

Anonymitet och Integritet inom Krypterade Chattjänster

Anonymity and Privacy within Encrypted Chat Services

**Namn: William Johnsson**

**Handledare: Henrik Efson**

**Teknikprogrammet, läsår 2021/2024**

# **FÖRORD**

Denna rapport är ett examensarbete vid NTI Kronhus, teknikprogrammet i Göteborg, som pågick under hösten och våren 2023/2024. Arbetet har varit intressant, lärorikt och intensivt. Jag har utvecklat, analyserat och studerat användandet och tekniken bakom krypterade chattjänster med fokus på anonymitet och integritet för användaren.

Jag har själv utformat och utfört projektet. Men under projektets gång har jag tagit hjälp av ett antal personer som jag skulle vilja rikta ett stort tack till.

Först och främst vill jag tacka alla de personer som tog sig tid att svara på frågor och ge feedback kring behovet av anonyma och krypterade chattjänster.

Jag vill även tacka de personer som hjälpte till att testa projektets applikationstjänst, både under utvecklingsfasen och till slut den färdiga produkten.

Sedan vill jag rikta ett stort tack till Björn Johnsson för korrekturläsning av rapporten och allmänt stöd under projektets utformning.

Till sist vill jag rikta ett stort tack till Henrik Efsom som gav mig handledning genom projektet och inte minst guidade mig avseende upplägg och struktur.

Göteborg 2024-02-22

# **SAMMANFATTNING**

Frågor om anonymitet och integritet är mer på agendan än någonsin, speciellt inom tjänster som hanterar kommunikation eller informationsutbyte i datavärlden. Företag och privatpersoner har idag blivit alltmer beroende av kommunikation på nätet där säkerheten är av högsta betydelse. För att lättare förstå de svårigheter och utmaningar som tjänster och företag möter, fokuserar det här projektet på att analysera och granska de svårigheter som dessa tjänster medför. Slutmålet var att utveckla en egen krypterad chattjänst med fokus på anonymitet, säkerhet och användarens integritet. Dessutom har arbetet tittat på det efterarbete som krävs för att säkerställa integriteten för slutanvändaren. Rapporten behandlar även de risker som dessa tjänster medför kring olika typer av attacker och risker som kan kopplas till olika typer av klienter.

Något som också berörs är betydelsen av att anonyma tjänster inte används för att begå brott.

Projektet heter SentinelNetGuard och består av olika delar. En webbplats, en skrivbordsapplikation, serverprogramvara och programvara för en proxyserver.

Arbetet resulterade i en större förståelse för de svårigheter och utmaningar som anonymitet och krypterade chattjänster medför. Speciellt tydligt blir det där man som klient är helt eller delvis anonym vilket försvårar administrationsarbetet.

Projektet har utförts dels i skolmiljö, bibliotek och i hemmet.

Förhoppningsvis kan rapporten bidra till en större och mer fördjupad förståelse för de utmaningar som anonymiteten i samband med användandet av olika typer av chattapplikationer innebär.

Projektet påverkades till viss del av att det genomfördes av en person. Hade mer tid eller resurser funnits hade projektet kunnat utökas ytterligare.

**Nyckelord:**

Analys, Krypterade tjänster, Anonymitet, Granskning, Personlig integritet, Säkerhet, End-to-end-kryptering, Oövervakad kommunikation, Säkerhetsaspekter på chattapplikationer, Kommunikation på internet

# **ABSTRACT**

Questions about anonymity and privacy are more on the agenda than ever, especially in services that handle communication or information exchange in the world of data. Today, companies and private individuals have become increasingly dependent on online communication where security is of utmost importance. To better understand the difficulties and challenges faced by services and companies, this project focuses on analyzing and reviewing the difficulties these services bring. The end goal is to develop an encrypted chat service with a focus on anonymity, security and user privacy. In addition, the work has looked at the post-work required to ensure the privacy of the end user. The report also deals with the risks that these services entail regarding different types of attacks and risks that can be linked to different types of clients.

Something that is also touched on is the importance of anonymous services not being used to commit crimes.

The project is called SentinelNetGuard and consists of different parts. A website, a desktop application, server software and proxy server software.

The work resulted in a greater understanding of the difficulties and challenges that anonymity and encrypted chat services entail. This becomes especially clear when you as a client are completely or partially anonymous, which makes administration work more difficult.

The project has been carried out partly in a school environment, library and at home.

Hopefully, the report can contribute to a greater and more in-depth understanding of the challenges that anonymity in connection with the use of different types of chat applications entails.

The project was affected to some extent by the fact that it was carried out by one person. Had more time or resources been available, the project could have been expanded further.

**Keywords**: Analysis, Encrypted services, Anonymity, Auditing, Personal privacy, Security, End-to-end encryption, Unmonitored communication, Security aspects of chat applications, Internet communication

# **INNEHÅLLSFÖRTECKNING**

[**FÖRORD 1**](#_8wslvp2d9rnc)

[**SAMMANFATTNING 2**](#_iiltx28ljig3)

[**ABSTRACT 3**](#_7ayaljcj95ga)

[**INNEHÅLLSFÖRTECKNING 4**](#_wajy24wh6isp)

[**Beteckningar 1**](#_3fbplhze0ave)

[**1. INLEDNING 1**](#_t8nmptclcdli)

[1.1. Idébeskrivning 1](#_kue1lt3iwrwb)

[1.2. Syfte 2](#_54aeo26o197d)

[1.3. Frågeställningar 2](#_7g2el5zfip62)

[1.4. Avgränsning 2](#_22e0lzw8enhj)

[**2. BAKGRUND 3**](#_8uvp5ajtmps9)

[2.1. Teori 3](#_dc33i8a2xw5w)

[2.1.1. Existerande Applikationer 3](#_mbw48w7j49ik)

[2.1.2. Säkerhetsaspekter 3](#_tf7m9r2hxu13)

[2.2. Tidigare forskning 5](#_4luqzriaz0w5)

[**3. METOD 6**](#_iyftjmrrrcra)

[3.1. Introduktion 6](#_i64dvtynrwo)

[3.1.1. Analys av redan existerande projekt 6](#_7qw132dcbrmk)

[3.1.2. Jämförelse 6](#_pe5s1lnkhaxn)

[3.2. Stegvis process 6](#_fusz8n4k8v46)

[3.2.1. Pseudokod 6](#_on4wip22p8wl)

[3.2.2. Första prototyp 6](#_boigbyamz4wz)

[3.2.3. Testning av första prototyp 7](#_s5urx9odt7fj)

[3.2.4. Utvärdering av test 7](#_ixkla419whex)

[3.2.5. Design 7](#_coxxehx58nyo)

[3.2.6. Förbättringar 7](#_16k99qj2zcd1)

[3.2.7. Testgrupper och intervjuer 8](#_g5gscdv856yu)

[3.3. Utvecklingsprocessen 8](#_jq3zy32wkcuk)

[3.3.1. Programmeringsspråk som används 8](#_e36pxcevz8p)

[3.3.1.1 Generellt 8](#_y8cpnbji8myk)

[3.3.1.2 Frontend 8](#_3fhd6235s2hv)

[3.3.1.3 Backend 8](#_vzu7y3fv5gj2)

[3.3.2. Hemsidan & Webbservern 8](#_2zgdluxam565)

[3.3.2.1 Applikationsprogrammeringsgränssnitt (API) 9](#_761twac8xrmw)

[3.3.2.2 Registrering på hemsidan 9](#_gz5bt5ioejy)

[3.3.2.2 Pre-defined Whitelist som funktionarlitet på hemsidan 9](#_el8wuz3i0xyd)

[3.3.2.2 Cookies på hemsidan 9](#_hp6tnv3twe3t)

[3.3.3. SentinelNetGuard Noder 10](#_b3nutids5rr3)

[3.3.3.1 Noder 10](#_1gywl9l2n8a1)

[3.3.3.2 Noders Tidsbestämda livslängd 10](#_72n4ykcags3s)

[3.3.4. Skrivbords Applikationen 10](#_41uenxfor0y7)

[3.3.4.1 SentinelNetGuard Secure Connect 10](#_84c7wmsgs655)

[3.4. Utvärderingsprocessen 11](#_kunundxi7jll)

[3.4.1. Utvärderingsanalys 11](#_932olf2ffjgb)

[3.4.2. Vidare analys 11](#_cbx3kdum612i)

[3.4.3. Slutgiltiga tester 11](#_vufdew8nhmsu)

[3.4.4. Slutanalys 12](#_n7jdneqia1om)

[**4. RESULTAT 13**](#_awjq5xeijmmh)

[4.1 Resultat av anonym tjänst 13](#_k2ae4l5dtuzb)

[4.1.1 Problemen med anonymitet 13](#_okk6gjutw6k4)

[4.1.2 Fördelar med anonymitet 13](#_ga862k4eqb0s)

[**5. DISKUSSION 14**](#_6ouzhidanc8r)

[5.1. Metoddiskussion 14](#_4gpol191iplm)

[5.2. Arbetsprocessen 14](#_4gpol191iplm)

[5.3. Teknikutvecklingsprocessen 15](#_4gpol191iplm)

[5.3.1 Utvecklingsprocessen av hemsidan & webbservern 16](#_9i1sj7rs8jl3)

[5.3.2 Utvecklingsprocessen av serverprogramvaran (Node) 16](#_w2dy525tdzrs)

[5.3.3 Utvecklingsprocessen av skrivbordsklienten 18](#_hil4xlrv0nib)

[5.4. Resultatdiskussion 19](#_4gpol191iplm)

[**6. SLUTSATSER 20**](#_v3gzok9sbfv0)

[**7. REFERENSER 21**](#_z8z5le3e6r4r)

[7.1 SentinelNetGuard länkar 21](#_yqi6b8ppczv7)

[7.2 Material Serverprogramvara 21](#_gb7d8xse5wte)

[7.3 Webbkällor 21](#_gb7d8xse5wte)

# 

# **Beteckningar**

Många av de begrepp som används i rapporten är mer kända under dess förkortningar så det kan vara bra att känna till följande:

**IP** Internet Protocol

**SNG** SentinelNetGuard

**HTTP** Hyper Text Transfer Protocol

**HTTPS** Hyper Text Transfer Protocol Secure

**HTML** HyperText Markup Language

**GUI** Graphical User Interface

**CSS** Cascading Style Sheets

**SCSS** Sassy Cascading Style Sheets

**API** Application Programming Interface

**Tor** The Onion Router

**DNS** Domain Name System

**PHP** Hypertext Preprocessor, tidigare Personal Home Page Tools

**VPN** Virtual Private Network

**RDBMS** Relational Database Management System

**SQL** Structured Query Language

**DB** Database

**WebRTC** Web Real-Time Communication

# **1. INLEDNING**

## 1.1. Idébeskrivning

De senaste åren har det gått att läsa mycket i nyheterna om krypterade chattprogram som används i olagliga syften. Problemet är att dessa chattprogram har funktioner för att skydda användarna men samma funktioner kan även användas i illegala syften. Därför är målet med detta projekt att undersöka och försöka lägga fram en idé om hur ett program som behåller de delar som erbjuder säkerhet och anonymitet för sina användare, men samtidigt motarbetar användning i olagliga syften. För att få svar på det behövde rapporten undersöka om det är möjligt att utföra krypterad och anonym kommunikation, men fortfarande uppehålla lag och ordning.

Målet med projektet är att man enkelt ska ha möjlighet att skapa en temporär applikation på en server under en viss specificerad tidsperiod, exempelvis för ett företags gruppmöte där viktig information ska diskuteras och där man inte litar på dagens chattprogram eftersom man inte har inblick i hur programmet fungerar. Det kan också bero på att man inte själv äger servrarna som hanterar datan. Genom att själva ha kontroll över hur länge chattrummet existerar, vilka som får tillgång, och var servern i sig befinner sig, så ökar det både säkerheten och förtroendet för “ hosten ”. Eftersom chattprogrammet kommer att vara designat så att informationen och kommunikationskanalen tas bort vid en angiven tid, så ska praktiskt sett all information som växlats försvinna och inte gå att återskapa. Tanken är att programvaran ska köras i en begränsad del av servern där den temporärt skapas och där allt skapas i ram, vilket försvinner när servern startas om.

## 1.2. Syfte

Syftet med projektet och undersökningen är att undersöka om säker kommunikation kan ske men där administrering finns med och kan undersöka om oönskad användning pågår och i sådana fall förhindrar olovlig användning av programvaran. Målet är att privatpersoner, företag och organisationer ska kunna ha möten utan att behöva oroa sig för intrång. Att sedan konversationerna som skett inte sparas ner och kan återskapas från fragment eller hårddiskar som återskapas är en del i att öka säkerheten.

## 1.3. Frågeställningar

* Är det möjligt för kommunikationsprogram att existera utan att användas olagligt?
* Vilka är företags och organisationers främsta farhågor när det gäller informationsstöld och dataintrång?
* Finns det en universell möjlighet att publicera programvaran på Linux-servrar?

## 1.4. Avgränsning

Eftersom projektet nu genomförs av en person är det svårt att få alla delar av projektet slutförda inom tidsramen vilket skulle kunna resultera i att vissa funktioner, såsom exempelvis kryptering inte implementeras fullständigt. Dessutom är programvaran även konfigurerad för att enbart använda sig av en databas under utvecklingsfasen av projektet. Emellertid var tanken att i den slutgiltiga produkten så skulle vanliga användare kunna tillhandahålla databasen själva.

# 2. BAKGRUND

## 2.1. Teori

Krypterade chattapplikationer, även ibland kallat kommunikations-applikationer, är programvaror som används för att underlätta kommunikation via text, röst eller video. Dessa applikationer använder sig ofta av avancerade krypteringsmetoder för att säkerställa att den överförda datan endast kan läsas av dess avsedda mottagare. Detta innebär att kommunikationen är skyddad från obehörig åtkomst, vilket ofta uppnås genom så kallad “end-to-end encryption”. Miljontals människor använder sig dagligen av applikationer på sina datorer och i sina mobiltelefoner som nyttjar denna teknik. Krypteringen i sig består av en mycket komplex kombination av matematik och teori, där algoritmer används för att omvandla den ursprungliga datan till en oigenkännlig form. Denna form, som endast kan avkrypteras av den avsedda mottagaren gör att informationen förblir säker och skyddad från obehörig insyn.

### 2.1.1. Existerande Applikationer

Ett exempel på en chattapplikation är Signal, som erbjuder både kryptering och integritetsskydd för sina användare. En annan är Whatsapp som även den använder sig av kryptering.

### 2.1.2. Säkerhetsaspekter

Säkerhetsaspekterna inom krypterade chattapplikationer är av avgörande betydelse för att säkerställa användares integritet och för att skydda kommunikationen från obehörig åtkomst. Dessa aspekter omfattar ett flertal centrala pelare som måste övervägas noga för att hanteringen ska förbli tillförlitlig och säker.

Följande är några av de väsentliga delar som borde övervägas:

**Vilken typ av kryptering som används:**

Krypteringsmetoden är något som bör matcha dess användningsområde. Där kan man behöva anpassa krypteringsmetoden för gruppchattar till skillnad från enskild kommunikation mellan två parter. Något som är värt att notera är att det även är möjligt att använda den klassiska tvåparts “end-to-end encryption” med “Security codes” i grupprum med flera personer, men där kan det bli en större utmaning beroende på olika omständigheter. Ett exempel på hur det kan gå till är med så kallad “Hybrid Encryption”. Den kombinerar både asymmetrisk kryptering och symmetrisk kryptering vilket leder till både relativt snabb men även säker växling av nycklar.

**Vilka parter som är involverade:**

När det kommer till säker kommunikation, vare sig det gäller inom företag eller mellan privatpersoner som bara vill kommunicera privat, så är det viktigt att överväga vilka parter som kommer att hantera eller ta del av kommunikationen.

Det är viktigt att man kan kommunicera och med säkerhet veta att den som mottar meddelandet faktiskt är den avsedde mottagaren. Tjänster såsom Whatsapp tillhandahåller möjligheten att i verkligheten utväxla koder med varandra för att säkerställa att koderna matchar. Matchar koderna, kommer data som för överförs vara “end-to-end encrypted” med den avsedda mottagaren. Whatsapp kallar detta för “Verify Security Code”, men benämningen kan skilja beroende på applikation.

**Vem eller vilka tillhandahåller servrar och “säkra” kommunikationstunnlar?**

Något som även är viktigt att tänka på är vem eller vilka som tillhandahåller servrarna som datan kommer att passera genom. Hur säkerställer man att servrarna inte har bakdörrar eller att kommunikationen som påstås vara krypterad faktiskt är det?

**Anonymitet, och vad för data?**

När det kommer till krypterade chattjänster och applikationer är anonymitet väsentligt. Beroende på användare, användningsområde och syfte kan graden av anonymitet variera. Om man ser generellt på anonyma tjänster tenderar de att lova mycket anonymitet men där insamling av persondata fortfarande sker. Till exempel är personuppgifter såsom telefonnummer, adress, namn och ibland även personnummer bland de uppgifter som ofta efterfrågas vid registrering. Många skulle påstå att dessa tjänster inte är “helt anonyma” eftersom större delar av den insamlade informationen går att spåra tillbaka till en person eller identitet. Medan andra påstår att vikten i anonymiteten istället ligger i att kommunikationen i sig är anonym för andra än avsändare och mottagare.

Ett tydligt exempel på en bransch där många företag aktivt försöker marknadsföra sig som anonyma är VPN-branschen. VPN står för “Virtual Private Network” och används för att gömma IP-adressen på dess användare och därmed till viss del gömma dess identitet på internet. De protokoll som används i VPN-tjänster krypterar även nätverkstrafiken vilket bidrar ytterligare till säkerhet.

Men problemet uppstår när tjänsterna kräver att registrering sker med personuppgifter. Uppgifterna kopplas sedan till betalning eller konto. Ett relativt nytt sätt att hantera personuppgifter är sättet som Mullvad VPN hanterar personuppgifter och registrering. Där de inte kräver att man uppger några personuppgifter alls utan där de istället förser användaren med ett unikt kontonummer som representerar användaren och användarens månadsabonnemang (Mullvad VPN, 9 februari 2024). Detta är en metod som även SentinelNetGuard använder sig av.

## 2.2. Tidigare forskning

Forskning kring anonymitet och personlig integritet har tidigare gjorts och exempelvis Dahlberg (2023) beskriver några av de integritetsrisker som TOR-nätverket för med sig, trots att det är ett nätverk som är utformat just för anonymisering. Där vissa attackvektorer såsom Tor’s DNS Cache och Timing Attacks på Tor’s DNS är delar som nämns i Dahlbergs rapport. Bland annat tas det upp hur dessa attacker kan förebyggas genom både designprinciper i webbläsare och ett mer säkert användande av Tor.

Tor kan användas till exempel på tjänster eller projekt där grundläggande designprinciper kan spela stor roll för slutanvändarens säkerhet och personliga integritet, och därmed anonymitet. På samma sätt kräver det att tjänster som fokuserar på anonymitet och personlig integritet utvecklas med ett tankesätt där man redan från början har individens integritet i åtanke.

Tor är ett anonymiseringsnätverk som konstant behöver uppdateras för att matcha dagens internet-spelplan. Tor har öppen källkod vilket betyder att vem som helst kan antingen bygga vidare eller kika på hur nätverket är uppbyggt. Fördelen med det är att man kan få direkt inblick i hur nätverket fungerar och opererar i praktiken. En nackdel är dock att desto större kodbas, desto större yta för potentiella attacker.

Med koppling till projektets frågeställning om anonymiserande tjänster och användningsområden är Tor projektet (The Tor Project) ett tydligt exempel på där en anonymiserad tjänst kan komma att användas till olagligheter.

Då Tor anonymiserar nätverkstrafik och förflyttar informationen via så kallade “relays” är det utifrån just Tor nätverkets struktur det blir svårare att identifiera vem som skickade vad för information och från var.

# **3. METOD**

## 3.1. Introduktion

### 3.1.1. Analys av redan existerande projekt

Introduktionen av projektet påbörjades med insamling och analys av redan existerande projekt. Exempelvis hur olika tekniker och standarder används och är implementerade i redan existerande anonymiserande tjänster, såsom Tor . Även anpassningar kring hur SentinelNetGuard-projektets uppbyggnad kunde komma att se ut framöver utifrån design och funktionalitet med hänvisning till andra redan existerande projekt.

### 3.1.2. Jämförelse

Arbetet jämförs med redan existerande projekt och noteringar med hur skillnader och likheter såg ut i jämförelse med dem. Det blev sedan utvärderat och användes i senare steg. Projektplanen kan komma att förändras i och med nya upptäckter.

## 3.2. Stegvis process

### 3.2.1. Pseudokod

Pseudokoden var första delen i processen då det var en av de viktigaste byggstenarna för senare utveckling vidare i processen. Pseudokoden som skrevs lade grund för de kommande prototyperna i 3.1.2. då de möjliggjorde och agerade mall för prototypen. Pseudokoden skapade en grundläggande plan för hela projektets uppbyggnad, som senare kom att utvecklas vidare och förstärkas med tiden.

### 3.2.2. Första prototyp

Första prototypen var en av de viktigaste delarna i denna stegvisa process. Prototypen kom senare många gånger att förändras allteftersom projektet gick framåt och när andra delar tillkom. Att ha en stabil prototyp var viktigt då det förenklade det senare arbetet med *3.2.3. Testning av första prototyp* och *3.2.6. Förbättringarna* då det minimerade återkommande buggar och problem som uppstod. Prototypen baserades på den tidigare pseudokoden i steg *3.2.1.*och var endast en grundläggande prototyp utan fokus på saker som design och användarvänlighet.

Första prototyperna hade till exempel inte någon GUI för användaren alls utan testerna utfördes först och främst i terminalen med undantag för direkt utmatning på hemsidan.

### 3.2.3. Testning av första prototyp

Testning av första prototypen kunde ses som en testperiod under projektets utformning där problem, buggar men även förbättringar noterades. Felen åtgärdades inte alltid direkt utan fick ibland vänta tills *3.1.6.* beroende på dess utsträckning och vad för typ av problem det handlade om. Var problemet relaterat till användarvänligheten av tjänsten prioriterades det oftast ner till *3.2.6* sektionen, då testningen fokuserade på själva funktionaliteten, säkerhetsperspektivet och den anonyma aspekten. Buggar som då inte producerade större besvär men som i framtiden kan komma att uppenbara sig noterades eller fixades direkt.

### 3.2.4. Utvärdering av test

Utvärderingen av testerna i föregående steg (3.2.3) bestod av analys och kritik mot delar inom den skapade prototypen. Analysen täckte stora delar såsom design, funktioner i stort, säkerhet, men även mer direkta delar såsom om prototypen fungerade tillräckligt bra eller om designen av programvaran inte var tillräcklig. Testerna (3.2.3) var utformade på ett vis där en lista med problem och förbättringar uppdagades. Vilket i sig kunde underlätta utvärderingen då alla problem och förbättringar fanns på en och samma plats.

Utvärderingen av testerna fördes senare vidare till förbättringar (3.2.6) där vidare arbete med att fixa problemen och förbättringarna började. Utvärdering skedde senare ännu en gång efter att problemen och förbättringarna hade blivit implementerade. Därefter fortsatte arbetet i vanligt ordning.

### 3.2.5. Design

Designen skapades utifrån användarvänlighet och beroende på vad testerna och utvärderingen gav (3.2.4) så modifierades designen i enlighet med det.

Designen i projektets utveckling fokuserade dock inte endast på användarvänlighet och GUI utan även mer konkret på designen av koden. Hur kod kan skrivas på ett optimalt och bra sätt där funktioner kan återanvändas och förfinas. Det har varit viktigt att rundprinciper inom programmering följs och att säkerhetsaspekter alltid beaktas när känsliga delar av projektet utvecklas och hanteras, däribland databasanslutning, kontohantering och liknande. Att enhetstester utförs är även viktigt för att säkerställa att alla funktioner och delar av koden fungerar som de ska och att oförutsägbart beteende kan undvikas så långt som möjligt.

### 3.2.6. Förbättringar

Projektet utformades med fokus på förbättringar. Förbättringarna kunde bestå av olika delar beroende på var dessa uppstod i metoden. Efter att en förbättring noterats och rättats till utifrån de steg ovan, kunde processen repeteras igen från *3.2.1*.

### 3.2.7. Testgrupper och intervjuer

Under projektets gång intervjuades personer kring användarvänligheten, designen men även projektet i sin helhet. Intervjuerna skedde parallellt med de skarpa testerna av programvaran där bland annat extern kommunikation utanför ett lokalt nätverk testades. Det vill säga att en nod som är hostad inom ett nätverk är åtkomlig utanför det interna LAN-nätverket.

## 3.3. Utvecklingsprocessen

### 3.3.1. Programmeringsspråk som används

#### 3.3.1.1 Generellt

Programmeringsspråken som kom att användas i projektet var PHP, Python, HTML, CSS/SCSS, Javascript, Rust, C++, SQL men även script språk såsom Bash.

#### 3.3.1.2 Frontend

Främst HTML, CSS, Javascript, men även lite SCSS förekom i frontend koden.

#### 3.3.1.3 Backend

Backend bestod av främst PHP och Rust, men även delar i Python förekom. Databas Programvaran bestod av MariaDB som är en officiell “fork” av RDBMS(MySQL relational database management system).

Serverns operativsystem som hostade prototypen i arbetet bestod av Ubuntu Server. Senare eftersom programvaran i projektet var menat att installeras snabbt universellt på alla Linux-baserade system, varierade programvaran beroende på vilken installation som var närvarande för tillfället. Tanken var även att serverinstallation varan skulle vara kompatibel med Raspberry Pi, och därmed vara så resurssnål som möjligt.

### 3.3.2. Hemsidan & Webbservern

Webbserver Programvaran som användes för hemsidan var Apache2 och backend koden som skrevs var majoriteten i PHP. Sedan användes HTML, CSS/SCSS och Javascript för att skapa utseendet och frontend funktionaliteten av hemsidan. PHP backend koden bestod till stor del av funktioner och kod som hanterade användarens konto och hanteringen av funktionaliteter såsom inlogg, hantering av “nodes”, och listor för “pre-defined whitelist”. Webbservern har även i projektet en uppgift att ha en API som även den var kodad i PHP. Där API:n bland annat hanterar returnering av data till skrivbords-applikationen SentinelNetGuard Secure Connect.

#### 3.3.2.1 Applikationsprogrammeringsgränssnitt (API)

API:n är uppbyggd så att den är beroende av GET-parametrar. Varje GET-parameter måste matcha server-sidans PHP-kod. Stora delar av API-koden är uppbyggd utifrån hur känslig eller viktig den data som efterfrågas är och det krävs alltid någon form av ytterligare data för att få den efterfrågade informationen. Om den inte matchar returneras ingenting, och förfrågan avslutas eller omdirigeras till den vanliga hemsidan. Beroende på var den ursprungliga förfrågan kom från hanteras svaret olika. Ett exempel på när detta sker är när aktuell API ska returnera kontodata där bland annat en nyckel som hanterar uppstart av noder finns. Nyckeln i sig används för att möjliggöra en relation mellan ett konto och en chatserver. Det klassas då som känslig data eftersom det i teorin skulle möjliggöra för tredje part att skapa noder under annans konto. För att API:n ska returnera denna nyckel krävs det att GET-förfrågan innehåller en parameter med en hemlighet som endast kontot själv skulle vetat om. Det resulterar i ett robust system som endast returnerar hemlig eller känslig information om den även får något i gengäld. På så vis säkerställs det att rätt uppgifter via API:n endast hamnar hos behöriga konton.

#### 3.3.2.2 Registrering på hemsidan

Registreringen av personkonto i projektet sker genom att man genererar ett konto. Efter att det skett ges ett unikt kontonummer som är bundet till just det kontot. Kontonumret kommer senare att användas för att skapa, logga in och för att chatta med andra klienter i noder eller chattrum. Notera att inga personuppgifter krävs vid registrering eller generering av konto. Kontonumret är den enda bindande datan som kan kopplas till kontot vilket ökar anonymiteten.

#### 3.3.2.2 Pre-defined Whitelist som funktionarlitet på hemsidan

Med “Pre-defined Whitelist” menas en funktionalitet i SentinelNetGuard där man redan innan man skapat ett chattrum kan fördefiniera konton som ska vara så kallade "White Listade" eller tillåtna att ansluta. Genom att skapa dessa listor kan tid sparas när väl skapandet av chattrummen sker. Whitelist i SNG hanteras generellt i webbservern och via hemsidan men kan även modifieras via API:n för desktop-applikationen.

#### 3.3.2.2 Cookies på hemsidan

För att underlätta inloggning i webbläsaren på hemsidan finns cookies tillgängliga. SentinelNetGuard använder sig av en så kallad session cookie som är en temporär “kaka” med textdata som sparas på användarens dator. Denna cookie består av en slumpmässig textbit som är sparad i databasen och byts ut var 30e dag om ingen inloggning har skett under den tidsperioden. Det finns även möjlighet för användaren att rensa och återställa sin session cookie vid behov eller om det finns risk att kontots session cookie har blivit stulen.

### 3.3.3. SentinelNetGuard Noder

#### 3.3.3.1 Noder

Noder i SentinelNetGuard är ett samlingsnamn för de temporära chattrum som en stor del av projektet bygger på. Koden som noderna körs i är Python vilket gör att den kan köras på i princip vilken dator eller kapabel enhet som helst med internetanslutning. På hemsidan efter inloggning finns ett färdigställt kommando för att installera noden på en molntjänst eller på en Linux-baserad server, men det går även på Windows utan det färdigställda kommandot. Koden för noderna tillhandahåller ett chattrum med “sockets” som hanterar kommunikationen mellan olika klienter. Efter uppstart ges administratören tillgång till en terminal där olika kommandon kan ges för olika adminverktyg.

#### 3.3.3.2 Noders Tidsbestämda livslängd

Varje nod har en tidsbestämd livslängd vilket bidrar till anonymiteten och säkerheten i chattrummet. Standardtiden på livslängden är 60 minuter men skulle administratören för noden vilja förlänga eller minska ner livslängden så är det möjligt.

### 3.3.4. Skrivbords Applikationen

#### 3.3.4.1 SentinelNetGuard Secure Connect

Namnet på chattapplikationen är SentinelNetGuard Secure Connect. Programmet kan installeras på både Linux-baserade system och på Windows-system. Applikationen är uppbyggd med hjälp av Tauri vilket är ett ramverk för skrivbordsapplikationer. Tauri använder sig av HTML, CSS och Javascript för frontend och Rust som backend. Frontend är det samlingsnamn för det som slutanvändaren ser och kan interagera med, medan backend består av mer grundliga icke synliga komponenter och processer.

Applikationen kräver endast ett kontonummer för inloggning där det sedan finns möjlighet till anslutning till en node eller ett chattrum. Skrivbordsapplikationen använder sig av API:n för att kommunicera med databasen och för inloggning.

När användaren anslutit till en node eller ett chattrum får denne tillgång till ett urval av kommandon och ett välkomstmeddelande med mer information om noden eller chattrummet.

## 3.4. Utvärderingsprocessen

### 3.4.1. Utvärderingsanalys

Projektet genomfördes utifrån planeringen och nådde de mål som var satta. Utöver projektmålen tillkom även möjliga förbättringar, exempelvis design-förbättringar som nämndes tidigare i rapporten. Vidare förbättringar riktade in sig mer på funktionaliteten av tjänsten där optimering var en stor del. En stor del av projektet bygger på att flera användare interagerar med varandra i så kallade noder eller chattrum. Beroende på vem eller vilka som använder sig av tjänsten så behöver noderna eller chattrummen dynamiskt anpassa serverns resurser. Om resurserna inte dynamiskt anpassar sig efter användarbehov riskerar noden eller chattrummet att få ökad latens eller i värsta fall att krascha.

### 3.4.2. Vidare analys

Med större resurser och mer tid till förfogande hade varje enskild del av projektet kunnat utvecklas ytterligare.

Om projektet hade påbörjats idag så skulle undersökningar kring vad för färgschema som skulle valts funnits med i projektplanen vilket hade underlättat och sparat onödigt arbete längre in.

Eftersom de tidiga prototyperna inte var speciellt representativa avseende slutprodukten borde mer tid lagts ner på de tidiga prototyperna vilket troligtvis hade sparat tid i de senare stegen då en tydligare och mer representativ struktur då funnits på plats tidigare.

### 3.4.3. Slutgiltiga tester

De slutgiltiga testerna utfördes i slutskedet av projektet. Testerna bestod mest av enskilda enhetstester av sektioner och funktioner i kod som tidigare inte hade testats. Vidare bestod testerna även av slutgiltiga prövningar kring hemsidans funktionalitet, utseende och hur designprinciperna som använts fungerar i praktiken när en användare interagerar med dem.

Testerna som utfördes på serverprogramvaran bestod av delar där programvaran till exempel testades för oförutsägbara mängder användare. Även meddelandehantering av användare som använder andra karaktärer i sina meddelanden testades. Några av de som specifikt testades var kyrilliska alfabetet, det traditionella kinesiska alfabetet, japanska tecken och det grekiska alfabetet.

Projektet visade sig klara av alla de stresstester med användare som projektet utsattes för. Tjänsten var välfungerande och följde alla de principer och designval som skapades under planeringsfasen. De problem och förbättringar som påpekades av testpersoner fixades och det bekräftade testerna.

### 3.4.4. Slutanalys

Synen på projektet som helhet är positiv. Testerna och analyserna kring arbetet där både rapporten och tjänstens framtagning inkluderas har gått bra. Projektet följde projektplanen som fanns och det resulterade i en färdig produkt och tjänst där nästan alla de delar som var planerade att finnas med gjorde det. Tjänsten som går under namnet SentinelNetGuard är en fullt fungerande krypterad anonym chattjänst där noder både kan skapas och tas bort, antingen manuellt eller automatiserat efter att dess livslängd har gått ut..

Det som blev tydligt och som framkommit är svårigheterna kring hur det är att ha en tjänst med fokus på anonymitet och integritet, men som fortfarande ger administratörer verktyg för att motverka felaktig användning.

På frågeställningen om det är möjligt att publicera serverprogramvaran universellt på Linux-baserade system så har detta lyckats mer än väl. Serverprogramvaran kan köras på både Windows och Linux-baserade system. Även skrivbordsapplikationen kan installeras och användas på både Windows och Linux-system.

# **4. RESULTAT**

## 4.1 Resultat av anonym tjänst

### 4.1.1 Problemen med anonymitet

Efter utvecklingen av tjänsten och projektet SentinelNetGuard framkom det att det inte är enkelt att ta fram en anonym tjänst med fokus på integritet som samtidigt har bra och starka administrationsverktyg. Då anonymitet, hel eller delvis, eliminerar många av de möjligheter en administratör skulle kunna verka utifrån. Som följd av detta underlättade det för vissa aktörer att undgå administratörens åtgärder genom att byta identitet, eller helt enkelt ignorera åtgärderna som administratörer utlyste. Detta eftersom tjänsten inte har någon som helst koppling till person eller identitet i verkligheten, då denne förblir anonym.

Ett ytterligare problem som kommer med anonymitet är att människor generellt inte tar lika stort ansvar.

### 4.1.2 Fördelar med anonymitet

Det finns många fördelar med anonymitet. Ett exempel på det är att människor som normalt sett inte vågar interagera och föra fram sina åsikter eller delta i en öppen diskussion under anonymitet kan känna en trygghet i att delta. Ytterligare bidrar anonymitet till att diskussioner kring frågor kan utvecklas vidare och ta helt nya former. Heta frågor där personer i vanliga fall skulle kunna känna sig antingen utsatta eller hotade blir med anonymitet ett mindre problem där fokuset istället kan ligga på information och diskussion. Yttrandefrihet och frispråkighet är centrala delar där anonymitet spelar en viktig roll. Frågor där kritik vanligtvis skulle blossa upp kan nu istället formas och utvecklas i en anonym form. Personer med utsatt position kan lättare dela med sig av tankar och åsikter så länge det inte bryter mot lagen. Det kan ske på ett mer öppet sätt där de i vanliga fall inte skulle kunnat interagera på grund av exempelvis deras politiska, geografiska, eller etniska bakgrund.

Internet har skapat en plattform för bland annat kritik och ifrågasättande kring samhällsfrågor som för samhället framåt. I anonyma kontexter kan det då vara lättare att dela med sig av kritik och frågor som berör känsliga områden utan att behöva känna skuld eller rädsla för påhopp eller hot som följd.

Integritet är en viktig punkt som nämns alltmer i media. Personlig integritet är den enskildes rätt att kontrollera vem som kan få tillgång till dennes privata information (Lagen.nu, u. å). Anonymitet bidrar till ökad integritet för individen. Speciellt när det kommer till dennes kommunikation på nätet.

# **5. DISKUSSION**

## 5.1. Metoddiskussion

Utifrån metoden och de givna förutsättningarna var metoden som användes under utvecklingen välfungerande. Då tidsramen och resurserna var begränsade skapade det en form av stress. Stressen i sig gjorde att delar inom metoden prioriterades framför andra.

Insamlingen av data innan arbetet utfördes på ett bra och välfungerande sätt.

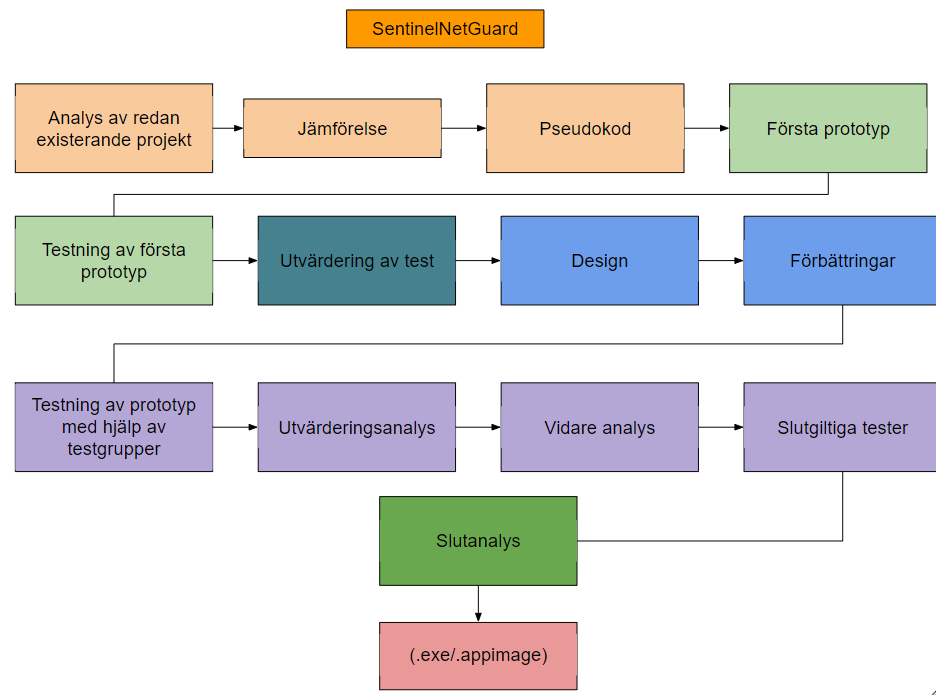
För att kunna utvärdera chattjänster och rapportens frågeställningar så utvecklades en tjänst. Tjänsten som bestod av hemsida, servermjukvara och skrivbordsapplikation arbetade tillsammans för att skapa en välfungerande och anonym chattjänst. Utifrån tjänsten kunde frågeställningarna lättare utvärderas eftersom alla aspekter som undersöktes fanns tillgängliga. Med hjälp av testpersoner kunde scenarion utspela sig i en miljö som speglar de tjänster som i dag finns offentligt tillgängliga på marknaden.

## 5.2. Arbetsprocessen

Arbetsprocessen kring projektet har i helhet varit välfungerande. Majoriteten av arbetet i stort har följt planeringen till punkt och pricka, medan andra delar har tillkommit under arbetets gång. Utvecklingen påbörjades med en grov skiss där grundstommen av projektet skapades först. Sedan tillkom allt fler funktioner och mer funktionalitet implementerades.

Tidsplanen var väl uttänkt och i avgränsningen nämns de problemen som potentiellt kunde uppstå med en begränsad arbetskraft på ett projekt som detta under den satta tidsplanen.

## 5.3. Teknikutvecklingsprocessen

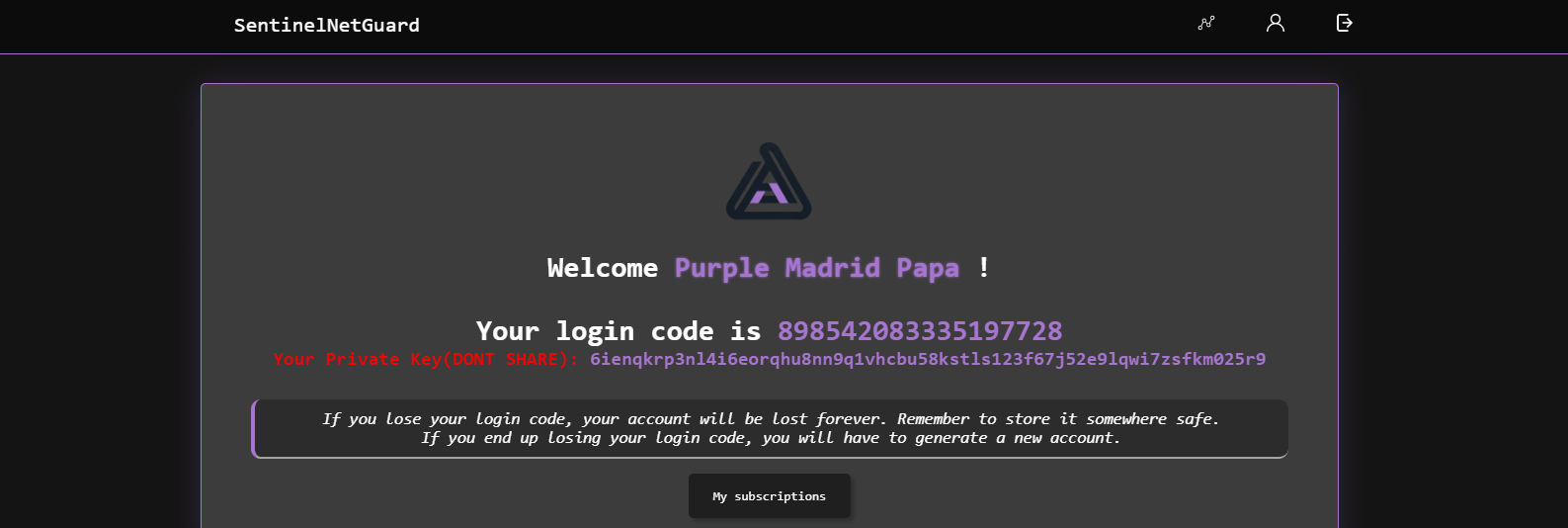


*Figur 5.3.6 (“Agile workmodel” över hela projektet)*

### 5.3.1 Utvecklingsprocessen av hemsidan & webbservern

Utvecklingen av hemsidan och webbservern utfördes enligt planeringen där skisser först skapades. Därefter började planeringen kring funktioner och hur tjänsten i stort skulle utformas. Olika funktionaliteter påbörjades och mycket pseudokod skrevs för att skapa en tydlig och klar struktur i projektet. Eftersom att webbservern även ansvarar för projektets API så arbetades det parallellt med funktioner och öppningar som skrivbords-applikationen skulle interagera med senare i processen. Utvecklingen fortsatte och det skapades en prototyp som senare byggdes på efter feedback från testpersoner och utvärderingar kring hur enkel hemsidan var att interagera med. Även färgschemat förändrades med tiden då ett flertal personer klagade och föreslog ändringar.

Ett problem som uppstod lite senare i utvecklingsprocessen var att webbservern blev tvungen att flyttas från en Raspberry PI 3 till en mer kapabel server med bättre hårdvara då det uppstod problem med latens och processorkraft vid databas kontakt via projektets API. Efter migreringen löstes problemet med latensen.

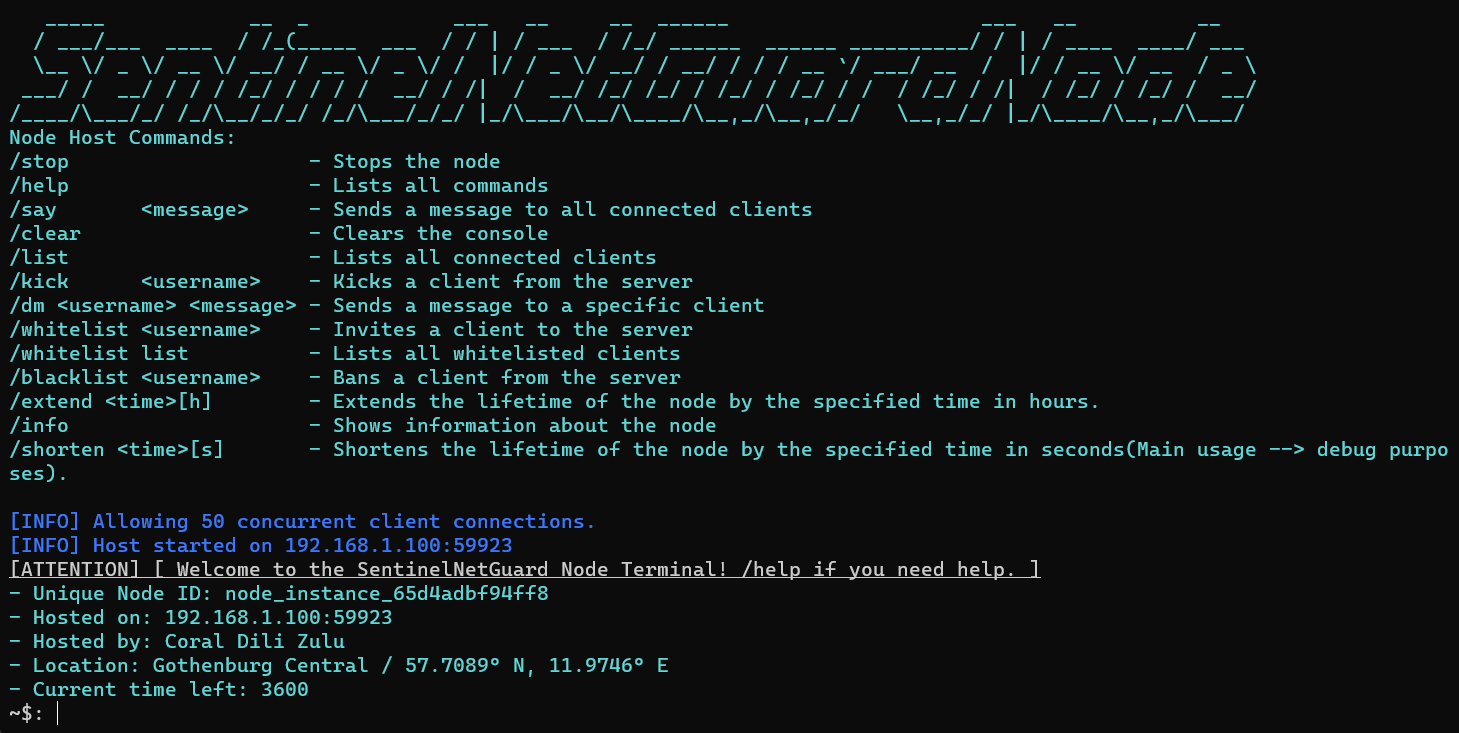


*Figur 5.3.1.1 (Figuren visar hemsidan efter inloggning och generation av konto, här visas användarens unika kontonummer, privata nyckel och användarnamn)*

### 5.3.2 Utvecklingsprocessen av serverprogramvaran (Node)

Serverprogramvaran som i projektet ansvarar för de så kallade noderna, vilket är ett annat namn för chattrummen i tjänsten, var en central del som krävde att strukturen ända från början av utvecklingen var robust. Programvaran interagerar med API:n genom förfrågningar. Där hämtar och lämnar den information om chattrummen till projektets databas. All anslutning till databasen skedde via API-förfrågningar vilket minimerade ytan för potentiella attacker då webbservern är den enda som har databasanslutning.

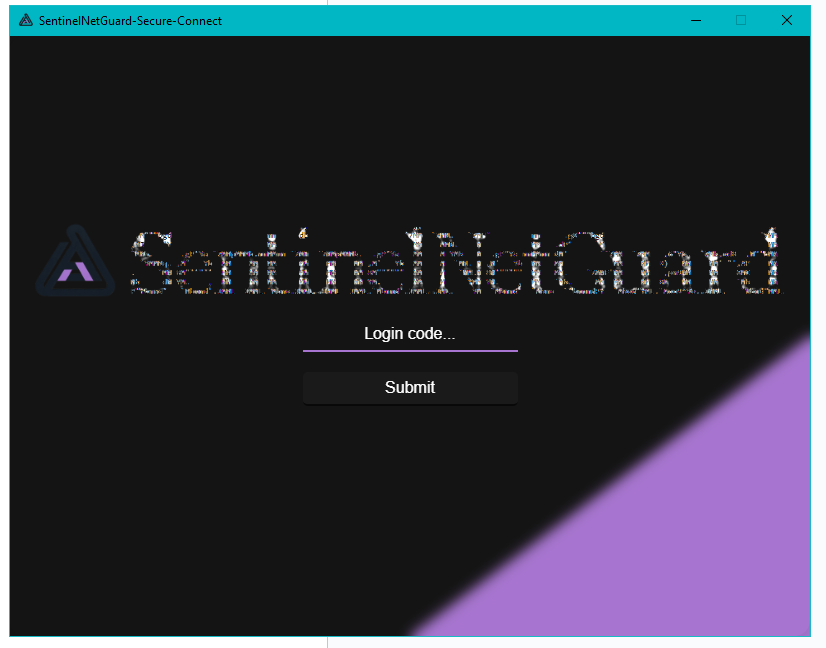
Ett problem som uppstod under skapandet av serverprogramvaran var hanteringen av klientanslutningar. Då noden behöver hålla koll på aktuella anslutningar av klienter så kräver det att nodens “socket” behåller anslutningen med alla klienternas enskilda anslutningar. Problem uppstod med latens och stunder av väntan då programvaran inte utnyttjade flertalet trådar, även kallat “multi-threading” för processen. Trådar eller threads delar upp arbetet på servern och gör det lättare för servern att hantera flera processer samtidigt. Dessutom användes så kallade “non blocking threads”. Vilket innebär att trådar fortsätter utföra sina uppgifter utan att vänta på att andra trådar ska slutföra sina operationer. Detta skiljer sig från den mer traditionella “blocking multi-threading”, där en tråd kan behöva vänta på resultatet av en annan tråd innan den kan fortsätta med sina egna operationer. Genom att använda flera trådar i noderna ökade det effektiviteten men även responsiviteten av programmet eftersom klienter kunde hanteras individuellt, och samtidigt.



*Figur 5.3.2.1 (Figuren visar skrivbordsapplikationen på Windows där ett kontonummer efterfrågas för inloggning)*

### 5.3.3 Utvecklingsprocessen av skrivbordsklienten

Processen kring utvecklingen av skrivbordsklienten skilde sig från utvecklingen av det övriga projektet. Då skrivbordsklienten använde sig av ett relativt nytt ramverk vid namn Tauri som möjliggör enkelt skapande av skrivbordsapplikationer med hjälp av HTML, CSS för frontend och Rust för backend (Tauri, u. å). En del problem uppstod när Rust skulle kommunicera med API:n. Rust lyckades inte kontakta webbservern och fick inte den returnerade datan som förväntades vid förfrågan. Detta löstes genom att byta modul i Rust till “http\_request”. Http\_request-modulen inkluderar den funktionaliteten som möjliggör för GET-förfrågningar.



*Figur 5.3.3.1 (Figuren visar skrivbordsapplikationen på Windows där ett kontonummer efterfrågas för inloggning)*

## 5.4. Resultatdiskussion

Målet med den del av projektet som täckte utvecklingen av tjänsten var att skapa en välfungerande chattjänst som både förblev anonym men även möjliggöra för den att vara kompatibel med majoriteten av Linux-baserade servrar. Det löstes genom att serverprogramvaran för noderna kördes i python-kod vilket stöds av nästan alla servrarna idag(Python Software Foundation, 2024). Resultatet var en välfungerande och smidig programvara som snabbt och smidigt möjliggjorde för både molnbaserad och hemma ägd uppstart av en chattserver.

Tjänsten som även inkluderade en skrivbordsapplikation fungerade även den som den skulle. Applikationen fungerade både internt på ett LAN-nätverk och externt med datorer och chattrum utanför det interna nätverket. Skrivbords-applikationen kunde installeras på Windows med hjälp av “.msi” och “.exe” installations-filer. Där Linux-baserade enheter istället använde sig av “.appimage” och “.deb”. Genom att göra applikationen tillgänglig på både Windows och Linux så bidrar det till att fler kan få tillgång och möjlighet att installera applikationen.

Med koppling till anonymitet och integritet påstår många att Linux är bättre säkerhetsmässigt än Windows där Linux har ett flertal fördelar när det kommer till makt över programvara och hårdvara. Linux-baserade system har bland annat ett inbyggt system där konton och rutiner automatiskt tilldelas låga behörigheter (Kaspersky, 2023).

# **6. SLUTSATSER**

Projektet och rapporten har hela tiden utgått från tre frågeställningar. Dessa frågeställningar har varit återkommande i både utvecklingen av tjänsten och programvara, men även av analyser kring tidigare arbeten, tjänster och tidigare forskning som är gjord inom området.

En slutsats som kan dras är att det finns en mängd svårigheter med att tillhandahålla och utveckla en anonym chattjänst som respekterar användarens integritet. Där många av de verktyg som vanligtvis kan användas för att moderera sådana tjänster tas bort eftersom att användare inte kan kopplas eller hållas ansvariga för deras beteenden.

Vidare påverkas individers, företags och tjänsters hotbild när det kommer till vilken typ av klienter som använder sig av tjänsten. Där exempelvis Tor-nätverket inte per automatik är en tjänst men mer en öppen programvara för vem som helst att ta del av, både används till bra delar såsom journalistik, men även andra delar som droghandel och olaglig aktivitet.

Möjligheten kring publicering av programvara på olika systemplattformar såsom Windows och Linux har under projektets gång testats. Slutsatsen som kan dras är att programvara var välfungerande på både Windows och på Linux-baserade system vilket bidrar till att programvaran kan användas i större utsträckning.

Då den här chatttjänsten i nuläget inte finns tillgänglig offentligt kunde många av de risker, problem och dilemman som skulle uppstått testas. Exempel på det är när kommunikation i anonyma noder mellan personer utvecklades, men då ingen utomstående kan ha insikt i vad som sker. Då blir det väldigt svårt för företaget eller tjänstens ägare att avgöra om det är laglig eller olaglig aktivitet som pågår i chattkonversationen. Detta är bara ett av de problem som anonyma chattrum för med sig. Ytterligare problem uppstår om det på något sätt skulle uppmärksammas att olaglig aktivitet pågick i ett chattrum, men där ingen av användarna kan identifieras eftersom att de inte kan kopplas till en identitet. Om man drar det steget längre kan användare i teorin även lägga till Tor routing som nämndes tidigare i rapporten på sin anslutning under skapandet av kontot eller anslutning till en chattnod. Detta försvårar arbetet ytterligare med att hitta dennes riktiga identitet om ingen annan data fångats upp. Såklart är detta endast en förenkling av hur det hela går till. För att få perspektiv på problemet är det inte bara att använda sig av Tor routing för att bli anonym, då många andra delar måste vägas in såsom så kallad fingerprinting av webbläsare (*eng. Fingerprinting*), vilket operativsystem som används, vad för typ av webbläsare, spårning av cookies, läcka via WebRTC och liknande.

Några lärdomar som kan medföras av arbetet är den stora komplexiteten av att utveckla en tjänst med fokus på anonymitet och utifrån användarens önskan om personlig integritet. Där användarens önskemål om anonymitet i många fall kan hamna i konflikt med tjänstens ägare eller administratörers förmåga att moderera. Det har även varit en utmaning med att utveckla ett så pass omfattande projekt där många delar måste samverka för att helheten ska fungera.

# **7. REFERENSER**

## 7.1 SentinelNetGuard länkar

**SentinelNetGuard. (u. å) *SentinelNetGuard.* Från** [**https://github.com/SentinelNetGuard/SentinelNetGuard**](https://github.com/SentinelNetGuard/SentinelNetGuard)

**SentinelNetGuard. (u. å) *SentinelNetGuard\_Node\_Setup.* Från**[**https://github.com/SentinelNetGuard/SentinelNetGuard\_Node\_Setup**](https://github.com/SentinelNetGuard/SentinelNetGuard_Node_Setup)

**SentinelNetGuard. (u. å) *SentinelNetGuard Secure Connect.* Från**[**https://github.com/wilarN/SentinelNetGuard/releases/**](https://github.com/wilarN/SentinelNetGuard/releases/)

**SentinelNetGuard. (u. å) *SentinelNetGuard Wiki.* Från**[**https://github.com/SentinelNetGuard/SentinelNetGuard/wiki**](https://github.com/SentinelNetGuard/SentinelNetGuard/wiki)

## 7.2 Material Serverprogramvara

**Redhat. (u. å) *Red Hat Enterprise Linux.* Från** [**https://www.redhat.com/en/technologies/linux-platforms/enterprise-linux**](https://www.redhat.com/en/technologies/linux-platforms/enterprise-linux)

**Ubuntu. (u. å) *Scale out with Ubuntu Server.* Från** [**https://ubuntu.com/server**](https://ubuntu.com/server)

## 7.3 Webbkällor

**Dahlberg, R. (2023). *On Certificate Transparency Verification and Unlinkability of Websites Visited by Tor Users*. [Doktorsavhandling, Karlstad Universitet].**

**Kaspersky. (2023, Augusti 16). *What is Linux and is it really secure?*. Från** [**https://www.kaspersky.com/resource-center/definitions/linux**](https://www.kaspersky.com/resource-center/definitions/linux)

**Lagen.nu. (u. å). *Personlig integritet.* Från** [**https://lagen.nu/begrepp/Personlig\_integritet**](https://lagen.nu/begrepp/Personlig_integritet)

**Mullvad VPN. (u.å). *Why privacy matters*. Från** [**https://mullvad.net/en/why-privacy-matters**](https://mullvad.net/en/why-privacy-matters)

**Mullvad VPN. (9 februari 2024) *No-logging of user activity policy.* Från** [**https://mullvad.net/en/help/no-logging-data-policy#numbered**](https://mullvad.net/en/help/no-logging-data-policy#numbered)

**Python Software Foundation. (2024, Februari 20). *Using Python on Unix platforms.* Från** [**https://docs.python.org/3/using/unix.html**](https://docs.python.org/3/using/unix.html)

**Tauri. (u. å). *What is Tauri?*. Från** [**https://tauri.app/about/intro**](https://tauri.app/about/intro)

**The Tor Project. (u. å). *Browse Privately. Explore Freely*. Från** [**https://www.torproject.org/**](https://www.torproject.org/)