vrednost. Ulaz u heš funkciju je proizvoljne dužine, ali izlaz je uvek fiksne dužine.

Slika 17. Struktura heš funkcije

Vrednosti koje vraća heš funkcija nazivaju se sažetkom poruke ili jednostavno heš vrednosti.

Tipične karakteristike heš funkcija su:

Izlaz fiksne dužine (heš vrednost)

- Heš funkcija pokriva podatke proizvoljne dužine do fiksne dužine. Ovaj proces se često naziva heširanjem podataka.
- Generalno, heš je mnogo manji od ulaznih podataka, pa se heš funkcije ponekad nazivaju funkcijama kompresije.
- Pošto je heš manji prikaz većih podataka, naziva se i sažetak.
- Heš funkcija sa n-bitnim izlazom se naziva n-bitna heš funkcija. Popularne heš funkcije generišu vrednosti između 160 i 512 bita.

Efikasnost rada

- Generalno za bilo koju heš funkciju h sa ulazom k, izračunavanje h(k) je brza operacija.
- Računarske heš funkcije su mnogo brže od simetrične enkripcije.

Da bi bila efikasan kriptografski alat, poželjno je da heš funkcija poseduje sledeća svojstva:

- Trebalo bi da bude računarski teško obrnuti heš funkciju. Drugim rečima, ako je heš funkcija h proizvela heš vrednost z, onda bi trebalo da bude težak proces pronalaženja bilo koje ulazne vrednosti k koja hešuje do z. Ovo svojstvo štiti od napadača koji ima samo heš vrednost i pokušava da pronađe ulaz, da će s obzirom na ulaz i njegov heš, biti teško pronaći drugi ulaz sa istim hešom. Drugim rečima, ako heš funkcija h za ulaz k proizvodi heš vrednost h(k), onda bi trebalo biti teško pronaći bilo koju drugu ulaznu vrednost i takvu da je h(i) = h(k).
- Pošto je heš funkcija funkcija kompresije sa fiksnom dužinom heša, nemoguće je da heš funkcija nema kolizije. Ovo svojstvo bez sudara samo potvrđuje da bi ove kolizije trebalo biti teško pronaći. Ovo svojstvo otežava napadaču da pronađe dve ulazne vrednosti sa istim hešom..

U srcu heširanja je matematička funkcija koja radi na dva bloka podataka fiksne veličine da bi kreirala heš kod. Ova heš funkcija čini deo algoritma heširanja.

Veličina svakog bloka podataka varira u zavisnosti od algoritma. Obično su veličine bloka od 128 bita do 512 bita.

Algoritam heširanja uključuje runde gore navedene heš funkcije poput blok šifre. Svaka runda uzima ulaz fiksne veličine, obično kombinaciju najnovijeg bloka poruka i izlaza poslednje runde.

Ovaj proces se ponavlja onoliko rundi koliko je potrebno za heširanje cele poruke.

Pošto heš vrednost prvog bloka poruke postaje ulaz za drugu heš operaciju, čiji izlaz menja rezultat treće operacije, i tako dalje. Ovaj efekat, poznat kao lavinski efekat heširanja.

Efekat lavine dovodi do suštinski različitih heš vrednosti za dve poruke koje se razlikuju čak i za jedan bit podataka.

Heš funkcija generiše heš kod radeći na dva bloka binarnih podataka fiksne dužine.

Algoritam heširanja je proces za korišćenje heš funkcije, koji navodi kako će poruka biti razbijena i kako su rezultati iz prethodnih blokova poruka povezani zajedno.

Najčešće korišćene hash funkcije su:

MD5 je bila najpopularnija i široko korišćena heš funkcija već nekoliko godina.

- MD familija se sastoji od heš funkcija MD2, MD4, MD5 i MD6. Usvojen je kao Internet standard RFC 1321. To je 128-bitna heš funkcija.
- MD5 se široko koristi u svetu softvera kako bi se obezbedio integritet prenete datoteke. Na primer, serveri datoteka često obezbeđuju unapred izračunati MD5 kontrolni zbir za datoteke, tako da korisnik može da uporedi kontrolni zbir preuzete datoteke sa njim.
- 2004. godine u MD5 su pronađene kolizije. Prijavljeno je da je analitički napad bio uspešan samo za sat vremena korišćenjem računarskog klastera. Ovaj napad kolizijom je doveo do kompromitovanja MD5 i stoga se više ne preporučuje za upotrebu.

Porodica SHA se sastoji od četiri SHA algoritma; SHA-0, SHA-1, SHA-2 i SHA-3. Iako iz iste porodice, postoje strukturno različite.

- Originalna verzija je SHA-0, 160-bitna heš funkcija, koju je objavio Nacionalni institut za standarde i tehnologiju (NIST) 1993. Imala je nekoliko slabosti i nije postala veoma popularna. Kasnije 1995. godine, SHA-1 je dizajniran da ispravi navodne slabosti SHA-0.
- SHA-1 je najčešće korišćena od postojećih SHA heš funkcija. Koristi se u nekoliko široko korišćenih aplikacija i protokola, uključujući SSL (Secure Socket Laier) bezbednost.
- Godine 2005. pronađen je metod za otkrivanje kolizija za SHA-1 u praktičnom

vremenskom okviru što dovodi u sumnju dugoročnu zapošljivost SHA-1.

- Porodica SHA-2 ima još četiri SHA varijante, SHA-224, SHA-256, SHA-384 i SHA-512 u zavisnosti od broja bitova u njihovoj heš vrednosti. Još uvek nisu prijavljeni uspešni napadi na SHA-2 heš funkciju.
- Iako je SHA-2 jaka heš funkcija. Iako značajno drugačiji, njegov osnovni dizajn i dalje prati dizajn SHA-1. Stoga je NIST pozvao na nove konkurentne dizajne heš funkcija.
- U oktobru 2012. NIST je izabrao Keccak algoritam kao novi SHA-3 standard. Keccak nudi mnoge prednosti, kao što su efikasne performanse i dobra otpornost na napade.

RIPEMD je akronim za RACE Integriti Primitives Evaluation Message Digest. Ovaj skup heš funkcija je dizajniran od strane otvorene istraživačke zajednice i generalno poznat kao porodica evropskih heš funkcija.

- Set uključuje RIPEMD, RIPEMD-128 i RIPEMD-160. Postoje i 256 i 320-bitne verzije ovog algoritma.
- Originalni RIPEMD (128 bita) je zasnovan na principima dizajna koji se koriste u MD4 i utvrđeno je da pruža upitnu sigurnost. RIPEMD 128-bitna verzija je došla kao brza zamena za prevazilaženje ranjivosti na originalnom RIPEMD-u.
- RIPEMD-160 je poboljšana verzija i najrasprostranjenija verzija u porodici. 256-bitne i 320-bitne verzije smanjuju šansu od slučajnog sudara, ali nemaju viši nivo bezbednosti u poređenju sa RIPEMD-128 i RIPEMD-160.

Whirlpool je 512-bitna heš funkcija.

- Izvodi se iz modifikovane verzije naprednog standarda šifrovanja (AES). Jedan od dizajnera bio je Vincent Rijmen, ko-kreator AES-a.
- Objavljene su tri verzije Vhirlpool-a; odnosno WHIRLPOOL-0, WHIRLPOOL-T i WHIRLPOOL.

4. Integrisanje sigurnog blokčejn interfejsa u arhitekturu interneta stvari

Korišćenje Blokčejn tehnologije u trenutnoj infrastrukturi interneta stvari pruža mreži mogućnost dodavanja ili uklanjanja bilo kog uređaja u mrežu i sprovođenje prilagođenih politika kontrole pristupa. U predloženom rešenju koristi se privatni blokčejn pristup.. Razlog za ovo je pomeranje kontrole mreže na jednu bezbednu tačku i ako neki od uređaja interneta stvari budu kompromitovani, to ne bi prouzrokovalo štetu u čitavoj mreži. Takođe, još jedna važna činjenica je da bi uređaj za pristup privatnoj blokčejn mreži trebao imati pozivnicu da bi joj se mogao pridružiti. Svi uređaji u ovoj mreži bili bi skriveni od javne mreže i da bi neko ili

nešto pristupilo pametnom uređaju trebalo bi da pošalje zahtev kontrolnoj tački blokčejn mreže.

Da bi jedan uređaj interneta stvari radio bez ikakvih problema i dobijao podatke iz usluge u oblaku, trebalo bi da uradi sledeće radnje. Da bi uređaj poslao zahtev servisu u oblaku i prvo dobio podatke, trebalo bi da pošalje zahtev na interfejs blokčejna koji potvrđuje autentičnost, beleži i sprovodi politike pristupa. Zahtev za blokčejn interfejs je jedna blockchain transakcija. Transakcija je uspešna ako je obavljena sa uređaja koji su odobreni u privatnoj mreži i ako obezbeđuje ispravan heš i podatke koji su potrebni za validaciju transakcije u blokčejnu. Jednom kada transakcija bude uspešna, blokčejn interfejs će poslati zahtev, to jest kopiju podataka tela bloka, servisu u oblaku. Usluga u oblaku bi odgovorila na blokčejn interfejs komandom. Blokčejn interfejs analizira zahtev iz usluge u oblaku i kreira novu transakciju na uređaju koji je zahtevao podatke.

Svi zahtevi unutar privatne mreže su transakcije i ne postoji način da budu izmišljeni ili promenjene. Svaka transakcija se čuva u centralnoj knjizi, bazi podataka koja može biti na interfejsu blokčejna. Blokčejn interfejs se može nalaziti na bilo kom računaru, a centralna knjiga može biti sačuvana na uređaju, serveru lokalne mreže ili u slučaju korišćenja u pametnim kućama gde ne postoji lokalni server, udaljena baza podataka koja se može šifrovati jakim algoritmom šifrovanja.

4.1. Generička šema predloženog rešenja

Pregled generičke šeme predloženog rešenja je sledeći:

- Svi uređaji internet stvari se nalaze iza blokčejn interfejsa kao članovi privatne blokčejn arhitekture.
- Privatni blokčejn interfejs je zadužen za autentifikaciju, validaciju i kontrolu pristupa svim uređajima iza sebe.
- Privatni blokčejn interfejs nudi pored blokčejn logike mogućnost šifrovanja odnosno dešifrovanja podataka naprednim kriptografskim algoritmima kao što su AES, DES, 3DES, RSA i drugi.
- Privatni blokčejn interfejs istovremeno radi kao sistem za otkrivanje upada (IDS) i sistem za prevenciju upada (IPS).
- Uređaji interneta stvari mogu lako da se dodaju i uklone iz mreže bez negativnih posledica za čitav sistem.
- Blokčejn interfejs u sebi sadrži zaštitni zid kao dodatni nivo filtriranja zahteva koji dolaze od nepoznatih izvora.

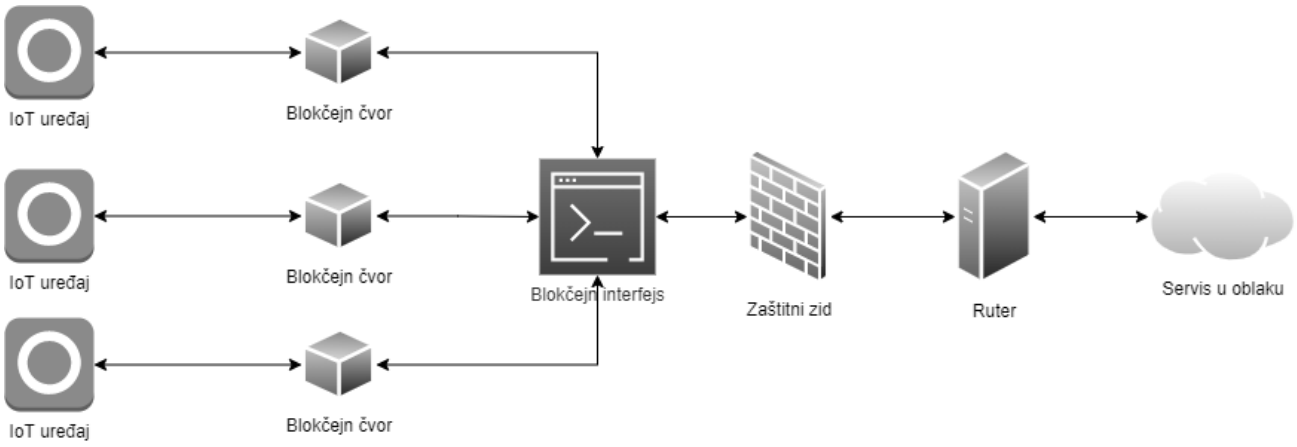
Privatna blokčejn arhitektura ima za cilj da omogući administratoru mreže da samostalno poziva, tj. dodaje uređaje. Cilj ovakvog pristupa je da sistem i dalje koristi decentralizovan pristup ali kontrola pristupa mreži je izuzetno visoka i onemogućeno je da bilo ko njoj pristupi ili je napusti bez toga da administrator sistema to odobri.

Mogućnost korišćenja dodatnih algoritama za šifrovanje odnosno dešifrovanje podataka kao što su AES, DES, 3DES, RSA i drugi, treba da doda dodatni nivo bezbednosti i da

spreči da komunikacija između uređaja interneta stvari i servisa u oblaku bude u otvorenom obliku. Cilj dodavanja enkripcije je da ukoliko servis u oblaku podržava neki od modernih algoritama za šifrovanje podataka da se ta opcija aktivira i zatim da blokčejn interfejs uradi šifrovanje podataka primenom odgovarajućeg algoritma.

Sistem za otkrivanje upada, vatreni zid kao i sistem za prevenciju upada su dodaci koji treba da spreče napade, odbiju pristupe blokčejn interfejsu sa neautorizovanih adresa, kao i da obaveste administratora mreže ukoliko se u privatnom blokčejnu, mreži iza blokčejn interfejsa, dešavaju nestandardni događaji i sumnja se na aktivni napad.

Iako su uređaji interneta stvari povezani jedni sa drugima kao i sa blokčejn interfejsom, oni treba da mogu lako da se dodaju kao i uklone iz nje bez negativnog uticaja na čitavu mrežu. Ovo se omogućava tako što kad se doda bilo koji uređaj on će da se poveže sa blokčejn interfejsom koji će potom da obavesti ostale članove mreže da je on dodat, isto tako i kad se ukloni iz mreže, blokčejn interfejs će da primeti da nema više tog člana i javiti ostalima da nema više učesnika u mreži.



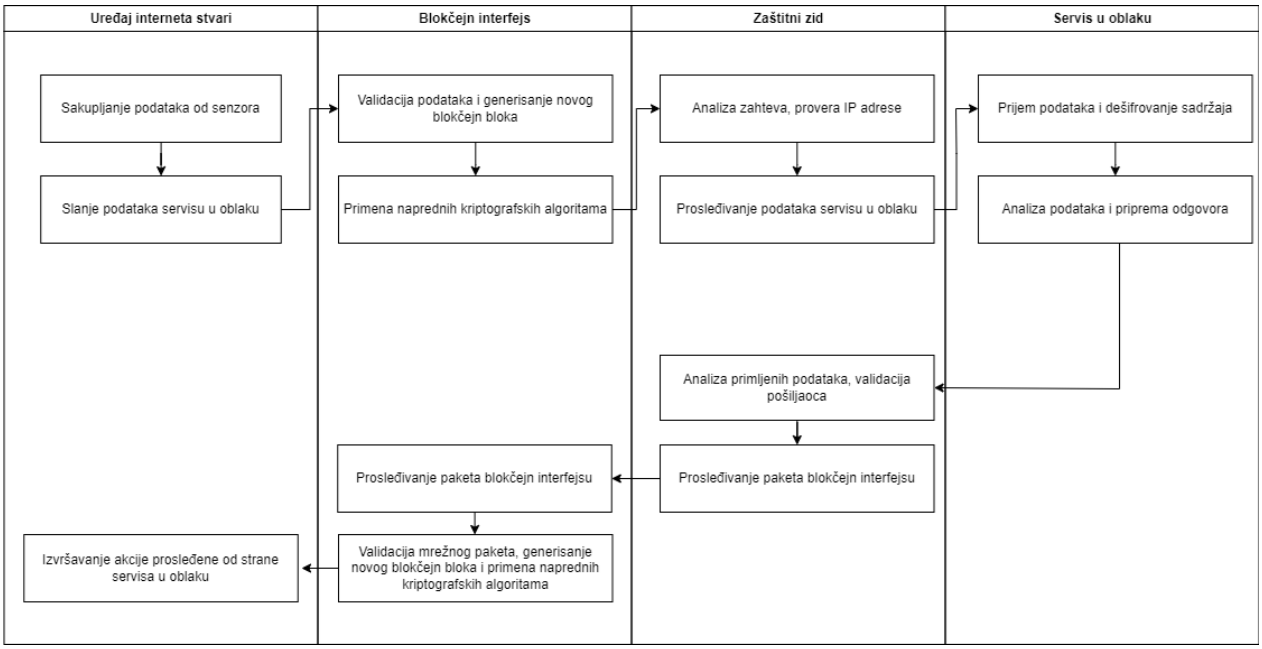
Slika 18. Struktura predloženog rešenja

Komunikacija između uređaja interneta stvari i blokčejn interfejsa može biti žična ili bežična. Blokčejn interfejs smatra da je svaki zahtev uređaja interneta stvari prema servisu u oblaku ili od servisa u oblaku prema uređaju nova transakcija. Blokčejn interfejs obrađuje zahteve, validira njihovu validnost i pravi nove blokčejn blokove. Svaki novi blok se čuva u centralnoj knjizi. Komunikacija između uređaja interneta stvari se takođe smatra transakcijama i prolazi kroz blokčejn interfejs. Ovo omogućava da postoji konstantni uvid u akcije koje se dešavaju unutar privatne blokčejn mreže kao i van nje.

Primer komunikacije između uređaja interneta stvari i servisa u oblaku je sledeći:

- Uređaj interneta stvari sakuplja informacija iz svog okruženja senzorima.
- Uređaj interneta stvari prosleđuje sakupljene informacije na dodatno obrađivanje servisu u oblaku.

- Da bi došlo do servisa u oblaku, blokčejn interfejs, validira zahtev, pravi novi blokčejn blok, dodaje ga u centralnu knjigu.
- Ukoliko postoji mogućnost, pre nego što se prosledi mrežni paket servisu u oblaku, sadržaj paketa se šifruje nekim od naprednih algoritama za šifrovanje (AES, DES, 3DES i drugi).
- Mrežni paket se prosleđuje servisu u oblaku gde oni, ukoliko je šifrovan, dešifruju ga, obrađuju podatke i dostavljaju nazad blokčejn interfejsu listu akcija ili obrađene informacije koje su od koristi uređaju ili korisniku usluge.
- Blokčejn interfejs poseduje dodatne nivoe bezbednosti tako da bilo ko ne može da odgovori njemu i da se primi taj paket. Ovo znači da blokčejn interfejs validira odgovor od servisa u oblaku, proverava da li je odgovor došao sa liste adresa koje su dozvoljene, da li je odgovor od ovog servisa očekivan i potom ga obrađuje. Pod obrađivanjem se podrazumeva dešifrovanje podataka ukoliko su šifrovani, pravljenje nove transakcije koja se upisuje u centralnu knjigu i potom prosleđivanje podataka uređaju interneta stvari koji je zahtevao uslugu.



Slika 19. Dijagram aktivnosti predloženog rešenja

Kao što je prikazano na slici 19., predloženo rešenje ima četiri problematična dela. Prvi fokus je na uređajima interneta stvari. Ovi uređaji koriste senzore za prikupljanje podataka i obradu podataka neophodnih za njihovo slanje na servise u oblaku. Zbog male procesorske snage ovih uređaja, podaci prikupljeni iz spoljnih izvora su loše šifrovani ili uopšte nisu šifrovani. Da bi se obezbedila bezbednost ovih podataka na internetu, predloženo rešenje mora da odradi dodatno šifrovanje podataka. Presretanje podataka je

druge tačke interesovanja. Blokčejn interfejs posmatra svaki mrežni zahtev kao novu transakciju. Svaka transakcija se čuva u centralnoj knjizi. Nakon što se nova transakcija sačuva, zahtev će biti prosleđen udaljenom serveru.

Za obradu ovih podataka sa rutera server ih snima lokalno. Podaci koji dolaze sa uređaja imaju sledeće zaglavlje zahteva i telo zahteva. Zaglavlje zahteva sadrži informacije kao što su URL zahteva, metod zahteva, statusni kod, verzija zahteva (HTTP/1.1 ili HTTP/2), informacije o kodiranju, informacije o korisničkom agentu, informacije o ovlašćenju i informacije o tipu sadržaja. Neki postojeća rešenja za obradu podataka uređaja interneta stvari da bi ispravno radila šalju dodatne informacije u zaglavlju.

Servisi u oblaku ne koristi većinu podataka u zaglavljima zahteva, tako da se mogu izostaviti. Telo zahteva ima podatke koji su potrebni za rad servisa u oblaku. Da bi sprečio curenje podataka, server izostavlja neiskorišćene podatke iz svakog zahteva servisu u oblaku. Svi zahtevi sa servera upućeni Internetu koriste HTTP/2 verziju protokola. Da bi se dodatno poboljšala bezbednost svakog zahteva, moguće je dodati sloj enkripcije na telo zahteva. To znači da ako udaljena usluga ima funkciju da koristi različite algoritame šifrovanja, server može da je doda ovde. Na primer, server može da generiše par RSA ključeva i da doda javni ključ na udaljenu uslugu ili da generiše bilo koji simetrični ključ za upotrebu sa AES, DES ili Triple DES. Novi pripremljeni zahtevi se sada mogu obraditi i poslati servisu u oblaku.

Zahtev se šalje sa servera na blokčejn interfejs koji ga zatim prosleđuje na internet. Jedina ulazna tačka za bilo koji uređaj interneta stvari u mreži su zahtevi za blokčein interfejs upućeni serveru. Udaljeni servis analizira podatke koje server šalje i šalje nazad odgovarajuću radnju koju uređaj treba da uradi. Ponovo, blokčejn interfejs prosleđuje ovaj zahtev serveru. Svaki zahtev za uređaj interneta stvari se prosleđuje serveru. Validnost zahteva se proverava na serveru. Server tada odgovara na sledeća pitanja:

- Da li serveri očekuju da udaljena usluga pošalje zahtev uređaju interneta stvari?
- Da li telo i zaglavlje zahteva sadrže sumnjive podatke?

Ako server zaključi da je zahtev validan, biće poslat na uređaju interneta stvari u lokalnoj mreži. Nakon toga će ovaj uređaj izvršiti radnju koju zahteva udaljeni servis.

4.2. Postavka eksperimentalnog okruženja

Tehnologije koje se koriste za razvoj i testiranje predloženog rešenja su sledeće:

- Programski jezik Javascript u Node.js okruženju
- Arduino ploča sa procesorom i senzorom za vlažnost vazduha i temperaturu
- Okruženje za razvoj zasnovano na Linux operativnom sistemu
- PM2 menadžer procesa za Node.js programski jezik

- Baza podataka u memoriji, zasnovana na JSON formatu

4.2.1. Node.js okruženje

Node.js je okruženje za izvršavanje Javascript programskog jezika. Zasnovan je na Chrome V8 engine-u. Asihnroni pristup i mogućnost izvršavanja više stvari odjednom je jedan od najvažnijih stavki Node.js-a. S obzirom da nema čekanja i vođen je događajima, kao i asinhroni pristup akcijama, čini ga trenutno najboljim okruženjem za razvoj aplikacija koje rade u realnom vremenu.

Node.js ima mogućnost da kreira servere, tako da može da zameni Apache, Nginx i druge vrste servera i bude rešenje za aplikacije koje nemaju mnogo procesorske snage da pokrenu njih a ponovo je potrebno serversko okruženje za rad. Pored toga izuzetno je moderan tako da se unutar njega već nalazi podrška za mnoge moderne kriptografske algoritme i postojeći bezbednosni problemi se uvek krpe. Skalabilnost je pozitivna strana Node.js-a. Kod se izvršava na jednoj niti.

Slede neke od važnih karakteristika koje čine Node.js izborom za razvoj modernik aplikacija:

- Asinhroni i vođen događajima – Svi API-ji biblioteke Node.js su asinhroni, odnosno ne blokiraju. To u suštini znači da server zasnovan na Node.js nikada ne čeka da API vrati podatke ili na izvršenje neke akcije. Server prelazi na sledeći API ili akciju nakon što ga pozove, a mehanizam obaveštenja o događajima Node.js pomaže serveru da dobije odgovor od prethodnog poziva API-ja.
- Veoma brz – Izgrađena na Google Chrome-ovom V8 JavaScript Engine-u, biblioteka Node.js je veoma brza u izvršavanju koda.
- Rad na jednoj niti, ali visoko skalabilni – Node.js koristi model sa jednom niti i petljom događaja. Mehanizam događaja pomaže serveru da odgovori na način bez blokiranja i čini server visoko skalabilnim za razliku od tradicionalnih servera koji kreiraju ograničene niti za obradu zahteva. Node.js koristi program sa jednom niti i isti program može da pruži uslugu za mnogo veći broj zahteva od tradicionalnih servera kao što je Apache HTTP server.
- Bez baferovanja – Node.js aplikacije nikada ne baferuju nikakve podatke. Ove aplikacije jednostavno šalju podatke u delovima.
- Licenca – Node.js je objavljen pod MIT licencom.

4.2.2. Arduino

Arduino je elektronska platforma otvorenog koda zasnovana na hardveru i softveru koji se lako koristi za pravljenje elektronskih projekata. Sve Arduino ploče imaju jednu zajedničku stvar a to je mikrokontroler. Mikrokontroler je u osnovi mali računar.

Sa Arduinom može se dizajnirati i praviti uređaji koji mogu da komuniciraju sa okruženjem. Arduino ploče su u osnovi alat za kontrolu elektronike. Oni su u stanju da čitaju ulaze sa svojim ugrađenim mikrokontrolerom (npr. svetlo na senzoru, objekat blizu senzora) i pretvaraju ga u izlaz (pokreću motor, zvone alarm, uključuju LED, prikazuju informacije na LCD-u).

Pored glavnog čipa mikrokontrolera, mikrokontroleru će biti potrebno mnogo različitih delova da bi funkcionisao. Ono što je Arduino uradio je da su uzeli sve bitne komponente mikrokontrolera i dizajnirali ga na način koji je veoma jednostavan za rad sa komadom PCB-a.

Arduino omogućava da se lako podesi da radi kao senzor koji koristimo u okruženju za testiranje i ponasa kao uređaj interneta stvari.

4.2.3. Linux okruženje

Okruženje u kom se razvija programski kod, radi testiranje kao i testira bezbednost predloženog rešenja je Ubuntu operativni sistem. Testiranje bezbednosti predloženog rešenja i izvršavanje napada se izvršava na Kali Linux-u.

Glavni razlog za korišćenje Linux operativnog sistema je da je moguće testirati i razvijati kod na istoj platformi na kojoj će se potom izvršavati u produkcionom okruženju. Linux dolazi sa setom paketa i omogućava programerima da lako upakuju svoje rešenje i testiraju pre nego što ga postave u produkcionom okruženju. Pored toga neće doći do nepoznatih grešaka usled razlika u funkcionisanju programskog koda između različitih programskih jezika.

Linux ne zahteva dosta memorije i procesorske snage za rad, a rešenja kao što su Ubuntu za internet stvari su specijalne edicije operativnog sistema ubuntu prilagođenje za rad na uređajima interneta stvari.

4.2.4. PM2 menadžer procesa

PM2 je menadžer procesa za Node.js aplikacije koje izvršavaju JavaScript sa ugrađenim regulatorom opterećenja. Omogućava da zauvek održava aplikacije u životu, da ih ponovo učita bez zastoja i da olakša uobičajene zadatke sistemskog administratora.

Neke od aktivnosti PM2 su sledeće:

- PM2 može ponovo da pokrene Node.js servis nakon što su se srušili.
- PM2 može ponovo da pokrene Node.js servis nakon ponovnog pokretanja servera.
- PM2 može da pokrene više procesa sa više jezgara CPU-a da bi postigao efekat kao što je Load Balancer.
- Ukoliko programski kod treba da se ažurira na serveru, poseduje dodatak koji omogućava da se server ne restartuje i gasi, već da se programski kod ažurira i potom prebaci server na novu verziju.

- Moguće je pokrenuti više servisa na više niti i time dobiti veće performanse samog servera.
- Moguće je podesiti da se server automatski restartuje na određene vremenske periode ukoliko je to potrebno.
- PM2 pruža mnogo informacija, uključujući broj ponovnog pokretanja, upotrebu CPU-a, upotrebu memorije, ID procesa itd.
- PM2 omogućava automatsko ponovno pokretanje pod određenim uslovima, kao što su „vreme rada“, „korišćenje memorije“ itd.
- PM2 može organizovati izveštaje, periodično podeliti izveštajei zadržati broj koji odredimo, izbrisati one prekoračene.
- PM2 pruža jednostavan metod postavljanja i podržava višestruke primene servera.

Razlozi za korišćenje PM2 su sledeći:

- Ponovno pokretanje nakon pada: PM2 nam omogućava da zadržimo procese u radu posle njihovog pada. Pokreće ih samostalno i izveštava administratora o potencijanim problemima.
- Daljinsko nadgledanje i upravljanje procesima: Moguće je uvek pristupiti korisničkom interfejsu koji dostavlja u svom paketu i imati uvid u resurse, statuse servisa, kao i potencijalne probleme koji se mogu javiti nakog dugog izvršavanja određenih procesa.

```
→ Projects pm2 start ecosystem.config.js
[PM2][WARN] Applications Custering, Clustering_Replcia not running, starting...
[PM2] App [Clustering_Replcia] launched (2 instances)
[PM2] App [Custering] launched (8 instances)
```

App name	id	mode	pid	status	restart	uptime	cpu	mem	user	watching
Clustering_Replcia	1	cluster	61672	online	0	0s	72%	78.0 MB	nikola_coders	disabled
Clustering_Replcia	3	cluster	61674	online	0	0s	61%	49.4 MB	nikola_coders	disabled
Custering	0	cluster	61671	online	0	0s	72%	78.0 MB	nikola_coders	disabled
Custering	2	cluster	61673	online	0	0s	61%	49.6 MB	nikola_coders	disabled
Custering	4	cluster	61677	online	0	0s	51%	45.2 MB	nikola_coders	disabled
Custering	5	cluster	61682	online	0	0s	39%	39.2 MB	nikola_coders	disabled
Custering	6	cluster	61687	online	0	0s	0%	37.8 MB	nikola_coders	disabled
Custering	7	cluster	61692	online	0	0s	0%	34.1 MB	nikola_coders	disabled
Custering	8	cluster	61697	online	0	0s	0%	32.7 MB	nikola_coders	disabled
Custering	9	cluster	61702	online	0	0s	0%	22.3 MB	nikola_coders	disabled

Slika 20. Prikaz informacija o aktivnim servisima u PM2 menadžeru

4.2.5. Baza podataka

Da bi predloženo rešenje funkcionisalo, potrebna je baza podataka. Sa obzirom da server treba da radi i pod velikim pritiskom, obslužuje dosta zahteva i uređaja komercijalna rešenja kao što su MySQL, MongoDB, MariaDB, PostgreSQL i druge.

Baza podataka za predloženo rešenje treba da bude lagana, tj. da ne troši mnogo memorije, ne zahteva dosta procesorske snage, a čuva informacije o rešenju, centralnu knjigu i ostale neophodne informacije za konfiguraciju. NoSQL baza podataka je najbolje rešenje koje može da se primeni u slučaju predloženog rešenja.

NoSQL baze podataka su dobre kada se bave velikim brojem distribuiranih podataka. Oni mogu da se pozabave problemima performansi velikih podataka bolje od relacionih baza podataka. Oni takođe dobro rade u analizi velikih nestrukturiranih skupova podataka i podataka na virtuelnim serverima u oblaku. Ove baze podataka se takođe mogu nazvati nerelacionim bazama podataka.

Dok se različite vrste baza podataka razlikuju po šemi, strukturi podataka i tipovima podataka koji im najviše odgovaraju, sve se sastoje od istih pet osnovnih komponenti:

- Hardver. Ovo je fizički uređaj na kojem radi softver baze podataka. Hardver baze podataka uključuje računare, servere i čvrste diskove.
- Softver. Softver ili aplikacija baze podataka daje korisnicima kontrolu nad bazom podataka. Softver sistema za upravljanje bazama podataka (DBMS) se koristi za upravljanje i kontrolu baza podataka.
- Podaci. Ovo su sirove informacije koje baza podataka čuva. Administratori baze podataka organizuju podatke kako bi ih učinili smislenijim.
- Jezik pristupa podacima. Ovo je programski jezik koji kontroliše bazu podataka. Programski jezik i DBMS moraju da rade zajedno.
- Procedure. Ova pravila određuju kako baza podataka funkcioniše i kako rukuje podacima.

JSON je tekstualni format podataka koji prati sintaksu JavaScript objekata, koju je popularizovao Daglas Krokford. Iako veoma podseća na literalnu sintaksu JavaScript objekata, može se koristiti nezavisno od JavaScript-a, a mnoga programska okruženja imaju mogućnost čitanja (raščlanjivanja) i generisanja JSON-a.

JSON postoji kao string - korisno kada je potrebno da se prenesu podaci preko mreže. Mora da se konvertuje u izvorni JavaScript objekat kada je potrebno da se pristupi podacima. Ovo nije veliki problem - JavaScript pruža globalni JSON objekat koji ima dostupne metode za pretvaranje između njih.

JSON string se može uskladištiti u sopstvenoj datoteci, koja je u osnovi samo tekstualna datoteka sa ekstenzijom .json i MIME tipom application/json.

Karakteristike JSON-a su sledeće:

- JSON je string sa određenim formatom podataka - sadrži samo svojstva, bez metoda.
- JSON zahteva da se koriste dvostruki navodniki oko stringova i imena svojstava. Pojedinačni navodnici nisu važeći osim što okružuju ceo JSON string.
- Čak i jedan pogrešno postavljeni zarez ili dvotačka može dovesti do toga da JSON datoteka ne funkcioniše.
- JSON zapravo može imati oblik bilo kog tipa podataka koji je važeći za uključivanje u JSON, a ne samo nizove ili objekte. Tako, na primer, jedan string ili broj bi bio važeći JSON.
- Za razliku od JavaScript koda u kojem svojstva objekta mogu biti bez navodnika, u JSON-u se kao svojstva mogu koristiti samo stringovi u navodnicima.

Korišćenje JSON formata podataka kao i NoSQL baze podataka je od velike važnosti zbog performansi, brzine i mogućnosti obrađivanja velike količine zahteva i uređaja unutar predloženog rešenja. Pristup u kome baza podataka neće blokirati Node.js okruženje i izvršavati si asinhrono bi bilo savršeno u ovom slučaju i omogućilo bi da baza podataka bude distribuirana i da ne usporava sistem.

4.3. Realizacija predloženog rešenja

Za realizaciju predloženog rešenja koristi se Node.js okruženje i JavaScript programski jezik. Realizacija predloženog rešenja se izvršava u sledećim koracima:

- Postavljanje jednostavnog Node.js servera.
- Dodavanje neophodnih ruta za prijem i slanje podataka kao i njihovo obrađivanje.
- Dodavanje blokčejn logike, kreiranje blokova, validacija transakcija.
- Dodavanje baze podataka koja će da radi kao centralna knjiga.
- Integrisanje u blokčejn logiku dodatne kriptografske algoritme, AES, DES, 3DES.
- Pravljenje jednostavnog korisničkog interfejsa gde će da se vidi iskorišćenost resursa i pregled stanja servera.
- Dodavanje zaštitnog zida u softveru.
- Dodavanje sistema za sprečavanje i otkrivanje upada.
- Čišćenje koda, provera mogućih unapređenja kao i praćenje nekih od predloga za bezbedno korišćenje Node.js okruženja.

- Primena PM2 menadžera i puštanje u rad.

Node.js okruženje omogućava postavljanje jednostavnog servera. Da bi se postavio server potrebno je definisati na kom portu će se izvršavati server. Da bi predloženo rešenje moglo da se izvrši tj. pokrene više puta i opsluži uređaje interneta stvari u različitim mrežama, ukoliko je potrebno, ovaj deo koda je postavljen da bude dinamični i bude u rasponu od porta 3000 do 4000.

Posle postavljanja servera, neophodno je dodati rute, kojima će moći da se obrađuju zahtevi poslati na server kao i da bi mogle dodatne akcije da se obrađuju i šalju podaci. Sledi lista neophodnih ruta za pravilno funkcionisanje predloženog rešenja:

- broadcast-and-join - ruta koja omogućava da se bilo koji uređaj interneta stvari po konekciji na blokčejn interfejs doda u mrežu, svi članovi mreže da budu obavešteni o njegovom prisustvu i učestvovanju u mreži kao i da taj uređaj bude obavešten o svim članovima mreže.

- blockchain - prezentaciona ruta, omogućava da se uvidi kakvo je stanje blokčejna u bilo kom trenutku rada, omogućava da kad se doda novi uređaj interneta stvari da bude obavešten o blokčejnu i da ima pristup centralnoj knjizi.

- transaction - ruta za pravljenje transakcija, svaki zahtev unutar i van mreže uz pomoć ove rute omogućava da se tretira kao transakcija. Svaka transakcija ima sledeće informacije u sebi, pošiljalac, primalac, vrsta transakcije (zahteva) kao i sadržaj tela zahteva koji se upisuje u polje za vrednost. Da bi mogao svaka transakcija da se validira uz nju se upisuje i jedinstveni heš za svaku transakciju.

- mine - da bi mogla svaka transakcija da se upiše u centralnu knjigu potrebno je njeno rudarenje. Rudarenje je u predloženom rešenju urađeno tako da se kroz njega uradi validacija transakcija, a nagrada za uspešno rudarenje je samo prezentaciona, nema monetarnu vrednost. Cilj rudarenja je da se uradi validacija i sačuva transakcija u centralnoj knjizi.

- register-and-broadcast-node - pri dodavanju novog uređaja interneta stvari ili povezivanja na novu mrežu moguće je da se registruje nov uređaj i javi svima da postoji novi član. Da bi se olakšao pristup i time ubrzala mreža ovo je ruta koja to omogućava. Pomoću ove rute moguće je sa vrlo malo resursa javiti svima da postoji novi član, a isto tako i lako javiti ukoliko taj član nije više u mreži.

- register-node - ova ruta omogućava da se doda novi član mreže bez obaveštenja da on postoji. Osnovna zamisao i primena ove rute je da ukoliko postavljamo novog člana mreže a on još nije aktivan ili u primeni, možemo privremeno da ga dodamo da tu postoji ali njegova komunikacija i članstvo u mreži su samo privremeni i ne može aktivno da učestvuje u transakcijama.

- register-bulk-nodes - kao prethodna ruta, ukoliko postoji potreba za dodavanjem velikog broja uređaja interneta stvari, moguće je preko ove rute dodati ih da budu članovi mreže ali neće aktivno učestovati u transakcijama.

Posle dodavanja ruta, neophodan korak ka daljem razvoju predloženog rešenja je dodavanja blokčejn logika. Blokčejn logika se sastoji iz dve klase, blok klase i blokčejn klase.

Blok klasa u sebi definiše sledeće stavke:

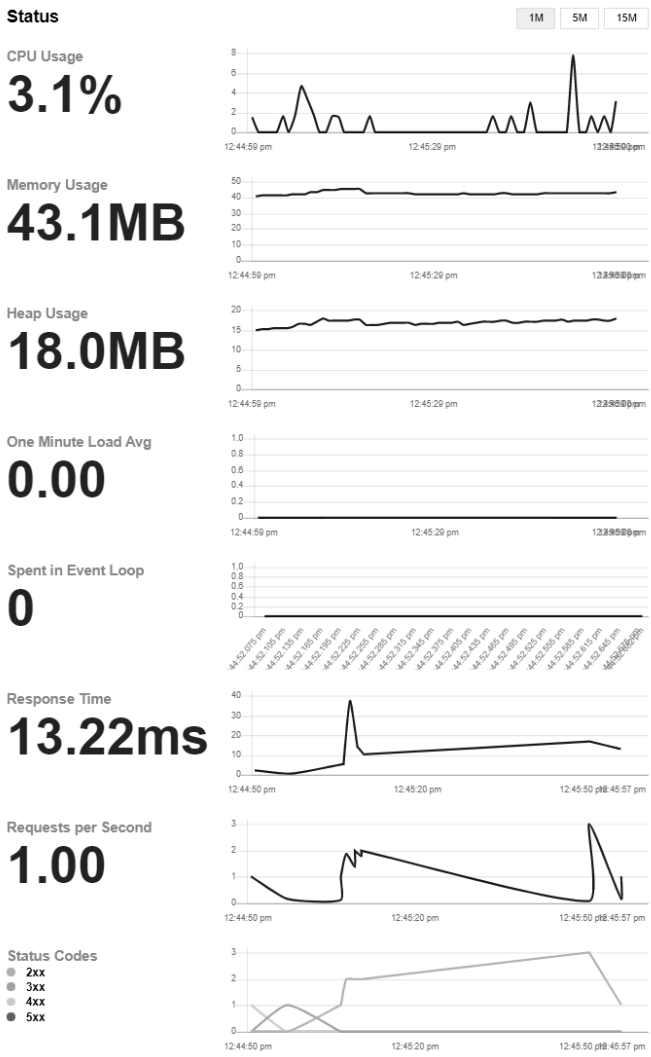
- index - pozicioni broj bloka, definiše poziciju bloka u blokčejnu
- timestamp - vremenska oznaka kada je blok napravljen
- transactions - istorjiski niz svih transakcija objedinjene u jednom bloku
- nonce - jedinstvena oznaka bloka
- hash - heš vrednost bloka, koristi se za validaciju
- prevBlockHash - heš vrednost prethodnog bloka

Blokčejn klasa je zadužena za kreiranje blokova, transakcija kao i njihovu validaciju. Da bi predloženo rešenje bilo bezbedno potrebno je da pri integrisanju blokčejn klase u rute bude pristup koji koristi privatne klase, da se kreira objekat klase i da on pri svakoj akciji čuva informacije samo u njoj.

Da bi blokčejn klasa radila, njoj je neophodna baza podataka. Centralna knjiga, koja čuva informacije o svim transakcijama, blokovima kao i akcijama objavljenim na blokčejn interfejsu mora da bude brza i da ne zahteve puno procesorske snage. Razvijena baza podataka u ovom slučaju je zasnovana na JSON formatu. Dve osnovne funkcionalnost koje poseduje baza je da pročita informacije iz baze podataka kao i da nadoda nove na postojeće. Zamišljeno rešenje ne treba da ima mogućnost nadogradnje podataka kao i modifikovanaj postojećih, tako da su te metode uklonjene zarad dobijanja pristupa koji ne zahteva puno procesorske memorije kao i memorije na tvrdom disku.

Node.js dolazi sa logikom za korišćenje modernih kriptografskih algoritama. Predloženo rešenje u sebi sadrži logiku za AES, DES i 3DES. Svaka od ovih rešenja svoj konačni šifrat dostavljaju u base64 formatu, da bi moglo da se sačuva u centralnu knjigu. Da bi mogli da definišemo za svaki uređaj interneta stvari, u bazu podataka se upisuje ključ kao i uređaj koji ga koristi. Ovo šifrovanje je primenjivo samo ukoliko servis u oblaku podržava šifrovanje podataka. Svaki algoritam za šifrovanje može da bude podešen po uređaju interneta stvari, njegov režim šifrovanja kao i veličina ključa.

Korisnički interfejs koji je prezentativnog karaktera je dostupan na pregled u bilo kom momentu korisniku blokčejn interfejsa. Ovaj interfejs omogućava da se uvidi da li blokčejn interfejs radi kako treba, da li postoje mogućnosti ili čak aktivni napadi na uređaj i kakva je iskorišćenost resursa unutar predloženog rešenja.



Slika 21. Statusni pregled iskorišćenosti resursa predloženog rešenja

Zaštitni zid u predloženom rešenju je specijalizovan za probleme koji je javljaju kao i da spreči napade poznate za internet stvari. Prvi nivo zaštite je zasnovan za limitiranju koje IP adrese mogu da pošalju zahtev i odgovore blokčejn interfejsu. Korisnik uređaja tj. servera ima mogućnost da u bilo kom momentu doda ili ukloni određene adrese sa spiska. Ovo treba da omogući da zahtevi ka servisu u oblaku mogu da se obave samo ka određenim servisima i obrnuto. Bilo ko sa strane ne treba niti ima potrebu da pristupa blokčejn interfejsu a time i uređajima interneta stvari koji se nalaze iza njega. Drugi nivo zaštite je zasnovan na tome koliko često ovi servisi mogu međusobno da šalju zahteve. Ukoliko se pošalje zahtev servisu u oblaku i oblak odgovori sa dosta zahteva ka uređaju interneta stvari, neki od tih zahteva ce biti odbijeni. Takođe ukoliko napadač pokuša da napadne neki od uređaja DDoS napadom, bice zaštićeni od tog napada. Treći nivo zaštite je u veličini paketa koji će moći da obrade. Moguće je uraditi podešavanje po uređaju interneta stvari kolika veličina paketa je dozvoljena.

Predzadnji korak u razvoju predloženog rešenja je pregled programskog koda, njegova evaluacija kao i unapređenje da se obezbedi bezbedno okruženje i ispoštuju predlozi za

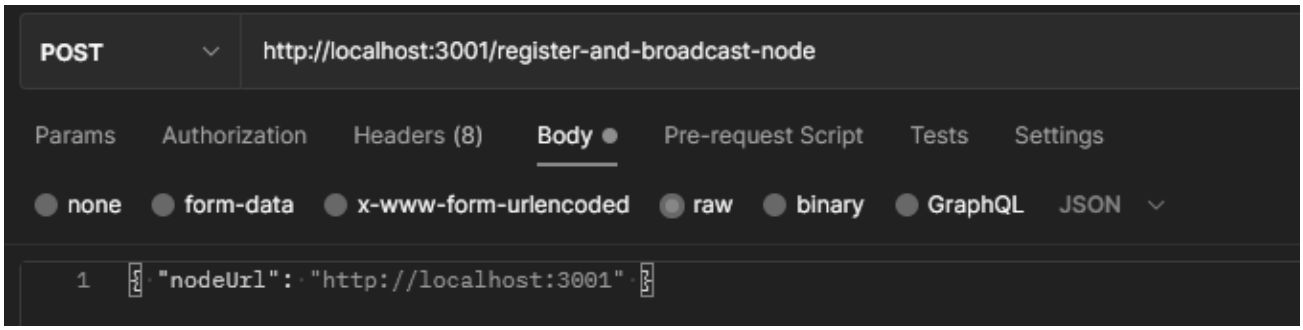
bezbedno programiranje u Node.js okruženju. Sledeće bezbedonosne mere su uključene:

- contentSecurityPolicy - postavlja zaglavlje Content-Securiti-Policy koje pomaže u ublažavanju cross-site scripting napada.
- expectCt - postavlja Expect-CT zaglavlje koje pomaže u ublažavanju pogrešno izdatih SSL sertifikata.
- frameguard - postavlja zaglavlje X-Frame-Options da bi vam pomoglo da ublažite Clickjacking napad.
- hidePoweredBy - uklanja X-Powered-By zaglavlje koje obaveštava o servisu koji koristi server (Apache, Nginx, Express i slično).
- hsts- postavlja zaglavlje Strict-Transport-Security koji tera da se koristi HTTPS protokol.
- ieNoOpen - postavlja zaglavlje X-Download-Options koji je specifičan za Internet Explorer 8 i sprečava preuzimanje ne bezbednog sadržaja.
- noSniff - postavlja zaglavlje X-Content-Type-Options. Sprečava pretraživanje sadržaja sajta koristeći MIME.
- permittedCrossDomainPolicies - postavlja zaglavlje X-Permitted-Cross-Domain-Policies. Sprečava učitavanje sadržaja sa drugog domena.
- referrerPolicy - postavlja zaglavlje Referrer-Policy koji definiše koje informacije su vidljive u Referrer zaglavlju paketa.
- xssFilter - isključuje korišćenje XSS-a tako što postavi vrednost zaglavlja X-XSS-Protection na 0.

Poslednji korak u razvoju predloženog rešenja je njegova postavka. PM2 proces menadžer se dodaje i aktivira na uređaju u klaster režimu. Klaster režim je poseban režim pri pokretanju aplikacije Node.js, on pokreće više procesa i balansira HTTP/TCP/UDP upite između njih. Ovo povećava ukupne performanse (za faktor od puta 10 na mašinama sa 16 jezgara) i pouzdanost (brže ponovno balansiranje utičnice u slučaju neobrađenih grešaka).

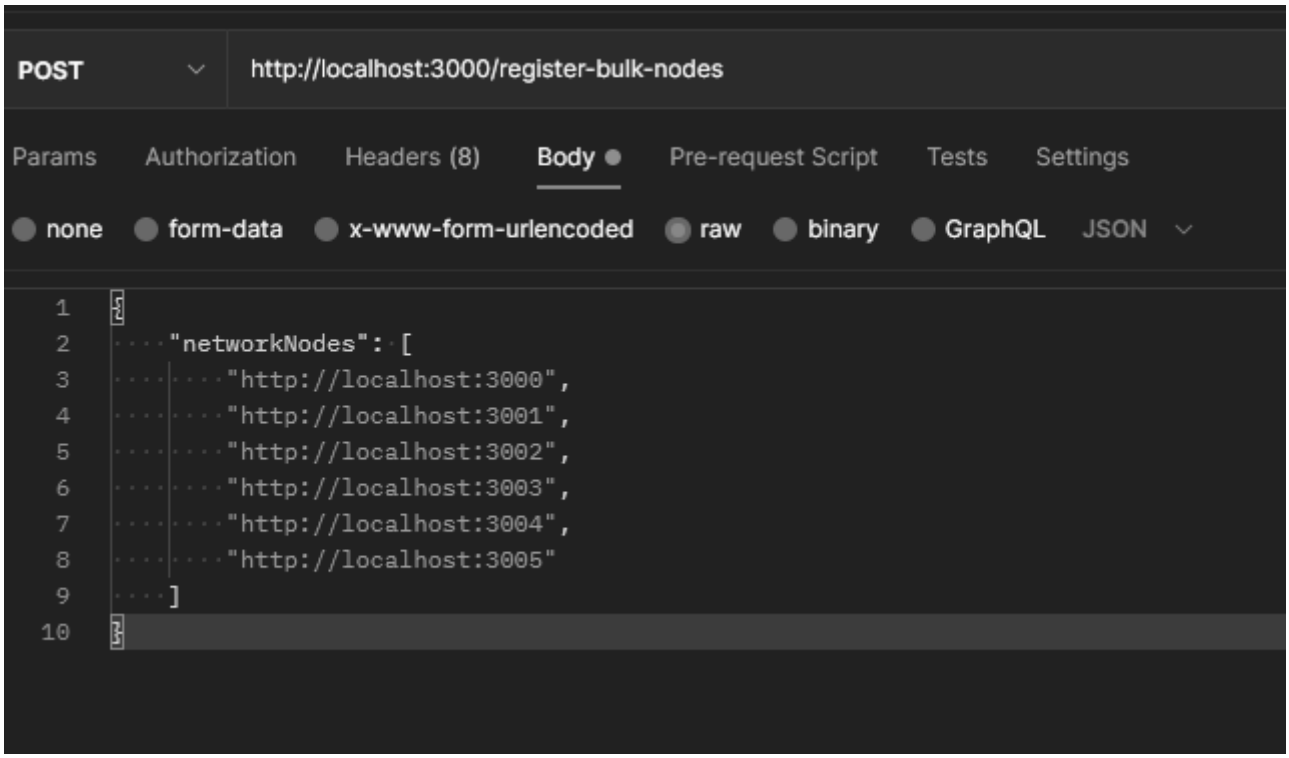
4.4. Pregled predloženog rešenja

Da bi predloženo rešenje moglo da bude bezbedno prvi korak u postavljanju je dodavanje uređaja interneta stvari. Kao što je prethodno rečeno postoji više načina da se dodaju. Jedan od njih je dodavanje jednog uređaja u mrežu. Za to se koristi akcija register-and-broadcast-node.



Slika 22. Dodavanje prvog uređaja interneta stvari u blokčejn mrežu

Na slici 23. prikazano je kako se dodaje uređaj. Slanjem POST zahteva na blokčejn interfejs, uređaj interneta stvari se dodaje u mrežu. Na slici se kaže da je blokčejn interfejs prvi uređaj ove mreže. On treba da ima uvid i kontrolu nad svim transakcijama u mreži. Sa dodavanjem u mrežu on dodaje prvi paket, tj. pravi transakciju i čuva je u bazu podataka.



Slika 23. Dodavanje uređaja interneta stvari u blokčejn mrežu

Nakon što je dodat prvi uređaj, potrebno je da se dodaju ostali uređaji. Na slici 24. prikazano je da se pomoću POST zahteva na rutu register-bulk-nodes, dodaju 5 novih uređaja. Blokčejn interfejs će proveriti postojanje ovih uređaja, da li se nalaze u istoj mreži i ako je tačno potom ih dodati u blokčejn mrežu. Svaki od ovih uređaja svakako može da bude nezavisno jedan od drugih uklonjen bez uticaja na čitavu mrežu.

Ukoliko uređaji koji su dodati podržavaju neki od naprednih kriptografskih algoritama, moguće je za svaki uređaj nezavisno dodati da primenjuju određeni algoritam, dodati ključeve i dodatne parametre.

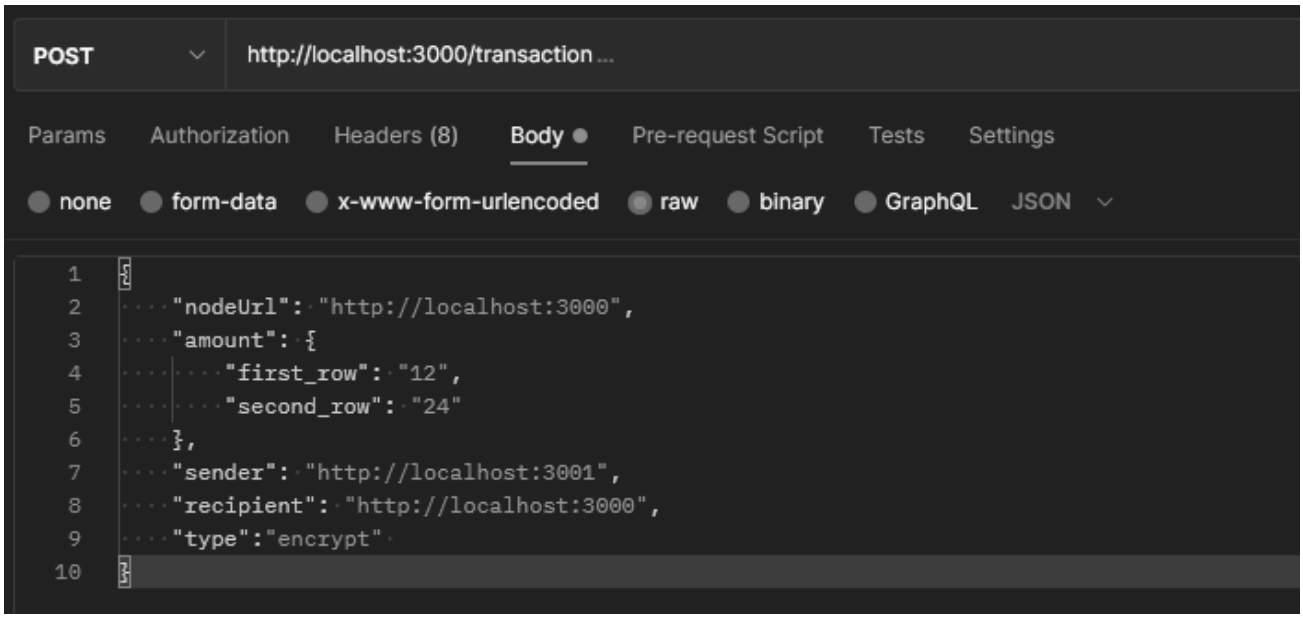
15 lines (15 sloc) | 300 Bytes

```
{
  "cipher": "DES",
  "AES": {
    "ENC_KEY": "bf3c199c2470cb477d907b1e0917c17b",
    "IV": "5183666c72eec9e4"
  },
  "DES": {
    "ENC_KEY": "bf3c199c2470cb477d907b1e0917c17b",
    "IV": "5183666c72eec9e4"
  },
  "TRIPLE_DES": {
    "ENC_KEY": "123456000000000000000000",
    "IV": "12345678"
  }
}
```

Slika 24. Dodavanje ključeva i neophodnih parametara za primenu naprednih kriptografskih algoritama

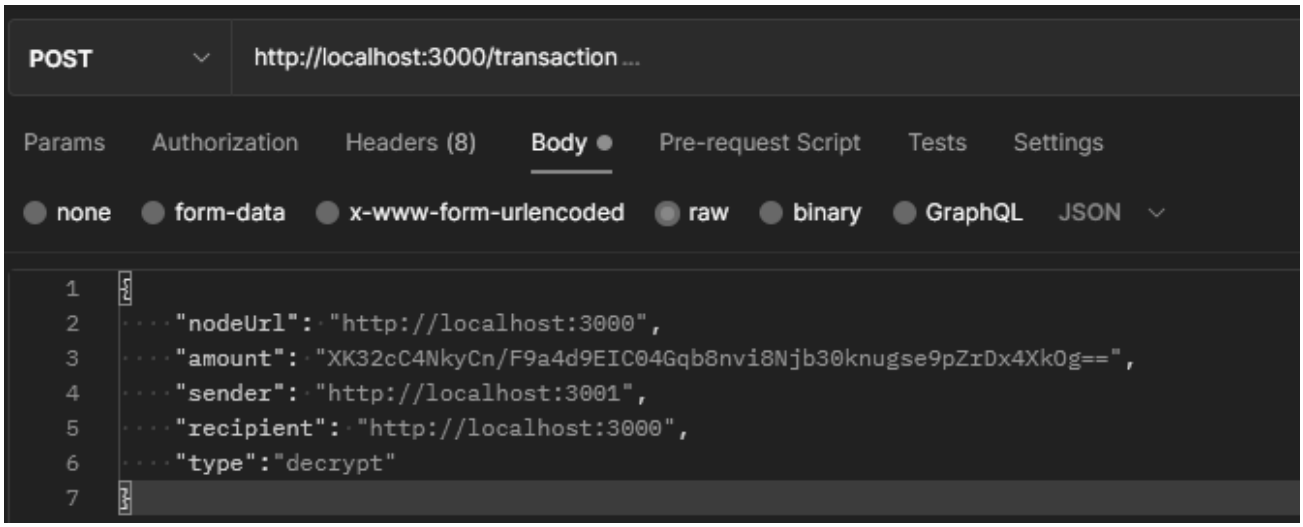
Na slici 25. prikazan je primer dodavanja ključeva i inicijalnih vektora za primenu naprednih kritograpfskih algoritama. cipher parametar određuje koji će algoritam biti u primeni, u slučaju slike 25. to će biti DES. Ostali parametri koji postoje za AES i TRIPLE_DES su samo prezentacioni. Naravno ovde važi da ključ i inicijani vektor takođe treba da postoje u servisu u oblaku, da bi mogla čitava komunikacija da funkcioniše.

Sledeći korak u komunikaciji nakon dodavanja ključeva je da se zahtev pošalje odnosno primi iz oblaka. Kada uređaj interneta stvari šalje zahtev, on problazi kroz blokčejn interfejs. U slučaju da je moguće obaviti primenu naprednih kriptografskih algoritama, oni se dodaju u ovom delu. Blokčejn interfejs upisuje transakciju u centralnu knjigu i kao šifrat je prosleđuje servisu u oblaku.



Slika 25. Upisivanje nove transakcije sa mogućnošću primene naprednog kriptografskog algoritma - šifrovanje

Na slici 24. prikazan je primer jedne transakcije. Svaka transakcija ima sledeće informacije u sebi, vrednost transakcije, komanda ili parametri koji se prosleđuju. Primalac, pošilalac i vrsta transakcije koja opisuuje da li se koristi metoda šifrovanja ili dešifrovanja, ukoliko je ovaj parametar nedostupan, šalju se podaci u otvorenom obliku.



Slika 26. Upisivanje nove transakcije koja je došla od servisa u oblaku i šifrovana je

Na slici 26. prikazana je transakcija koja se upisuje u centralnu knjigu. Blokčejn interfejs prepoznaje na osnovu primaoca koji je uređaj interneta stvari u pitanju, uzima njegov ključ i inicijalni vektor i na osnovu tih parametara dešifruje vrednost i prosleđuje parametre ili komandu tom uređaju. U centralnu knjigu se upisuje šifrovana vrednost, tako da ostaje informacija o tome kada je paket došao, šta je prosleđeno i kako izledaju podaci.

Nakon ovih akcija, blokčejn interfejs je u svakom momentu validirao od koga dolaze zahtevi, kome se šalju i da li postoje neki problematični parametri u paketu.

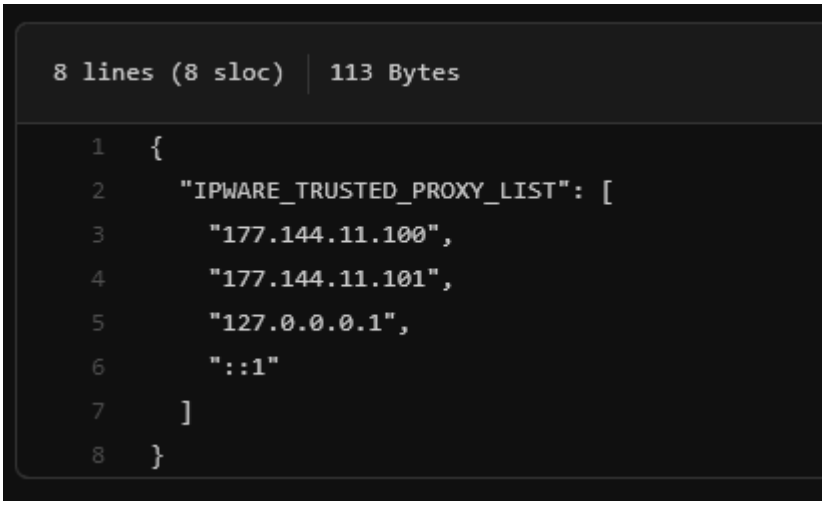
```
{
  - chain: [
      - [
          - {
                index: 1,
                timestamp: 1639913003577,
                transactions: [ ],
                nonce: 100,
                hash: "Genesis block",
                prevBlockHash: "0"
            },
          - {
                index: 2,
                timestamp: 1639913013469,
              - transactions: [
                  - {
                      - amount: {
                            first_row: "12",
                            second_row: "24"
                        },
                        sender: "http://localhost:3001",
                        recipient: "http://localhost:3000",
                        type: "decrypt"
                    },
                  - {
                        amount: 1,
                        sender: "00000",
                        recipient: "13cedea0-60be-11ec-b659-0ffd42801cd4"
                    }
                ],
                nonce: 301,
                hash: "0030393baf0cb2b0eecc596cfc402f4d7ded92aed7b76016179728ddeb7773cb",
                prevBlockHash: "Genesis block"
            },
          - {
                index: 3,
                timestamp: 1639913013851,
              - transactions: [
                  - {
                      - amount: {
                            first_row: "12",
                            second_row: "24"
                        },
                        sender: "http://localhost:3001",
                        recipient: "http://localhost:3000",
                        type: "decrypt"
                    },
                  - {
                        amount: 1,
                        sender: "00000",
                        recipient: "13cedea0-60be-11ec-b659-0ffd42801cd4"
                    }
                ],
```


Slika 27. Izgled blokčejna u centralnoj knjizi

Na slici 27. prikazan je izgled blokčejna u centralnoj knjizi tj. u bazi podataka. Svaki zahtev unutar mreže i van mreže se upisuje u izvornom obliku. U nizu podataka nalaze se paketi, svaki

ima svoju poziciju, vremensku oznaku kad je napravljen kao i informacije o transakciji.

Zadnji deo koji je potrebno pomenuti su opciona podešavanja u predloženom rešenju.



Slika 28. Opcioni podešavanja - dozvoljene adrese

Svaki uređaj interneta stvari koji je povezan na blokčejn interfejs i član je blokčejn mreže ima mogućnost da bude podešen tako da određene IP adrese budu dozvoljene z a komunikaciju. Na slici 28. prikazan je primer podešavanja za jedan uređaj, u nizu se nalaze dozvoljene adrese sa koji i na koje je moguće poslati pakete. Dodatne mogućnosti podešavanja su za svaki uređaj je veličina paketa kao i koliko često mogu da se šalju i primaju zahtevi.

4.5. Evaluacija sistema

U ovom delu rada razmatra se evaluacija blokčejn interfejsa i naprednih sistema zaštite koji se nalaze u njemu, predloženog rešenja. U nastavku ovog poglavlja dat je prikaz rezultata eksperimenata kao i pregled otpornosti na različite vrste napada.

Primenom bezbednosnih mehanizama i prećenjem blokčejn interfejsa utvrđeno je da je otporan na sledeće napade:

- DDoS - bezbednosni mehanizmi ugrađeni u blokčejn interfejs sprečavaju da se bilo koji zahtev koji dolazi do njega uopšte izvrši.
- Detekcija uređaja interneta stvari - nemoguće je da bilo ko otkrije koji se uređaji interneta stvari nalaze u mreži. Razlog za ovo je to što je svaki uređaj povezan na blokčejn interfejs i da bi neko dobio pristup njemu ili ga otkrio more da prođe kroz blokčejn interfejs.
- Prisluškivanje komunikacije - primenom naprednih algoritama za šifrovanje, sva komunikacija između blokčejn interfejsa, izlaskom van lokalne mreže, pa sve do servisa u oblaku je šifrovana. Pored naprednih algoritama za šifrovanje koristi se i HTTPS protokol za svu komunikaciju.
- Čovek u sredini - posmatranje IP adresa, primena ključeva kao i posmatranje

dodatnih parametara mrežnog paketa, onemogućava da bilo ko modifikuje pakete kao i da ih prosleđuje.

Pored prethodno pomenutih napada, sistem je otporan na bilo koju vrstu napada tipa ometanja, petljanja ili de-sinhronizacije. Baza podataka kao i blokčejn interfejs imaju informacije o svakom mrežnom paketu, kao deo transakcije. Ukoliko dođe do ometanja ili petljanja ovi paketi ce da budu odbačeni u procesu validacije podataka. Bilo koja vrsta de-sinhronizacije će zaustaviti transakcije i potom kad ponovo dođe do sinhronizacije, ovi podaci će da budu ponovo pozvani i validirani.

4.5.1. Prikaz rezultata

Najvažniji parametar koji je bitan za neometan rad predloženog rešenja je količina memorije koja je potrebna blokčejn interfejsu da obradi jedan uređaj interneta stvari. Sa obzirom da postoje limitacije u jačini hardvera koji se koristi, treba imati na umu da ovaj parametar, tj. količina radne memorije, ne treba da osciluje u svojoj potrošnji kao i da se adekvatno oslobađa nakon završenih akcija.

U tabelama ispod prikazana je iskorišćenost memorije blokčejn interfejsa pri preradi 20 zahteva koji dolaze od ili odlaze ka servisu u oblaku.

	heapTotal	heapUsed	external
1	180,61	153,12	15,24
2	180,64	154,21	15,24
3	180,65	154,22	15,24
4	180,64	154,22	15,24
5	180,61	154,23	15,24
6	180,64	154,22	15,24
7	180,61	154,23	15,24
8	180,66	154,24	15,24
9	101,64	95,65	12

10	101,67	95,65	12
11	101,69	95,65	12
12	101,65	95,65	12
13	101,64	95,65	12
14	101,67	95,65	12
15	102,08	95,65	10,44
16	102,09	95,65	10,44
17	101,77	95,65	10,44
18	101,65	95,65	10,44
19	101,65	95,65	10,44
20	101,65	95,65	10,44
Srednja vrednost	133,2955	117,23	12,828

Tabela 3. AES iskorišćenost memorije na 20 procesa.

	heapTotal	heapUsed	external
1	140,44	164,22	17,33
2	140,45	164,23	17,33
3	140,45	164,23	17,33
4	140,46	164,23	17,33
5	140,45	164,24	17,33

6	140,45	164,24	17,33
7	140,44	164,23	17,33
8	140,44	164,22	17,33
9	105,65	99,67	14,3
10	105,66	99,68	14,3
11	105,66	99,68	14,3
12	105,67	99,68	14,3
13	105,68	99,68	14,3
14	105,65	99,69	14,3
15	105,66	99,69	12,46
16	105,66	99,71	12,46
17	105,55	99,69	12,46
18	105,55	99,67	12,46
19	105,55	99,67	12,46
20	105,55	99,67	12,46
Sredjnja vrednost	119,5535	123,4631579	14,96

Tabela 4. DES iskorišćenost memorije na 20 procesa.

	heapTotal	heapUsed	external
1	110,94	132,25	8,46

2	110,94	132,25	8,46
3	110,94	132,25	8,46
4	110,94	132,25	8,46
5	110,94	132,25	8,46
6	110,94	132,25	8,46
7	110,94	132,25	8,46
8	110,94	132,25	8,46
9	95,34	84,43	5,54
10	95,34	84,43	5,54
11	95,34	84,43	5,54
12	95,34	84,43	5,54
13	95,34	84,43	5,54
14	95,34	84,43	5,54
15	95,34	84,43	7,85
16	95,34	84,43	7,85
17	95,34	84,43	7,85
18	95,34	84,43	7,85
19	95,34	84,43	7,85
20	95,34	84,43	7,85

Srednja vrednost	101,58	102,0478947	7,401

Tabela 5. 3DES iskorišćenost memorije na 20 procesa.

U tabelama 3. 4. i 5. prikazana je iskorišćenost memorije prilikom 20 procesa. Cilj ove analize je bio da se uoči potrošnja memorije, kao glavnih resursa prilikom aktivnog rada blokčejn interfejsa. U tabelama su definisani i posmatrani sledeći parametri:

- heapTotal - ukupna količina memorije koje je u primeni, zahtevana od virtualne mašine ali ne i iskorišćena.
- heapUsed - realna količina koliko memorije je zaista iskorišćeno pri aktivnom izvršavanju programskog koda.
- external - memorija koja se zahteva za izvršavanje spoljnih procesa, van programskog koda, a unutar virtualne mašine.

Ukupna iskorišćenost memorije i pri primeni naprednih algoritama za šifrovanje nije prešla preko 200 MB.

Sledeći bitan parametar koji je posmatran je iskorišćenost procesora. Da bi bilo što bliže realnom rešenju, procesor je testiran kroz emulacije različitog hardvera u virtualnim mašinama i limitaciji procesorske snage. Ovde se najbolji rezultati primete na procesoru jačine 1 GHz i da podržava rad na više niti.

Zaključak analize iskorišćenih resursa je da je Node.js optimalno rešenje za blokčejn interfejs. Limitacije u hardveru mogu da se zaobiđu i da se napravi uređaj koji će imati procesor jačine 1Ghz i ram memorijom preko 256 MB. Što se tiče memorije za upis podataka, tu mogu da postoje problemi ako se radi samo o lokalnoj bazi podataka. Pregledom transakcija koje se upisuju u lokalnu bazu, preporučeno rešenje bi bilo da se napravi pristup nekom servisu u oblaku gde bi se nalazila baza podataka koja je šifrovana naprednim algoritmom za šifrovanje.

4.5.2. Oblast primene predloženog rešenja

Predloženo rešenje ima široku oblast primene. Može se dodati u bilo koju mrežu i na njega povezati veliki broj uređaja interneta stvari. Predloženo rešenje može da pokrene nekoliko mreža na jednom hardveru. To omogućava da se uređaji interneta stvari kategorišu u različite mreže.

U oblasti mašinstva bi mogli različiti uređaji, senzori kao i mehanizmi za kontrolu postave u različite blokčejn mreže. Ovo bi administratoru dalo potpunu kontrolu i uvid u svaku mrežu zasebno.

U oblasti medicine, povećanje privatnosti kao i kontrole ko ima pristup određenim informacijama.

U oblasti agrikulture, uređaje interneta stvari potrebno je da budu kontrolisani, da budu zaštićeni od napada i da obavljaju neophodna merenja izuzetno tačno. Blokčejn mreža bi omogućila da ovi senzori olako komuniciraju između sebe kao i da njihova komunikacija bude bezbedna.

Pametni gradovi, pametne kuće, transport, sve su ovo oblasti primene gde je moguće dodatno povećati bezbednost i privatnost podataka. Predloženo rešenje bi u svakom od ovih oblasti mogao na neki način da unapredi postojeće bezbedonosne mehanizme, u nekim da omogući bolju komunikaciju između uređaja u istoj mreži i obezbedi da sama komunikacija bude tačnija i neometana.

5. Zaključak

U ovom poglavlju će se sumirati povećanje bezbednosti i privatnosti integrisanjem sigurnog blokčejn interfejsa u arhitekturu interneta stvari. Pored toga predložiće se u kojim poljima je moguće postići dodatne rezultate. Postavljeni ciljevi će se sumirati kroz navođenje postignutih rezultata.

5.1. Sumarni ciljevi istraživanja

Na početku istraživanja u ovom radu, uočeni su osnovni nedostaci u privatnosti i bezbednosti protokola za komunikaciju između uređaja interneta stvari. Nakon dodatnih istraživanja došlo je do zaključka da je moguće da se ostvari viši nivo bezbednosti i privatnosti korišćenjem blokčejn tehnologije.

Daljim radom, utvrđeno je da je moguće integrisati blokčejn tehnologije sa internetom stvari. Ovom sintezom dobija se povišen nivo privatnosti u komunikaciji između uređaja kao i visok nivo poverenja. Daljom analizom postavljeni su novi okviri za integraciju blokčejn tehnologije sa dodatnim mehanizmima odbrane, detekcije i prevencije napada u protokole interneta stvari.

Motivacija za ovo istražovanje potiče od uočenih problema koji postoje u svim uređajima interneta stvari. IoT hab koje obezbeđuje proizvođač (ako postoji) nudi malo ili nimalo bezbednosnih funkcija. Ova čvorišta uglavnom integrišu različite uređaje interneta stvari istog brenda. Drugi IoT centri se uglavnom koriste za posmatranje IoT uređaja u pametnoj kući i prikazivanje podataka koje pružaju na računaru. Rešenje je jednostavan interfejs pogodan za bilo koji IoT uređaj i mrežnu infrastrukturu. Korišćenje blokčejna kao dodatnog sloja sprečava druge napadače da pristupe pametnim uređajima. Glavna karakteristika ovog rada je da podržava bilo koji algoritam šifrovanja koji koriste udaljeni servisi u oblaku.

Osnovni cilj postavljen je razvoj sopstvenog interfejsa za integraciju blokčejn tehnologija sa internetom stvari. Za ostvarivanje ovog cilja bilo je neophodno analizirati tok mrežnih paketa između uređaja interneta stvari i servisa u oblaku. Unaprediti prethodno pomenutu komunikaciju i obezbediti bezbednost i privatnost u skladu sa kriptografskim principima.

5.2. Ostvareni rezultati i doprinosi

U ovom radu je predstavljeno i praktično realizovano sopstveno rešenje za povećanje bezbednosti i privatnosti integrisanjem sigurnog blokčejn interfejsa u arhitekturu interneta stvari. Osnovni doprinosi ovog rada su sledeći:

- Izvršen je pregled u oblasti, uočeni nedostaci u komunikaciji između uređaja interneta stvari i njihovoj interakciji sa servisima u oblaku.
- Izvršen je pregled postojećih programskih jezika, njihovih limitacija i urađena analiza koji bi bio najbolji za predloženo rešenje.
- Predložen je nov pristup za bazu podataka koja bi mogla da opsluži veliki broj uređaja bez uticaja na performanse predloženog rešenja.
- Predložen je sistem za prevenciju napada kao i sistem za detekciju napada koji bi se nalazio ispred blokčejn interfejsa kao dodatni nivo zaštite.
- Izvršen je pregled modernih kriptografskih rešenja i analiziran njihov uticaj na performanse sistema kao i stepen bezbednosti koji doprinosi predloženom rešenju.
- Dizajnirana je nova generička šema za integraciju blokčejn interfejsa sa dodatnim bezbedonosnim mehanizmima.
- Analiziran je doprinos rada u sprečavanju modernih napada na uređaje interneta stvari, njihovo prevenciji kao i unapređenju privatnosti koji je od velikog značaja posebno u okruženjima gde se radi sa informacijama od velikog značaja.
- Predloženo rešenje daje mogućnost administracije većeg broja blokčejn mreža sa jednog programskog / hardverskog rešenja kao i opsluživanje velikog broja uređaja. Dodavanje i uklanjanje uređaja interneta stvari bez uticaja na druge članove mreže kao i mogućnost podešavanja bezbedonosnih mehanizama svakog uređaja posebno.
- Jedinstveni doprinos ovog rada je integracija blokčejn tehnologija, validacije kao i mrežne strukture koje dolazi sa njom u internet stvari. Ovim pristupom povecana je inicijano predložena privatnost, kao i bezbednost podataka dodavanjem dodatnih kriptografskih algoritama.

5.3. Predlog daljeg rada

Primarno postavljeni zadaci su odgovoreni u ovom radu. Zadatak rada je bila integracija blokčejn interfejsa u arhitekturu interneta stvari. Rešenje je zasnovano na razvoju sopstvene integracije blokčejn klase za arhitekturu interneta stvari koje bi trebalo da bude optimalno za opsluživanje velikog broja uređaja kao i da doprinese kriptografskom unapređenju same arhitekture.

Prvi predlog daljeg razvoja bi bilo razvoj baze podataka koja bi se nalazila na udaljenom serveru ali bi bila dovoljno brza i koristila neki od naprednih algoritama za šifrovanje. Razlog za ovo je ukoliko je slučaj da postoji veliki broj uređaja interneta stvari iza blokčejn interfejsa,

transakcije će se stalno generisati i bivati upisane. Memorija koja se nalazi na harderskom rešenju blokčejn interfejsa je limitirana. Ukoliko bi baza podataka bila udaljena, mogla bi lako da bude skalirana i optimizovana za opsuživanje blokčejn interfejsa pri rastu centralne knjige. Još jedna važna stavka bi bila da ukoliko želimo da imamo još jednu blokčejn mrežu koja bi se nalazina van lokalne, a da koristi istu centralnu knjigu, ovo bi bilo optimalno rešenje. Na ovaj način bi mogli da vise različitih blokčejn mreža povežemo u jednu i stvorimo da se sve nalaze u jednoj.

Drugi predlog daljeg razvoja bi bila povećanje komunikacije sa proizođačima uređaja interneta stvari. Da ukoliko ne žele da koriste ili unaprede svoje sisteme daju mogućnost da servisi u oblaku mogu da koriste napredne kriptografske algoritme i da se napravi Plug&Play mogućnost za uređaje koji su povezani na blokčjen interfejs. To bi značilo da korisnik uređaja ne bi trebalo da bude upućen kako uređaj radi već samo da ga poveže u blokčejn mrežu i da se sami generišu neophodni parametri za bezbedan rad.

Treći predlog daljeg razvoja bi bio pregled optimalnog hardvera koji bi mogao da se lako proizvede i održava, a da može da opsluži uređaje i radi neometano. Ovde se govori i mobilnim procesorima kao i pločama koji bi bili laki za postavljanje i podešavanje za rad u svim situacijama, od pametne kuće pa sve do industrijskih pogona.

6. Dodatak

U dodatku su prikazani programski kodovi za potrebe ove disertacije.

```
const express = require("express");

const app = express();

const bodyParser = require("body-parser");

const DDOS = require("./anti-ddos");

const rateLimiter = require("./rate-limiter")(500, 900000);

const helmet = require("helmet");


const get_trusted_ip = require("ipware")().get_trusted_ip;

const                    proxy_list                    =
require("../firewall.json").IPWARE_TRUSTED_PROXY_LIST;


app.use(helmet.dnsPrefetchControl());

app.use(helmet.expectCt());
```

```
app.use(helmet.frameguard());
app.use(helmet.hidePoweredBy());
app.use(helmet.hsts());
app.use(helmet.ieNoOpen());
app.use(helmet.noSniff());
app.use(helmet.permittedCrossDomainPolicies());
app.use(helmet.referrerPolicy());
app.use(helmet.xssFilter());

app.use(function (req, res, next) {
  const ip_info = get_trusted_ip(req, proxy_list);

  if (proxy_list.includes(ip_info.clientIp)) {
    next();
  } else {
    res.end();
  }
});

app.use(function (req, res, next) {
  if (DDOS()) {
    res.send(503, "Server Too Busy");
  } else {
    next();
  }
});
```

```
rateLimiter.whitelist.push("127.0.0.1");
rateLimiter.blocked = function (req, res, next, remaining)
{
  res.send(
    429,
    "Too many requests have been made, " +
      "please wait " +
      remaining / 1000 +
      " seconds"
  );
};
app.use(
  bodyParser.urlencoded({
    extended: true,
  })
);
app.use(bodyParser.json());
app.use(require("./middleware-wrapper")());

app.use(
  express.urlencoded({
    extended: true,
    limit: "1kb",
  })
);
```

```
app.use(
  express.json({
    limit: "1kb",
  })
);


const port = process.argv[2];


require("./routes/index")(app);


app.listen(port, function () {
  console.log(`> listening on port ${port}...`);
});
```

Listing 1. Dinamična implementacija servera sa mogućnošću pokretanja više blokčejn interfejsa na jednoj hardverskoj implementaciji

```
const sha256 = require("sha256");
const nodeUrl = process.argv[3];


const AES = require("./ciphers/AES");
const DES = require("./ciphers/DES");
const TRIPLE_DES = require("./ciphers/Triple-DES");


const config = require("../config.json");
const Database = require("./db");
```

```
class Block {
  /**
   *
   * @param {int} index
   * @param {int} timestamp
   * @param {string} nonce
   * @param {string} prevBlockHash
   * @param {string} hash
   * @param {array} transactions
   */
  constructor(index, timestamp, nonce, prevBlockHash,
hash, transactions) {
    this.index = index;
    this.timestamp = timestamp;
    this.transactions = transactions;
    this.nonce = nonce;
    this.hash = hash;
    this.prevBlockHash = prevBlockHash;
  }
}

class Blockchain {
  constructor() {
    const chain = new Database().getBlockchain();
    if (chain) {
```

```
        this.chain = [new
  Database().getBlockchain()["chain"]];
      } else {
        this.chain = [];
      }

      this.pendingTransactions = [];

      this.nodeUrl = nodeUrl;
      this.networkNodes = [];

      this.creatNewBlock(100, "0", "Genesis block");
    }

    /**
     *
     * @param {int} nonce
     * @param {string} prevBlockHash
     * @param {string} hash
     * @returns
     */
    creatNewBlock(nonce, prevBlockHash, hash) {
      const newBlock = new Block(
        this.chain.length + 1,
        Date.now(),
        nonce,
```

```
      prevBlockHash,
      hash,
      this.pendingTransactions
    );

    this.pendingTransactions = [];
    this.chain.push(newBlock);

    return newBlock;
  }

  /**
   *
   * @returns
   */
  getLatestBlock() {
    return this.chain[this.chain.length - 1];
  }

  /**
   *
   * @param {object} amount
   * @param {string} sender
   * @param {string} recipient
   * @param {string} type
   * @returns
```

```
    */
   makeNewTransaction(amount, sender, recipient, type) {
     if (config.cipher == "AES") {
       const ENC_KEY = config.AES.ENC_KEY;
       const IV = config.AES.IV;

       if (type == "encrypt") {
         var AES_CLASS = new AES();
         amount = AES_CLASS.encrypt(JSON.stringify(amount),
ENC_KEY, IV);
       }

       if (type == "decrypt") {
         var AES_CLASS = new AES();
         amount = JSON.parse(AES_CLASS.decrypt(amount,
ENC_KEY, IV));
       }
     }

     if (config.cipher == "DES") {
       const ENC_KEY = config.TRIPLE_DES.ENC_KEY;
       const IV = config.TRIPLE_DES.IV;

       if (type == "encrypt") {
         var TRIPLE_DES_ClASS = new TRIPLE_DES();
         amount =
TRIPLE_DES_ClASS.encrypt(JSON.stringify(amount), ENC_KEY,
```

```
IV);
      }

      if (type == "decrypt") {
        var TRIPLE_DES_ClASS = new TRIPLE_DES();
        amount =
JSON.parse(TRIPLE_DES_ClASS.decrypt(amount, ENC_KEY, IV));
      }
    }

    if (config.cipher == "TripleDES") {
      const ENC_KEY = config.DES.ENC_KEY;
      const IV = config.DES.IV;

      if (type == "encrypt") {
        var DES_CLASS = new DES();
        amount = DES_CLASS.encrypt(JSON.stringify(amount),
ENC_KEY, IV);
      }

      if (type == "decrypt") {
        var DES_CLASS = new DES();
        amount = JSON.parse(DES_CLASS.decrypt(amount,
ENC_KEY, IV));
      }
    }
```

```
    const transaction = {
      amount: amount,
      sender: sender,
      recipient: recipient,
      type: type,
    };

    this.pendingTransactions.push(transaction);

    console.log(`>>> Transaction: ${amount} from ${sender}
to ${recipient}`);

    return this.getLatestBlock().index + 1;
  }

  /**
   *
   * @param {string} prevBlockHash
   * @param {string} currentBlock
   * @param {int} nonce
   * @returns
   */
  hashBlock(prevBlockHash, currentBlock, nonce) {
    const data = prevBlockHash +
JSON.stringify(currentBlock) + nonce;
    const hash = sha256(data);
    return hash;
```

```
    }


    /**
     *
     * @param {string} prevBlockHash
     * @param {string} currentBlockData
     * @returns
     */
    proofOfWork(prevBlockHash, currentBlockData) {
      let nonce = 0;
      let hash = this.hashBlock(prevBlockHash,
currentBlockData, nonce);

      while (hash.substring(0, 2) !== "00") {
        nonce++;
        hash = this.hashBlock(prevBlockHash,
currentBlockData, nonce);
      }

      return nonce;
    }
}

module.exports = Blockchain;
```

Listing 2. Blokčejn klasa optimizovana za rad sa uređajima interneta stvari sa logikom

za validaciju i dodavanje u centralnu knjigu.

```
const uuid = require("uuid/v1");
const nodeAddr = uuid();

const reqPromise = require("request-promise");

const Blockchain = require("../blockchain");
const bitcoin = new Blockchain();

const Database = require("../db");

module.exports = function (app) {
  app.post("/broadcast-and-join", function (req, res) {
    const nodeUrl = req.body.nodeURI;

    if (bitcoin.networkNodes.indexOf(nodeUrl) == -1) {
      bitcoin.networkNodes.push(nodeUrl);
    }

    const registerNodes = [];
    bitcoin.networkNodes.forEach((networkNode) => {
      const requestOptions = {
        uri: networkNode + "/register-node",
        method: "POST",
        body: { nodeUrl: nodeUrl },
```

```
          json: true,
        };

        registerNodes.push(reqPromise(requestOptions));
      });

      Promise.all(registerNodes)
        .then((data) => {
          const bulkRegisterOptions = {
            uri: nodeUrl + "/register-bulk-nodes",
            method: "POST",
            body: { networkNodes: [...bitcoin.networkNodes,
  bitcoin.nodeUrl] },
            json: true,
          };

          return reqPromise(bulkRegisterOptions);
        })
        .then((data) => {
          res.redirect("wallet");
        });
    });

    app.get("/blockchain", function (req, res) {
      const db = new Database();
      res.send(db.getBlockchain());
```

```
});


app.post("/transaction", function (req, res) {
  const blockIndex = bitcoin.makeNewTransaction(
    req.body.amount,
    req.body.sender,
    req.body.recipient,
    req.body.type
  );

  const latestBlock = bitcoin.getLatestBlock();
  const prevBlockHash = latestBlock.hash;
  const currentBlockData = {
    transactions: bitcoin.pendingTransactions,
    index: latestBlock.index + 1,
  };
  const nonce = bitcoin.proofOfWork(prevBlockHash,
currentBlockData);
  const blockHash = bitcoin.hashBlock(prevBlockHash,
currentBlockData, nonce);

  bitcoin.makeNewTransaction(1, "00000", nodeAddr);
  bitcoin.creatNewBlock(nonce, prevBlockHash,
blockHash);

  res.json({
    message: `Transaction is added to block with index:
```

```
${blockIndex}`,
    });
  });

  app.get("/mine", function (req, res) {
    const latestBlock = bitcoin.getLatestBlock();
    const prevBlockHash = latestBlock.hash;
    const currentBlockData = {
      transactions: bitcoin.pendingTransactions,
      index: latestBlock.index + 1,
    };
    const nonce = bitcoin.proofOfWork(prevBlockHash,
currentBlockData);
    const blockHash = bitcoin.hashBlock(prevBlockHash,
currentBlockData, nonce);

    bitcoin.makeNewTransaction(1, "00000", nodeAddr);

    const newBlock = bitcoin.creatNewBlock(nonce,
prevBlockHash, blockHash);

    const db = new Database();
    db.updateBlockchain(bitcoin);

    res.json({
      message: "Mining new Block successfully!",
      newBlock,
```

```
    });
  });

  app.post("/register-and-broadcast-node", function (req, res) {
    const nodeUrl = req.body.nodeUrl;

    if (bitcoin.networkNodes.indexOf(nodeUrl) == -1) {
      bitcoin.networkNodes.push(nodeUrl);
    }

    const registerNodes = [];
    bitcoin.networkNodes.forEach((networkNode) => {
      const requestOptions = {
        uri: networkNode + "/register-node",
        method: "POST",
        body: { nodeUrl: nodeUrl },
        json: true,
      };

      registerNodes.push(reqPromise(requestOptions));
    });

    Promise.all(registerNodes)
      .then((data) => {
        const bulkRegisterOptions = {
```

```
            uri: nodeUrl + "/register-bulk-nodes",
            method: "POST",
            body: { networkNodes: [...bitcoin.networkNodes,
  bitcoin.nodeUrl] },
            json: true,
          };

          return reqPromise(bulkRegisterOptions);
        })
        .then((data) => {
          res.json({
            message: "A node registers with network
  successfully!",
          });
        });
    });

    app.post("/register-node", function (req, res) {
      const nodeUrl = req.body.nodeUrl;

      if (
        bitcoin.networkNodes.indexOf(nodeUrl) == -1 &&
        bitcoin.nodeUrl !== nodeUrl
      ) {
        bitcoin.networkNodes.push(nodeUrl);

        res.json({
```

```
        message: "A node registers successfully!",
      });
    } else {
      res.json({
        message: "This node cannot register!",
      });
    }
  });

  app.post("/register-bulk-nodes", function (req, res) {
    const networkNodes = req.body.networkNodes;

    networkNodes.forEach((nodeUrl) => {
      if (
        bitcoin.networkNodes.indexOf(nodeUrl) == -1 &&
        bitcoin.nodeUrl !== nodeUrl
      ) {
        bitcoin.networkNodes.push(nodeUrl);
      }
    });

    res.json({
      message: "Registering bulk successfully!",
    });
  });
};
```

Listing 3. Rute neophodne za rad blokčejn interfejsa

```
/**
 *
 * @param {int} min
 * @param {int} max
 * @param {int} free
 */
function rateLimiter(min, max, free) {
  var instance = this;

  if (typeof min !== "number") min = 500;
  if (typeof max !== "number") max = 600000;
  if (typeof free !== "number") free = 2;
  if (min < 1) min = 1;
  if (max < min) max = min;
  if (free < 0) free = 0;

  var delays = [];

  while (free--) delays.push(0);

  delays.push(min);
  delays.push(min);
```

```
  while (true) {
    var value = delays[delays.length - 1] + delays[delays.length - 2];

    if (value > max) {
      delays.push(max);
      break;
    }

    delays.push(value);
  }

  setInterval(function () {
    var now = Date.now();
    for (var address in instance.addresses) {
      if (now - instance.addresses[address].lastAttempt > max)
        delete instance.addresses[address];
    }
  }, 1800000);

  this.addresses = {};

  this.whitelist = [];

  this.blocked = function (req, res, next, remaining) {
    res.send(
```

```
      429,
      "Too many requests have been made, " +
        "please wait " +
        remaining / 1000 +
        " seconds"
    );
  };

  this.reset = function (req) {
    var address;
    try {
      address =
        req.headers["x-forwarded-for"] ||
        req.connection.remoteAddress ||
        req.socket.remoteAddress ||
        req.connection.socket.remoteAddress;
    } catch (error) {}

    address && delete instance.addresses[address];
  };

  this.block = function (req, res, next) {
    var address;
    try {
      address =
        req.headers["x-forwarded-for"] ||
```

```
        req.connection.remoteAddress ||
        req.socket.remoteAddress ||
        req.connection.socket.remoteAddress;
    } catch (error) {}

    if (!address || instance.whitelist.indexOf(address) > -
1) {
      typeof next === "function" && next();
      return;
    }

    var fail = instance.addresses[address] || { count: 0,
lastAttempt: 0 };

    var remaining = fail.lastAttempt + delays[fail.count] -
Date.now();

    if (remaining > 0) instance.blocked(req, res, next,
remaining);
    else {
      fail.lastAttempt = Date.now();
      if (fail.count < delays.length - 1) fail.count++;

      instance.addresses[address] = fail;
      typeof next === "function" && next();
    }
  };
```

```
}


exports = module.exports = function (min, max, free) {
  return new rateLimiter(min, max, free);
};
```

Listing 4. Anti DDOS logika za prevenciju napada na blokčejn interfejs

```
var events = require("events");


var STANDARD_HIGHWATER = 70;
var STANDARD_INTERVAL = 500;
var LAG_EVENT = "LAG_EVENT";
var SMOOTHING_FACTOR = 1 / 3;
var lastTime;
var highWater = STANDARD_HIGHWATER;
var interval = STANDARD_INTERVAL;
var smoothingFactor = SMOOTHING_FACTOR;
var currentLag = 0;
var checkInterval;
var lagEventThreshold = -1;
var eventEmitter = new events.EventEmitter();


var DDOS = function () {
  var pctToBlock = (currentLag - highWater) / highWater;
```

```
  return Math.random() < pctToBlock;
};

/**
 *
 * @param {int} newInterval
 * @returns
 */
DDOS.interval = function (newInterval) {
  if (!newInterval) return interval;
  if (typeof newInterval !== "number")
    throw new Error("Interval must be a number.");

  newInterval = Math.round(newInterval);
  if (newInterval < 16)
    throw new Error("Interval should be greater than 16ms.");

  currentLag = 0;
  interval = newInterval;
  start();
  return interval;
};

DDOS.lag = function () {
  return Math.round(currentLag);
```

```
};


/**
 *
 * @param {int} newLag
 * @returns
 */
DDOS.maxLag = function (newLag) {
  if (!newLag) return highWater;


  if (typeof newLag !== "number") throw new Error("MaxLag must be a number.");
  newLag = Math.round(newLag);
  if (newLag < 10) throw new Error("Maximum lag should be greater than 10ms.");


  highWater = newLag;
  return highWater;
};


/**
 *
 * @param {int} newFactor
 * @returns
 */
DDOS.smoothingFactor = function (newFactor) {
  if (!newFactor) return smoothingFactor;
```

```
  if (typeof newFactor !== "number")
    throw new Error("NewFactor must be a number.");
  if (newFactor <= 0 || newFactor > 1)
    throw new Error("Smoothing factor should be in range
]0,1].");

  smoothingFactor = newFactor;
  return smoothingFactor;
};

DDOS.shutdown = function () {
  currentLag = 0;
  checkInterval = clearInterval(checkInterval);
  eventEmitter.removeAllListeners(LAG_EVENT);
};

DDOS.started = function () {
  return Boolean(checkInterval != null);
};

/**
 *
 * @param {object} fn
 * @param {int} threshold
 */
```

```
DDOS.onLag = function (fn, threshold) {
  if (typeof threshold === "number") {
    lagEventThreshold = threshold;
  } else {
    lagEventThreshold = DDOS.maxLag();
  }

  eventEmitter.on(LAG_EVENT, fn);
};

function start() {
  lastTime = Date.now();

  clearInterval(checkInterval);
  checkInterval = setInterval(function () {
    var now = Date.now();
    var lag = now - lastTime;
    lag = Math.max(0, lag - interval);
    currentLag = smoothingFactor * lag + (1 - smoothingFactor) * currentLag;
    lastTime = now;

    if (lagEventThreshold !== -1 && currentLag > lagEventThreshold) {
      eventEmitter.emit(LAG_EVENT, currentLag);
    }
  }, interval);
```

```
  checkInterval.unref();
}

start();

module.exports = DDOS;
```

Listing 5. Nastavak Anti DDOS logike kao i dodatni bezbedonosni mehanizmi za sprečavanje napada izviđanja kao i čoveka u sredini

U nastavku listinga definisani su dodatne metode za pomoć u radu kao što su provera sistema na performanse, da li postoje problematični parametri u paketima kao i izveštaji o metrici.

```
const pidusage = require("pidusage");
const os = require("os");
const v8 = require("v8");
const sendMetrics = require("./send-metrics");
const debug = require("debug")("express-status-monitor");

let eventLoopStats;

try {
  eventLoopStats = require("event-loop-stats");
} catch (error) {
  console.warn("event-loop-stats not found, ignoring event
```

```
loop metrics...");
}

/**
 *
 * @param {object} io
 * @param {object} span
 */
module.exports = (io, span) => {
  const defaultResponse = {
    2: 0,
    3: 0,
    4: 0,
    5: 0,
    count: 0,
    mean: 0,
    timestamp: Date.now(),
  };

  pidusage(process.pid, (err, stat) => {
    if (err) {
      debug(err);
      return;
    }

    const last = span.responses[span.responses.length - 1];
```

```
    stat.memory = stat.memory / 1024 / 1024;
    stat.load = os.loadavg();
    stat.timestamp = Date.now();
    stat.heap = v8.getHeapStatistics();

    if (eventLoopStats) {
      stat.loop = eventLoopStats.sense();
    }

    span.os.push(stat);
    if (
      !span.responses[0] ||
      (last.timestamp + span.interval) * 1000 < Date.now()
    ) {
      span.responses.push(defaultResponse);
    }

    if (span.os.length >= span.retention) span.os.shift();
    if (span.responses[0] && span.responses.length > span.retention)
      span.responses.shift();

    sendMetrics(io, span);
  });
};
```

Listing 6. Posmatrač sistema, generiše izveštaje o metrici sistema i obaveštava administratora ukoliko se dešava nešto što je nesvakidašnje

```
"use strict";


const axios = require("axios");


/**
 *
 * @param {object} promises
 * @returns
 */
function allSettled(promises) {
  const wrappedPromises = promises.map((p) =>
    Promise.resolve(p).then(
      (val) => ({ state: "fulfilled", value: val }),
      (err) => ({ state: "rejected", reason: err })
    )
  );


  return Promise.all(wrappedPromises);
}


/**
 *
```

```
 * @param {int} healthChecks
 * @returns
 */
module.exports = async (healthChecks) => {
  const checkPromises = [];

  (healthChecks || []).forEach((healthCheck) => {
    let uri =
`${healthCheck.protocol}://${healthCheck.host}`;

    if (healthCheck.port) {
      uri += `:${healthCheck.port}`;
    }

    uri += healthCheck.path;

    checkPromises.push(
      axios({
        url: uri,
        method: "GET",
      })
    );
  });

  const checkResults = [];
```

```
    return allSettled(checkPromises).then((results) => {
      results.forEach((result, index) => {
        if (result.state === "rejected") {
          checkResults.push({
            path: healthChecks[index].path,
            status: "failed",
          });
        } else {
          checkResults.push({
            path: healthChecks[index].path,
            status: "ok",
          });
        }
      });

      return checkResults;
    });
  };
```

Listing 7. Posmatrač mrežnih zahteva, analiza zahteva i proveravanje da li je došlo do bilo kakvih modifikacija

```
  /**
   *
   * @param {int} statusCode
   * @param {int} startTime
```

```
 * @param {int} spans
 */
module.exports = (statusCode, startTime, spans) => {
  const diff = process.hrtime(startTime);
  const responseTime = (diff[0] * 1e3 + diff[1]) * 1e-6;
  const category = Math.floor(statusCode / 100);

  spans.forEach((span) => {
    const last = span.responses[span.responses.length - 1];

    if (
      last !== undefined &&
      last.timestamp / 1000 + span.interval > Date.now() /
1000
    ) {
      last[category] += 1;
      last.count += 1;
      last.mean += (responseTime - last.mean) / last.count;
    } else {
      span.responses.push({
        2: category === 2 ? 1 : 0,
        3: category === 3 ? 1 : 0,
        4: category === 4 ? 1 : 0,
        5: category === 5 ? 1 : 0,
        count: 1,
        mean: responseTime,
```

```
        timestamp: Date.now(),
      });
    }
  });
};
```

Listing 8. Posmatrač zaglavlja paketa, validacija i analiza na modifikacije

```
const socketIo = require("socket.io");
const gatherOsMetrics = require("./gather-os-metrics");

let io;

const addSocketEvents = (socket, config) => {
  socket.emit("esm_start", config.spans);
  socket.on("esm_change", () => {
    socket.emit("esm_start", config.spans);
  });
};

/**
 *
 * @param {string} server
 * @param {object} config
 */
```

```
module.exports = (server, config) => {
  if (io === null || io === undefined) {
    if (config.websocket !== null) {
      io = config.websocket;
    } else {
      io = socketIo(server);
    }

    io.on("connection", (socket) => {
      if (config.authorize) {
        config
          .authorize(socket)
          .then((authorized) => {
            if (!authorized)
socket.disconnect("unauthorized");
            else addSocketEvents(socket, config);
          })
          .catch(() => socket.disconnect("unauthorized"));
      } else {
        addSocketEvents(socket, config);
      }
    });

    config.spans.forEach((span) => {
      span.os = [];
      span.responses = [];
```

```
      const interval = setInterval(
        () => gatherOsMetrics(io, span),
        span.interval * 1000
      );

      interval.unref();
    });
  }
};
```

Listing 9. Validator mrežnih paketa koji idu kroz sokete

```
const defaultConfig = require("./default-config");
/**
 *
 * @param {object} config
 * @returns
 */
module.exports = (config) => {
  if (!config) {
    return defaultConfig;
  }

  const mungeChartVisibility = (configChartVisibility) =>
{

Object.keys(defaultConfig.chartVisibility).forEach((key)
```

```
=> {
      if (configChartVisibility[key] === false) {
        defaultConfig.chartVisibility[key] = false;
      }
    });
    return defaultConfig.chartVisibility;
  };

  config.title =
    typeof config.title === "string" ? config.title :
defaultConfig.title;
  config.theme =
    typeof config.theme === "string" ? config.theme :
defaultConfig.theme;
  config.path =
    typeof config.path === "string" ? config.path :
defaultConfig.path;
  config.socketPath =
    typeof config.socketPath === "string"
      ? config.socketPath
      : defaultConfig.socketPath;
  config.spans =
    typeof config.spans === "object" ? config.spans :
defaultConfig.spans;
  config.port =
    typeof config.port === "number" ? config.port :
defaultConfig.port;
```

```
  config.websocket =
    typeof config.websocket === "object"
      ? config.websocket
      : defaultConfig.websocket;
  config.iframe =
    typeof config.iframe === "boolean" ? config.iframe :
defaultConfig.iframe;
  config.chartVisibility =
    typeof config.chartVisibility === "object"
      ? mungeChartVisibility(config.chartVisibility)
      : defaultConfig.chartVisibility;
  config.ignoreStartsWith =
    typeof config.path === "string"
      ? config.ignoreStartsWith
      : defaultConfig.ignoreStartsWith;

  config.healthChecks = Array.isArray(config.healthChecks)
    ? config.healthChecks
    : defaultConfig.healthChecks;

  return config;
};
```

Listing 10. Generalni validator sadržaja mrežnog paketa, njegovih parametara kao i sadržaja na potencijalne komande ili parametre koji mogu da naškode sistemu

Poslednji u listi listinga su metode za AES, DES i 3DES.

```
const crypto = require("crypto");


class AES {
  /**
   *
   * @param {string} val
   * @param {string} ENC_KEY
   * @param {string} IV
   * @returns
   */
  encrypt(val, ENC_KEY, IV) {
    let cipher = crypto.createCipheriv("aes-256-cbc",
ENC_KEY, IV);
    let encrypted = cipher.update(val, "utf8", "base64");
    encrypted += cipher.final("base64");


    return encrypted;
  }


  /**
   *
   * @param {string} encrypted
   * @param {string} ENC_KEY
   * @param {string} IV
   * @returns
```

```
    */
  decrypt(encrypted, ENC_KEY, IV) {
    let decipher = crypto.createDecipheriv("aes-256-cbc",
ENC_KEY, IV);
    let decrypted = decipher.update(encrypted, "base64",
"utf8");

    return decrypted + decipher.final("utf8");
  }
}

module.exports = AES;
```

Listing 11. AES klasa za šifrovanje i dešifrovanje podataka

```
const crypto = require("crypto");

class DES {
  /**
   *
   * @param {string} val
   * @param {string} ENC_KEY
   * @param {string} IV
   * @returns
   */
  encrypt(val, ENC_KEY, IV) {
    const cipher = crypto.createCipheriv("DES-CBC",
```

```
ENC_KEY, IV);
    let encrypted = cipher.update(val, "utf8", "hex");
    encrypted += cipher.final("hex");

    return encrypted;
  }

  /**
   *
   * @param {string} encrypted
   * @param {string} ENC_KEY
   * @param {string} IV
   * @returns
   */
  decrypt(encrypted, ENC_KEY, IV) {
    const decipher = crypto.createDecipheriv("DES-CBC",
ENC_KEY, IV);
    let decrypted = decipher.update(encrypted, "hex",
"utf8");
    decrypted += decipher.final("utf8");

    return decrypted;
  }
}
module.exports = DES;
```

Listing 12. DES klasa za šifrovanje i dešifrovanje podataka

```
const crypto = require("crypto");


class TRIPLE_DES {
  /**
   *
   * @param {string} text
   * @param {string} keyHex
   * @param {string} ivHex
   * @returns
   */
  decrypt(text, keyHex, ivHex) {
    ivHex = Buffer.from(ivHex, "utf8");
    keyHex = Buffer.from(keyHex, "utf8");


    const cipher = crypto.createDecipheriv("DES-EDE3-CBC",
keyHex, ivHex);
    let encrypted = cipher.update(text, "base64", "utf8");
    encrypted += cipher.final("utf8");


    return encrypted;
  }


  /**
   *
```

```
    * @param {string} encrypted
    * @param {string} keyHex
    * @param {string} ivHex
    * @returns
    */
  encrypt(encrypted, keyHex, ivHex) {
    ivHex = Buffer.from(ivHex, "utf8");
    keyHex = Buffer.from(keyHex, "utf8");

    const cipher = crypto.createCipheriv("DES-EDE3-CBC", keyHex, ivHex);
    let decrypted = cipher.update(encrypted, "utf8", "base64");
    decrypted += cipher.final("base64");

    return decrypted;
  }
}
module.exports = TRIPLE_DES;
```

Listing 13. 3DES klasa za šifrovanje i dešifrovanje podataka

Bibliografija

[1] Pavlović N., Šarac M., Adamović S., Sarčević M.,Khaleel A. and Maček N., An Approach to Adding Simple Interface as Security Gateway Architecture for IoT Device, Multimedia Tools and Applications, 2021

[2] C. Rong, G. Zhao, L. Yan, E. Cayirci and H. Cheng, "Computer and Information Security Handbook (Second Edition)", 2013, pp. 345-361 doi: 10.1016/B978-0-12-394397-2.00018-0

[3] Shouran, Z., Ashari, A., & Kuntoro, T. (2019). Internet of things (iot) of smart home: Privacy and security. International Journal of Computer Applications, 182(39), 3-8. doi:10.5120/ijca2019918450

[4] G. Kambourakis, C. Kolias and A. Stavrou, "The Mirai botnet and the IoT Zombie Armies," MILCOM 2017 - 2017 IEEE Military Communications Conference (MILCOM), Baltimore, MD, USA, 2017, pp. 267-272, doi: 10.1109/MILCOM.2017.8170867.

[5] Q. Lu, Z. Zhang, and S. Lü, "Home energy management in smart households: Optimal appliance scheduling model with photovoltaic energy storage system," Energy Reports, vol. 6, pp. 2450–2462, 2020.

[6] M. Y. Shabalov, Y. L. Zhukovskiy, A. D. Buldysko, B. Gil, and V. V. Starshaia, "The influence of technological changes in energy efficiency on the infrastructure deterioration in the energy sector," Energy Reports, vol. 7, pp. 2664–2680, 2021.

[7] M. Nawir, A. Amir, N. Yaakob and O. B. Lynn, "Internet of Things (IoT): Taxonomy of security attacks," 2016 3rd International Conference on Electronic Design (ICED), Phuket, 2016, pp. 321-326, doi: 10.1109/ICED.2016.7804660

[8] R. Mahmoud, T. Yousuf, F. Aloul and I. Zualkernan, "Cyber Security and internet of Things: Vulnerabilities, Threats, Intruders and Attacks," 2015 Journal of Cyber Security and Mobility, 2015, pp. 65-88, doi: 10.13052/jcsm2245-1439.414

[9] D. Usha and M. Bobby, "Privacy issues in smart home devices using internet of things – a survey", 2018 International Journal of Advanced Research 6(10): 566-568 doi: 10.21474/IJAR01/7839

[10] O. Alrawi, C. Lever, M. Antonakakis and F. Monrose, "SoK: Security Evalua-tion of Home-Based IoT Deployments," 2019 IEEE Symposium on Security and Privacy (SP), San Fransisco, CA, US, 2019 doi: 10.1109/SP.2019.00013

[11] Noura, H., Chehab, A., Sleem, L. et al. "One round cipher algorithm for multi-media IoT devices", Multimed Tools Appl 77, 18383–18413 (2018). doi: 10.1007/s11042-018-5660-y

[12] Thakur, Kutub. "Analysis of Denial of Services (DOS) Attacks and Prevention Tech-niques", International journal of engineering research and technology 4 (2015)

[13] M. Goyal and M. Dutta, "Intrusion Detection of Wormhole Attack in IoT: A Review," 2018 International Conference on Circuits and Systems in Digital Enterprise Technology (ICCSDET), Kottayam, India, 2018, pp. 1-5, doi: 10.1109/ICCSDET.2018.8821160.

[14] J. Buford, H. Yu and E. Keong Lua, "P2P Networking and Applications", 2008, San Francisco, CA, USA.

[15] A. Kavianpour and M. C. Anderson, "An Overview of Wireless Network Securi-ty," 2017 IEEE 4th International Conference on Cyber Security and Cloud Computing (CSCloud), New York, NY, 2017, pp. 306-309, doi: 10.1109/CSCloud.2017.45.

[16] Haroon Iqbal, Sameena Naaz, "Wireshark as a Tool for Detection of Various LAN Attacks," International Journal of Computer Sciences and Engineering, Vol.7, Is-sue.5, pp.833-837, 2019, doi: 10.26438/ijcse/v7i5.833837

[17] M. Faheem Mushtaq, S. Jamel, A. Hassan Disina, Z. A. Pindar, N. Shafinaz Ahmad Shakir and M. Mat Deris, "A Survey on the Cryptographic Encryption Algorithms", International Journal of Advanced Computer Science and Appli-cations(IJACSA), 8(11), 2017, doi: 10.14569/IJACSA.2017.081141

[18] H. Sun, D. Bonetta, C. Humer, and W. Binder, "Efficient dynamic analysis for Node.js", 2018, 27th International Conference on Compiler Construction (CC 2018). Association for Computing Machinery, New York, NY, USA, 196–206. doi: 10.1145/3178372.3179527

[19] R. Sarma and F. A. Barbhuiya, "Internet of Things: Attacks and Defences," 2019 7th International Conference on Smart Computing & Communications (ICSCC), Sarawak, Malaysia, Malaysia, 2019, pp. 1-5, doi: 10.1109/ICSCC.2019.8843649

[20] S. Cirani, G. Ferrari, N. Iotti and M. Picone, "The IoT hub: a fog node for seam-less management of heterogeneous connected smart objects," 2015 12th An-nual IEEE International Conference on Sensing, Communication, and Net-working - Workshops (SECON Workshops), Seattle, WA, 2015, pp. 1-6, doi: 10.1109/SECONW.2015.7328145

[21] Xin Zhou and Xiaofei Tang, "Research and implementation of RSA algorithm for encryption and decryption," Proceedings of 2011 6th International Forum on Strate-gic Technology, Harbin, Heilongjiang, 2011, pp. 1118-1121, doi: 10.1109/IFOST.2011.6021216

[22] B. Bhat, A. W. Ali and A. Gupta, "DES and AES performance evaluation," In-terna-

tional Conference on Computing, Communication & Automation, Noida, 2015, pp. 887-890, doi: 10.1109/CCAA.2015.7148500

[23] M. Andriansyah, M. Subali, I. Purwanto, S. A. Irianto and R. A. Pramono, "e-KTP as the basis of home security system using arduino UNO," 2017 4th In-ternational Conference on Computer Applications and Information Processing Technology (CAIPT), Kuta Bali, 2017, pp. 1-5, doi: 10.1109/CAIPT.2017.8320693.

[24] W. Mielczarek and T. Moń, "USB Data Capture and Analysis in Windows Using USBPcap and Wireshark," 2017, doi: 10.1007/978-3-319-19419-6_41

[25] Claudio Zunino, Adriano Valenzano, Roman Obermaisser, Stig Petersen, Factory Communications at the Dawn of the Fourth Industrial Revolution, Computer Standards & Interfaces, Volume 71, 2020.

[26] Kumar, N. and Mallick, P., 2018. Blockchain technology for security issues and challenges in IoT. Procedia Computer Science, 132, pp.1815-1823.

[27] Miraz, M., 2020. Blockchain Of Things (Bcot): The Fusion Of Blockchain And Iot Technologies. [online] Arxiv.org. Available at: <https://arxiv.org/pdf/1910.06898> [Accessed 1 November 2020].

[28] Hassija, V., Chamola, V., Saxena, V., Jain, D., Goyal, P. And Sikdar, B., 2020. A Survey On Iot Security: Application Areas, Security Threats, And Solution Architectures. [online] Ece.nus.edu.sg. Available at: <https://www.ece.nus.edu.sg/stfpage/bsikdar/papers/access_19.pdf> [Accessed 4 November 2020].

[29] S. Soni and B. Bhushan, "A Comprehensive survey on Blockchain: Working, security analysis, privacy threats and potential applications," 2019 2nd International Conference on Intelligent Computing, Instrumentation and Control Technologies (ICICICT), 2019, pp. 922-926, doi: 10.1109/ICICICT46008.2019.8993210.

[30] Andersen, M., Kolb, J., Chen, K., Fierro, G., Culler, D. and Popa, R., 2017. WAVE: A Decentralized Authorization System For Iot Via Blockchain Smart Contracts. [online] Www2.eecs.berkeley.edu. Available at: <https://www2.eecs.berkeley.edu/Pubs/TechRpts/2017/EECS-2017-234.pdf> [Accessed 4 November 2020].

[31] Atlam, H.F., Alenezi, A., Alassafi, M.O., & Wills, G. (2018). Blockchain with Internet of Things: Benefits, Challenges, and Future Directions. International Journal of Intelligent Systems and Applications, 10, 40-48.

[32] Makhdoom, Imran & Abolhasan, Mehran & Ni, Wei. (2018). Blockchain for IoT: The Challenges and a Way Forward. 594-605. 10.5220/0006905605940605.

[33] Panarello, A., Tapas, N., Merlino, G., Longo, F., & Puliafito, A. (2018). Blockchain and IoT Integration: A Systematic Survey. Sensors (Basel, Switzerland), 18.

[34] Dwivedi, A., G. Srivastava, Shalini Dhar and R. Singh. "A Decentralized Privacy-Preserving Healthcare Blockchain for IoT." Sensors (Basel, Switzerland) 19 (2019): n. pag.

[35] Tariq, Noshina & Asim, Muhammad & Al-Obeidat, Feras & Farooqi, Muhammad & Baker, Thar & Hammoudeh, Mohammad & Ghafir, Ibrahim. (2019). The Security of Big Data in Fog-Enabled IoT Applications Including Blockchain: A Survey. Sensors. 19. 1788. 10.3390/s19081788.

[36] Kim, S.-K.; Kim, U.-M.; Huh, J.-H. A Study on Improvement of Blockchain Application to Overcome Vulnerability of IoT Multiplatform Security. Energies 2019, 12, 402.

[37] F. A. Alaba, M. Othman, I. A. Hashem, and F. Alotaibi, "Internet of things security: A survey," Journal of Network and Computer Applications, vol. 88, pp. 10–28, 2017.

[38] D. E. Kouicem, A. Bouabdallah, and H. Lakhlef, "Internet of things security: A top-down survey," Computer Networks, vol. 141, pp. 199–221, 2018.

[39] I. Andrea, C. Chrysostomou, and G. Hadjichristofi, "Internet of things: Security vulnerabilities and challenges," 2015 IEEE Symposium on Computers and Communication (ISCC), 2015.

[40] P. C. van Oorschot and S. W. Smith, "The Internet of Things: Security Challenges," in IEEE Security & Privacy, vol. 17, no. 5, pp. 7-9, Sept.-Oct. 2019, doi: 10.1109/MSEC.2019.2925918.

[41] Chen, K., Zhang, S., Li, Z. et al. Internet-of-Things Security and Vulnerabilities: Taxonomy, Challenges, and Practice. J Hardw Syst Secur 2, 97–110 (2018). https://doi.org/10.1007/s41635-017-0029-7

[42] M. Aydos, Y. Vural, and A. Tekerek, "Assessing risks and threats with layered approach to internet of things security," Measurement and Control, vol. 52, no. 5-6, pp. 338–353, 2019.

[43] S. Elbouanani, M. A. E. Kiram and O. Achbarou, "Introduction to the Internet of Things security: Standardization and research challenges," 2015 11th International Conference on Information Assurance and Security (IAS), 2015, pp. 32-37, doi: 10.1109/ISIAS.2015.7492741.

[44] K. Tange, M. De Donno, X. Fafoutis and N. Dragoni, "A Systematic Survey of Industrial Internet of Things Security: Requirements and Fog Computing

Opportunities," in IEEE Communications Surveys & Tutorials, vol. 22, no. 4, pp. 2489-2520, Fourthquarter 2020, doi: 10.1109/COMST.2020.3011208.

[45] Li, S., Tryfonas, T. and Li, H. (2016), "The Internet of Things: a security point of view", Internet Research, Vol. 26 No. 2, pp. 337-359. https://doi.org/10.1108/IntR-07-2014-0173

[46] M. binti Mohamad Noor and W. H. Hassan, "Current research on internet of things (IOT) security: A survey," Computer Networks, vol. 148, pp. 283–294, 2019.

[47] M. M. Hossain, M. Fotouhi and R. Hasan, "Towards an Analysis of Security Issues, Challenges, and Open Problems in the Internet of Things," 2015 IEEE World Congress on Services, 2015, pp. 21-28, doi: 10.1109/SERVICES.2015.12.

[48] B. Alzahrani and N. Fotiou, "Enhancing internet of things security using software-defined networking," Journal of Systems Architecture, vol. 110, p. 101779, 2020.

[49] C. Maple, "Security and privacy in the internet of things," Journal of Cyber Policy, vol. 2, no. 2, pp. 155–184, 2017.

[50] J. Granjal, E. Monteiro and J. Sá Silva, "Security for the Internet of Things: A Survey of Existing Protocols and Open Research Issues," in IEEE Communications Surveys & Tutorials, vol. 17, no. 3, pp. 1294-1312, thirdquarter 2015, doi: 10.1109/COMST.2015.2388550.

[51] P. Mishra, A. Biswal, S. Garg, R. Lu, M. Tiwary and D. Puthal, "Software Defined Internet of Things Security: Properties, State of the Art, and Future Research," in IEEE Wireless Communications, vol. 27, no. 3, pp. 10-16, June 2020, doi: 10.1109/MWC.001.1900318.

[52] V. Adat and B. B. Gupta, "Security in internet of things: Issues, challenges, taxonomy, and architecture," Telecommunication Systems, vol. 67, no. 3, pp. 423–441, 2017.

[53] A. Sadeghi, C. Wachsmann and M. Waidner, "Security and privacy challenges in industrial Internet of Things," 2015 52nd ACM/EDAC/IEEE Design Automation Conference (DAC), 2015, pp. 1-6, doi: 10.1145/2744769.2747942..

[54] S. Sicari, A. Rizzardi, L. A. Grieco, and A. Coen-Porisini, "Security, privacy and trust in internet of things: The road ahead," Computer Networks, vol. 76, pp. 146–164, 2015.

[55] N. Kshetri, "Can Blockchain Strengthen the Internet of Things?," in IT Professional, vol. 19, no. 4, pp. 68-72, 2017, doi: 10.1109/MITP.2017.3051335.

[56] K. Sha, W. Wei, T. Andrew Yang, Z. Wang, and W. Shi, "On security challenges and open issues in internet of things," Future Generation Computer Systems, vol.

83, pp. 326–337, 2018.

[57] J. R. C. Nurse, S. Creese and D. De Roure, "Security Risk Assessment in Internet of Things Systems," in IT Professional, vol. 19, no. 5, pp. 20-26, 2017, doi: 10.1109/MITP.2017.3680959.

[58] Y. Yang, L. Wu, G. Yin, L. Li and H. Zhao, "A Survey on Security and Privacy Issues in Internet-of-Things," in IEEE Internet of Things Journal, vol. 4, no. 5, pp. 1250-1258, Oct. 2017, doi: 10.1109/JIOT.2017.2694844.

[59] P. I. Radoglou Grammatikis, P. G. Sarigiannidis, and I. D. Moscholios, “Securing the internet of things: Challenges, threats and solutions,” Internet of Things, vol. 5, pp. 41–70, 2019.

[60] E. Vasilomanolakis, J. Daubert, M. Luthra, V. Gazis, A. Wiesmaier and P. Kikiras, "On the Security and Privacy of Internet of Things Architectures and Systems," 2015 International Workshop on Secure Internet of Things (SIoT), 2015, pp. 49-57, doi: 10.1109/SIOT.2015.9.

[61] M. Elkhodr, S. Shahrestani, and H. Cheung, “The internet of things : New interoperability, management and security challenges,” International Journal of Network Security & Its Applications, vol. 8, no. 2, pp. 85–102, 2016.

[62] K. M. Sadique, R. Rahmani, and P. Johannesson, “Towards security on internet of things: Applications and challenges in Technology,” Procedia Computer Science, vol. 141, pp. 199–206, 2018.

[63] C. Tankard, “The security issues of the internet of things,” Computer Fraud & Security, vol. 2015, no. 9, pp. 11–14, 2015.

[64] R. Li, T. Song, B. Mei, H. Li, X. Cheng and L. Sun, "Blockchain for Large-Scale Internet of Things Data Storage and Protection," in IEEE Transactions on Services Computing, vol. 12, no. 5, pp. 762-771, 1 Sept.-Oct. 2019, doi: 10.1109/TSC.2018.2853167.

[65] M. Husamuddin and M. Qayyum, "Internet of Things: A study on security and privacy threats," 2017 2nd International Conference on Anti-Cyber Crimes (ICACC), 2017, pp. 93-97, doi: 10.1109/Anti-Cybercrime.2017.7905270.

[66] D. Mendez Mena, I. Papapanagiotou, and B. Yang, “Internet of things: Survey on security,” Information Security Journal: A Global Perspective, vol. 27, no. 3, pp. 162–182, 2018.

[67] S. Vashi, J. Ram, J. Modi, S. Verma and C. Prakash, "Internet of Things (IoT): A vision, architectural elements, and security issues," 2017 International Conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud) (I-SMAC), 2017, pp. 492-

496, doi: 10.1109/I-SMAC.2017.8058399.

[68] I. Yaqoob, E. Ahmed, M. H. Rehman, A. I. Ahmed, M. A. Al-garadi, M. Imran, and M. Guizani, “The rise of ransomware and emerging security challenges in the internet of things,” Computer Networks, vol. 129, pp. 444–458, 2017.

[69] L. Malina, J. Hajny, R. Fujdiak, and J. Hosek, “On perspective of security and privacy-preserving solutions in the internet of things,” Computer Networks, vol. 102, pp. 83–95, 2016.

[70] R. Billure, V. M. Tayur and Mahesh V, "Internet of Things - a study on the security challenges," 2015 IEEE International Advance Computing Conference (IACC), 2015, pp. 247-252, doi: 10.1109/IADCC.2015.7154707.

[71] A. E. Omolara, A. Alabdulatif, O. I. Abiodun, M. Alawida, A. Alabdulatif, W. H. Alshoura, and H. Arshad, “The internet of things security: A survey encompassing unexplored areas and New Insights,” Computers & Security, p. 102494, 2021.

[72] M. Khari, M. Kumar, S. Vij, P. Pandey and Vaishali, "Internet of Things: Proposed security aspects for digitizing the world," 2016 3rd International Conference on Computing for Sustainable Global Development (INDIACom), 2016, pp. 2165-2170.

[73] O. Alphand et al., "IoTChain: A blockchain security architecture for the Internet of Things," 2018 IEEE Wireless Communications and Networking Conference (WCNC), 2018, pp. 1-6, doi: 10.1109/WCNC.2018.8377385.

[74] S. Kraijak and P. Tuwanut, "A survey on internet of things architecture, protocols, possible applications, security, privacy, real-world implementation and future trends," 2015 IEEE 16th International Conference on Communication Technology (ICCT), 2015, pp. 26-31, doi: 10.1109/ICCT.2015.7399787.

[75] M. Banerjee, J. Lee, and K.-K. R. Choo, “A blockchain future for internet of things security: A position paper,” Digital Communications and Networks, vol. 4, no. 3, pp. 149–160, 2018.

[76] H. -N. Dai, Z. Zheng and Y. Zhang, "Blockchain for Internet of Things: A Survey," in IEEE Internet of Things Journal, vol. 6, no. 5, pp. 8076-8094, Oct. 2019, doi: 10.1109/JIOT.2019.2920987.

[77] L. Tseng, L. Wong, S. Otoum, M. Aloqaily and J. B. Othman, "Blockchain for Managing Heterogeneous Internet of Things: A Perspective Architecture," in IEEE Network, vol. 34, no. 1, pp. 16-23, January/February 2020, doi: 10.1109/MNET.001.1900103.

[78] W. Viriyasitavat, L. D. Xu, Z. Bi and D. Hoonsopon, "Blockchain Technology for

Applications in Internet of Things—Mapping From System Design Perspective," in IEEE Internet of Things Journal, vol. 6, no. 5, pp. 8155-8168, Oct. 2019, doi: 10.1109/JIOT.2019.2925825.

[79] Y. Gupta, R. Shorey, D. Kulkarni and J. Tew, "The applicability of blockchain in the Internet of Things," 2018 10th International Conference on Communication Systems & Networks (COMSNETS), 2018, pp. 561-564, doi: 10.1109/COMSNETS.2018.8328273.

[80] T. M. Fernández-Caramés and P. Fraga-Lamas, "A Review on the Use of Blockchain for the Internet of Things," in IEEE Access, vol. 6, pp. 32979-33001, 2018, doi: 10.1109/ACCESS.2018.2842685.

[81] M. A. Ferrag, M. Derdour, M. Mukherjee, A. Derhab, L. Maglaras and H. Janicke, "Blockchain Technologies for the Internet of Things: Research Issues and Challenges," in IEEE Internet of Things Journal, vol. 6, no. 2, pp. 2188-2204, April 2019, doi: 10.1109/JIOT.2018.2882794.

[82] A. D. Dwivedi, L. Malina, P. Dzurenda and G. Srivastava, "Optimized Blockchain Model for Internet of Things based Healthcare Applications," 2019 42nd International Conference on Telecommunications and Signal Processing (TSP), 2019, pp. 135-139, doi: 10.1109/TSP.2019.8769060.

[83] Y. Sun, L. Zhang, G. Feng, B. Yang, B. Cao and M. A. Imran, "Blockchain-Enabled Wireless Internet of Things: Performance Analysis and Optimal Communication Node Deployment," in IEEE Internet of Things Journal, vol. 6, no. 3, pp. 5791-5802, June 2019, doi: 10.1109/JIOT.2019.2905743.

[84] B. Alotaibi, "Utilizing Blockchain to Overcome Cyber Security Concerns in the Internet of Things: A Review," in IEEE Sensors Journal, vol. 19, no. 23, pp. 10953-10971, 1 Dec.1, 2019, doi: 10.1109/JSEN.2019.2935035.

[85] G. C. Polyzos and N. Fotiou, "Blockchain-Assisted Information Distribution for the Internet of Things," 2017 IEEE International Conference on Information Reuse and Integration (IRI), 2017, pp. 75-78, doi: 10.1109/IRI.2017.83.

[86] G. Sagirlar, B. Carminati, E. Ferrari, J. D. Sheehan and E. Ragnoli, "Hybrid-IoT: Hybrid Blockchain Architecture for Internet of Things - PoW Sub-Blockchains," 2018 IEEE International Conference on Internet of Things (iThings) and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom) and IEEE Smart Data (SmartData), 2018, pp. 1007-1016, doi: 10.1109/Cybermatics_2018.2018.00189.

[87] B. Cao et al., "When Internet of Things Meets Blockchain: Challenges in Distributed Consensus," in IEEE Network, vol. 33, no. 6, pp. 133-139, Nov.-Dec. 2019, doi: 10.1109/MNET.2019.1900002.

[88] S. Cha, J. Chen, C. Su and K. Yeh, "A Blockchain Connected Gateway for BLE-Based Devices in the Internet of Things," in IEEE Access, vol. 6, pp. 24639-24649, 2018, doi: 10.1109/ACCESS.2018.2799942.

[89] A. Rejeb, J. G. Keogh, and H. Treiblmaier, "Leveraging the internet of things and blockchain technology in Supply Chain Management," Future Internet, vol. 11, no. 7, p. 161, 2019.

[90] C. Ye, W. Cao, and S. Chen, "Security challenges of blockchain in internet of things: Systematic literature review," Transactions on Emerging Telecommunications Technologies, vol. 32, no. 8, 2020.

[91] M. Kamran, H. U. Khan, W. Nisar, M. Farooq, and S.-U. Rehman, "Blockchain and internet of things: A bibliometric study," Computers & Electrical Engineering, vol. 81, p. 106525, 2020.

[92] H. Honar Pajooh, M. Rashid, F. Alam, and S. Demidenko, "Multi-layer blockchain-based security architecture for internet of things," Sensors, vol. 21, no. 3, p. 772, 2021.

[93] X. Zhu and Y. Badr, "Identity management systems for the internet of things: A survey towards Blockchain Solutions," Sensors, vol. 18, no. 12, p. 4215, 2018.

[94] Y. Liu, K. Wang, K. Qian, M. Du and S. Guo, "Tornado: Enabling Blockchain in Heterogeneous Internet of Things Through a Space-Structured Approach," in IEEE Internet of Things Journal, vol. 7, no. 2, pp. 1273-1286, Feb. 2020, doi: 10.1109/JIOT.2019.2954128.

[95] C. Ge, Z. Liu, and L. Fang, "A blockchain based decentralized data security mechanism for the internet of things," Journal of Parallel and Distributed Computing, vol. 141, pp. 1–9, 2020.

[96] O. Cheikhrouhou and A. Koubâa, "BlockLoc: Secure Localization in the Internet of Things using Blockchain," 2019 15th International Wireless Communications & Mobile Computing Conference (IWCMC), 2019, pp. 629-634, doi: 10.1109/IWCMC.2019.8766440.

[97] B. Cao, X. Wang, W. Zhang, H. Song and Z. Lv, "A Many-Objective Optimization Model of Industrial Internet of Things Based on Private Blockchain," in IEEE Network, vol. 34, no. 5, pp. 78-83, September/October 2020, doi: 10.1109/MNET.011.1900536.

[98] B. Yu, J. Wright, S. Nepal, L. Zhu, J. Liu and R. Ranjan, "IoTChain: Establishing Trust in the Internet of Things Ecosystem Using Blockchain," in IEEE Cloud Computing, vol. 5, no. 4, pp. 12-23, Jul./Aug. 2018, doi: 10.1109/MCC.2018.043221010.

[99] W. Viriyasitavat, L. Da Xu, Z. Bi and A. Sapsomboon, "New Blockchain-Based Architecture for Service Interoperations in Internet of Things," in IEEE Transactions on Computational Social Systems, vol. 6, no. 4, pp. 739-748, Aug. 2019, doi: 10.1109/TCSS.2019.2924442.

[100] N. Fabiano, "Internet of Things and Blockchain: Legal Issues and Privacy. The Challenge for a Privacy Standard," 2017 IEEE International Conference on Internet of Things (iThings) and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom) and IEEE Smart Data (SmartData), 2017, pp. 727-734, doi: 10.1109/iThings-GreenCom-CPSCom-SmartData.2017.112.

[101] X. Ding, J. Guo, D. Li and W. Wu, "An Incentive Mechanism for Building a Secure Blockchain-Based Internet of Things," in IEEE Transactions on Network Science and Engineering, vol. 8, no. 1, pp. 477-487, 1 Jan.-March 2021, doi: 10.1109/TNSE.2020.3040446.

[102] N. Fabiano, "The Internet of Things ecosystem: The blockchain and privacy issues. The challenge for a global privacy standard," 2017 International Conference on Internet of Things for the Global Community (IoTGC), 2017, pp. 1-7, doi: 10.1109/IoTGC.2017.8008970.

[103] G. Ali, N. Ahmad, Y. Cao, M. Asif, H. Cruickshank, and Q. E. Ali, “Blockchain based permission delegation and access control in internet of things (baci),” Computers & Security, vol. 86, pp. 318–334, 2019.

[104] Y. Cao, F. Jia and G. Manogaran, "Efficient Traceability Systems of Steel Products Using Blockchain-Based Industrial Internet of Things," in IEEE Transactions on Industrial Informatics, vol. 16, no. 9, pp. 6004-6012, Sept. 2020, doi: 10.1109/TII.2019.2942211.

[105] Bo Tang, Hongjuan Kang, Jingwen Fan, Qi Li, and Ravi Sandhu. 2019. IoT Passport: A Blockchain-Based Trust Framework for Collaborative Internet-of-Things. In Proceedings of the 24th ACM Symposium on Access Control Models and Technologies (SACMAT '19). Association for Computing Machinery, New York, NY, USA, 83–92. DOI:https://doi.org/10.1145/3322431.3326327

[106] S. Pal, T. Rabehaja, A. Hill, M. Hitchens and V. Varadharajan, "On the Integration of Blockchain to the Internet of Things for Enabling Access Right Delegation," in IEEE Internet of Things Journal, vol. 7, no. 4, pp. 2630-2639, April 2020, doi: 10.1109/JIOT.2019.2952141.

[107] Rui Zhang, Rui Xue, and Ling Liu. 2019. Security and Privacy on Blockchain. ACM Comput. Surv. 52, 3, Article 51 (July 2019), 34 pages. DOI:https://doi.org/10.1145/3316481

[108] H. Halpin and M. Piekarska, "Introduction to Security and Privacy on the Blockchain," 2017 IEEE European Symposium on Security and Privacy Workshops (EuroS&PW), 2017, pp. 1-3, doi: 10.1109/EuroSPW.2017.43.

[109] A. Dorri, S. S. Kanhere, R. Jurdak and P. Gauravaram, "Blockchain for IoT security and privacy: The case study of a smart home," 2017 IEEE International Conference on Pervasive Computing and Communications Workshops (PerCom Workshops), 2017, pp. 618-623, doi: 10.1109/PERCOMW.2017.7917634.

[110] Y. Yu, Y. Li, J. Tian and J. Liu, "Blockchain-Based Solutions to Security and Privacy Issues in the Internet of Things," in IEEE Wireless Communications, vol. 25, no. 6, pp. 12-18, December 2018, doi: 10.1109/MWC.2017.1800116.

[111] E. Zaghloul, T. Li, M. W. Mutka and J. Ren, "Bitcoin and Blockchain: Security and Privacy," in IEEE Internet of Things Journal, vol. 7, no. 10, pp. 10288-10313, Oct. 2020, doi: 10.1109/JIOT.2020.3004273.

[112] B. K. Mohanta, D. Jena, S. Ramasubbareddy, M. Daneshmand and A. H. Gandomi, "Addressing Security and Privacy Issues of IoT Using Blockchain Technology," in IEEE Internet of Things Journal, vol. 8, no. 2, pp. 881-888, 15 Jan.15, 2021, doi: 10.1109/JIOT.2020.3008906.

[113] J. Wan, J. Li, M. Imran, D. Li and Fazal-e-Amin, "A Blockchain-Based Solution for Enhancing Security and Privacy in Smart Factory," in IEEE Transactions on Industrial Informatics, vol. 15, no. 6, pp. 3652-3660, June 2019, doi: 10.1109/TII.2019.2894573.

[114] M. A. Ferrag, L. Shu, X. Yang, A. Derhab and L. Maglaras, "Security and Privacy for Green IoT-Based Agriculture: Review, Blockchain Solutions, and Challenges," in IEEE Access, vol. 8, pp. 32031-32053, 2020, doi: 10.1109/ACCESS.2020.2973178.

[115] R. Henry, A. Herzberg and A. Kate, "Blockchain Access Privacy: Challenges and Directions," in IEEE Security & Privacy, vol. 16, no. 4, pp. 38-45, July/August 2018, doi: 10.1109/MSP.2018.3111245.

[116] O. B. Mora, R. Rivera, V. M. Larios, J. R. Beltrán-Ramírez, R. Maciel and A. Ochoa, "A Use Case in Cybersecurity based in Blockchain to deal with the security and privacy of citizens and Smart Cities Cyberinfrastructures," 2018 IEEE International Smart Cities Conference (ISC2), 2018, pp. 1-4, doi: 10.1109/ISC2.2018.8656694.

[117] K. Azbeg, O. Ouchetto, S. J. Andaloussi, L. Fetjah and A. Sekkaki, "Blockchain and IoT for Security and Privacy: A Platform for Diabetes Self-management," 2018 4th International Conference on Cloud Computing Technologies and Applications (Cloudtech), 2018, pp. 1-5, doi: 10.1109/CloudTech.2018.8713343.

[118] A. S. Sani et al., "Xyreum: A High-Performance and Scalable Blockchain for IIoT Security and Privacy," 2019 IEEE 39th International Conference on Distributed Computing Systems (ICDCS), 2019, pp. 1920-1930, doi: 10.1109/ICDCS.2019.00190.

[119] G. Zyskind, O. Nathan and A. '. Pentland, "Decentralizing Privacy: Using Blockchain to Protect Personal Data," 2015 IEEE Security and Privacy Workshops, 2015, pp. 180-184, doi: 10.1109/SPW.2015.27.

[120] B. K. Mohanta, U. Satapathy, S. S. Panda and D. Jena, "A Novel Approach to Solve Security and Privacy Issues for IoT Applications Using Blockchain," 2019 International Conference on Information Technology (ICIT), 2019, pp. 394-399, doi: 10.1109/ICIT48102.2019.00076.

[121] F. Chen, Z. Xiao, L. Cui, Q. Lin, J. Li, and S. Yu, "Blockchain for internet of things applications: A review and open issues," Journal of Network and Computer Applications, vol. 172, p. 102839, 2020.

[122] Alfandi, O., Khanji, S., Ahmad, L. et al. A survey on boosting IoT security and privacy through blockchain. Cluster Comput 24, 37–55 (2021). https://doi.org/10.1007/s10586-020-03137-8

[123] O. Oksiiuk and I. Dmyrieva, "Security and privacy issues of blockchain technology," 2020 IEEE 15th International Conference on Advanced Trends in Radioelectronics, Telecommunications and Computer Engineering (TCSET), 2020, pp. 1-5, doi: 10.1109/TCSET49122.2020.235489.

[124] B. S. Egala, A. K. Pradhan, V. Badarla and S. P. Mohanty, "Fortified-Chain: A Blockchain-Based Framework for Security and Privacy-Assured Internet of Medical Things With Effective Access Control," in IEEE Internet of Things Journal, vol. 8, no. 14, pp. 11717-11731, 15 July15, 2021, doi: 10.1109/JIOT.2021.3058946.

[125] T. Nguyen, N. Tran, L. Loven, J. Partala, M. Kechadi and S. Pirttikangas, "Privacy-Aware Blockchain Innovation for 6G: Challenges and Opportunities," 2020 2nd 6G Wireless Summit (6G SUMMIT), 2020, pp. 1-5, doi: 10.1109/6GSUMMIT49458.2020.9083832.

[126] A. Kosba, A. Miller, E. Shi, Z. Wen and C. Papamanthou, "Hawk: The Blockchain Model of Cryptography and Privacy-Preserving Smart Contracts," 2016 IEEE Symposium on Security and Privacy (SP), 2016, pp. 839-858, doi: 10.1109/SP.2016.55.

[127] T. Ahram, A. Sargolzaei, S. Sargolzaei, J. Daniels and B. Amaba, "Blockchain technology innovations," 2017 IEEE Technology & Engineering Management Conference (TEMSCON), 2017, pp. 137-141, doi: 10.1109/TEMSCON.2017.7998367.

[128] D. Arora, S. Gautham, H. Gupta and B. Bhushan, "Blockchain-based Security Solutions to Preserve Data Privacy And Integrity," 2019 International Conference on Computing, Communication, and Intelligent Systems (ICCCIS), 2019, pp. 468-472, doi: 10.1109/ICCCIS48478.2019.8974503.

[129] W. Gao, W. G. Hatcher and W. Yu, "A Survey of Blockchain: Techniques, Applications, and Challenges," 2018 27th International Conference on Computer Communication and Networks (ICCCN), 2018, pp. 1-11, doi: 10.1109/ICCCN.2018.8487348.

[130] R. Mahmoud, T. Yousuf, F. Aloul and I. Zualkernan, "Cyber Security and internet of Things: Vulnerabilities, Threats, Intruders and Attacks," 2015 Journal of Cyber Security and Mobility, 2015, pp. 65-88, doi: 10.13052/jcsm2245-1439.414