Razvoj sustava za računalnu sigurnost prema arhitekturi nultog povjerenja

Borina, Mateo

Sveučilište Jurja Dobrile u Puli
Fakultet informatike u Puli

Mateo Borina

Razvoj sustava za računalnu sigurnost prema arhitekturi nultog povjerenja

Diplomski rad

Pula, lipanj 2024.

Sveučilište Jurja Dobrile u Puli
Fakultet informatike u Puli

Razvoj sustava za računalnu sigurnost prema arhitekturi nultog povjerenja

Diplomski rad

Mateo Borina
0303088140, redovan student
Studijski smjer: diplomski sveučilišni studij Informatika
Znanstveno područje: Društvene znanosti
Znanstveno polje: Informacijske i komunikacijske znanosti
Znanstvena grana: Informacijski sustavi i informatologija
Mentor: doc. dr. sc. Nikola Tanković

Pula, lipanj 2024. godine

IZJAVA O AKADEMSKOJ ČESTITOSTI

Ja, dolje potpisani _____, kandidat za magistra _____
ovime izjavljujem da je ovaj Diplomski rad rezultat isključivo mojega vlastitog rada, da se temelji na mojim istraživanjima te da se oslanja na objavljenu literaturu kao što to pokazuju korištene bilješke i bibliografija. Izjavljujem da niti jedan dio Diplomskog rada nije napisan na nedozvoljeni način, odnosno da je prepisan iz kojega necitiranog rada, te da ikoji dio rada krši bilo čija autorska prava. Izjavljujem, također, da nijedan dio rada nije iskorišten za koji drugi rad pri bilo kojoj drugoj visokoškolskoj, znanstvenoj ili radnoj ustanovi.

Student

_____

U Puli, _____

IZJAVA O KORIŠTENJU AUTORSKOG DJELA

Ja, _____ dajem odobrenje Sveučilištu Jurja Dobrile u Puli, kao nositelju prava iskorištavanja, da moj diplomski rad pod nazivom

_____

_____

koristi na način da gore navedeno autorsko djelo, kao cjeloviti tekst trajno objavi u javnoj internetskoj bazi Sveučilišne knjižnice Sveučilišta Jurja Dobrile u Puli te kopira u javnu internetsku bazu završnih radova Nacionalne i sveučilišne knjižnice (stavljanje na raspolaganje javnosti), sve u skladu s Zakonom o autorskom pravu i drugim srodnim pravima i dobrom akademskom praksom, a radi promicanja otvorenoga, slobodnoga pristupa znanstvenim informacijama.

Za korištenje autorskog djela na gore navedeni način ne potražujem naknadu.

U Puli, _____

Potpis

_____

Sažetak i ključne riječi

U današnjem digitalnom dobu, računalna sigurnost postaje ključna za zaštitu podataka i sustava od sve složenijih prijetnji. Arhitektura nultog povjerenja (*Zero Trust Architecture*, ZTA) predstavlja suvremeni pristup sigurnosti koji pretpostavlja da nijedan entitet unutar mreže nije automatski pouzdan. U okviru ovog rada razvijen je sustav koji obuhvaća osam mikroservisnih aplikacija unutar ZTA, pružajući sveobuhvatnu zaštitu i kontrolu pristupa. Mikroservisi sustava aplikacija postavljeni su na *Kubernetes*. Razvoj je bio uspješan, doprinoseći području računalne sigurnosti kroz implementaciju otvorenog koda i demonstraciju naprednih sigurnosnih funkcionalnosti. Sustav ističe važnost modularnosti i fleksibilnosti, omogućavajući korisnicima da koriste njegove dijelove kao vanjske servise u vlastitim aplikacijama. Također, sustav služi kao detaljan primjer primjene različitih sigurnosnih algoritama. Kôd i detaljne upute za implementaciju dostupni su na *Github* repozitoriju: https://github.com/b0rke-mborina/zta-cybersec-suite, dok se *Docker* images mogu pronaći na *Docker Hubu*: https://hub.docker.com/repository/docker/b0rkemborina/zta-cybersec-suite/general.

Ključne riječi: računalna sigurnost, arhitektura nultog povjerenja, sustav aplikacija za računalnu sigurnost, *Docker*, *Kubernetes*

Abstract and keywords

In today's digital age, computer security is becoming essential to protect data and systems from all the more complex threats. Zero Trust Architecture (ZTA) represents a modern approach to security that assumes that no entity within the network is automatically trusted. As part of this work, a system was developed that includes eight microservice applications within ZTA, providing comprehensive protection and access control. Dockerized application system microservices are deployed on Kubernetes. The development was successful, contributing to the field of computer security through open source implementation and demonstration of advanced security functionalities. The system emphasizes the importance of modularity and flexibility, allowing users to use parts of it as external services in their own applications. Also, the system serves as a detailed example of the application of various security algorithms. Code and detailed deployment instructions are available on the Github repository: https://github.com/b0rkemborina/zta-cybersec-suite, while Docker images can be found on the Docker Hub: https://hub.docker.com/ repository/docker/b0rkemborina/zta-cybersec-suite/general.

Keywords: cybersecurity, zero-trust architecture, application system for cybersecurity, Docker, Kubernetes

Sadržaj

1 Uvod

U suvremenom digitalnom okruženju, sigurnost informacijskih sustava postaje sve važnija tema. Razvoj novih tehnologija i stalno rastući broj kibernetičkih prijetnji zahtijevaju inovativne pristupe kako bi se osigurala zaštita podataka i resursa. Jedan od takvih pristupa je arhitektura nultog povjerenja (*Zero Trust Architecture*, ZTA) [1], koja mijenja postojeće paradigme računalne sigurnosti. Cilj ovog rada je istražiti arhitekturu nultog povjerenja, predstaviti njezine ključne koncepte i razviti sustav aplikacija za računalnu sigurnost temeljenog na toj arhitekturi. Također, nakon razvoja sustava analizom raspraviti o ključnim prednostima ove arhitekture, izazovima i implementiranim rješenjima.

1.1 Pozadina i kontekst

Postojeći modeli sigurnosti oslanjaju se na perimetarsku obranu, gdje se povjerenje dodjeljuje svim korisnicima i uređajima unutar granica mreže [2]. Ova metoda pretpostavlja da su svi unutar mrežnog perimetra pouzdani, dok se vanjski entiteti smatraju prijetnjama. Perimetarska obrana uključuje zaštitne mjere poput vatrozida, mrežnih pregrada i sigurnosnih sustava za otkrivanje upada, koji služe kao prva linija obrane protiv vanjskih napada. Međutim, ovaj je pristup pokazao nedovoljnim u borbi protiv sve češćih i sofisticiranijih kibernetičkih napada. Napadi poput *phishinga*, *ransomwarea* i napada iznutra (*insider threats*) lako mogu zaobići perimetarske obrane. Jednom kada napadači uspiju prodrijeti unutar mreže, mogu se slobodno kretati i pristupati osjetljivim podacima zbog implicitnog povjerenja unutar mreže. Dodatno, usvajanje *cloud* tehnologija i porast broja povezanih IoT uređaja (*Internet of Things*) čine perimetarsku obranu još izazovnijom i manje učinkovitom [3].

Arhitektura nultog povjerenja (*Zero Trust Architecture*, ZTA) temelji se na principu da nikome i ničemu ne treba automatski vjerovati, bilo unutar ili izvan mreže [4]. Svaki zahtjev za pristupom mora se provjeriti, a pristup se omogućava samo na temelju stroge autentifikacije i autorizacije. U ZTA modelu, sigurnosna politika primjenjuje se na svakom sloju mreže, a pristup se dodjeljuje na najmanjoj potrebnoj razini (princip najmanjih privilegija).

Implementacija arhitekture nultog povjerenja zahtijeva nekoliko ključnih značajki [5]. Sve mrežne aktivnosti moraju biti vidljive i praćene u stvarnom vremenu, što omogućava detekciju neobičnih ili zlonamjernih aktivnosti. Autentifikacija mora biti višefaktorska i kontinuirana, osiguravajući da svaki korisnik ili uređaj ima pravu razinu pristupa. Mikrosegmentacija mreže omogućava kontrolu pristupa s minimalnim dopuštenjima, ograničavajući neometano kretanje napadača unutar mreže.

U arhitekturi nultog povjerenja također mogu biti integrirane tehnologije kao što su umjetna inteligencija (AI) i strojno učenje (ML) za prepoznavanje uzoraka i predviđanje potencijalnih prijetnji [6]. Ovo omogućava proaktivni pristup sigurnosti, gdje sustav može automatski odgovoriti na prijetnje u stvarnom vremenu. Važnost stalne edukacije zaposlenika o važnosti sigurnosnih protokola i najboljih praksi izražena je pri implementeciji spomenute arhitekture.

1.2 Iskaz problema

Današnji informacijski sustavi suočeni su s brojnim sigurnosnim prijetnjama, uključujući napadače ili zloćudne aktere, maliciozni softver, unutarnje prijetnje i druge oblike kibernetičkih napada. Ovi napadi postaju sve sofisticiraniji i češći, što predstavlja ozbiljan rizik za kompanije svih veličina i sektora. Gubitak podataka, financijski gubici, uništenje reputacije i pravne posljedice samo su neki od mogućih ishoda uspješnog kibernetičkog napada. Postojeće sigurnosne mjere, koje se oslanjaju na perimetarsku obranu, često nisu dovoljne za zaštitu osjetljivih podataka i kritičnih resursa. U ovim modelima, mrežni perimetar se štiti vatrozidima, antivirusnim softverom i sustavima za otkrivanje upada [2]. Međutim, jednom kada napadači uspiju probiti vanjske obrambene slojeve, imaju slobodan pristup unutarnjim resursima zbog pretpostavke povjerenja unutar perimetra. Dodatno, ovaj pristup ne uzima u obzir mogućnost unutarnjih prijetnji, kao što su kompromitirani korisnički računi. To je posebno velik problem iz razloga što su ljudi najslabija karika u obrani od kibernetičkih napada [7]. Problem leži u pretpostavci povjerenja unutar perimetra mreže, što napadačima omogućava lakši pristup jednom kada probiju vanjsku obranu sustava. Mnogi sigurnosni incidenti rezultat su upravo takvih propusta, gdje kompromitirani račun može neometano pristupiti osjetljivim informacijama. Tradicionalni sigurnosni modeli također ne pružaju dovoljno fleksibilnosti za upravljanje modernim radnim okruženjima koja uključuju udaljeni rad, mobilne uređaje i cloud usluge.

Ovaj diplomski rad namjerava riješiti spomenute nedostatke implementacijom nultog povjerenja kao temeljne sigurnosne paradigme. To će se učiniti uklanjanjem implicitnog povjerenja unutar mreže i primjenjivanjem strogih mjera autentifikacije i autorizacije na svaki upit, bez obzira na njegov izvor. Svaki korisnik i svaki uređaj bit će

podvrgnuti strogim provjerama prije obavljanja zadataka. Osim toga, omogućit će se kontinuirano praćenje mrežnih aktivnosti, što će pomagati u prepoznavanju i odgovaranju na prijetnje. Ovim će se pristupom također mikrosegmentirati mrežu, što će ograničavati kretanje napadača i smanjiti površinu napada.

1.3 Ciljevi diplomskog rada

Glavni cilj ovog diplomskog rada je razviti sustav aplikacija za računalnu sigurnost koji je u potpunosti usklađen s principima arhitekture nultog povjerenja. Međutim, postoji i više specifičnih ciljeva koje se želi ostvariti u sklopu ovog diplomskog rada.

Prvi specifični cilj diplomskog rada je analiza postojećih sigurnosnih modela i njihovih nedostataka. Ova analiza će uključivati pregled trenutnih pristupa računalnoj sigurnosti, identificiranje njihovih slabosti i procjenu koliko su učinkoviti u zaštiti od suvremenih prijetnji. Time, može se razumjeti gdje stariji modeli sigurnosti zakazuju i kako nulto povjerenje može popuniti te praznine.

Sljedeći cilj je definiranje ključnih elemenata arhitekture nultog povjerenja. To uključuje uspostavljanje jasnih smjernica i principa koji čine temelj ove arhitekture, poput stalne autentifikacije, minimalnog privilegiranog pristupa i stroge kontrole pristupa resursima. Definiranje ovih elemenata ključno je za izgradnju sustava koji može dosljedno primjenjivati principe nultog povjerenja [1].

Treći cilj je predstavljanje dizajna sustava te njegova implementacija i razvoj. Kako bi se ostvario, potrebno je definirati zahtjeve sustava, navesti komponente, objasniti komunikaciju te razmotriti sigurnosna pitanja općeg pregleda dizajna. U implementaciji važno je navesti tehnologije i obrazložiti rješenja izazova na koje se naišlo. Učinkovitost implementacije izvijestit će se pomoću testiranja.

Konačno, posljednji cilj je evaluacija programskog proizvoda i zaključak. Tijekom evaluacije, bit će važno zabilježiti sva ograničenja sustava, kao i usporediti ga s postojećim proizvodima. Zaključak će sadržavati doprinose, praktične primjene i ideje za budući rad.

1.4 Opseg i ograničenja

Opseg ovog rada obuhvaća teorijsku i praktičnu analizu arhitekture nultog povjerenja u kontekstu računalne sigurnosti. Teorijski dio rada će se baviti detaljnim pregledom postojećih sigurnosnih modela i koncepata, s naglaskom na njihove nedostatke i načine na koje arhitektura nultog povjerenja može riješiti te probleme. Praktični dio

rada će se fokusirati na razvoj i analizu sustava aplikacija koji primjenjuje principe nultog povjerenja. Sustav aplikacija sastojati će se od osam mikroservisnih aplikacija iz domene računalne sigurnosti s mikroservisima koji dopunjuju arhitekturu nultog povjerenja.

Jedno od glavnih ograničenja ovog diplomskog rada je vremenski okvir unutar kojeg se rad piše. S obzirom da se rad piše u sklopu diplomskog rada, vremenski su resursi ograničeni. Student ima određeni period za završetak istraživanja, razvoja programskog rješenja, testiranja i pisanja konačnog izvještaja. Iako se diplomski rad radi pod mentorstvom iskusnog stručnjaka, postoji mogućnost da će stručnost biti smanjena pošto rad piše jedan student. Mentorstvo pruža smjernice i podršku, ali student će samostalno provoditi razvoj i analizu. Tehnologije korištene u diplomskom radu bit će ograničene na open source i besplatne alate. Iako su open source tehnologije često vrlo moćne i fleksibilne, ovo ograničenje može značiti da neće biti moguće koristiti najnovije ili najnaprednije komercijalne alate i softvere koji bi mogli ponuditi dodatne funkcionalnosti ili jednostavnost korištenja.

Unatoč ovim ograničenjima, identifikacija i dokumentacija izazova koji se mogu pojaviti tijekom razvoja i testiranja sustava mogu pružiti vrijedne smjernice za buduća istraživanja i implementacije, čime se doprinosi ukupnom napretku na području računalne sigurnosti.

1.5 Značaj diplomskog rada

Razvoj sustava za računalnu sigurnost temeljenog na arhitekturi nultog povjerenja ima značajne implikacije za unapređenje zaštite podataka i infrastrukture. Implementacija nultog povjerenja može značajno smanjiti rizik od kibernetičkih napada, poboljšati sigurnosne prakse i osigurati bolju zaštitu osjetljivih informacija. Ovaj diplomski rad doprinosi znanstvenom i praktičnom razumijevanju nultog povjerenja, pružajući smjernice za buduće implementacije i istraživanja u području računalne sigurnosti.

2 Pregled literature

U ovom poglavlju analizirat će se postojeći pristupi i rješenja u domeni računalne sigurnosti s posebnim naglaskom na arhitekturu nultog povjerenja. Pregled literature omogućit će dublje razumijevanje trenutno dostupnih sigurnosnih sustava i tehnologija, identificiranje njihovih prednosti i nedostataka te uočavanje praznina koje bi arhitektura nultog povjerenja mogla popuniti. Osim toga, analizirat će se povezani radovi i rješenja koja su već implementirana u praksi kako bi se dobio uvid u učinkovitost različitih pristupa.

Prvo slijedi pregled postojećih sustava aplikacija u domeni računalne sigurnosti, gdje se istražuju modeli i tehnike koje su trenutno u upotrebi. Zatim će se detaljno analizirati arhitektura nultog povjerenja u računalnoj sigurnosti, zajedno s njezinim ključnim principima i načinima primjene. Na kraju, identificirat će se izazovi s kojima se suočavaju trenutni pristupi računalnoj sigurnosti i kako arhitektura nultog povjerenja može doprinijeti njihovom rješavanju.

2.1 Pregled postojećih programskih rješenja u domeni računalne sigurnosti

Postoje mnoga programska rješenja u domeni računalne sigurnosti, koja su razvijena s ciljem poboljšanja zaštite informatičkih sustava i podataka. Ona variraju od jednostavnih alata za zaštitu pojedinačnih uređaja do složenih sustava za upravljanje sigurnosnim prijetnjama na mrežnoj razini. U ovom potpoglavlju, pregledat će se neka od značajnih rješenja otvorenog koda koja su se pojavila u literaturi.

Prva dva rada koja će se razmotriti razvijena su s ciljem izrade scenarija računalnih prijetnji. *"EZSetup: A Novel Tool for Cybersecurity Practices Utilizing Cloud Resources"*[8] opisuje alat za upravljanje virtualnim okruženjima za vježbe računalne sigurnosti. *EZSetup* je web alat koji omogućava kreiranje i upravljanje korisnički definiranih virtualnih okruženja za različite vrste vježbi iz područja računalne sigurnosti na zahtjev i u velikom opsegu. On se ističe po tome što ne ovisi o određenom tipu *cloud* platforme ili tehnologije, već može komunicirati s više *cloudova* istovremeno. Njegova jednostavna prilagodljivost i smanjenje administrativnog opterećenja čine ga izuzetno korisnim za stvaranje i korištenje okruženja za vježbe kroz pažljivo dizajnirana *web* sučelja. Eksperimentalni rezultati pokazuju pozitivan učinak i sugeriraju da se *EZSetup* može primijeniti i u drugim područjima računalnih znanosti i inženjeringa. Slično tome, *"A Cloud-based platform for the emulation of complex cybersecurity scenarios"* [9] opisuje *SmallWorld*, skalabilnu softversku platformu koja reproducira realistične scenarije uranjanjem stvarnih sustava u softverski definirano vir-

tualno okruženje. *SmallWorld* omogućava procjenu, podučavanje i učenje aspekata računalne sigurnosti kroz inovativne virtualizacijske i simulacijske tehnike. Jedna od ključnih značajki *SmallWorlda* je podrška za dizajniranje i izgradnju složenih, dinamičnih scenarija s autonomnim softverskim agentima koji mogu simulirati ponašanje ljudskih korisnika ili zlonamjernih aplikacija, čime se stvara realističnija testna okolina. Praktična primjena *SmallWorlda* demonstrirana je kroz dva realistična studija slučaja.

Jedan od važnih radova u području zaštite pametnih kuća je *"Defensive Programming for Smart Home Cybersecurity"* [10]. U ovom radu, autori predlažu pristup za nadzor sustava u domeni pametnih kuća koristeći paradigmu defenzivnog programiranja u kombinaciji sa *Shodan API*-jem. Ovaj pristup omogućava bolji nadzor i zaštitu pametnih kuća identificiranjem i odgovaranjem na potencijalne sigurnosne prijetnje u stvarnom vremenu. Strojno učenje također ima značajnu ulogu u modernim rješenjima za računalnu sigurnost, kao što je prikazano u radu *"Towards a Machine Learning Based Situational Awareness Framework for Cybersecurity: An SDN Implementation"* [11]. Ovaj rad predstavlja okvir za situacijsku svjesnost temeljen na strojnom učenju koji koristi SDN (*Software-Defined Networking*) paradigmu za detekciju mrežnih entiteta i njihovu procjenu u odnosu na poznate ranjivosti. Sustav kontinuirano prati procijenjene entitete koristeći IDS (*Intrusion Detection System*) temeljen na strojnome učenju, koji je treniran s poboljšanim skupom podataka. Rezultati pokazuju povećanje točnosti predikcija za više od 4% u odnosu na konvencionalne metode. Razvoj sustava za upravljanje računalnom sigurnošću detaljno je opisan u radu *"Cybersecurity Management System: Defense and Response"* [12]. CMSDR je cloud softverska aplikacija koja pruža i "obranu" i "odgovor" na napade, s edukativnim materijalima i primjerima koji korisnicima pomažu učiti o različitim vrstama napada. Aplikacija također omogućava računalno potpomognuto izvješćivanje i obavještavanje kako bi organizacije mogle učinkovitije odgovoriti na tekuće incidente. CMSDR je univerzalna aplikacija koja se može koristiti na bilo kojoj platformi s *web* preglednikom.

Posljednja dva rada koja će se razmotriti fokusiraju se na razvoj sustava za treniranje o računalnoj sigurnosti. *"ICSTASY: An Integrated Cybersecurity Training System for Military Personnel"* [13] opisuje scenarijski baziranu, interaktivnu i imerzivnu platformu za obuku iz računalne sigurnosti. ICSTASY podržava različite značajke treninga, a kroz demonstraciju prototipa dokazano je da može osigurati učinkovitu i realističnu obuku ne samo za vojno okruženje već i za privatni sektor. *"Cybersafe: Gamifying Cybersecurity Training with a Training App"* [14] predstavlja mobilnu aplikaciju dizajniranu za prepoznavanje i borbu protiv socijalno-inženjerskih napada. Aplikacija *Cybersafe* uvodi koncept gamifikacije u tradicionalnu obuku iz računalne sigurnosti, čineći je

pristupačnom za sve dobne skupine. Aplikacija koristi interaktivne kvizove kako bi procijenila sposobnost korisnika da prepoznaju prijetnje, a nalazi iz istraživanja pružaju vrijedan resurs za trenere, razvojne programere aplikacija i organizacije koje teže sigurnom digitalnom okruženju.

2.2 Arhitektura nultog povjerenja u računalnoj sigurnosti

U literaturi se objašnjava kako arhitektura nultog povjerenja (*Zero Trust Architecture*, ZTA) funkcionira i koje su njezine ključne komponente. U radu *"Autonomic Security for Zero Trust Networks"* [15], predstavljeni su rezultati eksperimentalnog testiranja autonomnog kontrolnog sustava temeljenog na OODA (*Observe, Orient, Decide, Act*) okviru. Ovaj sustav modelira osnovne građevne jedinice za predloženu mrežu podatkovnog centra u oblaku s nultim povjerenjem. Testovi su pokazali poboljšane vremena odgovora na prijetnje zahvaljujući automatiziranom upravljanju identitetom i autentifikacijom paketa, uz dinamično upravljanje osam različitih razina povjerenja u mreži. Razvijeni softver za parsiranje i orkestraciju dnevnika, zajedno s alatima za upravljanje dnevnicima otvorenog koda, omogućava koordiniranu i integriranu reakciju na prijetnje iz vatrozida, autentifikacijskih prolaza i drugih mrežnih uređaja, što predstavlja značajno poboljšanje u odnosu na konvencionalne metode. Rad *"Access Control Policy Enforcement for Zero-Trust-Networking"* [16] opisuje okvir za provođenje politika koji rješava mnoge otvorene izazove utemeljene na pristupu kontrole pristupa zasnovanom na riziku za ZTN (*Zero-Trust Networking*). Autori specificiraju dizajn potrebnih jezika za politike, uključujući generički jezik za politike vatrozida, te mehanizam za mapiranje tih pravila na specifičnu sintaksu vatrozida i njihovu implementaciju. Održivost dizajna prikazana je putem koncepta. U radu *"eZTrust: Network-Independent Zero-Trust Perimeterization for Microservices"* [17], predlaže se *eZTrust*, pristup neovisan o mreži za perimetarizaciju mikroservisa. *eZTrust* omogućuje korisnicima podatkovnih centara izražavanje politika kontrole pristupa na temelju identiteta radnih opterećenja, te omogućuje operatorima podatkovnih centara pouzdanu i efikasnu provedbu tih pravila. Korištenjem eBPF (*extended Berkeley Packet Filter*), ovaj pristup prati autentične identitete radnih opterećenja te primjenjuje označavanje i verifikaciju po svakom paketu. Evaluacija prototipa pokazuje da *eZTrust* ostvaruje 2-5 puta nižu latenciju paketa i 1.5-2.5 puta manje opterećenje procesora u usporedbi s tradicionalnim metodama. U radu *"Protection of Sensitive Data in Zero Trust Model"* [18], predstavljen je novi model za kontrolu pristupa osjetljivim podacima unutar ZTM-a (*Zero Trust Model*). U ovom modelu, ne postoji zadano povjerenje ni za unutarnje ni za vanjske strane mreže, a prijetnje osjetljivim podacima mogu dolaziti izvan i unutar zone sustava. Model uvodi *proxy* za kontrolu pristupa koji štiti osjetljive podatke analizirajući zahtjeve za pristup, tip korisnika, uređaja, aplikacije i podataka. Ovaj model se

pokazao vrlo učinkovitim u poboljšanju sigurnosti osjetljivih podataka od neovlaštenog pristupa i manipulacije. Rad *"Network Segmentation and Zero Trust Architectures"* [19] naglašava da segmentacija mreže i ZTA predstavljaju sigurnosne pristupe koji nadmašuju pristup tvrđave. Segmentacija dijeli mrežu i primjenjuje pristup tvrđave na svaki dio mreže, dok ZTA ne vjeruje mreži i oslanja se na krajnje točke za sigurnost. Iako se ova dva pristupa ne mogu direktno kombinirati zbog temeljnih nekompatibilnosti, mogu se koristiti u hibridnom načinu rada gdje različiti dijelovi poduzeća koriste različite pristupe. ZTA može poboljšati segmentaciju pružanjem sigurnosti unutar pojedinačnih segmenata, dok segmentacija može poboljšati performanse, smanjiti *broadcast* promet i smanjiti troškove. Konačni cilj hibridnog rješenja za sigurnost je ZTA.

2.3 Postojeće slične implementacije

Implementacije sličnih rješenja arhitekture nultog povjerenja u literaturi pruže korisne uvide i primjere za daljnja istraživanja i razvoj. U radu *"Zero Trust Network Security Model in containerized environments"* [20] istražuje se i implementira operativne kontrole potrebne za primjenu modela nultog povjerenja u kontejneriziranim okruženjima. Cilj je ublažiti curenje podataka tijekom *"east-west"* prometa između mikroservisa. Ovaj rad pokazuje da je uz odgovarajuće alate moguće zaštititi podatke u prijenosu i regulirati promet kako bi se utvrdilo gdje, kako i kada dolazi do napada. Implementacijom koncepta, autori demonstriraju kako regulacija prometa i zaštita podataka može učinkovito spriječiti napade poput *man-in-the-middle* napada. Rad *"Building A Zero Trust Architecture Using Kubernetes"* [21], koji je jako sličan ovom diplomskom radu, istražuje nulto povjerenje kao novi pristup mrežnoj sigurnosti, s usredotočenošću na implementaciju pomoću *Kubernetesa*. Rad analizira prethodna istraživanja i implementacije arhitekture nultog povjerenja te razmatra sigurnost na svim slojevima OSI modela. Autori ističu prednosti i nedostatke ovog pristupa te pokazuju kako kontejneri mogu odgovoriti na različite vrste napada u mrežnom okruženju. U radu *"Zero-Trust Model for Smart Manufacturing Industry"* [22] detaljno se istražuje i dokumentira pristup nultog povjerenja u kontekstu pametne proizvodne industrije. Ovaj rad uključuje pregled postojećih kibernetičkih sigurnosnih rješenja i predlaže dizajn modela nultog povjerenja za infrastrukturu na licu mjesta i u oblaku. Razmatraju se različita sigurnosna rješenja kao što su mikro-segmentacija industrijske mreže, otkrivanje uređaja i alati za upravljanje usklađenošću, koji su ključni za postizanje potpune sigurnosti nultog povjerenja. *"Implementing a Zero Trust Environment for an Existing On-premises Cloud Solution"* [23] projekt je dizajna i implementacije sigurnog sustava za rukovanje i zaštitu osobnih podataka. Ovaj rad naglašava važnost jakih sigurnosnih mjera i strogih kontrola pristupa. Predloženo rješenje uključuje integraciju s postojećom infrastrukturom u oblaku, korištenjem alata poput *Keycloak* za upravljanje

identitetom, *GitLab* za hosting koda, *GPG* za potpisivanje komitova i *OpenVPN* za mrežni pristup. Autori ističu da ovaj sustav pruža sveobuhvatnu zaštitu podataka, autentifikaciju korisnika i mrežnu sigurnost, te predlažu buduća poboljšanja i alternative. U radu *"Zero Trust Architecture in a Multi-Cloud environment"* [24] istražuje se implementacija arhitekture nultog povjerenja u *multi-cloud* okruženju. Cilj je razumjeti izvedivost i dizajnerske smjernice za autentifikaciju radnih opterećenja u takvim okruženjima. Kroz proučavanje različitih aspekata računalstva (*cloud computing*) i *multi-cloud* okruženja, te implementacijom dokaza koncepta pomoću *Istio service mesha*, autori pokazuju kako se arhitektura nultog povjerenja može uspostaviti s relativnom lakoćom. Ovaj pristup poboljšava funkcionalnost klastera omogućujući vidljivost i upravljanje prometom te učinkovito rješava sigurnosne i upravljačke izazove karakteristične za takva okruženja.

2.4 Izazovi u arhitekturi nultog povjerenja

Arhitektura nultog povjerenja (*Zero Trust Architecture*, ZTA) predstavlja jedan od najnovijih pristupa u računalnoj sigurnosti. S obzirom na to, ovaj pristup donosi niz izazova, posebno u složenim okruženjima kao što su *multi-cloud* infrastrukture.

U radu *"The Challenge of Achieving Zero Trust Remote Access in Multi-Cloud Environment"* [25], autori ističu kako je sve veći broj ljudi *online* doveo do novih sigurnosnih prijetnji, čineći tradicionalne sigurnosne modele neodrživima. Moderni tehnološki i radni okviri, poput nultog povjerenja, donose nove prepreke za mrežnu sigurnost. Dok su neke napredne tehnološke organizacije poput *Google*-a već usvojile principe nultog povjerenja, mnoge tvrtke još uvijek pokušavaju prilagoditi ovaj model svojim potrebama. Model nultog povjerenja dokazano učinkovito rješava probleme za koje je osmišljen, a sve više firmi implementira njegove koncepte djelomično ili u cijelosti, što je važan trend. Rad *"Migrating to Zero Trust Architecture: Reviews and Challenges"* [26] detaljno raspravlja o konceptu i primjeni ZTA. Jedan od glavnih izazova jest nedostatak standardizacije i problem zaključavanja dobavljača (*vendor lock-in*). U radu se također iznosi kratak pregled koraka i razmatranja koje treba uzeti u obzir prilikom prelaska s perimetrijske sigurnosne arhitekture na ZTA. Prema radu *"Never Trust, Always Verify: A Multivocal Literature Review on Current Knowledge and Research Gaps of Zero-trust"* [27], akademska literatura uglavnom se fokusira na arhitekturu i poboljšanje performansi nultog povjerenja, dok se praksa orijentira na organizacijske prednosti i strategije migracije. Međutim, ekonomske analize i studije vezane za korisnike su zanemarene. Autori ističu potrebu za daljnjim istraživanjima kako bi se omogućila šira primjena ZTA prema devet istraživačkih pitanja, koja su autori identificirali te koja mogu voditi buduća istraživanja i pridonijeti zrelosti ovog

područja. Rad *"Automation and Orchestration of Zero Trust Architecture: Potential Solutions and Challenges"* [28] naglašava kako ZTA rješava sigurnosne probleme prisutne u perimetrijskim sigurnosnim arhitekturama poput unutarnjih napada. Korištenje AI tehnologija dodatno potiče implementaciju i razvoj ZTA. Međutim, malo modela nultog povjerenja koristi AI tehnike za automatizaciju, što predstavlja priliku za buduća istraživanja. Prema radu *"The Challenges and Processes of Achieving Optimal Implementation of Zero Trust Architecture in Workplace"* [29], tradicionalni pristupi mrežnoj sigurnosti više nisu održivi zbog novih radnih modela poput BYOD i udaljenog rada. Prijelaz na ZTA zahtijeva promjenu paradigme i dobro osmišljene planove za maksimiziranje koristi i smanjenje rizika. Rad identificira kritične izazove i nudi procese i strateške sheme za postizanje optimalne implementacije ZTA u radnim okruženjima.

3 Teorijski okvir

Teorijski okvir ovog istraživanja pruža temelj za razumijevanje i analizu sustava aplikacija u domeni računalne sigurnosti, s posebnim fokusom na arhitekturu nultog povjerenja. Konceptualizacija sustava aplikacija u računalnoj sigurnosti uključuje definiranje osnovnih pojmova, identificiranje ključnih komponenti i opisivanje načina na koji ovi sustavi djeluju u zaštiti informacija i resursa [30]. Osnovni pojmovi poput autentifikacije, autorizacije, enkripcije i integriteta podataka predstavljaju temeljne elemente svakog sigurnosnog sustava. Identifikacija ključnih komponenti podrazumijeva razumijevanje hardverskih i softverskih elemenata koji čine sigurnosni sustav, kao i načina njihove interakcije i integracije. Ovaj dio istraživanja omogućava detaljno razumijevanje trenutnih sigurnosnih modela i njihovih operativnih mehanizama, što je ključno za usporedbu s novim pristupima kao što je nulto povjerenje.

3.1 Teorijske osnove i okviri

Teorijske osnove i okviri u kontekstu arhitekture nultog povjerenja igraju ključnu ulogu u razumijevanju temeljnih principa i koncepta u osnovi ovog sigurnosnog modela. Arhitektura nultog povjerenja (ZTA) [1] predstavlja promjenu u odnosu na tradicionalne pristupe sigurnosti koji se oslanjaju na perimetarsku obranu i pretpostavku unutarnjeg povjerenja. Umjesto toga, ZTA uvodi princip da niti jedan korisnik ili uređaj, bez obzira na njihovu lokaciju unutar ili izvan mreže, ne bi trebao automatski biti vjerodostojan. Osnovna pretpostavka ZTA je da svaki zahtjev za pristup resursima mora proći kroz proces autentifikacije i autorizacije [4]. Ovaj pristup omogućava preciznu kontrolu pristupa na temelju identiteta, uloge, konteksta pristupa i stanja uređaja. Tako se smanjuje rizik od neovlaštenog pristupa i minimizira potencijalna šteta.

Ključne teorijske osnove, odnosno principi ili koncepti [5], arhitekture nultog povjerenja su:

- Princip kontinuirane autentifikacije i autorizacije
- Princip najmanjih privilegija
- Koncept mikrosegmentacije mreže
- Koncept stalnog praćenja i analize aktivnosti
- Princip osiguranja krajnjih točaka
- Princip primjene snažne enkripcije podataka

Implementacija ovih principa zahtijeva integraciju raznovrsnih sigurnosnih tehnologija i alata te promjenu tradicionalnih pristupa sigurnosti [30]. Njihova pravilna primjena može značajno povećati sigurnost informacijskih sustava, pružajući snažan i prilagodljiv sigurnosni okvir koji je sposoban odgovoriti na sve prijetnje suvremenog digitalnog okruženja.

3.1.1 Načela arhitekture nultog povjerenja

Prvo načelo arhitekture nultog povjerenja je kontinuirana autentifikacija i autorizacija [5]. Svaki korisnik, uređaj ili aplikacija moraju neprestano dokazivati svoj identitet i prava pristupa tijekom cijele sesije. Ovo se provodi kroz kontinuirani proces autentifikacije koji smanjuje rizik od zloupotrebe autentifikacijskih podataka čak i u slučaju ugrožavanja sigurnosti. Osim autentifikacije, autorizacija se također primjenjuje strogo, osiguravajući da svaki pristup resursima bude u skladu s definiranim sigurnosnim pravilima i politikama organizacije.

Princip najmanjih privilegija u *Zero Trust Architecture* (ZTA) zahtijeva da svaki korisnik, uređaj i aplikacija imaju samo onoliko pristupa koliko im je potrebno za obavljanje njihovih zadataka. Ovaj princip minimizira rizik od neovlaštenog pristupa i potencijalne štete u slučaju ugrožavanja sigurnosti. Primjenom principa najmanjih privilegija, ZTA osigurava da čak i ako dođe do sigurnosnog incidenta, šteta je ograničena na najmanji mogući opseg.

Mikrosegmentacija je još jedno ključno načelo ZTA koje doprinosi visokoj razini sigurnosti. Podrazumijeva podjelu mreže na manje, izolirane segmente s precizno definiranim pravilima pristupa. Svaki segment djeluje kao svojevrsna zona, što znači da čak i ako napadač uspije probiti jedan segment, njegovo daljnje kretanje unutar mreže je znatno otežano. Mikrosegmentacija smanjuje rizik od bočnog (*lateral*) kretanja napadača i povećava ukupnu otpornost mrežne infrastrukture.

Stalno praćenje i analiza aktivnosti unutar mreže su ključni za otkrivanje i reakciju na sigurnosne prijetnje u stvarnom vremenu. Ovo načelo uključuje upotrebu naprednih alata za praćenje mrežnog prometa, analizu korisničkog ponašanja i detekciju anomalija koje mogu ukazivati na potencijalne sigurnosne incidente. Kontinuirano praćenje omogućava organizaciji brzu reakciju na sigurnosne prijetnje, minimizirajući potencijalne štete i vrijeme potrebno za otkrivanje problema.

Osiguranje krajnjih točaka, kao što su računala, mobilni uređaji i IoT uređaji, ključan je element u ZTA. Uključuje primjenu različitih sigurnosnih mjera poput enkripcije podataka, provjere integriteta uređaja te korištenje antivirusnih i *antimalware* rješenja. Cilj je osigurati da krajnje točke budu zaštićene od različitih vrsta napada i da ne predstavljaju ulaznu točku za ugrožavanje sigurnosti ostatka mreže.

Primjena snažne enkripcije podataka u ZTA osigurava da podaci ostanu povjerljivi i nečitljivi čak i ako dođu u ruke neovlaštenih osoba ili napadača. Enkripcija je ključna zaštita podataka u prijenosu i mirovanju te osigurava da se informacije sigurno razmjenjuju između različitih dijelova sustava, čime se smanjuje rizik od curenja podataka ili neovlaštenog pristupa.

3.2 Integracija nultog povjerenja u sustavima računalne sigurnosti

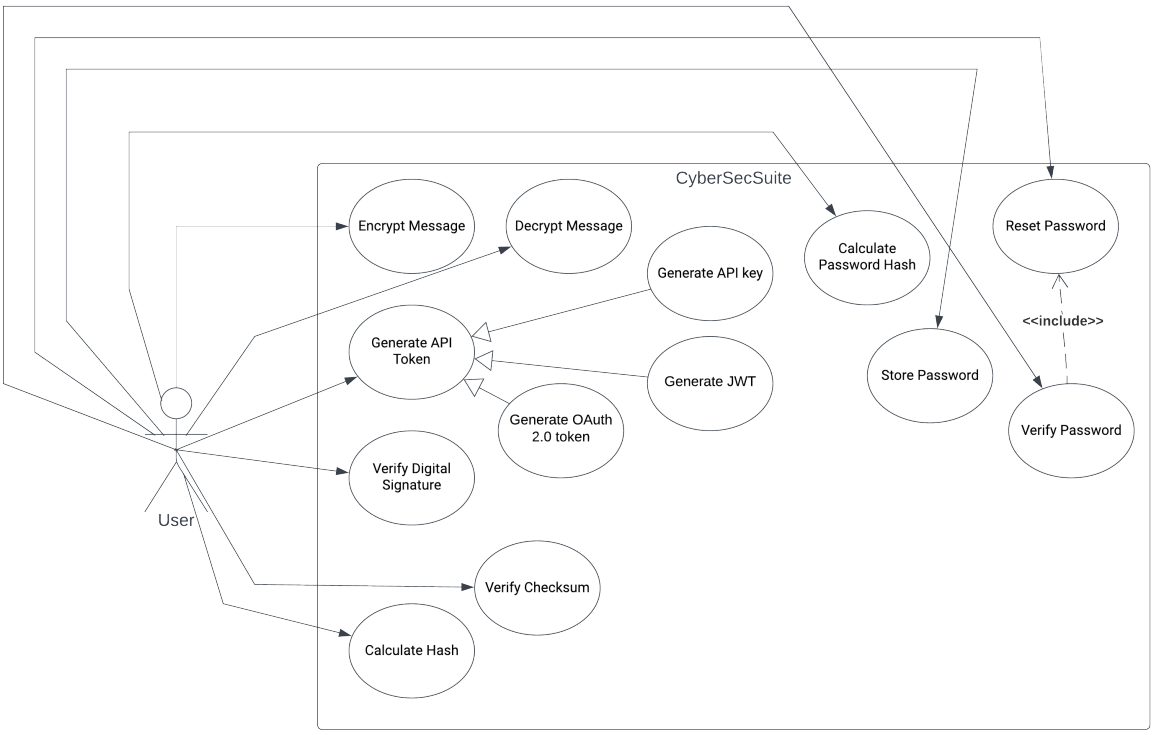
Integracija arhitekture nultog povjerenja u sustave računalne sigurnosti zahtijeva stručnost razvojnog tima kako bi se osigurala njena uspješna implementacija [31]. Ključni aspekti integracije uključuju prilagodbu postojećih infrastruktura, implementaciju novih tehnologija te sustavno upravljanje promjenama kako bi se osiguralo da sve komponente sustava rade u skladu s principima nultog povjerenja.

Timovi za razvoj moraju surađivati s različitim odjelima kako bi osigurali da implementacija ne narušava operativne procese organizacije. To uključuje usklađivanje s IT operacijama, sigurnosnim timovima i korisničkom podrškom kako bi se osigurala integracija bez prekida u radu. Timovi također moraju osigurati da su sve promjene dobro dokumentirane i komunikacija s ključnim dionicima je transparentna kako bi se izbjegle nepredviđene situacije tijekom implementacije. Stručnost razvojnih programera igra ključnu ulogu u uspješnoj integraciji nultog povjerenja. Razumijevanje principa sigurnosti, kriptografije, mrežnih protokola i infrastrukturnih komponenti nužno je za implementaciju sigurnosnih pravila, mikrosegmentaciju mreže te konfiguraciju sustava za stalno praćenje i analizu [32]. Sposobnost razumijevanja složenih arhitektonskih zahtjeva i prilagodba na specifične potrebe organizacije ključni su faktori uspjeha u implementaciji nultog povjerenja. Razvojni programeri također moraju biti stručni u korištenju alata za upravljanje kontejnerima poput *Docker*-a i orkestracije kontejnera kao što je *Kubernetes*. Ovi alati omogućuju fleksibilnost u implementaciji mikrosegmentacije, upravljanju politikama pristupa te omogućuju automatizaciju sigurnosnih procesa [28]. Sposobnost integracije s *cloud* platformama i ostalim digitalnim infrastrukturama također je važna za prilagodbu nultog povjerenja dinamičkim okruženjima i skalabilnost sigurnosnih rješenja prema potrebama kompanije.

Implementacija nultog povjerenja zahtijeva sveobuhvatan pristup koji uključuje ne samo tehničke aspekte već i organizacijske i ljudske resurse [26]. Uspješna integracija ovisi o sposobnosti timova za razvoj da rade zajedno, prilagode se promjenama i primijene najbolje prakse u sigurnosnom inženjeringu. Kroz jasno definirane procese, stručnost i koordinaciju, može se postići visoka razina sigurnosti i zaštite.

3.3 Konceptualizacija sustava aplikacija u domeni računalne sigurnosti

U domeni računalne sigurnosti, razvit će se sustav aplikacija u obliku API sučelja, koje će služiti korisnicima i njihovim aplikacijama. Tako će se korisnicima omogućiti implementaciju sigurnosnih funkcionalnosti u njihovim aplikacijama. Kako bi se olakšala integracija s različitim aplikacijama i platformama, funkcionalnosti će biti dostupne kroz standardizirana API sučelja. API sučelja pružat će razne sigurnosne usluge, omogućavajući korisnicima i njihovim aplikacijama da ih koriste bez potrebe za detaljnim razumijevanjem sigurnosnih protokola. Kroz njih, korisnici će moći dodati napredne sigurnosne funkcionalnosti svojim aplikacijama na učinkovit način. Ovaj sustav osiguravat će visoku razinu sigurnosti te fleksibilnost i jednostavnost integracije. Dijagram slučajeva upotrebe (use case) nalazi se na slici.



Slika 1: Dijagram slučajeva upotrebe (use case) sustava aplikacija

U domeni računalne sigurnosti plan je razviti sljedeće aplikacije za osiguravanje različitih aspekata:

- Aplikacija za kriptografiju - osigurava povjerljivost i integritet podataka kroz enkripciju i dekripciju informacija
- Generator autentikacije - koristi se za generiranje sigurnih autentifikacijskih kodova ili tokena koji se koriste za verifikaciju identiteta korisnika

- Aplikacija za verifikaciju digitalnog potpisa - osigurava autentičnost i integritet elektroničkih dokumenata provjerom digitalnih potpisa
- Aplikacija za *hashiranje* i *checksum* - omogućava provjeru integriteta podataka generiranjem jedinstvenih *hash* vrijednosti koje se koriste za detekciju promjena u podacima
- Upravitelj lozinki - aplikacija za sigurno pohranjivanje i upravljanje lozinkama korisnika
- Aplikacija za spremanje datoteka - služi sigurnoj pohrani datoteka koja koristi enkripciju za zaštitu pohranjenih informacija
- Aplikacija za maskiranje podataka - koristi se za zaštitu osjetljivih informacija kroz zamjenu stvarnih podataka fiktivnim podacima tijekom korištenja
- Aplikacija za dijeljenje podataka o kibernetičkim prijetnjama - omogućuje kompanijama i pojedncima razmjenu informacija o prijetnjama kako bi se unaprijedile sigurnosne mjere

Kao zaštitu sustava, razvit će se mikroservisi za arhitekturu nultog povjerenja, koji će igrati ključnu ulogu u sigurnosti sustava. Ti mikroservisi su:

- Mikroservis za upravljanje (*governance*) - upravljačka komponenta koja definira i provodi sigurnosne politike unutar sustava te rješava probleme
- Mikroservis za autentikaciju (IAM) - servis za upravljanje identitetima
- Mikroservis za segmentaciju mreže - razdvaja mrežu na manje segmente kako bi se ograničio pristup u slučaju napada
- Mikroservis za kontrolu pristupa - osigurava da korisnici imaju pristup samo onim resursima koji su im potrebni za rad
- Mikroservis za komunikaciju (*tunnelling*) - omogućava sigurne komunikacijske kanale između različitih dijelova sustava
- Mikroservis za kriptografiju - pruža kriptografske operacije drugim serivisima
- Mikroservis za praćenje (*monitoring*) - kontinuirano praćenje aktivnosti u sustavu radi otkrivanja prijetnji

U sustavu provodit će se vanjska i unutarnja autorizacija za kontrolu pristupa resursima. Vanjska autorizacija odnosi se na provjeru prava pristupa korisnika izvan organizacijskog perimetra, dok unutarnja autorizacija provjerava prava pristupa unutar sustava. Sustav će biti postavljen pomoću *Dockera* [33] i *Kubernetes* [34] te omogućit će

efikasnu implementaciju, skalabilnost i upravljanje aplikacijama. *Docker* omogućava kreiranje izoliranih kontejnera za različite komponente aplikacija, dok *Kubernetes* pruža orkestraciju kontejnera, omogućavajući automatsko skaliranje, implementaciju i upravljanje aplikacijama u distribuiranom okruženju.

4 Dizajn i arhitektura sustava

Dizajn i arhitektura sustava predstavljaju ključni korak u implementaciji arhitekture nultog povjerenja (ZTA) u praktičnom okruženju računalne sigurnosti. Ovaj dio istraživanja usredotočuje se na planiranje strukture sustava koji će primjenjivati principe ZTA kako bi se osigurala visoka razina sigurnosti i zaštite resursa organizacije.

Centralni cilj dizajna i arhitekture je stvoriti fleksibilan i skalabilan okvir koji će omogućiti preciznu kontrolu pristupa, kontinuirano praćenje aktivnosti i efikasno upravljanje sigurnosnim politikama. Uzimajući u obzir kompleksnost suvremenih mrežnih okruženja i raznolikost tehnoloških infrastruktura, ključno je uskladiti teorijske principe ZTA s praktičnim zahtjevima implementacije [30].

Ovo poglavlje će razmotriti osnovne komponente arhitekture sustava, integraciju ključnih sigurnosnih tehnologija te metodologije razvoja i testiranja koje će osigurati uspješnu implementaciju ZTA.

4.1 Zahtjevi sustava

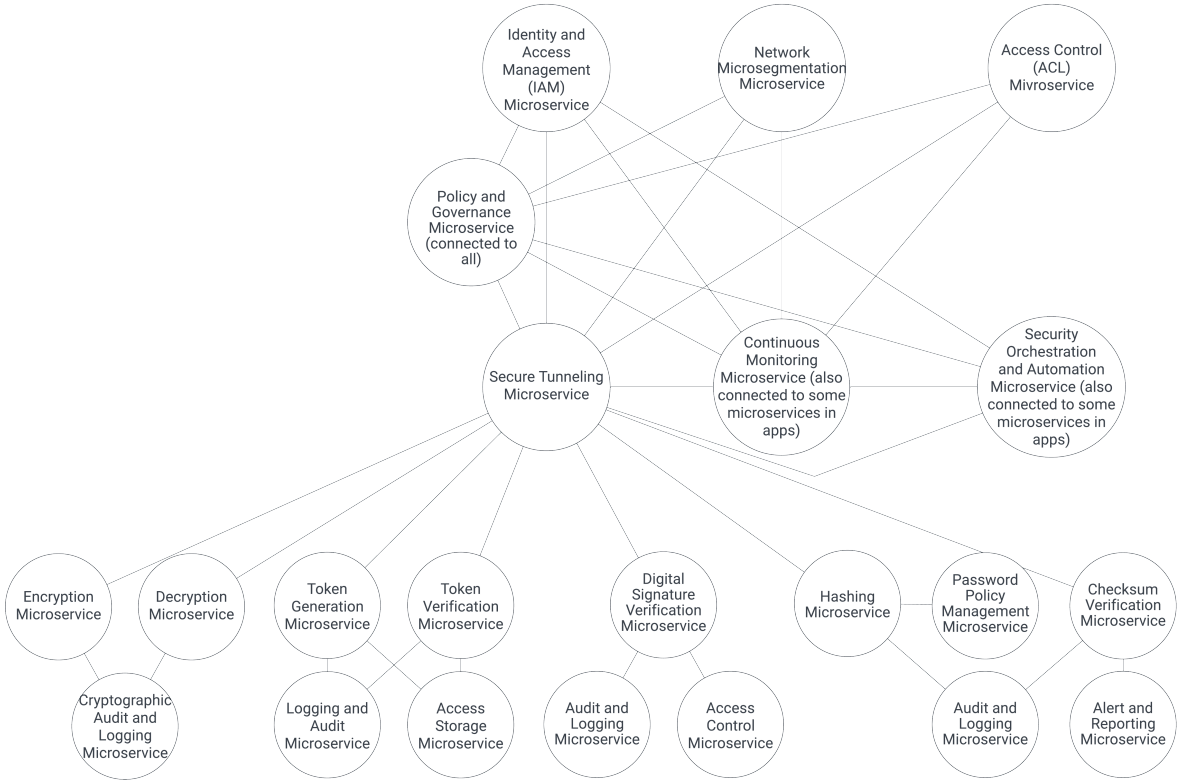
Sustav koji se dizajnira mora biti u obliku API servisa. Tako će se omogućiti komunikacija i integracija s drugim aplikacijama. Sustav, također, mora biti u stanju prihvatiti i obraditi mnogo upita, te slati odgovore na njih. On mora podržavati raznolike značajke i sastojati se od više mikroservisnih aplikacija, od kojih svaka ima posebnu funkcionalnost u domeni računalne sigurnosti. U ovom slučaju bit će razvijeno osam takvih aplikacija. Ove aplikacije će biti dizajnirane za specifične sigurnosne procese kao što su kriptografija, generiranje autentikacijskih tokena, sigurno spremanje datoteka, verifikacija digitalnih potpisa i slično.

Za izgradnju arhitekture nultog povjerenja, potrebni su ZTA mikroservisi. Oni će osigurati strogu autentikaciju i autorizaciju za svaki zahtjev koji dolazi izvana, što znači da će svi pristupi resursima biti provjereni prije odobrenja. Ovo će se koristiti kako bi se osiguralo da se svaki korisnik ili aplikacija identificira na siguran način prije pristupa osjetljivim podacima ili funkcionalnostima sustava. Implementacija će biti postavljena u *Docker* kontejnere radi lakšeg upravljanja i skaliranja aplikacija. *Kubernetes* će se koristiti za orkestraciju tih kontejnera te za provođenje autentikacije između servisa unutar arhitekture.

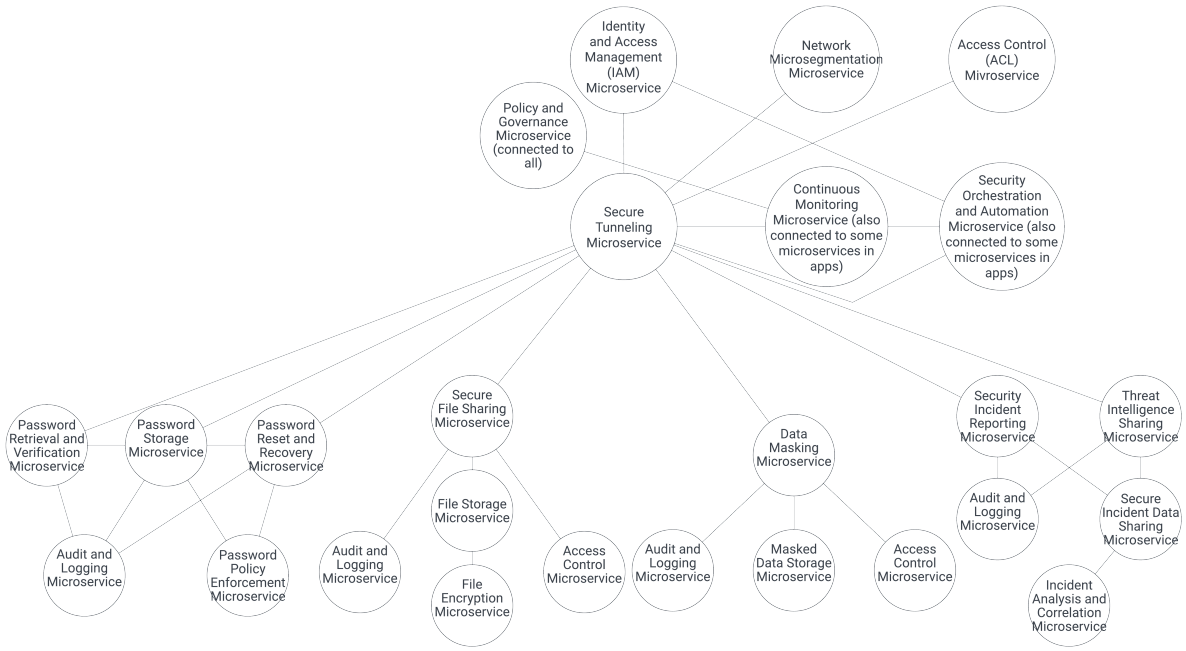
Ovakav dizajn omogućava fleksibilnost u upravljanju resursima, osigurava visoku dostupnost aplikacija i poboljšava sigurnost cijelog sustava kroz integraciju ZTA principa. Zadovoljavanje ovih zahtjeva u dizajn sustava osigurava da ZTA bude učinkovito implementirana i da osigura osjetljive informacije i resurse.

4.2 Pregled arhitekture

Arhitektura sustava je organizirana u osam ključnih aplikacija, svaka sa svojim specifičnim funkcijama u domeni računalne sigurnosti. U sustavu postoje specifični mikroservisi neophodni za arhitekturu nultog povjerenja. U slikama se može vidjeti arhitektura sustava aplikacija.



Slika 2: Arhitektura sustava aplikacija - ZTA mikroservisi i aplikacije 1 - 4.



Slika 3: Arhitektura sustava aplikacija - ZTA mikroservisi i aplikacije 5 - 8.

4.2.1 Aplikacija za modernu kriptografiju

Služi za šifriranje i dešifriranje podataka kako bi se osigurala njihova povjerljivost i integritet. Koristi se za zaštitu osjetljivih informacija prilikom njihovog prijenosa ili pohrane.

Aplikacija za modernu kriptografiju sastoji se od sljedećih mikroservisa:

1. Mikroservis za enkripciju

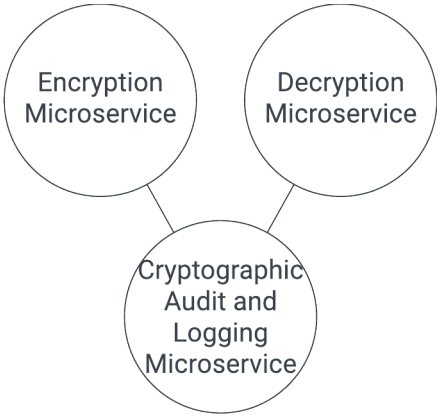
   Služi za enkripciju podataka.

2. Mikroservis za dekripciju

   Služi za dekripciju podataka.

3. Mikroservis za bilježenje

   Koristi se za praćenje i zapisivanje različitih događaja, aktivnosti i informacija koje se generiraju unutar aplikacije.



Slika 4: Arhitektura aplikacije za modernu kriptografiju

4.2.2 Generator autentikacijskih ključeva i tokena

Ova aplikacija generira i provejrava autentikacijske ključeve, tokene i druge oblike identifikatora koji se koriste za provjeru identiteta i autorizaciju pristupa raznim servisima ili sustavima.

Generator autentikacijskih ključeva i tokena sastoji se od sljedećih mikroservisa:

1. Mikroservis za generiranje

   Služi za generiranje autentikacijskih ključeva i tokena.

2. Mikroservis za verifikaciju

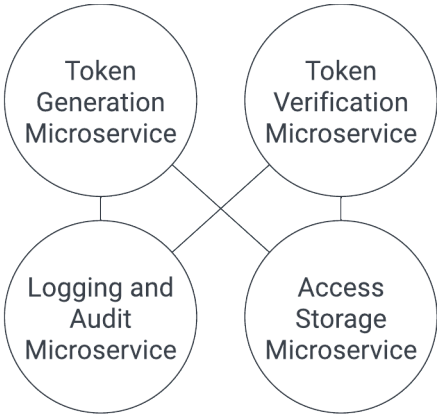
   Služi za verifikaciju ili provjeru autentikacijskih ključeva i tokena.

3. Mikroservis za spremanje

   Koristi se za upravljanje pohranom podataka i omogućuje ostalim mikroservisima pohranjivanje, dohvaćanje i ažuriranje podataka.

4. Mikroservis za bilježenje

   Koristi se za praćenje i zapisivanje različitih događaja, aktivnosti i informacija koje se generiraju unutar aplikacije.



Slika 5: Arhitektura generatora autentikacijskih ključeva i tokena

4.2.3 Aplikacija za verifikaciju digitalnih potpisa

Koristi se za provjeru autentičnosti digitalnih potpisa kako bi se osiguralo da su elektronički dokumenti ili transakcije autentični i da nisu ni promjenjeni ni krivotvoreni.

Aplikacija za verifikaciju digitalnih potpisa sastoji se od sljedećih mikroservisa:

1. Mikroservis za verifikaciju

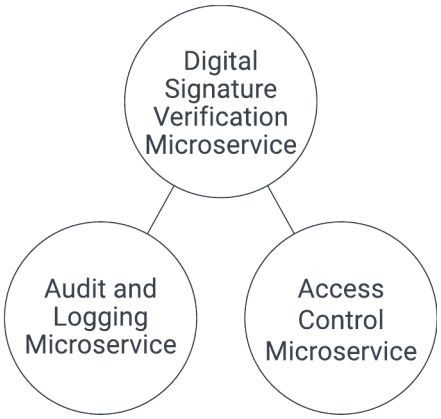
   Služi za verifikaciju digitalnih potpisa.

2. Mikroservis za kontrolu pristupa

   Koristi se za definiranje, upravljanje i provođenje pravila koja određuju tko ima pravo pristupa resursima aplikacije.

3. Mikroservis za bilježenje

   Koristi se za praćenje i zapisivanje različitih događaja, aktivnosti i informacija koje se generiraju unutar aplikacije.



Slika 6: Arhitektura aplikacije za verifikaciju digitalnih potpisa

4.2.4 Aplikacija za hashiranje i checksum

Ova aplikacija služi za generiranje i provjeru *hash* vrijednosti radi osiguranja integriteta podataka. Generaranje *hash* vrijednosti odnosi se na pretvaranje podataka u fiksni skup znakova, a provjera na korištenje *checksuma* za provjeru integriteta podataka [35].

Aplikacija za *hashiranje* i *checksum* sastoji se od sljedećih mikroservisa:

1. Mikroservis za *hashiranje*

   Služi za *hashiranje* podataka.

2. Mikroservis za verifikaciju *checksuma*

   Služi za verifikaciju *checksuma*.

3. Mikroservis za uzbunjivanje i izvješćivanje

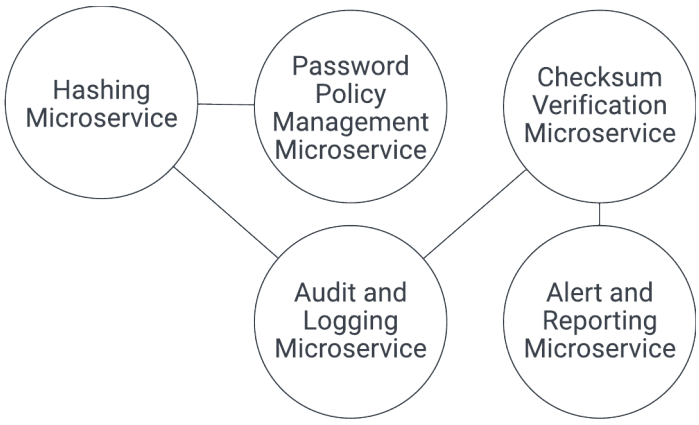
   Koristi se za automatizirano obavještavanje o problemima ili incidentima pri neuspješnoj verifikaciji *checksuma*.

4. Mikroservis za provođenje uvjeta

   Koristi se za definiranje, upravljanje i provođenje pravila i politika vezanih uz lozinke unutar aplikacije.

5. Mikroservis za bilježenje

   Koristi se za praćenje i zapisivanje različitih događaja, aktivnosti i informacija koje se generiraju unutar aplikacije.



Slika 7: Arhitektura aplikacije za *hashiranje* i *checksum*

4.2.5 Upravitelj lozinki

Ova aplikacija omogućuje sigurno pohranjivanje, generiranje i upravljanje lozinkama za različite račune. Cilj je olakšati korištenje jakih, jedinstvenih lozinki.

Upravitelj lozinki sastoji se od sljedećih mikroservisa:

1. Mikroservis za spremanje

   Služi za spremanje lozinki, upravljanje pohranom podataka i omogućuje ostalim mikroservisima pohranjivanje, dohvaćanje i ažuriranje podataka.

2. Mikroservis za verifikaciju

   Služi za verifikaciju lozinki.

3. Mikroservis za ponovo postavljanje

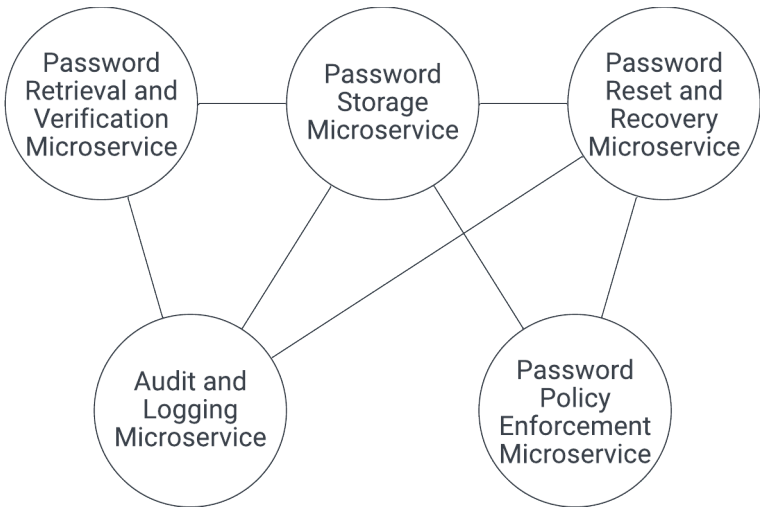
   Služi za ponovo postavljanje lozinki.

4. Mikroservis za provođenje uvjeta

   Koristi se za definiranje, upravljanje i provođenje pravila i politika vezanih uz lozinke unutar aplikacije.

5. Mikroservis za bilježenje

   Koristi se za praćenje i zapisivanje različitih događaja, aktivnosti i informacija koje se generiraju unutar aplikacije.



Slika 8: Arhitektura upravitelja lozinki

4.2.6 Aplikacija za sigurno spremanje datoteka

Koristi se za enkripciju i zaštitu osjetljivih datoteka kako bi se spriječio neovlašten pristup ili gubitak.

Aplikacija za sigurno spremanje datoteka sastoji se od sljedećih mikroservisa:

1. Mikroservis za dijeljenje

   Služi za prihvaćanje i slanje podataka o datotekama korisniku.

2. Mikroservis za spremanje

   Koristi se za upravljanje pohranom podataka i omogućuje ostalim mikroservisima pohranjivanje, dohvaćanje i ažuriranje podataka.

3. Mikroservis za kriptografiju

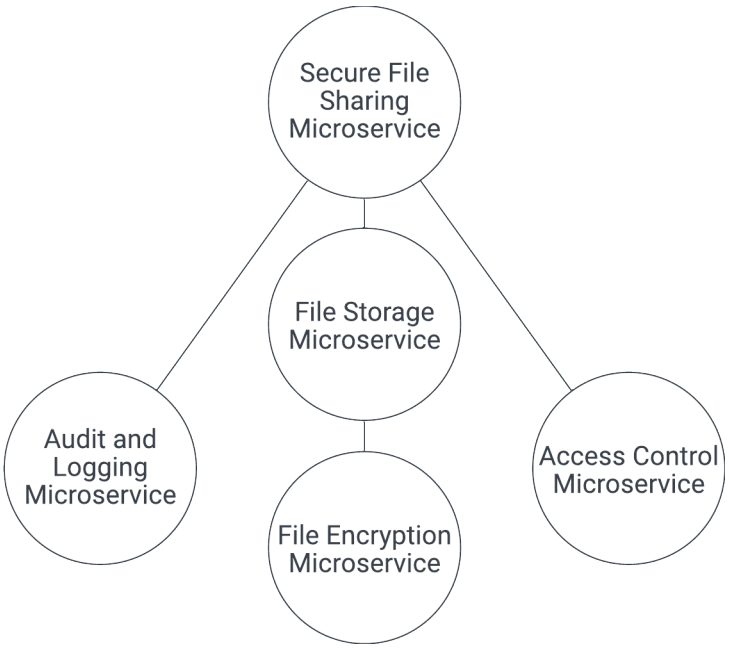
   Koristi se za pružanje sigurnosnih funkcionalnosti poput enkripcije i dekripcije prilikom spremanja i dohvaćanja podataka.

4. Mikroservis za kontrolu pristupa

   Koristi se za definiranje, upravljanje i provođenje pravila koja određuju tko ima pravo pristupa resursima aplikacije.

5. Mikroservis za bilježenje

   Koristi se za praćenje i zapisivanje različitih događaja, aktivnosti i informacija koje se generiraju unutar aplikacije.



Slika 9: Arhitektura aplikacije za sigurno spremanje datoteka

4.2.7 Aplikacija za maskiranje podataka

Služi za maskiranje ili anonimizaciju podataka kako bi se zaštitila privatnost subjekata unutar podataka ili osigurala usklađenost s regulativama o zaštiti privatnosti.

Aplikacija za maskiranje podataka sastoji se od sljedećih mikroservisa:

1. Mikroservis za maskiranje podataka

   Služi za maskiranje podataka i slanje originalnih podataka korisniku.

2. Mikroservis za spremanje

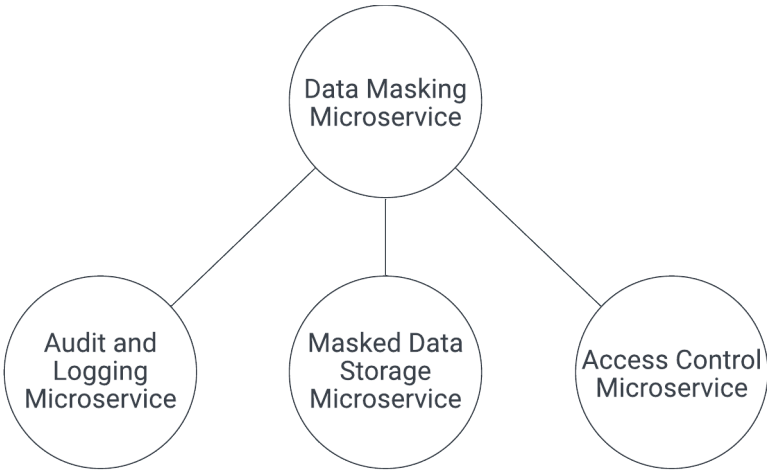
   Koristi se za upravljanje pohranom podataka i omogućuje ostalim mikroservisima pohranjivanje, dohvaćanje i ažuriranje podataka.

3. Mikroservis za kontrolu pristupa

   Koristi se za definiranje, upravljanje i provođenje pravila koja određuju tko ima pravo pristupa resursima aplikacije.

4. Mikroservis za bilježenje

   Koristi se za praćenje i zapisivanje različitih događaja, aktivnosti i informacija koje se generiraju unutar aplikacije.



Slika 10: Arhitektura aplikacije za maskiranje podataka

4.2.8 Aplikacija za dijeljenje podataka o kibernetičkim prijetnjama

Ova aplikacija omogućuje dijeljenje informacija o trenutnim kibernetičkim prijetnjama i sigurnosnim incidentima radi bolje zaštite i odgovora na sigurnosne prijetnje.

Aplikacija za dijeljenje podataka o kibernetičkim prijetnjama sastoji se od sljedećih mikroservisa:

1. Mikroservis za izvješćivanje o sigurnosnim incidentima

   Služi za prihvaćanje podataka o sigurnosnim incidentima korisnika.

2. Mikroservis za dijeljenje podataka o kibernetičkim prijetnjama

   Služi za slanje podataka o kibernetičkim prijetnjama korisniku.

3. Mikroservis za dijeljenje podataka

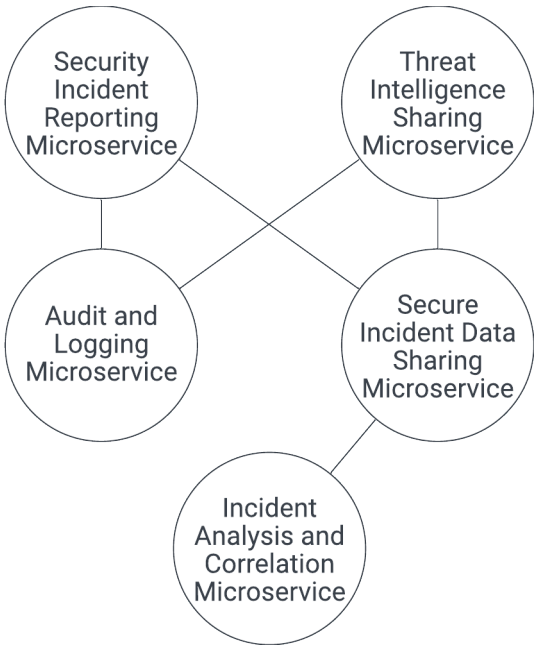
   Koristi se za upravljanje pohranom podataka i omogućuje ostalim mikroservisima pohranjivanje, dohvaćanje i ažuriranje podataka.

4. Mikroservis za analizu

   Koristi se za analizu sigurnosnih incidenata i provjeru jesu li oni vezani za ovaj sustav.

5. Mikroservis za bilježenje

   Koristi se za praćenje i zapisivanje različitih događaja, aktivnosti i informacija koje se generiraju unutar aplikacije.



Slika 11: Arhitektura aplikacije za dijeljenje podataka o kibernetičkim prijetnjama

4.2.9 Servisi arhitekture nultog povjerenja

Mikroservisi arhitekture nultog povjerenja služe kao temelj za sigurnost u ovom sustavu [36], fokusirajući se na minimiziranje povjerenja između komponenti. Svaki mikroservis obavlja specifičnu ulogu kao što su autentikacija, autorizacija, enkripcija podataka, segmentacija mreže te praćenje događaja, osiguravajući tako integritet, povjerljivost i dostupnost podataka uz visoku razinu zaštite od potencijalnih kibernetičkih prijetnji. U sustav implementirani su sljedeći mikroservisi arhitekture nultog povjerenja:

1. Mikroservis za upravljanje

Služi za koordinaciju, nadzor, upravljanje i rješavanje problema pri drugim mikroservisima unutar sustava.

2. Mikroservis za identitete i autentikaciju

   Koristi se za autentikaciju korisnika.

3. Mikroservis za segmentaciju mreže

   Služi za segmentaciju mreže ograničavanjem korisničkog pristupa samo jednoj aplikaciji sustava.

4. Mikroservis za kontrolu pristupa

   Koristi se za definiranje, upravljanje i provođenje pravila koja određuju tko ima pravo pristupa resursima aplikacije.

5. Mikroservis za sigurnu komunikaciju

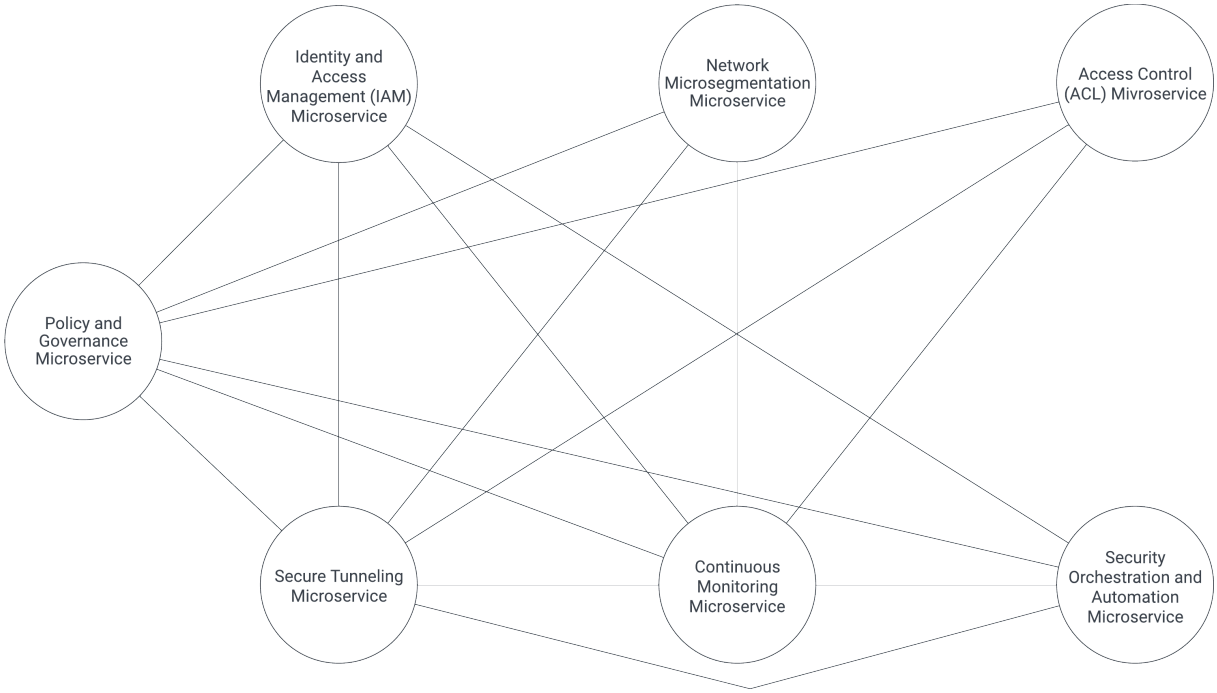
   Služi za komunikaciju između ostalih mikroservisa u sustavu i obavlja autorizaciju korisnika prilikom upita.

6. Mikroservis za kriptografiju

   Koristi se za pružanje funkcionalnosti poput enkripcije, dekripcije i *hashiranja* svim ostalim mikroservisima u sustavu.

7. Mikroservis za praćenje

   Služi za praćenje i zapisivanje različitih događaja, aktivnosti i informacija koje se generiraju unutar sustava.



Slika 12: Arhitektura servisa nultog povjerenja

4.3 Komponente i moduli

Kroz modularni dizajn, sustav se dijeli na manje, nezavisne jedinice koje zajednički ostvaruju kompleksne funkcionalnosti, omogućavajući bolju organizaciju, lakše održavanje i veću fleksibilnost [36]. Svaki mikroservis unutar sustava je samostalan, što znači da može neovisno funkcionirati i komunicirati s drugim dijelovima sustava putem jasno definiranih sučelja. Svaki mikroservis sadrži potrebne funkcije u posebnoj datoteci, čime se postiže visoka razina ponovne upotrebljivosti koda. Takva organizacija ne samo da pojednostavljuje razvoj i održavanje, već također omogućuje lakše prilagodbe i proširenja funkcionalnosti prema potrebi.

Mikroservisi su dizajnirani da budu kohezivni, što znači da su svi dijelovi mikroservisa usko povezani i surađuju kako bi ostvarili specifične ciljeve. Oni djeluju kao povezani sustav, gdje svaki doprinosi ukupnoj funkcionalnosti i stabilnosti. Ova struktura omogućuje da se sustav može nositi s manjim greškama i izbjegava potpuni pad zbog lokaliziranih problema. Međutim, unutar sustava postoje jedinstvene točke kvarova (*Single points of failure*, SPOF). To su mikroservis za kriptografiju i mikroservis za upravljanje u arhitekturi nultog povjerenja. Iako su ti mikroservisi kritični za rad sustava, sustav je konstruiran na način da može izdržati manje greške, zahvaljujući redundantnosti ostalih komponenti.

Sustav je osmišljen tako da njegove komponente budu samostalne, lako zamjenjive i unapređivane. Svaka komponenta može se zasebno testirati, što omogućuje detaljno provjeravanje ispravnosti i funkcionalnosti prije nego što se integrira u cjelinu. Takav pristup smanjuje rizik od neočekivanih grešaka i olakšava održavanje sustava tijekom vremena. Mikroservisi unutar sustava su izrazito kohezivni i skalabilni, omogućujući sustavu da se lako prilagođava promjenama i rastu. Informacije unutar sustava su skrivene, čime se postiže visoka razina sigurnosti, omogućujući da se promjene unutar jedne komponente ne odražavaju negativno na ostatak sustava.

4.4 Protok podataka i komunikacija

U arhitekturi ovog sustava, protok podataka i komunikacija igraju ključnu ulogu u osiguravanju efikasnosti i pouzdanosti. Sustav koristi HTTP protokol [37] za komunikaciju među mikroservisima, što znači da je potreban odgovor na svaki upit. Iako bi za neke servise RPC (*Remote Procedure Call*) protokol [38] bio bolji izbor zbog svoje efikasnosti, odlučeno je da se za HTTP radi jednostavnosti implementacije i održavanja homogenosti sustava.

Svaki mikroservis unutar sustava ima jedno ili više strogo definiranih sučelja koja služe za komunikaciju s drugim mikroservisima. Ova sučelja omogućuju jasnu i preciznu razmjenu informacija, čime se smanjuje mogućnost grešaka i poboljšava sigurnost. Podaci se najprije primaju na vanjskim mikroservisima koji su zaduženi za interakciju s vanjskim svijetom, a zatim se šalju prema unutarnjim mikroservisima koji obrađuju i pohranjuju podatke. Podaci se većinom dijele otvorenim tekstom (*plaintext*), što omogućuje jednostavnu i brzu razmjenu informacija. Međutim, zbog sigurnosnih izazova koje takav pristup nosi, ponekad se koristi enkripcija za osjetljive podatke. Dodatno, ključevi, tajne i enkriptirani tekstovi se kodiraju pomoću *base64* [39] prije slanja ili primanja. Tako je osigurana dodatna razina sigurnosti tijekom prijenosa podataka. Iako se podaci većinom prenose kao otvoreni tekst, svi podaci se pohranjuju u enkriptiranom obliku kako bi se osigurala njihova zaštita i integritet.

4.5 Razmatranja sigurnosnog dizajna

U dizajnu i arhitekturi sustava, sigurnost je od ključne važnosti kako bi se osigurala zaštita podataka i stabilnost sustava. Pristup sigurnosnom dizajnu ovog sustava uključuje niz mjera koje pozitivno utječu na sigurnost sustava, ali postoje i određena ograničenja koja je potrebno razmotriti.

Svaki upit u sustavu se validira na modelima korištenjem *Pydantic* biblioteke [40], što osigurava da su svi ulazni podaci ispravno oblikovani i sigurni za daljnju obradu. Uz to, svaki niz znakova (*string*) koji ulazi u sustav provjerava se kako bi se osiguralo da ne sadrži nedopuštene znakove, čime se smanjuje rizik od napada kao što su SQL injekcije (*SQL injection*) [41]. Pripremljeni SQL upiti s parametrima koriste se za rad s bazama podataka. To dodatno povećava sigurnost kao glavni način sprječavanja SQL injekcija. Svi podaci u sustavu spremaju se u enkriptiranom obliku, što osigurava njihovu zaštitu u slučaju neovlaštenog pristupa bazama podataka. Dostupnost vanjskih endpointa vanjskih mikroservisa izvan sustava ograničena je korištenjem *Kubernetes* tehnologije [34], čime se smanjuje površina napada. Samo određeni mikroservisi mogu komunicirati međusobno unutar sustava. Ovo je također osigurano pomoću *Kubernetesa* pa osigurava da određenom mikroservisu unutar sustava mogu pristupati samo autorizirani mikroservisi, čime se povećava sigurnost unutarnjih komunikacija.

U dizajniranom sustavu postoje određena ograničenja koja narušavaju sigurnost. Za ključeve i ostale tajne koriste se *placeholderi*, što znači da ih je potrebno zamijeniti pravim vrijednostima prije korištenja u stvarnoj situaciji. Ovo predstavlja sigurnosni rizik ukoliko se *placeholderi* ne zamijene pravilno. Sustav nema *frontend*, što znači da nije dodatno osiguran prijenos podataka između korisnika i vanjskih granica sustava. To je ranjivost i može biti problem ukoliko napadač prestretne podatke tijekom prijenosa [42]. Većina podataka unutar sustava dijeli se u otvorenom obliku (*plaintext*) zbog performansi, što predstavlja sigurnosni rizik u slučaju da netko neovlašteno pristupi mreži unutar sustava. Također, nije moguće ograničiti pristup određenoj aplikaciji specifičnim korisnicima ili grupama korisnika, nego se ograničenja pristupa mogu primijeniti samo na cijeli sustav, što smanjuje snagu kontrole pristupa.

Mjere sigurnosti i ograničenja nalaze se u tablici.

Tablica 1: Tablični popis mjera sigurnosti i ograničenja.

Mjere sigurnosti	Ograničenja
Validacija podataka upita korištenjem modela podataka uz *Pydantic* bibilioteku	Za ključeve i ostale tajne koriste se *placeholderi*
Validacija i sanitizacija nizova znakova (*stringova*) korištenjem *regexa* (*regular expression*)	Sustav nema *frontend*
Rad s korisničkim podacima u bazama podataka korištenjem pripremljenih SQL upita s parametrima	Većina podataka unutar sustava dijeli se u otvorenom obliku (*plaintext*)

Mjere sigurnosti	Ograničenja
Zaštita od neovlaštenog pristupa podacima iz baza podataka enkiptiranjem podataka prije spremanja	Nije moguće ograničiti pristup određenoj aplikaciji specifičnim korisnicima ili grupama korisnika, nego samo cijelom sustavu, odnosno svim aplikacijama
Dostupnost vanjskih pristupnih točki (*end-points*) vanjskih mikroservisa izvan sustava ograničena je korištenjem *Kubernetes* tehnologije	
Povećavanje sigurnosti unutarnjih komunikacija ograničavanjem komunikacija tako da određeni mikroservisi mogu komunicirati međusobno unutar sustava	

5 Implementacija i razvoj

Implementacija i razvoj predstavljaju ključnu fazu u stvaranju softverskog sustava, gdje se ideje i dizajnerski planovi pretvaraju u funkcionalni proizvod [43]. Oni zahtijevaju pažljivu selekciju tehnologija, definiranje softverskih komponenti, suočavanje s različitim izazovima te primjenu učinkovitih rješenja. Kroz temeljito testiranje, osigurava se da sustav zadovoljava sve postavljene zahtjeve i da je sve kvalitetno izgrađeno. U ovom poglavlju, detaljno su obrađene korištene tehnologije, opisane ključne softverske komponente, analizirani izazovi koji su se pojavili i prikazana implementirana rješenja. Također, objašnjeni su procesi testiranja koji su neophodni za postizanje pouzdanosti i stabilnosti sustava.

5.1 Tehnologije i softverske komponente

U procesu implementacije i razvoja sustava, ključna uloga pripala je izboru tehnologija koje omogućuju brz i efikasan razvoj, skalabilnost i sigurnost. Za programski jezik odabran je *Python* [44], poznat po svojoj jednostavnosti, čitljivosti koda i velikom broju standardnih biblioteka. *Python* pruža fleksibilnost i snažnu podršku za asinkrono programiranje, što je bitno za izgradnju sustava koji može učinkovito upravljati velikim brojem istodobnih zahtjeva. Za upravljanje bazama podataka koristi se *SQLite* [45], lagana i pouzdana relacijska baza podataka. *SQLite* je idealan za ovaj sustav zbog jednostavnog načina pohrane i pristupa podacima bez potrebe za postavljanjem složenih sustava upravljanja bazama podataka. U kombinaciji s *aiosqlite* bibliotekom [46], omogućeno je asinkrono upravljanje bazama podataka, čime se dodatno povećava učinkovitost sustava.

Razvoj *web* aplikacije oslanja se na nekoliko ključnih biblioteka. *Uvicorn* [47] i *FastAPI* [48] čine temelj *web* servera i pružaju brze i moderne alate za izradu API sučelja. *Pydantic* [40] se koristi za validaciju podataka iz upita. *Aiohttp* [49] i *asyncio* [50] biblioteke omogućuju asinkrono programiranje upita. Za sigurnost i enkripciju podataka koriste se *pycryptodome* [51], *jwt* [52], *cryptography* [53] i *bcrypt* [54]. *Pycryptodome* i *cryptography* pružaju alate za šifriranje i dešifriranje podataka. Tako se štite osjetljive informacija. Biblioteka *jwt* (*JSON Web Token*) koristi se manipulaciju istoimenim tokenima za autentifikaciju, dok *bcrypt* služi za sigurno pohranjivanje lozinki. *Faker* biblioteka [55] omogućuje generiranje lažnih podataka za potrebe maskiranja u aplikaciji za maskiranje podataka. *Pika* biblioteka [56] koristi se za rad s *RabbitMQ* [57]. Ona omogućuje pouzdano slanje obavijesti adminu sustava.

Kao tehnologiju za izolaciju mikroservisa korišten je *Docker* [33]. Napravljeni su *Docker images* koji omogućuju dosljednost okruženja, lako postavljanje varijabli okruženja i jednostavnu distribuciju aplikacije. Postavljanje svih dijelova sustava na *Kubernetes* [34] omogućilo je održavanje, automatsko skaliranje i visoku dostupnost aplikacije.

5.2 Izazovi i implementirana rješenja

U procesu implementacije i razvoja sustava, suočeno je s nizom tehničkih izazova koji su zahtijevali promišljena rješenja kako bi se osigurala sigurnost, pouzdanost i efikasnost sustava. U nastavku su detaljno opisani ključni izazovi i implementirana rješenja.

Jedan od ključnih izazova bio je zaštita od *SQL injection* napada, što je ozbiljna sigurnosna prijetnja za svaki sustav koji koristi SQL baze podataka. Implementirano rješenje uključivalo je upotrebu pripremljenih SQL upita sa parametrima. To sprječava direktno umetanje zlonamjernih kodova unutar SQL upita. Dodatno, implementirano je provjeravanje nizova znakova (*string*) pomoću regularnih izraza (*regex, regular expression*), kako bi se filtrirali i validirali ulazni podaci prije nego što se proslijede bazi podataka.

Validacija upita predstavljala je još jedan značajan izazov, posebno u kontekstu osiguravanja da svi podaci koji ulaze u sustav budu točni i ispravni. Kao rješenje, korištena je *Pydantic* biblioteka kojom su se jednostavno i efikasno validirali podaci pomoću modela definiranih u *Pythonu*. *Pydantic* omogućava automatsku provjeru tipova podataka, te osigurava da svi podaci ispunjavaju specificirane uvjete prije nego što se dalje obrade.

Rješavanje kobnih problema koji mogu srušiti sustav zahtijevalo je poseban pristup. U tu svrhu, razvijen je mikroservis za upravljanje koji je zadužen za otkrivanje i rješavanje problema unutar sustava. Ovaj mikroservis čeka na upit sa problemom ostalih komponenti sustava i rješava ih. Ukoliko mikroservis nije u mogućnosti riješiti problem, šalje se obavijest adminu. Time se značajno smanjuje vrijeme zastoja i poboljšava funkcioniranje sustava. Implementacijom sustava za slanje obavijesti administratorima omogućavajući pravovremene reakcije i poduzimanje potrebnih mjera za otklanjanje problema.

Enkripcija podataka prije spremanja također je bila kritična komponenta sigurnosne arhitekture sustava. Kako bi se osigurala privatnost i sigurnost podataka, implementiran je mikroservis za kriptografiju koji služi cijelom sustavu. Ovaj mikroservis pruža funkcionalnosti za enkripciju i dekripciju podataka, koristeći najnovije kriptografske algoritme. Uz pomoć njega se osjetljivi podaci štite od neovlaštenog pristupa.

Posljednji, ali ne manje važan izazov, bio je osigurati visoke performanse i modularnost sustava. Rješenje za ovaj izazov pronađeno je u *dockerizaciji* i korištenju *Kubernetes* platforme. *Dockerizacijom* se napravilo pakiranje aplikacija u kontejnere da bi se osiguralo konzistentno okruženje za razvoj, testiranje i produkciju. S *Kubernetes* se, s druge strane, napravilo povezivanje kontejnera te se tako omogućilo skaliranje, upravljanje i održavanje aplikacija na distribuiranoj infrastrukturi. Ova kombinacija tehnologija značajno je pojednostavila upravljanje resursima i omogućila lakše održavanje i nadogradnju sustava.

Izazovi i implementirana rješenja nalaze se u tablici.

Tablica 2: Tablični popisi izazova i implementiranih rješenja.

Izazovi	**Implementirana rješenja**
Zaštita od *SQL injection* napada	Upotreba pripremljenih SQL upita sa parametrima i provjeravanje nizova znakova (*string*) pomoću regularnih izraza (*regex, regular expression*)
Validacija upita	Validacija podataka pomoću modela definiranih u *Pythonu* korištenjem *Pydantic* biblioteke
Rješavanje kobnih problema koji mogu srušiti sustav	Mikroservis za upravljanje koji je zadužen za otkrivanje i rješavanje problema unutar sustava te slanje obavijesti adminu
Privatnost i sigurnost spremljenih podataka	Enkripcija podataka prije spremanja korištenjem mikroservisa za kriptografiju
Osiguravanje visokih performansi i modularnosti sustava	*Docker* i *Kubernetes*

5.3 Testiranje i validacija

Kako bi se osiguralo da programsko rješenje radi kako treba, potrebno ga je temeljito testirati. Testiranje je ključni korak u razvoju softvera koji omogućava identificiranje i ispravljanje grešaka prije nego što sustav bude implementiran u produkcijsko okruženje. Kvalitetno testiranje osigurava da sve funkcionalnosti sustava rade ispravno i prema očekivanjima, čime se povećava pouzdanost i sigurnost aplikacije.

Za testiranje sustava korišten je *Pytest* [58], popularni okvir (*framework*) za testiranje u *Pythonu*. *Pytest* omogućava pisanje jednostavnih testova. Prednosti korištenja *Pytesta* uključuju jednostavnost sintakse, mogućnost automatskog otkrivanja testova, te podršku za složenije scenarije testiranja. Korištenjem *Pytesta*, osigurava se da svi dijelovi koda rade u skladu s definiranim specifikacijama. U procesu testiranja testirani su odgovori na poslane upite. Ovo je uključivalo provjeru da li sustav ispravno obrađuje različite vrste upita te da li odgovori zadovoljavaju očekivane kriterije. Testirali su se scenariji poput uspješnog izvršavanja upita, rukovanja pogreškama, te validacije ulaznih podataka.

6 Evaluacija

6.1 Usporedba s postojećim rješenjima

Prilikom evaluacije razvijenog sustava, važno je usporediti ga s postojećim rješenjima na tržištu kako bi se procijenila njegova kvaliteta, sigurnost i jedinstvenost. Ovaj se sustav po kvaliteti može usporediti sa svim postojećim rješenjima. Implementacija sustava je izvedena pozorno, s naglaskom na pouzdanost, učinkovitost i preciznost, što ga čini konkurentnim u odnosu na druge dostupne opcije. Kao i većina postojećih rješenja, ovaj sustav je usredotočen na sigurnost sustava slijedeći *Zero Trust Architecture* (ZTA) koncepte, odnosno načela. Ova paradigma sigurnosti ima sposobnost da minimizira rizike povezane s povjerenjem u postojeće sigurnosne modele [30]. Implementacija ZTA principa osigurava da svaki zahtjev za pristup bude provjeren, što dodatno povećava ukupnu sigurnost sustava. Jedan od ključnih aspekata koji ovaj rad čini jedinstvenim je činjenica da je ovo jedini pronađeni sustav aplikacija otvorenog koda koji pruža jako specifične funkcionalnosti implementiran kao servis za korištenje u drugim aplikacijama. Ostali sustavi pružaju vrlo opće funkcionalnosti. Uz to, ovaj sustav, u odnosu na druge, omogućava visoku razinu fleksibilnosti i prilagodljivosti u integraciji tih specifičnih funkcionalnosti. Ove funkcionalnosti, odnosno algoritmi, su implementirane tako da zadovoljavaju najviše standarde industrije, pružajući korisnicima pouzdane sigurnosne usluge. Ova jedinstvena kombinacija open source pristupa i specijaliziranih sigurnosnih algoritama čini ovo programsko rješenje posebno vrijednim resursom za razvojne timove koji traže fleksibilna i pouzdana sigurnosna rješenja. Time se ova implementacija ne samo ističe kao pouzdana alternativa postojećim rješenjima, već također nudi jedinstvene prednosti koje ga čine iznimno vrijednim doprinosom zajednici.

6.2 Ograničenja razvijenog sustava

Kroz proces evaluacije sustava, identificirano je nekoliko ograničenja koja utječu na njegovu funkcionalnost i pouzdanost. Jedan od glavnih nedostataka je prisutnost jedinstvenih točaka kvarova (*single points of failure*). Ovi pojedinačni elementi sustava, ako zakažu, mogu uzrokovati prestanak rada cijelog sustava, što predstavlja značajan rizik za stabilnost i dostupnost usluge. Kontrola pristupa u trenutnom sustavu također ima svoja ograničenja. U slučaju kvara dijela sustava, sustav nije sposoban ograničiti pristup samo tom dijelu; umjesto toga, pristup cijelom sustavu se automatski ograničava. Ovo može dovesti do nepotrebnog prekida rada za korisnike koji možda i ne koriste pogođeni dio sustava, smanjujući ukupnu dostupnost i pouzdanost sustava.

Također, sustav još uvijek nije potpuno spreman za postavljanje na poslužitelj. U trenutnom stanju, sustav koristi *placeholdere* kao vrijednosti za ključeve i tajne unutar enkripcijskih algoritama. Ovo je privremeno rješenje koje mora biti zamijenjeno stvarnim, sigurnim ključevima i tajnama prije nego što sustav bude spreman za produkcijsku upotrebu.

Iako sustav pruža određene funkcionalnosti u domeni računalne sigurnosti, on ne pokriva sve moguće algoritme. To ostavlja prostor za proširenje, što znači da bi buduće verzije sustava mogle uključiti dodatne algoritme i time povećati njegovu korisnost i primjenjivost.

Jedan od značajnih nedostataka je i nedostatak podrške za automatsko ponovno pokretanje dijelova sustava u slučaju njihovog pada. Sustav nema ugrađene mehanizme za detekciju i automatsko ponovno pokretanje problematičnih komponenti, što zahtijeva ručnu intervenciju za vraćanje sustava u funkcionalno stanje nakon kvara. Dodatno, nije razvijen automatiziran način postavljanja cijelog sustava. Razlog za to leži u činjenici da to nije bio fokus ovog diplomskog rada. Implementacija automatiziranog postavljanja bi značajno olakšala implementaciju i održavanje sustava, ali zahtijeva dodatno vrijeme i resurse za razvoj.

7 Zaključak

Uspješno je implementiran sustav aplikacija u domeni računalne sigurnosti, čime je ostvaren značajan napredak u zaštiti informacija i sustava od potencijalnih prijetnji. Tijekom razvoja poštovana su sva načela arhitekture nultog povjerenja, čime je osigurano da svaki aspekt sustava zadovoljava najviše standarde sigurnosti. Arhitektura nultog povjerenja, koja se temelji na pretpostavci da nijedan entitet unutar mreže nije automatski pouzdan, primijenjena je dosljedno, omogućujući minimalizirajući rizik od neovlaštenih upada. Ovo načelo dodatno je pojačalo otpornost sustava na potencijalne prijetnje, osiguravajući snažnu zaštitu podataka. Programsko rješenje postavljeno je u *Docker* i na *Kubernetes*, što je omogućilo jednostavniju implementaciju, skalabilnost i upravljivost sustava. Korištenje *Docker* kontejnera omogućilo je izolaciju aplikacija i jednostavnije upravljanje ovisnostima (*dependencies*), dok je *Kubernetes* osigurao orkestraciju kontejnera i automatizaciju procesa implementacije i održavanja.

Sustav aplikacija ističe se među sličnim postojećim rješenjima funkcionalnostima i opsegom. Integracija naprednih sigurnosnih mjera, zajedno s fleksibilnom i skalabilnom infrastrukturom, omogućila je stvaranje rješenja koje zadovoljava potrebe korisnika. Specifične funkcionalnosti ovog sustava čine ga posebnim u usporedbi s konkurentskim rješenjima.

7.1 Doprinosi području

Pokazan je jedan od načina kako implementirati arhitekturu nultog povjerenja, čime je pružen značajan doprinos području računalne sigurnosti. Ova arhitektura, koja pretpostavlja da nijedan entitet unutar mreže nije automatski pouzdan, demonstrirana je kroz konkretan sustav koji implementira stroge sigurnosne mjere i kontrolu pristupa. Ovaj pristup osigurava visoku razinu zaštite podataka, minimizirajući rizike od neovlaštenog pristupa i unutarnjih prijetnji. Primjena nultog povjerenja u praktičnom okruženju pruža dragocjene smjernice i primjer drugim stručnjacima i organizacijama u nastojanju da unaprijede svoje sigurnosne protokole.

Razvijen je sustav aplikacija u domeni računalne sigurnosti otvorenog koda, što predstavlja korak naprijed u dostupnosti naprednih sigurnosnih rješenja. Kroz otvoreni kod, ova aplikacija omogućava široj zajednici stručnjaka i istraživača da pregledaju, modificiraju i unapređuju rješenje, potičući inovacije i kolaboraciju unutar zajednice računalne sigurnosti. Ovaj doprinos omogućava širenje najboljih praksi i pomaže u izgradnji sigurnijeg digitalnog okruženja za sve korisnike.

7.2 Praktične primjene

Razni dijelovi sustava korisnici mogu koristiti kao vanjski servis u razvoju vlastitih aplikacija, čime se omogućava integracija naprednih sigurnosnih funkcionalnosti u različite softverske projekte. Ovaj pristup pruža fleksibilnost i prilagodljivost, omogućavajući korisnicima da iskoriste specifične komponente sustava koje najbolje odgovaraju njihovim potrebama. Korištenjem ovih servisa, programeri mogu značajno smanjiti vrijeme i resurse potrebne za implementaciju vlastitih sigurnosnih rješenja, dok istovremeno osiguravaju visoku razinu zaštite podataka i sustava.

Sustav također služi kao opširan primjer funkcioniranja različitih algoritama u domeni računalne sigurnosti. Integracija i demonstracija širokog spektra algoritama unutar jednog sustava omogućava korisnicima da prouče i razumiju kako ti algoritmi rade u stvarnim scenarijima. Ova karakteristika čini sustav vrijednim obrazovnim alatom, korisnim za studente, istraživače i profesionalce koji žele proširiti svoje znanje i vještine u području računalne sigurnosti. Analiza i eksperimentiranje s različitim algoritmima unutar ovog sustava pomaže u stjecanju dubljeg uvida u njihove prednosti, nedostatke i načine primjene u praksi.

7.3 Implikacije i područja za budući razvoj

Sljedeći korak u razvoju bio bi poboljšavanje same arhitekture sustava kako bi sustav mogao normalno funkcionirati iako je djelomično u kvaru, pomoću poboljšane kontrole pristupa. Implementacija dodatnih mehanizama kontrole pristupa poput ograničavanja pristupa određenoj aplikaciji omogućila bi dostupnost sustava čak i u slučaju nepredviđenih problema. Također, bilo bi dobro poboljšati samu arhitekturu sustava da se izbjegnu jedinstvene točke kvarova. Trenutna infrastruktura može se unaprijediti distribucijom kritičnih komponenti i uvođenjem redundantnih sustava koji preuzimaju funkcije u slučaju kvara pojedinih dijelova. Ovakav pristup omogućio bi kontinuiranu dostupnost i smanjio rizik od prekida rada, što je ključno za aplikacije koje zahtijevaju visoku razinu pouzdanosti i sigurnosti.

Predlaže se proširenje sustava tako da se podrže ostali algoritmi koji još nisu implementirani. Integracija novih algoritama omogućila bi korisnicima pristup širem spektru sigurnosnih alata i tehnika, čime bi se dodatno unaprijedila svestranost. Proširenje podrške za različite algoritme također bi omogućilo korisnicima više opcija prilagodbi sigurnosnih rješenja potrebama svojih projekata.

Popis literature

[1] S. P. Marsh, S. Univ. (GB). Department of Computing Science, Mathematics, U. of Stirling. Department of Computing Science, i Mathematics, *Formalising trust as a computational concept*. u Technical report (university of stirling. Department of computing science and mathematics). University of Stirling, 1994. [Na internetu]. Available: https://books.google.hr/books?id=lrfONwAACAAJ

[2] U. Göksel, M. ALKAN, İ. A. Doğru, i M. Dörterler, „Perimeter network security solutions: A survey“, u *2019 3rd international symposium on multidisciplinary studies and innovative technologies (ISMSIT)*, IEEE, 2019, str. 1–6.

[3] M. Abomhara i G. M. Køien, „Cyber security and the internet of things: Vulnerabilities, threats, intruders and attacks.“, *J. Cyber Secur. Mobil.*, sv. 4, izd. 1, str. 65–88, 2015.

[4] S. Rose, O. Borchert, S. Mitchell, i S. Connelly, „Zero trust architecture“, kol. 2020, doi: 10.6028/nist.sp.800-207.

[5] A. Kerman, O. Borchert, S. Rose, A. Tan, i others, „Implementing a zero trust architecture“, *National Institute of Standards and Technology (NIST)*, sv. 75, 2020.

[6] Q. Jin i L. Wang, „Zero-trust based distributed collaborative dynamic access control scheme with deep multi-agent reinforcement learning“, *EAI Endorsed Transactions on Security and Safety*, sv. 8, izd. 27, 2020.

[7] M. Hall, „Why people are key to cyber-security“, *Network Security*, sv. 2016, izd. 6, str. 9–10, 2016, doi: https://doi.org/10.1016/S1353-4858(16)30057-5.

[8] Y. Li, D. Nguyen, i M. Xie, „EZSetup: A novel tool for cybersecurity practices utilizing cloud resources“, *Proceedings of the 18th Annual Conference on Information Technology Education*, ruj. 2017, doi: 10.1145/3125659.3125699.

[9] A. Furfaro, A. Piccolo, A. Parise, L. Argento, i D. Saccà, „A cloud-based platform for the emulation of complex cybersecurity scenarios“, *Future Generation Computer Systems*, sv. 89, str. 791–803, pros. 2018, doi: 10.1016/j.future.2018.07.025.

[10] M. T. Rossi, R. Greca, L. Iovino, G. Giacinto, i A. Bertoli, „Defensive programming for smart home cybersecurity“, *2020 IEEE European Symposium on Security and Privacy Workshops (EuroS&PW)*, ruj. 2020, doi: 10.1109/eurospw51379.2020.00087.

[11] Y. Nikoloudakis *i sur.*, „Towards a machine learning based situational awareness framework for cybersecurity: An sdn implementation“, *Sensors*, sv. 21, izd. 14, str. 4939, srp. 2021, doi: 10.3390/s21144939.

[12] C. Huang, „Cybersecurity management system: Defense and response“, stu. 2022.

[13] D. Lee, D. Kim, C. Lee, M. K. Ahn, i W. Lee, „ICSTASY: An integrated cybersecurity training system for military personnel“, *IEEE Access*, sv. 10, str. 62232–62246, 2022, doi: 10.1109/access.2022.3182383.

[14] C. Roque, G. Moodley, i S. Mandal, „Cybersafe: Gamifying cybersecurity training with a training app“, *International Conference on Cyber Warfare and Security*, sv. 19, izd. 1, str. 299–307, ožu. 2024, doi: 10.34190/iccws.19.1.2198.

[15] D. Eidle, S. Y. Ni, C. DeCusatis, i A. Sager, „Autonomic security for zero trust networks“, *2017 IEEE 8th Annual Ubiquitous Computing, Electronics and Mobile Communication Conference (UEMCON)*, lis. 2017, doi: 10.1109/uemcon.2017.8249053.

[16] R. Vanickis, P. Jacob, S. Dehghanzadeh, i B. Lee, „Access control policy enforcement for zero-trust-networking“, *2018 29th Irish Signals and Systems Conference (ISSC)*, lip. 2018, doi: 10.1109/issc.2018.8585365.

[17] Z. Zaheer, H. Chang, S. Mukherjee, i J. Van der Merwe, „EZTrust: Network-independent zero-trust perimeterization for microservices“, *Proceedings of the 2019 ACM Symposium on SDN Research*, tra. 2019, doi: 10.1145/3314148.3314349.

[18] I. Ahmed, T. Nahar, S. S. Urmi, i K. A. Taher, „Protection of sensitive data in zero trust model“, *Proceedings of the International Conference on Computing Advancements*, sij. 2020, doi: 10.1145/3377049.3377114.

[19] W. R. Simpson i K. E. Foltz, „Network segmentation and zero trust architectures“, *Lecture Notes in Engineering and Computer Science: Proceedings of The World Congress on Engineering 2021, London, U.K.*, str. 201–206, srp. 2021.

[20] C. de Weever i M. Andreou, „Zero trust network security model in containerized environments“, velj. 2020.

[21] D. D'Silva i D. D. Ambawade, „Building a zero trust architecture using kubernetes“, *2021 6th International Conference for Convergence in Technology (I2CT)*, tra. 2021, doi: 10.1109/i2ct51068.2021.9418203.

[22] B. Paul i M. Rao, „Zero-trust model for smart manufacturing industry“, *Applied Sciences*, sv. 13, izd. 1, str. 221, pros. 2022, doi: 10.3390/app13010221.

[23] L. EKMAN i V. PERO, „Implementing a zero trust environment for an existing on-premises cloud solution“, 2023.

[24] A. MARTIRADONNA, „Zero trust architecture in a multi-cloud environment“, srp. 2023.

[25] V. N. Chimakurthi, „The challenge of achieving zero trust remote access in multi-cloud environment“, *ABC Journal of Advanced Research*, sv. 9, izd. 2, str. 89–102, pros. 2020, doi: 10.18034/abcjar.v9i2.608.

[26] S. Teerakanok, T. Uehara, i A. Inomata, „Migrating to zero trust architecture: Reviews and challenges“, *Security and Communication Networks*, sv. 2021, str. 1–10, svi. 2021, doi: 10.1155/2021/9947347.

[27] C. Buck, C. Olenberger, A. Schweizer, F. Völter, i T. Eymann, „Never trust, always verify: A multivocal literature review on current knowledge and research gaps of zero-trust“, *Computers & Security*, sv. 110, str. 102436, stu. 2021, doi: 10.1016/j.cose.2021.102436.

[28] Y. Cao, S. R. Pokhrel, Y. Zhu, R. Doss, i G. Li, „Automation and orchestration of zero trust architecture: Potential solutions and challenges“, *Machine Intelligence Research*, sv. 21, izd. 2, str. 294–317, sij. 2024, doi: 10.1007/s11633-023-1456-2.

[29] C. Uwaoma, „The challenges and processes of achieving optimal implementation of zero trust architecture in workplace“. EasyChair Preprint no. 10878.

[30] J. Garbis i J. W. Chapman, *Zero trust security: An enterprise guide*. Apress Berkeley, CA, 2021.

[31] J. Kindervag i others, „Build security into your network's dna: The zero trust network architecture“, *Forrester Research Inc*, sv. 27, str. 1–16, 2010.

[32] S. Mehraj i M. T. Banday, „Establishing a zero trust strategy in cloud computing environment“, u *2020 international conference on computer communication and informatics (ICCCI)*, IEEE, 2020, str. 1–6.

[33] D. Merkel, „Docker: Lightweight linux containers for consistent development and deployment“, *Linux J.*, sv. 2014, izd. 239, ožu. 2014.

[34] L. Mercl, V. Sobeslav, i P. Mikulecky, „Design of Reactive Systems for Control Network Traffic on the Kubernetes Platform“, u *Studies in computational intelligence*, 2019.

[35] K. Aggarwal i H. K. Verma, „Hash_RC6 — variable length hash algorithm using RC6“, u *2015 international conference on advances in computer engineering and applications*, 2015, str. 450–456. doi: 10.1109/ICACEA.2015.7164747.

[36] S. Newman, *Building microservices*, 1st izd. O'Reilly Media, Inc., 2015.

[37] E. Wilde, „Hypertext transfer protocol (HTTP)“, u *Wilde's WWW: Technical foundations of the world wide web*, Berlin, Heidelberg: Springer Berlin Heidelberg, 1999, str. 53–135. doi: 10.1007/978-3-642-95855-7_4.

[38] R. Thurlow, „RPC: Remote Procedure Call Protocol Specification Version 2“. u Request for comments. RFC 5531; RFC Editor, svi. 2009. doi: 10.17487/RFC5531.

[39] S. Josefsson, „The Base16, Base32, and Base64 Data Encodings“. u Request for comments. RFC 4648; RFC Editor, lis. 2006. doi: 10.17487/RFC4648.

[40] „Pydantic“. [Na internetu]. Available: https://docs.pydantic.dev/latest/

[41] M. A. Mohd Yunus, M. Brohan, N. Mohd Nawi, E. Salwana, N. Najib, i C. Liang, „Review of SQL injection : Problems and prevention“, *JOIV : International Journal on Informatics Visualization*, sv. 2, str. 215, lip. 2018, doi: 10.30630/joiv.2.3-2.144.

[42] F. Androulidakis Iosifand Kioupakis, „Interception of computer data“, u *Industrial espionage and technical surveillance counter measurers*, Cham: Springer International Publishing, 2016, str. 23–36. doi: 10.1007/978-3-319-28666-2_3.

[43] L. Hotz, A. Gunter, i T. Krebs, „A knowledge-based product derivation process and some ideas how to integrate product development“, u *Proc. Of software variability management workshop*, 2003, str. 136–140.

[44] G. Van Rossum i F. L. Drake Jr, *Python reference manual*. Centrum voor Wiskunde en Informatica Amsterdam, 1995.

[45] K. P. Gaffney, M. Prammer, L. Brasfield, D. R. Hipp, D. Kennedy, i J. M. Patel, „SQLite: Past, present, and future“, *Proc. VLDB Endow.*, sv. 15, izd. 12, str. 3535–3547, kol. 2022, doi: 10.14778/3554821.3554842.

[46] „API reference - aiosqlite documentation“. [Na internetu]. Available: https://aiosqlite.omnilib.dev/en/stable/api.html

[47] „Uvicorn“. [Na internetu]. Available: https://www.uvicorn.org/

[48] „FastAPI“. [Na internetu]. Available: https://fastapi.tiangolo.com/

[49] „Client reference - aiohttp 3.9.5 documentation“. [Na internetu]. Available: https://docs.aiohttp.org/en/stable/client_reference.html

[50] „Asyncio documentation“. [Na internetu]. Available: https://docs.python.org/3/library/asyncio.html

[51] „Welcome to PyCryptodome’s documentation - PyCryptodome 3.210b0 documentation“. [Na internetu]. Available: https://pycryptodome.readthedocs.io/en/latest/

[52] „PyPI jwt“. [Na internetu]. Available: https://pypi.org/project/jwt/

[53] „PyPI cryptography“. [Na internetu]. Available: https://pypi.org/project/cryptography/

[54] „PyPI bcrypt“. [Na internetu]. Available: https://pypi.org/project/bcrypt/

[55] „Welcome to faker’s documentation! - faker 25.9.1 documentation“. [Na internetu]. Available: https://faker.readthedocs.io/en/master/

[56] „Pika | read the docs“. [Na internetu]. Available: https://readthedocs.org/projects/pika/

[57] „RabbitMQ RSS“. [Na internetu]. Available: https://www.rabbitmq.com/docs

[58] H. Krekel, B. Oliveira, R. Pfannschmidt, F. Bruynooghe, B. Laugher, i F. Bruhin, „Pytest x.y“. 2004. [Na internetu]. Available: https://github.com/pytest-dev/pytest

Popis tablica

Popis slika