CHALMERS
UNIVERSITY OF TECHNOLOGY

UNIVERSITY OF GOTHENBURG

Krypterad men spårbar: Hur WiFi-trafik kan avslöja användarbeteenden

Kandidatarbete på institutionen för data- och informationsteknik

Amos Alkebro
Abd Rahman Alkhatib
Helena Nguyen
Marcella Toft
Frida Wallström
William Åkerström

Institutionen för data- och informationsteknik
CHALMERS TEKNISKA HÖGSKOLA
GÖTEBORGS UNIVERSITET
Göteborg, Sverige 2026

Kandidatarbete 2026

Krypterad men spårbar: Hur WiFi-trafik kan avslöja användarbeteenden

Amos Alkebro
Abd Rahman Alkhatib
Helena Nguyen
Marcella Toft
Frida Wallström
William Åkerström

UNIVERSITY OF
GOTHENBURG

CHALMERS
UNIVERSITY OF TECHNOLOGY

Institutionen för data- och informationsteknik
Chalmers Tekniska Högskola
Göteborgs Universitet
Göteborg, Sverige 2026

Krypterad men spårbar: Hur WiFi-trafik kan avslöja användarbeteenden
Amos Alkebro Abd Rahman Alkhatib Helena Nguyen Marcella Toft Frida Wallström William Åkerström

Handledare: Romaric Duvignau, Institutionen för data- och informationsteknik
Examinatorer: Arne Linde och Patrik Jansson, Institutionen för data- och informationsteknik
Rättande lärare: Magnus Almgren, Institutionen för data- och informationsteknik

Kandidatarbete 2026
Institutionen för data- och informationsteknik
Chalmers Tekniska Högskola och Göteborgs Universitet
SE-412 96 Göteborg
Telefon +46 31 772 1000

Skriven i LaTeX
Göteborg, Sverige 2026

Abstract

This study investigates what information about devices and user behavior can be extracted through passive analysis of wireless network traffic. By developing a prototype system for data collection and traffic analysis, the feasibility of identifying device type, operating system, currently active application, user activities, and behavioral patterns over time is examined. The results indicate that it is possible, to some extent, to identify device type and operating system based on observable traffic patterns. However, the identification of user activities and specific applications remains uncertain and is affected by traffic variability and limited access to representative datasets. Furthermore, general behavioral patterns, such as recurring periods of activity, can be inferred, albeit with limited accuracy. Several factors influencing the reliability of the analysis are identified, including variations in signal strength and limitations inherent in passive traffic collection. These factors complicate the consistent association of observed traffic with individual devices over time. Overall, the study demonstrates that network traffic analysis can provide insights into device characteristics and user behavior, even without access to the content of communication. At the same time, the accuracy is limited, and the results should be interpreted with caution, particularly when generalizing to more complex environments. The study thus highlights both the possibilities and limitations of this type of analysis, as well as the associated privacy implications.

Keywords: WiFi, Network security, encryption, fingerprinting, metadata, passive analysis, network traffic, user behavior, machine learning, privacy

Sammandrag

Denna studie undersöker vilken information om enheter och användarbeteenden som kan utvinnas genom passiv analys av trådlös nätverkstrafik. Genom att utveckla ett prototypsystem för insamling och analys av trafikdata analyseras möjligheten att identifiera enhetstyp, operativsystem, använd applikation, användaraktiviteter och beteendemönster över tid. Resultaten visar att det i viss utsträckning är möjligt att identifiera enhetstyp och operativsystem baserat på observerbara trafikmönster. Däremot är identifiering av användaraktiviteter och specifika applikationer mer osäker och påverkas av variationer i trafik samt begränsad tillgång till representativa datamängder. Vidare kan övergripande beteendemönster, som återkommande aktivitetsperioder, urskiljas, men med begränsad noggrannhet. Studien identifierar flera faktorer som påverkar analysens tillförlitlighet, däribland variationer i signalstyrka och begränsningar i passiv trafikinsamling. Dessa faktorer försvårar en entydig koppling mellan observerad trafik och enskilda enheter över tid. Sammantaget visar arbetet att analys av nätverkstrafik kan ge viss insikt i enheters egenskaper och användarbeteenden, även utan åtkomst till kommunikationens innehåll. Däremot är noggrannheten begränsad, och resultaten bör tolkas med försiktighet, särskilt vid generalisering till mer komplexa miljöer. Studien belyser därmed både möjligheter och begränsningar med denna typ av analys, samt de integritetsimplikationer som kan uppstå.

Nyckelord: WiFi, nätverkssäkerhet, kryptering, identifiering, metadata, passiv analys, nätverkstrafik, användarbeteenden, maskininlärning, integritet

Tackord

Vi vill rikta ett varmt tack till vår handledare Romaric Duvignau för vägledningen och den konstruktiva återkopplingen vi fått under arbetets gång. Hans engagemang har varit till stor hjälp genom hela projektet. Dessutom vill vi tacka Joachim Toft och Erik Wallström för deras stöd och insiktsfulla kommentarer som har hjälpt till med att höja arbetets kvalitet.

Amos Alkebro, Abd Rahman Alkhatib, Helena Nguyen,
Marcella Toft, Frida Wallström, William Åkerström,
Göteborg, juni 2026

Lista över akronymer

Nedan är en lista över akronymer som används i denna rapport, tillsammans med deras betydelse och en kort förklaring.

Akronym	**Fullständig benämning**
CSV	Comma-Separated Values
dBm	Decibel-milliwatt
GDPR	General Data Protection Regulation
HTML	Hypertext Markup Language
HTTP	Hypertext Transfer Protocol
IEEE	Institute of Electrical and Electronics Engineers
IP	Internet Protocol
JSON	JavaScript Object Notation
LEK	Lag om Elektronisk Kommunikation
MAC	Media Access Control
OS	Operating System
OUI	Organizationally Unique Identifiers
PCAP (filformat)	Packet Capture (file format)

Innehåll

Figurer

Tabeller

1
Inledning

Att bli observerad i den fysiska världen väcker ofta en omedelbar reaktion. En person som följer efter någon på gatan eller övervakar var någon befinner sig uppfattas snabbt som ett intrång i den personliga integriteten. I den digitala världen sker liknande former av övervakning, men betydligt mer diskret. Enheter uppkopplade till ett nätverk lämnar kontinuerligt efter sig spår, ofta utan att användarna själva är medvetna om det.

Trådlösa nätverk utgör idag en central del av den digitala infrastrukturen, där ett stort antal enheter på universitet, arbetsplatser och i offentliga miljöer dagligen ansluter sig. Varje anslutning genererar spår i form av exempelvis tidsmönster och trafikfrekvens. Genom analys av dessa spår kan det vara möjligt att identifiera enheter och dra slutsatser om användarbeteenden, även när innehållet i kommunikationen inte kan avläsas.

Många moderna operativsystem har samtidigt infört integritetsskyddande mekanismer som syftar till att försvåra identifiering och spårning av enskilda enheter. Tidigare forskning har dock visat att det fortfarande är möjligt att utvinna detaljerad information om användare och enheter genom analys av observerad nätverkstrafik [1]. Denna typ av analys kan genomföras passivt, vilket innebär att nätverkstrafiken observeras utan att kommunikationen påverkas. Eftersom analysen inte förändrar trafiken lämnas inga tydliga spår, vilket gör den svår att upptäcka och därmed möjlig att genomföra utan den observerades kännedom.

Passiv analys kan dessutom i många fall genomföras med relativt enkla medel. En vanlig dator med stöd för avlyssning av nätverkstrafik och kostnadsfri programvara för trafikanalys kan vara tillräckligt för att observera kommunikation i närliggande trådlösa nätverk. Möjligheten att observera och analysera nätverkstrafik är därmed inte begränsad till avancerade eller resursstarka aktörer.

Detta kandidatarbete syftar till att undersöka vilken information som kan utvinnas genom passiv analys av trådlös nätverkstrafik. Arbetet omfattar både insamling och analys av trådlösa paket samt utveckling av ett system för att visualisera och analysera trafikmönster i nätverket.

1.1 Bakgrund

Information som följer med nätverkstrafik kan avslöja uppgifter om både enheter och deras användare. Detta medför betydande integritets- och säkerhetsrisker, eftersom observerad trafik kan användas för att identifiera enheter, kartlägga användarbeteenden och genomföra riktade attacker. Frågan om vilken information som kan utvinnas ur nätverkstrafik har därför blivit allt mer relevant i takt med att trådlösa nätverk och integritetsskyddande mekanismer fortsätter att utvecklas.

1.1.1 Förundersökning

I början av projektet genomfördes en förstudie i form av en Google-enkät för att undersöka deltagarnas kunskap, säkerhetsmedvetenhet och uppfattningar kring nätverkssäkerhet. Enkäten omfattade frågor som teknisk bakgrund, användning av nätverk och förståelse för krypterad kommunikation. Vidare behandlades frågor om vilken typ av information deltagarna trodde kunde utvinnas genom avlyssning av nätverkstrafik och hur trygga de kände sig vid uppkoppling. Se bilaga A.1 för länk till enkäten och dess svar.

Deltagarna var ungefär 50 stycken, varav 35 stycken var studenter på Chalmers Tekniska Högskola. Många deltagare bedömde sin tekniska kunskap som hög, men saknade ändå förståelse för vilken information som kan utvinnas ur trådlös nätverkstrafik. 53% av deltagarna angav att de kände sig oroliga eller osäkra när de var uppkopplade till nätverk. Samtidigt uppgav flera deltagare att de antingen uppfattade risken för avlyssning som låg eller inte betraktade avlyssning som problematiskt, eftersom de inte ansåg sig inneha någon känslig information.

Resultaten av enkäten pekar på att många saknar en grundläggande förståelse för hur datakommunikation fungerar och vetskap om de risker som är förknippade med nätverksuppkopplingar. Enkäten belyser även ett annat problem: en likgiltighet inför den information som exponeras vid datakommunikation. Denna inställning kan sannolikt härledas till bristande kunskap om vilken information som relativt enkelt kan spåras och analyseras. Sammantaget tyder resultaten på ett behov av att undersöka hur osäkra trådlösa nätverk är och vilka risker som finns.

1.1.2 Informationsutvinning ur nätverkstrafik

Tidigare forskning visar att observerad nätverkstrafik, trots kryptering, kan avslöja betydande mängder information om både enheter och deras användare. Trafikmönster och annan observerbar information kan exempelvis användas för att identifiera enhetstyp, operativsystem och återkommande beteendemönster [1, 2].

Att utvinna spår från nätverkstrafik skulle medföra flera integritets- och säkerhetsrisker. Om en angripare exempelvis kan identifiera operativsystemet och versionen på en enhet kan systemspecifika sårbarheter utnyttjas för att genomföra riktade cyberattacker [2]. Kombinationen av enhetsprofilering och dess återkommande tra-

fikmönster kan dessutom användas för att spåra och särskilja specifika enheter över tid, vilket kan leda till en avanonymisering av användare [3].

Observerad nätverkstrafik riskerar även att ge insikt i användarnas aktiviteter och rutiner. Tidpunkter för nätverksaktivitet och återkommande trafikmönster skulle exempelvis kunna användas för att uppskatta när enheter används, identifiera olika typer av aktiviteter och kartlägga dagliga beteendemönster [1, 4]. Detta innebär att omfattande information om användare kan utvinnas trots att innehållet i kommunikationen inte är tillgängligt, något som resultaten från förundersökningen indikerar att många användare har begränsad medvetenhet om.

1.2 Syfte

Syftet med detta arbete är att undersöka vilken information om enheter och deras användare som kan utvinnas genom passiv analys av observerad trafik i trådlösa nätverk. Arbetet fokuserar särskilt på den information som kan utläsas ur mönster och paket i krypterad nätverkstrafik, samt vilka integritetsrisker detta kan medföra.

I syfte att tydliggöra projektets inriktning och mål formuleras följande fyra frågeställningar:

- Med vilken sannolikhet är det möjligt att identifiera vilken applikation som genererar nätverkstrafiken?
- Är det möjligt att identifiera enhetstyp och operativsystem utifrån nätverkstrafik?
- Kan olika typer av användaraktiviteter, som videouppspelning eller näthandel, urskiljas genom analys av nätverkstrafik?
- Vilka beteendemönster över tid, exempelvis återkommande rutiner och tidpunkter för aktivitet, kan rekonstrueras genom analys av nätverkstrafik?

1.3 Genomförande

Arbetet baserades huvudsakligen på kvantitativ analys av observerad nätverkstrafik. Analysen möjliggjordes genom utveckling av en webbapplikation som kunde samla in och analysera nätverkstrafik med olika funktioner för bearbetning samt visualisering. För att undersöka möjligheten att identifiera applikationer användes maskininlärning för klassificering baserad på statistiska och tidsrelaterade egenskaper i trafiken. Identifiering av operativsystem och enhetstyp tillämpades genom analys av mönster och karakteristiska drag hos anslutna enheter. Utöver detta genomfördes kvalitativ analys av visualiserad trafikdata för att undersöka om användaraktiviteter och beteendemönster över tid kunde urskiljas och tolkas.

1.4 Avgränsningar och begränsningar

Studien behandlar enbart nätverkstrafik som förmedlas via trådlösa anslutningar. Trafik över kabelbundna anslutningar ingår inte, eftersom fokus ligger på scenarier där nätverkstrafik i praktiken är mer tillgänglig för passiv observation. Arbetet omfattar inte heller försök att kringgå krypteringsmekanismer, utan fokuserar enbart på den information som kan observeras utan åtkomst till trafikens innehåll.

Arbetet genomförs i enlighet med gällande lagstiftning, däribland lagen om elektronisk kommunikation (LEK) [5] och EU:s dataskyddsförordning (GDPR) [6]. Datainsamlingen begränsas därför till en kontrollerad labbmiljö för att minimera insamling av trafik från utomstående användare. Information som kan kopplas till enskilda individer behandlas inte i rapporten.

Projektets genomförande begränsas även av tillgängliga resurser, främst med avseende på tillgången till en varierad uppsättning testenheter. Testmiljön bygger till stor del på gruppmedlemmarnas privata utrustning, och avgränsas därför huvudsakligen till mobilenheter baserade på Android och iOS och datorer med operativsystemet Linux. Detta påverkar i sin tur resultatens representativitet och generaliserbarhet. All datainsamling och automatiserad analys genomförs även på datorer med Linuxbaserade operativsystem, eftersom denna miljö ger direkt åtkomst till nätverkskortet samt stöd för de verktyg och bibliotek som används för insamling och analys av nätverkspaket.

I moderna nätverk förekommer tekniker som syftar till att minska möjligheten till spårning, exempelvis samlade överföringar av paket och paketutfyllnad. Effekten av sådana tekniker förekommer inte i de använda testenheterna och analyseras inte i denna studie, eftersom de ligger utanför arbetets huvudsakliga fokus och endast förekommer i begränsad omfattning i praktiken.

2
Teori

Detta kapitel presenterar den teoretiska grund som krävs för att förstå de tekniker och metoder som används i arbetet. Först behandlas grundläggande datakommunikation, med fokus på trådlösa nätverk, adressering och observerbar information i nätverkstrafik. Därefter beskrivs principerna för systemdesign samt de verktyg, bibliotek och analysmetoder som ligger till grund för implementationen.

2.1 Grundläggande datakommunikation

För att kunna analysera trafik i trådlösa nätverk krävs en grundläggande förståelse för hur datakommunikation och nätverksprotokoll fungerar. Detta avsnitt beskriver därför centrala koncept som datapaket, adressering och trafikanalys, med särskilt fokus på observerbar information i trådlösa nätverksmiljöer.

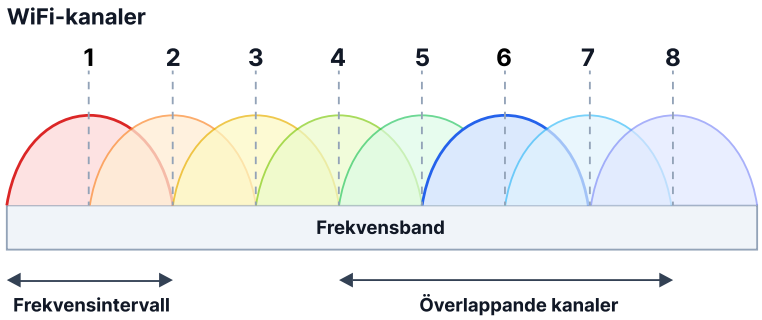
2.1.1 Nätverksstruktur och trådlös kommunikation

Nätverksanalys förutsätter en grundläggande förståelse för hur datakommunikation fungerar och nätverksinfrastruktur är uppbyggd. Kommunikation via WiFi regleras av IEEE 802.11-standarden [7], där IEEE står för Institute of Electrical and Electronics Engineers [8]. Standarden beskriver hur trådlös kommunikation ska struktureras, bland annat hur enheter ansluter till nätverk, skickar datapaket och hanterar radiovågor. För att enheter ska kunna kommunicera effektivt måste de följa samma standard, vilket säkerställer kompatibilitet mellan olika enheter [9, 10].

Kommunikation i nätverk delas upp i flera lager där varje lager ansvarar för olika delar av överföringen [11]. Inom datakommunikation överförs information i form av dataenheter, det vill säga avgränsade block av data som skickas mellan nätverksenheter såsom datorer och mobiltelefoner. Benämningen för dessa dataenheter varierar beroende på vilket lager som avses. I denna rapport används därför begreppet datapaket som ett generellt begrepp för dataenheter oavsett lager. Ett datapaket består förenklat av ett pakethuvud och en nyttolast [12]. Pakethuvudet innehåller styrinformation som krävs för att kommunikationen ska fungera korrekt, exempelvis avsändar- och mottagaradress, medan nyttolasten utgör den faktiska data som överförs. Strukturellt kan ett datapaket liknas vid ett brev, där nyttolasten motsvarar innehållet och pakethuvudet motsvarar kuvertet med adressinformation.

WiFi bygger på kommunikation via radiovågor mellan klientenheter och accesspunkter [7]. Klientenheter kan exempelvis vara datorer, mobiltelefoner eller andra uppkopplade enheter, medan accesspunkten fungerar som en brygga mellan de anslutna enheterna och publika nätverk. När en klientenhet kommunicerar omvandlas digital information till radiovågor som skickas till accesspunkten, som därefter vidarebefordrar trafiken inom nätverket eller till det publika nätverket. Nätverk kan förenklat delas in i lokala och publika nätverk [7]. Ett lokalt nätverk är begränsat till ett geografiskt område, exempelvis ett hem, universitet eller en arbetsplats, där anslutna enheter kan kommunicera direkt med varandra. Publika nätverk, som internet, används istället för kommunikation mellan olika lokala nätverk och möjliggör global uppkoppling mellan enheter och tjänster.

Flera trådlösa nätverk och enheter kan behöva kommunicera samtidigt inom samma geografiska område. För att möjliggöra detta delas nätverken upp i olika kanaler, där varje kanal motsvarar ett specifikt frekvensintervall inom ett frekvensband. Ett frekvensband utgör ett intervall av radiofrekvenser som används för trådlös kommunikation. Beroende på frekvensbandets egenskaper kan kanaler vara antingen överlappande eller separerade från varandra, se figur 2.1 [7]. Uppdelningen i kanaler minskar risken för störningar genom att kommunikationen fördelas över olika frekvenser, men interferens mellan nätverk kan fortfarande uppstå. Detta är särskilt vanligt i miljöer med hög nätverkstäthet, där flera nätverk kan använda samma eller överlappande kanaler samtidigt. Sådana störningar kan påverka både trafikens kvalitet och den data som fångas upp vid nätverksanalys.



Figur 2.1: Visualisering av kanaluppdelning inom ett frekvensband. Varje siffra representerar en WiFi-kanal och figuren visar hur närliggande kanaler kan överlappa i frekvensintervall.

2.1.2 Adressering och MAC-adresslumpning

Datakommunikation kräver mekanismer för adressering och dirigering så att datapaket kan skickas till rätt mottagare mellan olika enheter och nätverk. Detta sker

genom användning av flera typer av adresser som används på olika nivåer i kommunikationen. Två centrala adresstyper är IP-adresser (Internet Protocol) [12] och MAC-adresser (Media Access Control), som fyller olika funktioner i nätverket.

När datapaket skickas över nätverk används IP-adresser för att avgöra vart trafiken ska vidarebefordras [12]. I lokala nätverk används ofta privata IP-adresser, medan publika IP-adresser används vid kommunikation utanför det lokala nätverket [13]. Flera enheter inom samma lokala nätverk kan därför dela samma publika IP-adress vid kommunikation mot internet [14].

MAC-adressen kan också användas för att identifiera den specifika mottagarenheten i ett lokalt nätverk. En MAC-adress är avsedd att vara unik och är kopplad till en enhets nätverkskort. Den består av 48 bitar och delas vanligtvis upp i två delar: den första delen utgör ett OUI-prefix (Organizationally Unique Identifier), vilket identifierar tillverkaren av nätverkskortet, medan den andra delen används för att särskilja enskilda nätverksgränssnitt [15].

Både MAC- och IP-adresser kan användas för att identifiera och följa enheter över tid, särskilt i trådlösa nätverk där de kan observeras av andra enheter inom räckvidd. Moderna operativsystem har därför infört olika integritetsskyddande mekanismer, exempelvis MAC-adresslumpning [16]. Tekniken innebär att enhetens faktiska MAC-adress tillfälligt ersätts med en slumpmässigt genererad adress, vilket försvårar spårning och direkt identifiering av enheter.

Ett kännetecken för slumpgenererade adresser är att den lokalt administrerade biten i adressens första byte är satt. Detta visar att adressen har genererats eller modifierats via mjukvara istället för att vara fabriksinställd av tillverkaren. I hexadecimal notation innebär detta att den andra hexadecimala siffran antar värdena 2, 6, A eller E, och övriga delar av adressen genereras slumpmässigt [16].

2.1.3 Analys av nätverkstrafik

Ett centralt skydd för användarens integritet på ett nätverk är kryptering. Moderna applikationer och tjänster använder ofta krypterade protokoll för att skydda innehållet i kommunikationen från utomstående parter [17]. Kryptering minskar risken att en obehörig tredje part kan läsa meddelanden, inloggningsuppgifter eller annan information som utbyts mellan applikationer [18]. Kryptering innebär dock inte att allt informationsläckage försvinner.

Metadata utgör den information som beskriver kommunikationen snarare än dess innehåll. Den är till skillnad från nyttolasten, fortsatt synlig även vid kryptering av nätverkstrafik. Eftersom trafiken måste kunna vidarebefordras korrekt krävs att viss trafikrelaterad information förblir tillgänglig, vilket innebär att metadatan inte kan döljas utan att påverka kommunikationen. Denna information omfattar bland annat adressering och paketstorlekar [19]. En observatör kan därmed analysera metadata och dra slutsatser om kommunikationen utan att behöva dekryptera nyttolasten.

Trafikanalys kan delas in i passiv och aktiv analys beroende på om observatören påverkar kommunikationen eller enbart observerar den. Aktiv trafikanalys innebär att observatören aktivt påverkar nätverkstrafiken, exempelvis genom att skicka specifika förfrågningar, skapa artificiella fördröjningar eller på annat sätt interagera med enheter i nätverket. Aktiv analys kan ge mer precisa och djupgående insikter, men är samtidigt mer resurskrävande och lättare att upptäcka. Passiv trafikanalys innebär istället att nätverkstrafik observeras utan att befintliga paket modifieras. Analysen baseras därmed främst på metadata som redan finns tillgänglig i nätverket, och eftersom kommunikationen inte påverkas är passiv analys i regel svår att upptäcka för de observerade enheterna [19].

En typ av sådan metadata-analys är genom analys av de sökförfrågningar som enheter skickar ut för att identifiera tillgängliga nätverk. Dessa sökförfrågningar utgörs av nätverkspaket som används för att upptäcka närliggande accesspunkter. I sökförfrågningar finns strukturerade datafält, kallade informationselement. Varje element har ett unikt ID-nummer, och den ordning i vilken dessa ID-nummer förekommer bildar en sekvens. Elementen beskriver vilka funktioner en enhet stödjer och dess nätverkskrav [20]. Sökförfrågningar skickas ofta okrypterat, vilket innebär att de kan observeras och analyseras av andra enheter inom räckvidd. De skickas även periodiskt när enheten inte aktivt ansluter till ett nätverk, vilket bidrar till ett kontinuerligt läckage av metadata i nätverksmiljön [21].

2.2 Design- och implementationsprinciper

Utveckling av mjukvarusystem bygger inte enbart på funktionella krav, utan även på principer som syftar till att göra systemen begripliga, utbyggbara och resurseffektiva. I system som hanterar kontinuerliga dataströmmar och realtidsvisualisering ställs särskilda krav på hur olika delar av systemet struktureras och samverkar.

Modularitet är en arkitekturell princip inom mjukvarudesign där ett system delas upp i moduler. Varje modul har ett eget väldefinierat ansvarsområde. Eftersom modulerna är relativt oberoende av varandra möjliggörs isolering av delar för testning och felsökning. Detta förenklar hanteringen av systemets komplexitet och möjliggör att enskilda delar kan vidareutvecklas eller förändras utan att påverka övriga delar av systemet i onödan [22].

Ett sätt att utnyttja principen om modularitet är genom observatörsmönstret. Observatörsmönstret är ett designmönster som används för att hantera kommunikation mellan olika delar av ett system. Mönstret bygger på två huvudroller: publicerare och observatör. Ofta finns en publicerare och flera observatörer. Observatörerna kan prenumerera och avprenumerera från publiceraren beroende på om de vill ta del av dess information. Medan publiceraren skickar ut uppdateringar till observatörerna när ny information tillkommer. Syftet är att låta en komponent sprida information till andra komponenter på ett modulärt sätt. Publiceraren behöver inte känna till detaljer om observatörerna, och observatörerna agerar dessutom oberoende av

varandra. Detta skapar ett centraliserat sätt att distribuera information till flera delar av ett system och möjliggör att komponenter kan läggas till eller tas bort utan att övriga delar påverkas [23].

2.3 Verktyg, program och bibliotek

Insamling, analys och visualisering av nätverksdata bygger på flera digitala verktyg, program och bibliotek. Detta avsnitt beskriver de tekniker som ligger till grund för arbetets implementation.

2.3.1 Insamling och analys av nätverkstrafik

För att en enhet ska kunna kommunicera över ett nätverk krävs ett nätverkskort som hanterar sändning och mottagning av data. Vanligtvis filtrerar nätverkskortet bort all trafik som inte är adresserad till enheten, så att endast relevant trafik skickas vidare till operativsystemet. Vissa nätverkskort kan dock konfigureras till ett så kallat övervakningsläge [24]. I övervakningsläget inaktiveras paketfiltreringen, vilket innebär att all observerbar trådlös trafik vidarebefordras till operativsystemet, även trafik som inte är adresserad till den aktuella enheten. För att aktivera övervakningsläget kan verktyget `Airmon-ng` användas [25]. Därefter kan en trafikavlyssnare användas för att samla in och analysera paket som förekommer i den trådlösa miljön.

Ett sätt att implementera trafikavlyssnare är genom att använda bibliotek, som exempelvis `Scapy`. `Scapy` kan användas för både aktiv paketgenerering och passiv insamling av nätverkstrafik. Vid användning tillsammans med ett nätverkskort i övervakningsläge kan `Scapy` spara fångad nätverkstrafik, vilket möjliggör analys av nätverkspaketen [26]. Ett verktyg som ofta används för analys av nätverkspaket är Wireshark. Wireshark är en programvara som används inom flera områden, främst för att analysera nätverkstrafik. Programvaran erbjuder ett grafiskt användargränssnitt för inspektion och visualisering av fångad nätverkstrafik [27].

`Mac-vendor-lookup` är ett Python-bibliotek som möjliggör identifiering av tillverkare baserat på en given MAC-adress. Biblioteket använder en lokalt lagrad kopia av IEEE:s OUI-prefixdatabas, vilket gör det möjligt att genomföra identifiering utan nätverksåtkomst. Metoden baseras på MAC-adressens OUI-prefix för att avgöra den ursprungliga tillverkaren. Informationen är dock begränsad till de prefix som finns registrerade i databasen [28].

2.3.2 Webbaserad visualisering

En webbserver hanterar kommunikation mellan klient- och serversida genom att ta emot inkommande förfrågningar och skicka tillbaka svar i form av exempelvis webbsidor eller data. I mer komplexa applikationer används ofta webbserverramverk för att förenkla utvecklingen och strukturera applikationens logik. Dessa ramverk erbjuder färdiga funktioner för hantering av förfrågningar och svar, vilket minskar behovet av att implementera grundläggande serverfunktionalitet från grunden. Ett

exempel på ett sådant ramverk är `Flask`, ett Pythonbaserat ramverk som kännetecknas av sin flexibilitet och enkelhet [29].

En WebSocket-anslutning är en beständig kommunikationskanal som möjliggör tvåvägskommunikation i realtid mellan klient och server. Både klient och server kan skicka data över samma anslutning när ny information finns tillgänglig [30]. Detta skiljer sig från traditionell HTTP-kommunikation (Hypertext Transfer Protocol), där klienten normalt initierar varje förfrågan, vilket gör protokollet mindre effektivt vid frekvent dataöverföring. `Socket.IO` är ett JavaScript-baserat bibliotek som utnyttjar WebSocket-anslutningar för att hantera kommunikationen i en webbserver. `Socket.IO` kan användas i Python genom tredjepartsbiblioteket `Python-socketio` [31].

Ett annat JavaScript-baserat bibliotek är `Chart.js`, som istället används för att visualisera data i form av diagram i webbläsaren. Det stödjer flera typer av diagram, som exempelvis linje-, stapel- och cirkeldiagram. En central funktion är möjligheten att dynamiskt uppdatera diagram baserat på nya datavärden, vilket gör det lämpligt för visualisering av föränderliga datamängder i realtidssystem [32].

2.3.3 Git och GitLab

Git är ett versionshanteringsverktyg som används för att spara, spåra och hantera ändringar i kod över tid. Systemet gör det möjligt att se tidigare versioner, följa utvecklingen av projekt och vid behov återgå till en tidigare version. Git möjliggör även att flera användare parallellt kan arbeta på olika kodfiler eller på en och samma kodfil, genom så kallade grenar [33]. För att ha en central kodbas kan plattformen GitLab användas [34].

2.4 Maskininlärning och dataanalys

Maskininlärning bygger på algoritmer som analyserar data för att identifiera mönster och samband. Genom denna process skapas en maskininlärningsmodell som kan användas för att klassificera observationer utifrån fördefinierade klasser och identifierade mönster. Modellen kan därefter tillämpas på ny data för att göra prediktioner utan att vara explicit programmerad för varje enskilt fall [35].

En maskininlärningsmodell tränas på en datamängd och en samling attribut, även kallade egenskaper. Inför träningen delas datamängden upp i tränings- och testdata. Träningsdatan används för att lära modellen identifiera mönster, medan testdatan används för att utvärdera hur väl modellen fungerar. Attributen är mätbara eller konstruerade egenskaper hos datamängden som används som indata till modellen. Valet av attribut är centralt eftersom det påverkar modellens förmåga att särskilja olika klasser. Vissa egenskaper kan exempelvis vara överflödiga och ha låg klassificeringsförmåga, vilket kan påverka modellens träffsäkerhet negativt. Av denna anledning är det viktigt att identifiera de egenskaper som bäst fångar skillnader mellan datapunkter [36].

Ett mått för att utvärdera prestandan hos en maskininlärningsmodell är träffsäkerhet. Träffsäkerhet anger andelen korrekta prediktioner som en modell gör i förhållande till det totala antalet observationer. Detta mått är dock mindre tillförlitligt vid obalanserade datamängder, eftersom modellen kan uppnå en hög träffsäkerhet genom att konsekvent förutsäga den mest frekvent förekommande klassen [37]. Ett annat vanligt utvärderingsmått är precision, vilket anger andelen sant positiva förutsägelser i förhållande till alla observationer som modellen har klassificerat som positiva. Med andra ord beskriver precision hur många av de positiva prediktionerna som faktiskt är korrekta. Detta mått är särskilt viktigt i situationer där falskt positiva klassificeringar kan få betydande konsekvenser, eftersom det ger en indikation på hur tillförlitliga modellens positiva prediktioner är.

För att träna en modell kan `Scikit-learn` användas. `Scikit-learn` är ett bibliotek i Python som innehåller olika algoritmer för klassificering [38]. En vanlig maskininlärningsalgoritm, som finns inbyggd i `Scikit-learn`, är Random Forest-algoritmen. Random Forest-algoritmen är en maskininlärningsmetod som bygger på flera beslutsträd. Varje träd tränas på ett slumpmässigt urval av den givna data och en slumpmässig delmängd av de givna attributen. Den slutliga förutsägelsen fås genom att kombinera resultaten från alla träd, genom att den klass som flest träd förutsäger väljs. Random Forest-algoritmen är alltså en typ av modell som klassificerar data genom att kombinera resultaten från flera beslutsträd, var och en uppbyggd av regelbaserade uppdelningar, tills en klass fastställs [39].

2.5 Statistisk analys

Inom statistisk analys finns ett flertal mått för att beskriva datapunkter och deras relation till varandra. Ett mått är kurtosis, vilket beskriver hur mycket en fördelnings ändar avviker från en normalfördelning. Måttet kan användas för att identifiera om datan innehåller fler extrema värden än väntat. Ett högt kurtosisvärde indikerar fler sådana extremvärden [40].

Skevhet beskriver asymmetrin i en fördelning relativt dess medelvärde. Ett positivt värde innebär att fördelningen har en längre ände åt höger, det vill säga att det förekommer fler extremvärden över medelvärdet, medan en negativ skevhet innebär motsatsen [40].

Ankomstojämnhet är ett statistiskt mått som beskriver hur jämnt paket anländer över tid. Paketströmmen kan representeras som en tidsserie, där tidsintervallen mellan efterföljande paket beräknas och analyseras. Ett högt värde på ankomstojämnhet indikerar att paketen tenderar att anlända i korta, intensiva grupper med tydliga perioder av hög aktivitet följt av perioder med låg eller ingen aktivitet. Ett lägre värde indikerar däremot ett mer jämnt och stabilt mönster. Ankomstojämnhet, a, kan beräknas enligt ekvation (2.1).

$$a = \frac{\sigma - \mu}{\sigma + \mu} \tag{2.1}$$

I ekvationen representerar μ medelvärdet av tidsintervallen mellan paket, medan σ representerar standardavvikelsen för dessa intervall.

Jon Kleinbergs algoritm är en annan metod för att beräkna ankomstojämnhet [41]. Algoritmen gör detta genom att modellera datan som ett system, och sedan växla mellan olika tillstånd med olika aktivitetsnivåer. En implementation av algoritmen finns i Python-biblioteket `burst_detection` [42].

3
Metod

Detta kapitel beskriver de metoder och tekniska lösningar som användes för att samla in, bearbeta och analysera nätverkstrafik. Inledningsvis presenteras den övergripande metodiken och de analysmetoder som utvecklades för identifiering av applikationer, operativsystem, enhetstyper och användarbeteenden. Därefter beskrivs implementationen av det verktyg som integrerar systemets analysfunktioner samt hur dessa utvecklades och utvärderades.

3.1 Metodik

För att undersöka möjligheten att identifiera vilken applikation som används på en enhet utifrån dess nätverkstrafik tillämpades maskininlärning. En maskininlärningsmodell tränades på egenskaper baserade på ankomstojämnhet och trafikvolym från en datamängd bestående av insamlad nätverkstrafik. Modellen användes därefter för att i realtid prediktera vilken applikation som genererade trafiken.

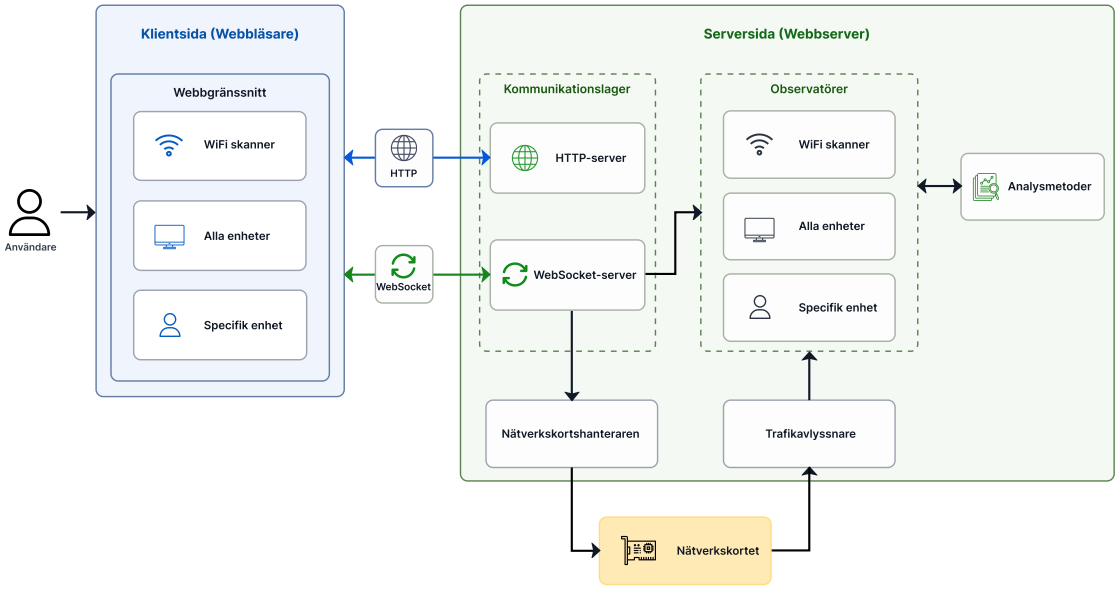
Två metoder användes för identifiering av operativsystem. Den första metoden baserades på mönstermatchning med hjälp av informationselement från sökförfrågningar i ett trådlöst nätverk. Den andra metoden undersökte möjligheten att använda maskininlärning för ändamålet, där en modell tränades på statistiska egenskaper baserade på tidsmönster i insamlad nätverkstrafik. För identifiering av enhetstyp användes signalstyrka som indikator, där en varierande signalstyrka tolkades som att enheten var i rörelse och därmed sannolikt en mobilenhet.

Användaraktiviteter analyserades genom att urskilja olika typer av aktiviteter utifrån nätverkstrafikens egenskaper. Trafiken kategoriserades i grupper baserade på egenskaper som paketfrekvens och ankomstojämnhet. Utöver enskilda aktiviteter undersöktes även beteendemönster genom att studera hur den samlade WiFi-trafiken från flera enheter förändrades över längre tidsperioder.

Ett gemensamt analysverktyg utvecklades för att möjliggöra en systematisk, samlad och automatiserad utvärdering av nätverkstrafik. Genom att integrera samtliga metoder i ett och samma system kunde nätverkstrafik både samlas in och analyseras i realtid. Detta möjliggjorde parallell identifiering av applikation, operativsystem, enhetstyp och användarbeteende utifrån samma paketström.

3.2 Analysverktyget

Detta avsnitt behandlar analysverktygets systemarkitektur samt den underliggande processen för insamling, bearbetning och visualisering av nätverkstrafik. Inledningsvis beskrivs arkitekturen på klient- och serversidan. Därefter redogörs för hur nätverkspaket fångas upp och distribueras genom systemet, samt hur data lagras och förbereds för analys. Avsnittet avslutas med en genomgång av webbapplikationens olika vyer och deras respektive funktioner.



Figur 3.1: Schematisk översikt över analysverktygets arkitektur och kommunikationsflöde mellan klient och server. Till vänster visas klientsidan i form av ett webbaserat gränssnitt med tre sidor. Dessa kommunicerar med serversidan via HTTP och WebSocket-anslutningar. Serversidan visas till höger och där hanteras bakgrundslogik för varje sida, med motsvarande uppdelning.

3.2.1 Systemets arkitektur och designval

Analysverktygets övergripande arkitektur visas i figur 3.1. Systemet är uppdelat i en klientsida och en serversida som tillsammans möjliggör realtidsbaserad analys och visualisering av trådlös nätverkstrafik.

Klientdelen består av ett webbaserat grafiskt användargränssnitt med tre sidor: *WiFi-skanner*, *Alla enheter* och *Specifik enhet*. Se bilaga C.1, C.2 respektive C.3 för respektive sidas utseende ur användarperspektiv. WiFi-skannern används för att identifiera och välja tillgängliga trådlösa nätverk att analysera. Sidan *Alla enheter* ger en samlad översikt av identifierade enheter, det vill säga alla enheter som skickar eller tar emot nätverkspaket inom det valda nätverket. Sidan *Specifik enhet* visar istället mer detaljerad information om en enskild enhets kommunikationsmönster

och identifierade egenskaper.

Webbgränssnittet är implementerat med HTML-ramar för struktur och JavaScript-baserade komponenter för interaktivitet. För visualisering av tidsserier och statistiska mönster används `Chart.js`, vilket möjliggör dynamiska grafer direkt i webbläsaren. Ett webbaserat gränssnitt valdes för analysverktygets eftersom webbteknologier erbjuder väletablerade bibliotek för visualisering och interaktivitet, exempelvis i jämförelse med ett terminalbaserat gränssnitt. En grafisk och användarvänlig presentation underlättar tolkningen av insamlad data, vilket i sin tur gör det möjligt att fokusera mer på själva analysen.

På serversidan finns tre observatörer, som motsvarar för varje klientsida, och ansvarar för den bakomliggande logiken, se figur 3.1. Det finns också en nätverkskortshanterare som konfigurerar nätverkskortet till övervakningsläge och hanterar kanalväxling under nätverksskanning. Alla paket som samlas in av nätverkskortet hanteras i sin tur av trafikavlyssnaren, som distribuerar paketen vidare till observatörerna för bearbetning. Observatörerna använder därefter olika analysmetoder för att identifiera operativsystem, enhetstyper, applikationer och användarmönster baserat på observerad nätverkstrafik.

När analysen är utförd skickas informationen från webbservern till webbgränssnittet. Webbservern är implementerad med `Flask` och `Socket.IO`. `Flask` ansvarar för att hantera HTTP-rutter och leverans av webbgränssnittets sidor, medan `Socket.IO` hanterar realtidskommunikationen via WebSocket-anslutningar mellan klient- och serversidan. Detta innebär en hybridarkitektur där initial inläsning av webbsida sker via HTTP, medan all dynamisk dataöverföring sker via WebSocket-kanaler.

WebSocket-baserad kommunikation valdes eftersom data endast överförs när ny information finns tillgänglig, vilket reducerar onödig nätverkstrafik och minskar belastningen på systemet. Denna anslutning utgör en beständig kommunikationskanal som möjliggör kontinuerlig överföring av nätverksdata i realtid, utan behov av upprepade HTTP-förfrågningar. Meddelanden som skickas över WebSocket-anslutningen är i JSON-format (JavaScript Object Notation), vilket ger ett enkelt, flexibelt och språkoberoende sätt att strukturera data.

3.2.2 Systemets arbetsflöde

Före analys skickar användaren en förfrågan till systemet om att aktivera övervakningsläget. Förfrågningen består av ett meddelande som skickas över WebSocket-kanalen till Nätverkskortshanteraren. Nätverkskortshanteraren konfigurerar nätverkskortet till övervakningsläge genom ett `Airmon-ng`-kommando. Nätverkskortet kan därefter avlyssna all nätverkstrafik inom dess räckvidd. Den insamlade trafiken avläses och hanteras därefter av en trafikavlyssnare som använder sig av biblioteket `Scapy`.

Trafikavlyssnaren är implementerad enligt observatörsmönstret och fungerar som en publicerare där varje insamlat nätverkspaket vidarebefordras till samtliga observatörer. Systemet har tre analyskomponenter som agerar observatörer, vilket innebär att de kan bearbeta trafiken parallellt. Trafikavlyssnaren är alltså inte beroende av analyskomponenterna, utan ansvarar enbart för insamling och distribution av data. Analyskomponenter kan därför läggas till, modifieras eller tas bort utan att påverka insamlingsprocessen.

Efter aktivering av övervakningsläget kan systemets *WiFi-skanner* initieras. Skannern identifierar tillgängliga trådlösa nätverk genom kanalbaserad skanning över ett fördefinierat intervall av WiFi-kanaler, där kanaler med frekvensmässigt överlapp exkluderas. För varje kanal omkonfigureras nätverkskortet via nätverkskortshanteraren, och nätverkspaket analyseras för att identifiera närliggande accesspunkter. Varje upptäckt nätverk representeras som ett unikt objekt baserat på accesspunktens MAC-adress och tillhörande nätverksnamn, och visas tillsammans med dess signalstyrka och kanalinformation. Denna information visualiseras i webbgränssnittet där användaren kan välja ett specifikt nätverk för vidare analys.

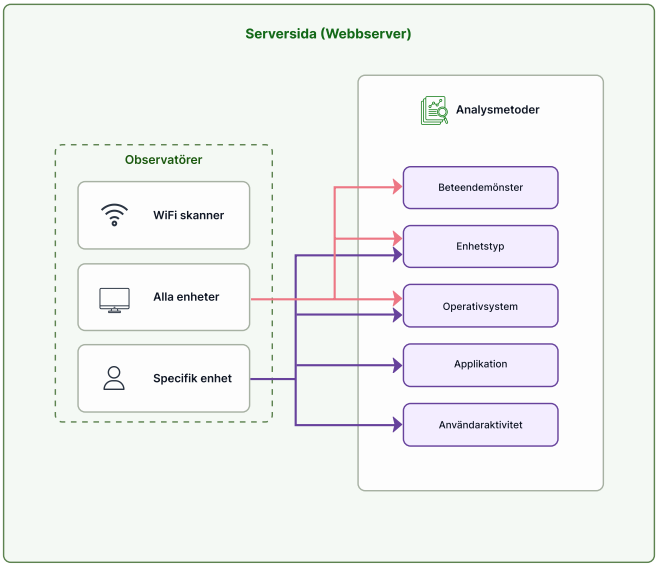
När ett nätverk har valts kan observatörerna *Alla enheter* och *Specifik enhet* börja ta emot data från trafikavlyssnaren och utföra analyser av den observerade nätverkstrafiken. Resultaten skickas därefter via WebSocket-anslutningen till webbgränssnittet där de visualiseras för användaren. Dessa vyer representerar en gradvis fördjupning av analysen, från en övergripande nätverksöversikt till detaljerad analys av individuella enheter. Arbetsflödet består alltså av att först välja ett nätverk, därefter analysera samtliga enheter inom nätverket och slutligen fördjupa analysen genom att välja en specifik enhet.

När användaren navigerar till sidan *Alla enheter*, registrerar sig *Alla enheter* som observatör och börjar ta emot nätverkspaket från trafikavlyssnaren. Paketen sorteras baserat på MAC-adressen, vilket innebär att varje enhet tilldelas en egen profil som uppdateras kontinuerligt utifrån observerad trafik. För varje enhet beräknas nätverksbaserade egenskaper som ankomstojämnhet, trafikvolym och paketfrekvens. Utöver dessa statistiska mått genomförs även identifiering av operativsystem, enhetstyp och beteendemönster baserat på observerade nätverksmönster, se figur 3.2.

Om en specifik enhet på sidan *Alla enheter* väljs, övergår systemet från sammanställd analys till mer detaljerad analys. På denna sida sammanställs resultat från flera analysmetoder: identifiering av applikation, operativsystem, enhetstyp och användaraktivitet, enligt figur 3.2. Systemet genererar även visualiseringar av nätverksaktivitet över tid, vilket möjliggör analys av kommunikationsmönster, signalvariation och aktivitetsmönster. Detta ger en djupare förståelse för en enhets beteende och kommunikation inom nätverket.

För att möjliggöra analyser sparas insamlad nätverkstrafik, som innehåller både strukturell och tidsbaserad information om enheter, för vidare bearbetning. Central information som MAC-adresser och tidsstämplar sparas i CSV-filer (Comma-

Separated Values), ett filformat där data lagras som textvärden separerade med kommatecken. Detta val av lagringsformat motiveras av dess enkelhet och kompatibilitet med efterföljande analysverktyg. CSV-filer genereras dynamiskt under körning och kan därefter användas för att rekonstruera tidsserier som ligger till grund för visualisering, statistisk analys och modellbaserad tolkning av nätverksbeteenden.



Figur 3.2: Översikt över funktionaliteten i vyerna *Alla enheter* och *Specifik enhet*. Pilarna visar hur respektive vy är kopplad till analysmetoderna.

3.3 Identifiering av applikation

För att undersöka om nätverkstrafik kan användas för att identifiera vilken applikation som används på en avlyssnad enhet utvecklades funktionalitet för realtidsklassificering. Systemet analyserade nätverkstrafik från enheten och avgjorde vilken av fem utvalda applikationer som sannolikt hade genererat den, med hjälp av maskininlärning. Maskininlärning valdes eftersom nätverkstrafik kan innehålla komplexa och svårtolkade mönster som kan vara svåra att identifiera manuellt. De applikationer som inkluderades var YouTube, TikTok, Instagram, Facebook och LinkedIn. Dessa valdes eftersom de är vanligt förekommande och fanns representerade i den datamängd som användes för att träna modellen.

3.3.1 Träning av modell

Metoden för att träna modellen baserades till stor del på den metod som presenteras i Classifying 5G Encrypted Packet Traces [43], med undantag för vissa maskininlärningsattribut som valdes bort efter experimentell utvärdering. Modellen tränades även på artikelns tillhörande datamängd, som samlades in år 2023. Datamängden består av nätverkstrafik mellan en router och mobiltelefoner av modellerna Sony

Xperia Z3 och Motorola G7, där varje applikation analyseras separat och utan samtidig trafik från andra applikationer. Datamängden innehåller cirka 200 pcap-filer per applikation, där PCAP (packet capture) är ett filformat för lagring av inspelad nätverkstrafik. Varje fil omfattar omkring 60 sekunder insamlad trafik där aktiviteter, som videouppspelning och bläddring i flöden, utförs inom respektive applikation.

För att möjliggöra modellträning förbehandlades datamängden i flera steg. Varje PCAP-fil delades först upp i segment om 30 sekunder. Denna segmentlängd valdes för att säkerställa att varje segment innehöll tillräckligt med data för att modellen skulle kunna identifiera relevanta mönster. Därefter tillämpades ett glidande tidsfönster inom varje segment, där fönstret förflyttades i steg om 50 millisekunder för att aggregera trafiken över tid. Trafiken delades upp i två riktningar: från mobil till router och från router till mobil. Den förstnämnda representerades med negativa värden och den sistnämnda med positiva, vilket möjliggör att båda riktningar kan hanteras inom samma segment utan separata variabler.

Perioder av intensiv nätverkstrafik kunde identifieras genom beräkning av ankomstojämnhet. För detta användes Kleinbergs algoritm via biblioteket `burst-detection`, eftersom det erbjuder en färdig implementation. Analys av ankomstojämnhet möjliggjorde undersökning av hur nätverkstrafiken varierade över tid, exempelvis med avseende på volym och riktning. För varje segment extraherades därefter ett antal statistiska och trafikrelaterade egenskaper, baserade på både ankomstojämnhet och total trafikvolym. Egenskaperna som analyserades var följande:

- Medelvärde, varians och standardavvikelse
- Skevhet och kurtosis
- Percentilbaserade mått
- Maximi- och minimivärden
- Median absolutavvikelse

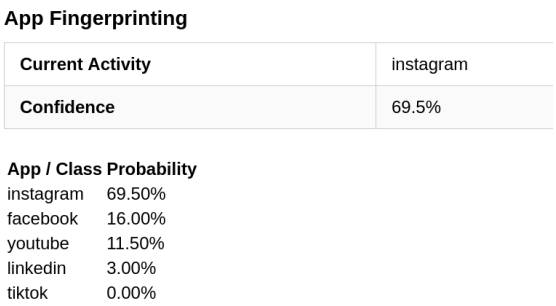
Datamängden delades därefter upp i tränings- och testdata med en fördelning på 75% respektive 25%. Denna fördelning ansågs ge tillräckligt med data för träning samtidigt som en representativ mängd kunde reserveras för utvärdering av modellen. Klassificeringen utfördes med en Random Forest-algoritm från `Scikit-learn` bestående av 200 träd, utan begränsning av trädens djup och med balanserade klassvikter för att hantera eventuell klassobalans i datamängden. Dessa parametrar valdes genom iterativ justering och utvärdering av olika konfigurationer i syfte att identifiera de som gav bäst klassificeringsresultat. Random Forest-algoritmen valdes eftersom den lämpar sig för datamängder med många egenskaper och är enkel att implementera. Efter träning av modellen användes den för att prediktera klasser för testdatan, och dess prestanda utvärderades med hjälp av måtten precision och träffsäkerhet.

3.3.2 Användning av modell i realtid

Den tränade modellen integrerades i komponenten *Specifik enhet*. Vid körning samlades nätverkspaket kontinuerligt in och lagrades i en buffert. Den inkommande datan bearbetades sedan på motsvarande sätt som träningsdatan, där paketen filtrerades

och analyserades med ett glidande tidsfönster på 30 sekunder. Därefter delades trafiken upp baserat på riktning, varefter analys av ankomstojämnhet utfördes. För att säkerställa stabila prediktioner filtrerades fönster med otillräcklig trafikmängd bort. När tillräcklig mängd trafik hade ackumulerats, extraherades samma uppsättning egenskaper som vid träning och användes som indata till modellen. Modellen beräknade därefter en sannolikhet för varje applikation, och den applikation som fick högst sannolikhet tolkades som modellens prediktion.

Modellens resultat visualiserades på sidan *Specifik enhet* genom en tabell som visade varje applikation och dess tillhörande sannolikhet, samt genom en presentation av den predikterade applikationen, enligt figur 3.3. Denna vy uppdaterades med en frekvens på 30 sekunder.



Figur 3.3: Visualisering av modellens resultat i analysverktyget.

3.3.3 Realtidstestning av modell

Utvärdering av modellens träffsäkerhet genomfördes på nätverkstrafik genererad av en Android-baserad mobiltelefon av märket Xiaomi, modell Poco F7. För varje applikation (YouTube, TikTok, Instagram, Facebook och LinkedIn) genomfördes tre separata tester. För Facebook och Instagram utfördes dessutom ytterligare tre tester för att undersöka en alternativ användarhandling. Varje test bestod av två minuters användning av applikationen, där modellens prediktion noterades var 30:e sekund. Detta resulterade i totalt 12 prediktioner per applikation, med undantag för Facebook och Instagram som istället gav 24 prediktioner vardera. Under testerna utfördes samma typer av användarhandlingar som modellen tidigare tränats på, för att säkerställa jämförbarhet mellan realtidsdata och träningsdata. För Instagram och Facebook inkluderade testerna bläddring i flödet och visning av 'Reels'. För LinkedIn och TikTok testades enbart bläddring i flödet, medan videouppspelning användes för YouTube.

För att minimera störningar och öka testernas tillförlitlighet genomfördes samtliga mätningar utan att köra andra program i bakgrunden. Endast en av de fem utvalda applikationerna var aktiv vid varje testtillfälle. Under varje test samlades nätverkstrafik in och bearbetades i realtid av systemet. För varje intervall genererades prediktioner från modellen, tillsammans med tillhörande sannolikheter.

3.4 Identifiering av operativsystem

Identifiering av operativsystem begränsades i detta arbete till iOS och Android och genomfördes huvudsakligen genom analys av sökförfrågningar och informationselement. Denna metod, benämnd Metod A, baserades på sekvenser av informationselement i sökförfrågningar som jämfördes med referensmönster för att identifiera operativsystemet [44]. Utöver detta undersöktes en alternativ metod baserad på maskininlärning för att analysera om operativsystem kunde identifieras utifrån trafikmönster i nätverksflöden. Denna metod, benämnd Metod B, byggde på maskininlärning där en Random Forest-modell tränades på nätverkstrafik för att undersöka om operativsystem kunde klassificeras.

3.4.1 Metod A

Identifiering av operativsystem på olika enheter genomfördes genom analys av sökförfrågningar med fokus på informationselement. Nätverkstrafik från enheter samlades in och undersöktes manuellt med hjälp av Wireshark. Från varje paket extraherades MAC-adress och sekvensen av informationselement. Informationselementsekvensen användes som en signatur för enheten. Signaturen jämfördes därefter med referensmönster för iOS och Android som tagits fram genom analys av informationselementsekvenser från flera enheter, vilket möjliggjorde identifiering av operativsystemet.

Analysen genomfördes iterativt genom att först undersöka informationselementstrukturen för en enskild enhet och därefter jämföra resultaten med flera andra enheter som använde samma operativsystem. Genom att identifiera återkommande mönster i ordning och förekomst av informationselement kunde specifika regler för operativsystemsidentifiering formuleras. För iOS observerades att de inledande elementen i sekvensen följde prefixet (0, 1, 50), att minst ett av elementen 45 eller 127 förekom, samt att en Apple-specifik OUI återfanns i sökförfrågningarna. För Android observerades istället att sekvensen inleddes med antingen (0, 1, 50, 45) eller (0, 1, 50, 3, 45), samt att elementen 127 och 221 förekom i sekvensen. Dessa mönster användes som grund för en identifieringsmetod baserad på mönstermatchning av informationsselementsekvenser. En matchning fastställdes när en sekvens uppfyllde de regler som identifierats för respektive operativsystem.

Som komplement till mönstermatchningen användes biblioteket `Mac-vendor-lookup` för att koppla MAC-adresser till respektive tillverkare utifrån adressens OUI-prefix. Den identifierade tillverkaren användes därefter för att göra en förenklad uppskattning av operativsystemet. Exempelvis antogs Apple-enheter använda iOS, medan Samsung-enheter antogs använda Android. Denna metod fungerade dock endast som ett komplement till den huvudsakliga identifieringsmetoden, då OUI-prefix inte är tillgängliga i alla MAC-adresser. Även om OUI-prefixet är tillgängligt betyder det dessutom inte att den nödvändigtvis är tillförlitlig, exempelvis om MAC-adressen är slumpmässigt genererad.

För att identifiera slumpmässigt genererade MAC-adresser implementerades en kontrollmekanism. Detta gjordes genom analys av den lokalt administrerade biten i MAC-adressen. Om en MAC-adress identifierades som slumpmässigt genererad markerades detta, och tillverkarinformationen uteslöts som grund för identifiering av operativsystemet. I sådana fall baserades identifieringen enbart på resultat från analysen av informationselementet. Om även den analysen misslyckades, markerades operativsystemet som okänt.

Systemet testades initialt på 27 enheter, både nya och äldre modeller, för att säkerställa att det fungerade korrekt. Därefter genomfördes ytterligare tester på två enheter, av varje operativsystem, för att verifiera att systemet fungerade korrekt i samtliga fall. Inom respektive operativsystem hade en enhet MAC-adresslumpning aktiverad, medan den andra inte hade det.

Visualiseringen för operativsystemsidentifiering bestod av en kolumn i en tabell på sidan *Alla enheter*, samt en textrad på sidan *Specifik enhet*. Båda uppdaterades automatiskt och visade det identifierade operativsystemet.

3.4.2 Metod B

Syftet med Metod B är att undersöka om operativsystem kan identifieras genom maskininlärning. För detta tränades en Random Forest-modell på observerbara trafikmönster i nätverksflöden. Metoden bestod av fem huvudsteg: sökning efter en lämplig datamängd, dataförberedelse, skapande av egenskaper, modellträning och utvärdering.

Arbetet inleddes med att söka efter datamängder som innehöll både nätverkstrafik och tillhörande information om enheternas operativsystem. Den datamängd som identifierades bestod av CSV-filer innehållande nätverksflöden tillsammans med tillhörande metadata samt referensetiketter som angav vilket operativsystem enheten använde. Datamängden användes som grund för att undersöka om operativsystem kunde identifieras med hjälp av maskininlärning. Under arbetets gång konstaterades dock att datamängden inte innehöll tillräckligt detaljerad information för att möjliggöra en tillförlitlig analys.

För att simulera ett realistiskt scenario med passiv avlyssning filtrerades datamängden, där endast följande behölls: starttid för flödet, sluttid för flödet, sessions-ID och referensetiketter. Flödena grupperades sedan per enhet med hjälp av sessions-ID. För varje enhet beräknades antalet flöden och statistiska mått, som medelvärde samt minsta och största varaktighet. Dessa användes därefter som attribut till Random Forest-modellen. Slutligen utvärderades den tränade modellen för att bedöma dess förmåga att klassificera operativsystem korrekt.

3.5 Identifiering av enhetstyp

Identifiering av enhetstyp baserades på analys av variationer i signalstyrka över tid. I detta arbete avses med enhetstyp distinktionen mellan mobila enheter, såsom mobiltelefoner och surfplattor, och icke-mobila enheter, såsom stationära datorer. Metoden utgår från att olika typer av enheter ofta uppvisar olika rörelsemönster i ett nätverk. Mobila enheter, som exempelvis mobiltelefoner, förflyttas ofta mellan olika positioner, vilket kan ge upphov till större variationer i signalstyrka. Stationära och bärbara datorer tenderar däremot att användas från mer statiska positioner och uppvisar därför vanligtvis mindre variation under samma tidsperiod. Metoden bygger på en generalisering och är därför inte fullt tillförlitlig. Bärbara datorer kan exempelvis vara i rörelse, samtidigt som mobiltelefoner kan vara stillastående under längre perioder. Syftet med metoden var därför inte att exakt identifiera en specifik enhetstyp, utan snarare att möjliggöra en grov klassificering baserad på enhetens rörelsemönster.

För att uppskatta rörelse analyserades signalstyrkan av inkommande paket över ett tidsfönster på 12 sekunder. Paketen delades in i grupper utifrån uppmätt signalstyrka, där mätvärden med liknande nivåer placerades tillsammans. En ny mätning inkluderades om den låg tillräckligt nära det aktuella medianvärdet för den befintliga samlingen av mätvärden. I implementationen användes en gräns på 3 dBm. Gränsen fastställdes empiriskt genom att mäta signalstyrkan hos stillastående enheter, där det observerades att mätvärdena för dessa enheter typiskt varierade inom ett intervall på ± 3 dBm kring en stabil nivå. Värdet valdes därför som den minsta gräns som tillät gruppering av mätvärden utan att naturliga fluktuationer hos stillastående enheter bröt upp grupperingen i onödan. Om skillnaden översteg denna gräns avslutades den aktuella gruppen och en ny påbörjades.

En sekvens betraktades som stabil när den innehöll minst tre mätvärden och en varaktighet på minst en sekund. Detta gjorde att kortvariga och tillfälliga avvikelser filtrerades bort. För varje stabil sekvens beräknades därefter medianen av sekvensens signalvärden. Medianen användes eftersom den gav ett robust mått på den signalnivå som mätningarna huvudsakligen koncentrerades kring.

Variationen i signalstyrka definierades som skillnaden mellan den högsta och lägsta stabila signalnivån inom tidsfönstret, enligt ekvation 3.1.

$$V = \max(c_1, c_2, \ldots, c_k) - \min(c_1, c_2, \ldots, c_k), \tag{3.1}$$

I ekvationen är V variationen i signalstyrka och $c_1, c_2, \ldots, c_k$ de stabila signalnivåerna.

Ett lågt värde på V innebar att signalstyrkan var relativt konstant, medan ett högre värde innebar större variation i signalnivån över tid. Enheten klassificerades som rörlig om signalstyrkevariationen V översteg 6 dBm, annars klassificerades enheten som stationär. Tröskelvärdet fastställdes genom empirisk optimering, där ett intervall av kandidatvärden utvärderades och träffsäkerheten mättes för varje värde.

Resultaten visade att träffsäkerheten ökade upp till 6 dBm och därefter minskade, varför detta värde valdes. Eftersom signalstyrkan kunde variera även för stationära enheter innebär valet av ett relativt högt tröskelvärde att systemet i osäkra fall hellre klassificerar en enhet som stationär än som rörlig, vilket minskar risken för falska rörelseindikationer.

Resultatet av klassificeringen presenteras både på sidan *Alla enheter* och sidan *Specifik enhet*. Signalstyrkevariationen V och enhetens klassificering som stationär eller rörlig visades i separata kolumner. Klassificeringen användes därefter som en indikator på möjlig enhetstyp.

För att testa identifieringen av enhetstyp användes fyra enheter: två mobila och två stationära. De mobila enheterna testades i två scenarion: stationärt, där enheten placerades still på ett bord, och i rörelse, där enheten bars med under förflyttning i rummet. De stationära enheterna testades enbart i det stationära scenariot, eftersom dessa per definition inte förflyttas. Varje test varade i 30 sekunder och upprepades 12 gånger per enhet och scenario. I de fall där klassificeringen växlade mellan stationär och i rörelse under testets gång baserades resultatet på det rörelsetillstånd som enheten klassificerades som under den största delen av testperioden.

3.6 Identifiering av användaraktiviteter och beteendemönster

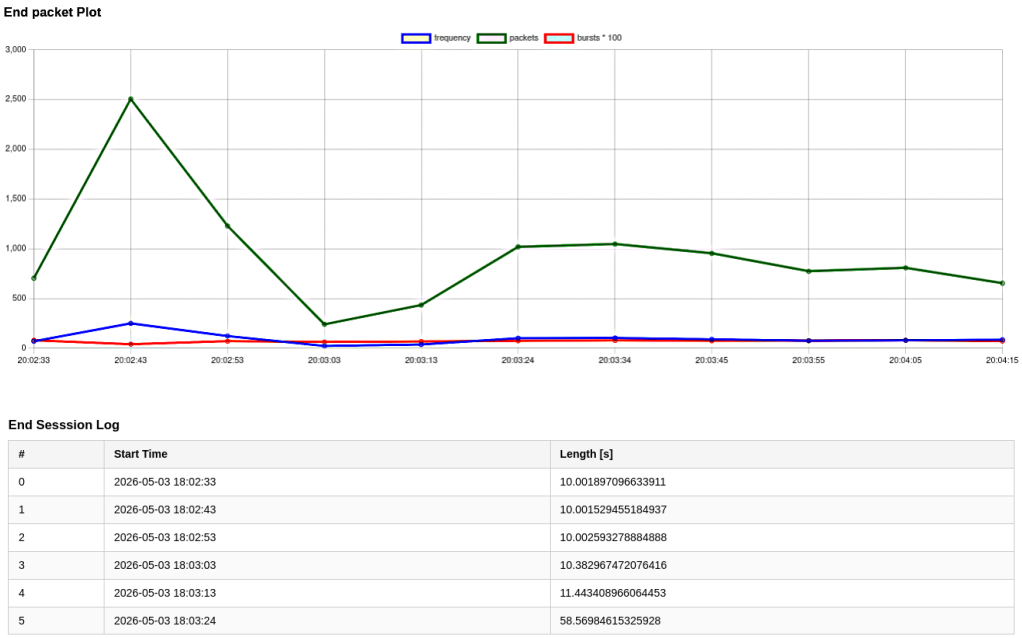
Syftet med denna del av arbetet var att undersöka om användaraktiviteter och beteendemönster kunde urskiljas genom passiv analys av nätverkstrafik över tid, med målet att identifiera när användaren byter aktivitet. För detta utvecklades funktioner som analyserade förändringar i trafikmönster med hjälp av frekvens, det vill säga antalet paket per sekund, samt ankomstojämnhet.

Systemet samlade in nätverkstrafik och analyserade paketens tidsstämplar tillsammans med tillhörande MAC-adresser. Trafiken delades först upp genom att identifiera längre avbrott i kommunikationen, exempelvis när enheter kopplade från nätverket eller var inaktiva under en längre period. Ett tidsintervall på en minut mellan två paket användes som gräns för att påbörja en ny gruppering av trafiken.

Därefter analyserades paketfrekvensen inom varje gruppering. Frekvens definierades som det genomsnittliga antalet paket per sekund beräknat över intervall om tio sekunder. Ett längre mätintervall användes för att minska påverkan från tillfälliga variationer och ankomstojämnheter i trafiken. Förändringar i frekvens användes sedan för att identifiera mer avgränsande trafiksegment inom de redan etablerade grupperingarna. Efter prototyptester fastställdes en förändring på 50% som gränsvärde för att identifiera ett nytt segment. Dessa segment benämndes sessioner.

Visualiseringen av användaraktiviteter består av en graf placerad i vyn *Specifik enhet*, där frekvens, ankomstojämnhet och antal paket visas över tid. Visualiseringen

innehåller även en tabell med identifierade sessioner tillsammans med tillhörande starttid och sessionsnummer, se figur 3.4. Grafen och tabellen uppdaterades automatiskt via `Socket.IO` efter varje mätning.



End Sesssion Log

#	Start Time	Length [s]
0	2026-05-03 18:02:33	10.001897096633911
1	2026-05-03 18:02:43	10.001529455184937
2	2026-05-03 18:02:53	10.002593278884888
3	2026-05-03 18:03:03	10.382967472076416
4	2026-05-03 18:03:13	11.443408966064453
5	2026-05-03 18:03:24	58.56984615325928

Figur 3.4: Visualiseringen av sessioner. Högst upp visas en graf med paket (grön), frekvens (blå) och ankomstojämnhet (röd) över tid. Värdena samlades in var tionde sekund. Nedanför visas en tabell med de uppmätta sessionerna under mätningen.

Funktionen för analys av beteendemönster bygger vidare på samma principer som användes vid identifiering av användaraktiviteter, men tillämpades istället på flera enheter samtidigt. Sessioner beräknades därför för samtliga enheter som var anslutna till nätverket och presenterades i en tabell tillsammans med tillhörande tidsinformation. Utöver detta visades även en graf över paketaktiviteten för enheterna över tid. Visualiseringen placerades i vyn *Alla enheter* och uppdaterades automatiskt via `Socket.IO` under mätningens gång.

Den ursprungliga tanken var att frekvens, frekvensförändringar och ankomstojämnhet kunde användas för att identifiera enskilda användaraktiviteter, där sessioner skulle markera övergångar mellan dessa. Om en användare exempelvis först läste en nyhetsartikel och därefter började titta på film förväntades detta ge upphov till separata sessioner med olika trafikmönster. Under prototyptester observerades dock att många aktiviteter genererade överlappande och varierande trafikmönster. Några tydligt avgränsade eller konsekvent unika mönster för enskilda aktiviteter kunde därför inte identifieras. Analysens fokus ändrades därför till att istället undersöka om aktiviteter kunde grupperas utifrån övergripande trafikmönster snarare än identifieras individuellt. Sessionerna behölls dock i verktyget som stöd för manuell analys samt visualisering av trafik över tid.

3.6.1 Testning och analys av användaraktiviteter

Testningen av användaraktiviteter inleddes med att kartlägga hur olika typer av användning påverkade nätverkstrafiken. Ett antal vanliga användaraktiviteter valdes ut för att undersöka om de gav upphov till återkommande trafikmönster som kunde urskiljas genom passiv trafikanalys. Följande aktiviteter testades:

- Uppspelning av film via Netflix
- Uppspelning av videor via YouTube
- Uppspelning av program via TV4 Play
- Direktsändning via SVT Play
- Bläddring i Instagram
- Videosamtal via WhatsApp
- Näthandling via Amazon
- Bläddring på Aftonbladet
- Läsning av forskningsartiklar via Wikipedia

Samtliga tester var cirka två minuter långa och genomfördes på en surfplatta med Android som operativsystem. Surfplattan var ansluten till ett trådlöst nätverk som skapades genom internetdelning från en mobiltelefon med Android.

Vid varje test startades den avsedda aktiviteten på surfplattan innan nätverkstrafiken började samlas in. Varje aktivitet testades vid tre separata tillfällen för att undersöka om liknande trafikmönster kunde observeras mellan mätningarna. Utifrån resultaten identifierades tre övergripande grupper av användaraktiviteter baserat på trafikens karaktär:

- **Grupp 1:** Videouppspelning, direktsändning och videosamtal.
- **Grupp 2:** Webbsurfning, näthandel och bläddring i sociala medier eller nyhetssidor.
- **Grupp 3:** Lågintensiv aktivitet, exempelvis läsning av forskningsartiklar eller redan nedladdat material, samt vistelse på statiska webbsidor utan reklam.

Grupperna definierades därefter utifrån observerade egenskaper i nätverkstrafiken:

- **Grupp 1:**
  - Paketantal: över 700 paket per 10 sekunder med relativt jämnt trafikflöde
  - Frekvens: stabil
  - Ankomstojämnhet: stabil
- **Grupp 2:**
  - Paketantal: cirka 100–2000 paket per 10 sekunder med stora variationer
  - Frekvens: varierande
  - Ankomstojämnhet: varierande
- **Grupp 3:**
  - Paketantal: under 200 paket per 10 sekunder
  - Frekvens: stabil
  - Ankomstojämnhet: varierande

För aktiviteter som innehöll videouppspelning, exempelvis Netflix, YouTube, SVT Play och TV4 Play, observerades variationer mellan de olika mätningarna på två minuter. Därför genomfördes även kompletterande tester med en varaktighet på sex minuter för dessa aktiviteter. Testupplägget var identiskt med tidigare mätningar och användes för att analysera hur trafikmönstren förändrades över längre tidsperioder. Resultaten från dessa tester användes dock inte som grund för gruppindelningen.

Efter den inledande kartläggningen genomfördes 27 ytterligare tester för att undersöka hur väl de identifierade grupperna kunde särskiljas. Under varje test utförde en person någon av aktiviteterna ovan på testenheten medan nätverkstrafiken samlades in under cirka två minuter. Därefter analyserade en annan person visualiseringen av nätverkstrafiken och försökte avgöra vilken aktivitetsgrupp som bäst motsvarade det observerade trafikmönstret. Utöver de tre grupperna kunde även alternativet '?' anges om inget mönster bedömdes passa in på trafiken.

Resultaten kompletterades med en kvalitativ analys av samtliga visualiseringar. Analysen fokuserade på att identifiera både likheter och skillnader mellan mätningar av samma aktivitet samt mellan olika aktiviteter. Även skillnader mellan två- och sexminutersmätningarna analyserades för att undersöka hur trafikmönstren förändrades över tid.

3.6.2 Testning och analys av beteendemönster

Testningen av beteendemönster syftade till att undersöka hur förändringar i aktivitet på nätverksnivå kunde visualiseras när flera enheter observerades samtidigt. För att simulera ett scenario motsvarande början och slutet av en arbetsdag anslöts tio enheter successivt till samma trådlösa nätverk under en period på två minuter. Enheterna förblev därefter anslutna under cirka tre minuter innan de successivt kopplades från nätverket under ytterligare två minuter.

Under testet samlades nätverkstrafik kontinuerligt in och visualiserades i form av en graf över paketaktivitet för samtliga enheter. Visualiseringen sparades därefter för vidare analys av hur förändringar i nätverksaktivitet kunde kopplas till användarnas beteendemönster på gruppnivå.

3.7 Utvecklingsprocess

Utvecklingen av analysverktyget bedrevs iterativt, där funktionalitet successivt byggdes ut genom mindre förbättringar och avgränsade implementationer snarare än i ett enda sammanhängande utvecklingssteg. Ett sådant arbetssätt var särskilt lämpligt i ett projekt där flera komponenter påverkade varandra, exempelvis paketinsamling, statistikberäkning och visualisering. Förändringar i en del av systemet kunde därmed införas gradvis och följas av anpassningar i övriga delar utan att omfattande omstruktureringar krävdes. Samtidigt möjliggjorde arbetssättet en kontinuerlig utvärdering av vilka funktioner som var relevanta för projektets syfte.

För att säkerställa en långsiktigt hållbar kodbas prioriterades tydlig strukturering, konsekvent namngivning och kommentarer där dessa bidrog till att förklara funktionalitet eller designval. Koden granskades kontinuerligt för att identifiera möjligheter till förenkling, uppdelning och omstrukturering. Detta var särskilt viktigt i takt med att ny funktionalitet successivt tillfördes och tidigare lösningar behövde anpassas till en mer etablerad systemstruktur.

För versionshantering och samarbete användes Git tillsammans med GitLab som gemensam plattform. Utvecklingen genomfördes i separata grenar där nya funktioner, förbättringar och korrigeringar utvecklades isolerat från den centrala kodbasen. Detta möjliggjorde parallell utveckling av flera delar av systemet utan att påverka övriga komponenter. Innan ny funktionalitet integrerades i huvudgrenen genomfördes kodgranskning av en gruppmedlem som inte deltagit i implementationen. Arbetssättet bidrog till att identifiera fel, diskutera alternativa lösningar och förbättra den övergripande kodkvaliteten.

3.8 Etiska överväganden i arbetet

Projektet bygger på insamling och analys av nätverkstrafik, vilket innebär att arbetet berör integritetsrelaterade och etiska frågor. Datainsamlingen begränsades därför till kontrollerade testmiljöer där endast gruppmedlemmarnas egna enheter eller enheter tillhörande personer som lämnat informerat samtycke användes.

Insamlingen genomfördes i isolerade nätverksmiljöer, exempelvis genom internetdelning från mobiltelefoner, för att minimera risken att extern trafik skulle inkluderas i analysen. Vid passiv avlyssning av trådlösa nätverk finns dock en risk att trafik från närliggande nätverk eller överlappande kanaler fångas upp. Sådan trafik inkluderades inte i den fortsatta bearbetningen och användes inte i analysen.

Analysen begränsades till observerbar metadata, exempelvis tidsstämplar, paketstorlekar och adresseringsrelaterad information. Ingen analys av nyttolast genomfördes och inga försök gjordes att dekryptera trafik eller kringgå krypteringsmekanismer.

Insamlad data hanterades enbart inom projektgruppen och användes endast för projektets syfte. Information som kunde kopplas till enskilda individer inkluderades inte i rapporten. Hanteringen av materialet utformades med hänsyn till principerna om dataminimering, samtycke och ändamålsbegränsning i enlighet med dataskyddsförordningen GDPR.

4
Resultat

I detta kapitel presenteras resultaten från de analyser och tester som genomfördes i projektet. Resultaten redovisas separat för respektive delområde och omfattar både kvantitativa mått och observationer från manuella tester.

4.1 Identifiering av applikation

Den tränade modellen uppnådde en träffsäkerhet på cirka 82%. Precisionen för varje applikation redovisas i tabell 4.1.

Tabell 4.1: Klassificeringsresultat för modellen baserat på träningsdata. Precisionen redovisas per applikation. Viktat medelvärde är medelvärdet där klasser viktas efter antal observationer.

Klass	**Precision**	**Antal**
Facebook	0,84	136
Instagram	0,84	173
LinkedIn	0,81	130
TikTok	0,86	122
YouTube	0,80	169
Viktat medelvärde	0,83	730

För applikationerna Instagram (IG) och Facebook (FB) uppnås relativt stabila resultat med låg felklassificering. För de resterande applikationerna, YouTube (YT), TikTok (TT) och LinkedIn (LI), observeras däremot omfattande felklassificeringar, där aktiviteterna ofta identifierades som Instagram. Facebook och Instagram förväxlades i två av tolv fall vardera, där den ena klassificerades som den andra. Resultaten från realtidstestningen av modellen visas i tabell 4.2.

Tabell 4.2: Resultat från realtidstester av modellens klassificering för olika applikationer. Faktisk applikation avser den applikation som användes, medan uppskattad applikation avser den klass som modellen predikterade med högst sannolikhet. Felklassificering definieras som andelen felaktiga klassificeringar.

Applikation	Aktivitet	**Sannolikhet för faktisk applikation (%)**	Sannolikhet för uppskattad applikation (%)	Felklassificering (%)
FB	Flöde	**40,5**	FB: 40,5	17
FB	'Reels'	**54,8**	FB: 54,8	17
IG	Flöde	**55,0**	IG: 55,0	17
IG	'Reels'	**51,8**	IG: 51,8	17
LI	Flöde	**12**	IG/FB: 52,8	100
TT	Flöde	**0**	IG: 58,5	100
YT	Video	**10,75**	IG: 65,0	100

4.2 Identifiering av operativsystem

I Metod A kunde operativsystemet identifieras korrekt för samtliga 27 testade enheter. De fyra kompletterande testerna visade dessutom att operativsystemet kunde identifieras korrekt både för enheter med och utan aktiv MAC-adresslumpning. Resultaten visade även att enhetens tillverkare kunde identifieras i de fall där OUI-prefixet var tillgängligt. En fullständig sammanställning av testerna presenteras i tabell B.1, medan kompletterande resultat redovisas i figur B.1. Metod B klassificerade ungefär hälften av Android- och iOS-enheterna korrekt.

4.3 Identifiering av enhetstyp

Resultatet visar att stationära enheter klassificerades mer tillförlitligt än rörliga enheter. Detta innebär att metoden var bättre på att undvika falska indikationer på rörelse än på att upptäcka samtliga fall där en enhet faktiskt var i rörelse, se tabell 4.3.

Tabell 4.3: Resultat från manuella tester av klassificering av rörelsetillstånd hos enheter. Korrekt identifierat avser andelen tester där verktyget lyckades klassificera enheten som rörlig eller stationär. De två efterföljande kolumnerna visar antalet tester som klassificerades som respektive rörelsetillstånd.

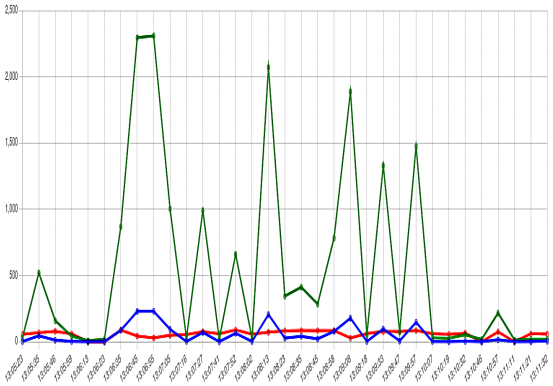
Grupp	**Korrekt identifierat (%)**	Klassificerad som i rörelse (antal)	Klassificerad som stationär (antal)
Mobil 1 i rörelse	**50**	6 av 12	6 av 12
Mobil 1 stillastående	**100**	0 av 12	12 av 12
Mobil 2 i rörelse	**58,3**	7 av 12	5 av 12
Mobil 2 stillastående	**100**	0 av 12	12 av 12
Stationär dator 1	**100**	0 av 12	12 av 12
Stationär dator 2	**100**	0 av 12	12 av 12

4.4 Identifiering av olika användaraktiviteter

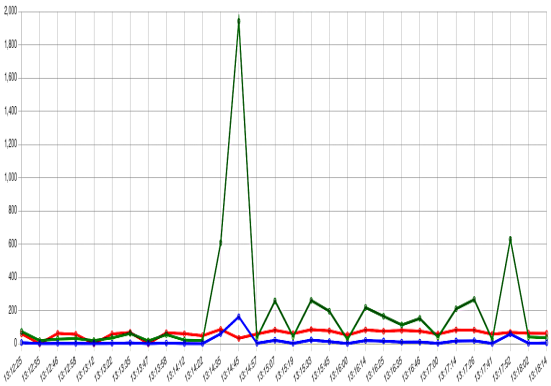
Resultaten från testerna av användaraktiviteter redovisas i tabell 4.4. Grupperna kunde i vissa fall urskiljas utifrån de observerade trafikmönstren, särskilt för grupp 1 och grupp 3. För sexminutersmätningarna observerades att filmuppspelning uppvisade betydande variationer i trafikflöde över tid, se figur 4.1. Detta skiljde sig från tvåminutersmätningarna som användes vid gruppindelningen. Andra användaraktiviteter, som videosamtal eller artikelläsning, uppvisade däremot tydligare och mer stabila trafikmönster, se figur 4.2.

Tabell 4.4: Resultat från tester av hur väl de definierade grupperna av användaraktiviteter kunde urskiljas utifrån observerade trafikmönster. Falska positiva avser fall där en grupp felaktigt identifierades, medan falska negativa avser fall där gruppen inte identifierades trots att den förekom.

Grupp	**Korrekt identifierat (%)**	Falska positiva (antal)	Falska negativa (antal)
1	**83,3**	4 av 15	1 av 12
2	**58,3**	1 av 15	5 av 12
3	**100**	0 av 24	0 av 3

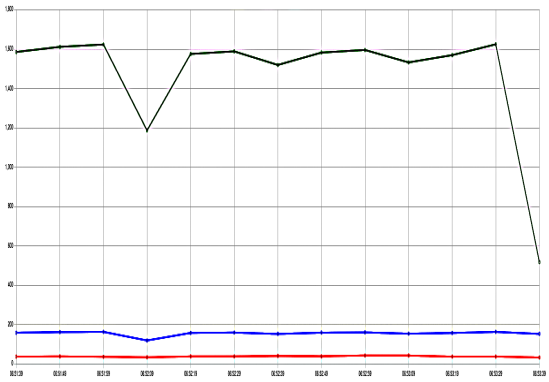


(a) Första mätningen med flera toppar i trafikflödet.

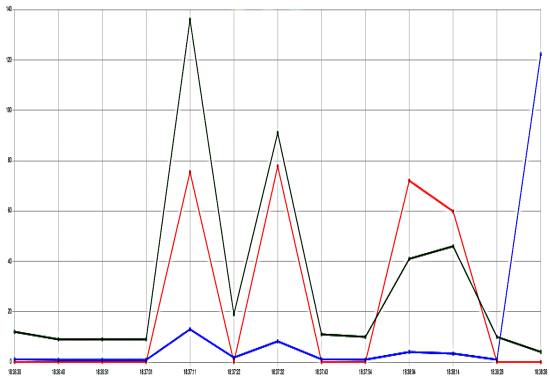


(b) Andra mätningen med en dominerande topp och i övrigt låg trafiknivå.

Figur 4.1: Trafikmönster under två sexminutersmätningar vid filmuppspelning på Netflix. Mätningarna visar variationer i trafikflödets mönster mellan olika mätningar. Grafen visar paket (grön), frekvens (blå) och ankomstojämnhet (röd) över tid.



(a) Trafikmönster vid videosamtal på WhatsApp med relativt stabil trafikintensitet.

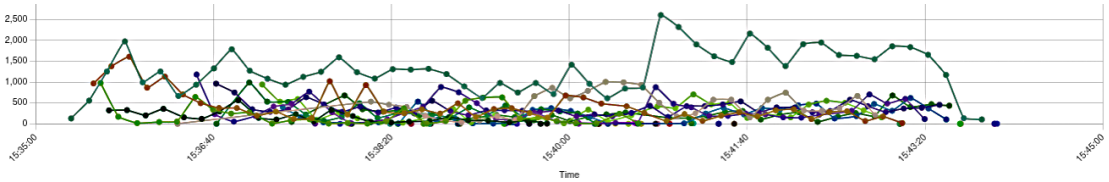


(b) Trafikmönster vid artikelläsning med låg trafikintensitet.

Figur 4.2: Trafikmönster vid videosamtal respektive artikelläsning. Aktiviteterna uppvisar relativt stabila trafikmönster jämfört med filmuppspelning. Grafen visar paket (grön), frekvens (blå) och ankomstojämnhet (röd) över tid.

4.5 Identifiering av beteendemönster

Resultaten från simuleringen av en arbetsdag redovisas i figur 4.3.



Figur 4.3: Trafikmönster för flera enheter under en längre tidsperiod. Variationer i nätverkstrafiken observeras när enheter ansluter till och kopplas från nätverket. Majoriteten av trafik från andra nätverk på samma kanal har filtrerats bort.

5
Diskussion

Detta kapitel diskuterar arbetets resultat i relation till projektets frågeställningar. Fokus ligger på att analysera resultatens tillförlitlighet, metodernas begränsningar och i vilken utsträckning resultaten kan generaliseras. Kapitlet behandlar även möjliga vidareutvecklingar samt etiska implikationer av den genomförda analysen.

5.1 Identifiering av applikation

Resultaten från identifiering av applikation visar att maskininlärningsmodellen har en begränsad förmåga att urskilja de testade applikationerna i realtidstesterna. Modellen uppvisar störst träffsäkerhet vid identifiering av Instagram, där den genomsnittliga träffsäkerheten är cirka 53%. Samtidigt förekommer en tendens att andra applikationer felklassificeras som Instagram, där modellen i ungefär 69% av fallen predikterar Instagram. Av dessa prediktioner är enbart 35% korrekta. Detta indikerar en systematisk överrepresentation av Instagram i modellens prediktioner, vilket tyder på att resultaten för Instagram delvis kan vara påverkade av denna överrepresentation snarare än enbart distinkta trafikmönster.

Facebook är den applikation som, förutom Instagram, kunde urskiljas med högst sannolikhet, med en träffsäkerhet på cirka 48%. Detta är högre än en slumpmässig klassificering, som hade varit 20% om Instagrams överrepresentation bortses från. De fall där Facebook felklassificerades, klassificerades det istället som Instagram, samtidigt som Instagram felklassificerades som Facebook i två av testerna. Detta indikerar att förväxlingen mellan applikationerna inte enbart kan förklaras av en överrepresentation av Instagram i modellens prediktioner.

En möjlig förklaring till den observerade förväxlingsgraden mellan klasserna är att Facebook och Instagram, båda utvecklade av Meta, använder liknande funktioner och medietyper, exempelvis 'Reels'. Detta kan leda till att applikationerna genererar liknande nätverkstrafikmönster, vilket försvårar modellens möjlighet att urskilja dem baserat på trafikbaserade attribut. Samtidigt visar den relativt höga träffsäkerheten för Facebook att det, trots dessa likheter, finns vissa distinkta egenskaper i nätverkstrafiken som modellen kan utnyttja för klassificering.

TikTok uppvisade låg identifieringsgrad. Detta kan delvis bero på att TikTok är den nyaste av de testade applikationerna och sannolikt har förändrats sedan datamängden samlades in. Förändringar i funktionalitet kan därmed ha lett till skillnader

i trafikmönster, vilket innebär att de aktuella observationerna inte fullt ut överensstämmer med de mönster som modellen tränats på. Samtidigt uppvisar även äldre applikationer såsom LinkedIn och YouTube låg identifieringsgrad, vilket indikerar att ålder i sig inte är en tillräcklig förklaring till modellens svårigheter.

Den tränade modellen uppnådde en relativt hög träffsäkerhet på 82% på träningsdatan, vilket indikerar att den kunde identifiera mönster i den datamängd den tränades på. Precisionen var även generellt hög för de enskilda applikationerna, vilket tyder på att modellen i många fall gjorde korrekta klassificeringar. Skillnaden mellan modellens prestanda på träningsdatan och resultaten från realtidstesterna tyder på att modellen inte fullt ut generaliserade till den nyinsamlade trafiken. En möjlig förklaring är att applikationernas trafikmönster skiljer sig från de förhållanden som fanns vid insamlingen av träningsdatan. Träningsdatan samlades in år 2023, medan realtidstesterna genomfördes tre år senare. Det är därmed rimligt att anta att förändringar i applikationernas trafikmönster har skett över tid, vilket i sin tur kan ha påverkat modellens förmåga att klassificera trafiken korrekt.

En annan faktor som kan påverka modellens klassificeringsförmåga är skillnader i insamlingsmiljö. Datamängden genererades med hjälp av två mobila enheter, medan testningen genomfördes på en tredje enhet. Detta kan introducera variationer i nätverkstrafikens egenskaper, exempelvis i hur data överförs och tidsmönster i kommunikationen. Bakgrundstrafik från enheten utgör dessutom en osäkerhetsfaktor, eftersom olika mobila enheter och operativsystem kan generera varierande nivåer och typer av bakgrundsaktivitet. Detta kan bidra med ytterligare brus i nätverkstrafiken och därmed påverka modellens möjlighet att urskilja applikationernas trafikmönster.

Vidare observerades under realtidstestning att klassificeringarna från modellen varierade beroende på användarbeteende, exempelvis bläddringshastighet. Bläddringshastigheten kan påverka mängden trafik som genereras, vilket i sin tur påverkar trafikmönstret och därmed klassificeringen. Detta visar att modellen inte enbart påverkas av vilken applikation som används, utan även av hur applikationen används. Eftersom användarscenarierna inte kunde kontrolleras fullt ut kan detta ha bidragit till variationen i resultaten.

Ytterligare en osäkerhetsfaktor är att både modellkonfigurationen och egenskaper relaterade till ankomstojämnhet är beroende av ett antal parametrar. Antalet och definitionen av detekterade trafiktoppar påverkas exempelvis av valda tröskelvärden och beräkningsinställningar, vilket innebär att modellens resultat delvis påverkas av hur dessa egenskaper definieras. Även själva maskininlärningsmodellen innehåller flera parametrar, som antal träd i Random Forest-modellen samt hur djupt varje beslutsträd tillåts växa. Dessa val påverkar modellens komplexitet och därmed dess förmåga att generalisera från träningsdatan.

Trots dessa variationer och osäkerhetsfaktorer visar resultaten att modellen i vissa fall, exempelvis för Facebook, uppnår en precision högre än en slumpmässig nivå. Detta indikerar att det finns strukturer i datan som kan utnyttjas för klassificering,

även under varierande förhållanden.

5.2 Identifiering av operativsystem

Resultaten visar att metod A korrekt identifierade alla de testade enheternas operativsystem. Detta tyder på att mönster i informationselementsekvenser innehåller användbar information för att skilja mellan iOS- och Android-baserade enheter. Resultatet är särskilt intressant eftersom identifieringen även fungerade för enheter med slumpmässigt genererade MAC-adresser, där OUI-baserad identifiering är begränsad.

Samtidigt finns det begränsningar som påverkar resultatens generaliserbarhet. En sådan begränsning är att testningen, trots att samtliga enheter identifierades korrekt, endast omfattade 27 enheter från ett begränsat antal tillverkare och modeller. Detta innebär att det inte kan garanteras att samma mönster kan hittas hos andra, framtida eller äldre modeller. Samtidigt tyder det faktum att identifieringen fungerade för äldre modeller i testningen på att mönstren inte enbart är begränsade till de senaste versionerna.

En ytterligare osäkerhetsfaktor är att informationselementsekvenser inte är standardiserade för identifiering av operativsystem och kan därmed förändras över tid. Förändringar i tillverkarens implementationer eller framtida integritetsfunktioner skulle kunna påverka metodens tillförlitlighet och minska möjligheten att identifiera operativsystem genom denna typ av metadataanalys.

Identifiering av tillverkare baserat på MAC-adresser visade sig även ha begränsningar. Metoden bygger på OUI-prefix och en databas över kända adressintervall, vilket innebär att identifieringen blir osäker när adressen är slumpmässigt genererad eller när databasen saknar information. Tillverkaridentifieringen fungerade därför främst som ett komplement till informationselements-sekvensanalysen snarare än som en självständig identifieringsmetod.

Metod B syftade till att undersöka om maskininlärning kan användas för att identifiera operativsystem utifrån nätverkstrafik. Modellen visade en viss förmåga att klassificera Android och iOS korrekt, då mer än hälften av enheterna identifierades rätt för båda klasserna. Jämfört med Metod A, som identifierade operativsystem med fullständig träffsäkerhet, var prestandan för Metod B betydligt lägre. Därför valdes Metod A som grund för systemet, och Metod B uteslöts från vidare användning och analys.

Den största begränsningen var tillgången till en lämplig datamängd. Den använda datamängden saknade tillräckligt relevant information, då endast tidsmönster i trafiken användes som underlag. Detta begränsade modellens förmåga att urskilja operativsystem på ett tillförlitligt sätt. Resultaten innebär dock inte att maskininlärning är olämpligt för uppgiften, utan snarare att bättre och mer detaljerad data sannolikt krävs för ett bättre resultat.

5.3 Identifiering av enhetstyp

Resultaten visar att variationer i signalstyrka kan användas för att uppskatta en enhets rörelsetillstånd och därigenom ge en begränsad indikation på enhetstyp. Vid identifiering av stationära enheter uppnådde verktyget 100% träffsäkerhet. För de mobila enheterna i rörelse var träffsäkerheten däremot lägre, där ungefär hälften av fallen identifierades korrekt. Resultaten tyder därmed på att metoden i större utsträckning kan bekräfta att en enhet är stillastående, men ger inte ett tillförlitligt sätt avgöra om en enhet är i rörelse.

Den lägre träffsäkerheten för mobila enheter kan delvis förklaras av det konservativa tröskelvärde som användes för signalvariation. Detta kan kopplas till metodens utformning, där tröskelvärdet valdes för att minska risken att stationära enheter med naturliga signalvariationer felaktigt skulle klassificeras som rörliga. Samtidigt innebar detta att mobila enheter behövde uppvisa tydliga och stabila förändringar i signalstyrka för att klassificeras som rörliga. Den lägre träffsäkerheten för mobila enheter kan därför förstås som en konsekvens av metodens avvägning mellan att minska falska positiva klassificeringar och att upptäcka faktisk rörelse.

Det är även viktigt att notera att signalstyrkan påverkas av faktorer utöver fysisk rörelse, exempelvis hinder i miljö och avstånd till mottagaren. Variationer i signalstyrka motsvarar därför inte nödvändigtvis förändringar i enhetens position. Testerna genomfördes även under gynnsamma och kontrollerade förhållanden, där avstånd och hinder inte utgjorde ett problem. I mer verklighetstrogna nätverksmiljöer är det sannolikt att träffsäkerheten hade varit lägre. Sammantaget innebär detta att enhetstyp inte kan identifieras tillförlitligt utifrån enbart signalstyrkevariation. Verktyget ger en indikation om rörelsetillstånd, men uppnår inte tillräcklig träffsäkerhet för att utgöra ett självständigt identifieringsverktyg.

5.4 Identifiering av användaraktiviteter

Resultaten visar att användaraktiviteter till viss del kan identifieras genom analys av nätverkstrafik, men att träffsäkerheten är begränsad. Vissa aktiviteter, exempelvis videosamtal, genererade tydliga mönster med ett jämnt och högt paketflöde, vilket gjorde dem enkla att urskilja. Andra aktiviteter uppvisade däremot liknande trafikmönster. Trafik från näthandel kunde exempelvis likna trafik från sociala medier eller nyhetssidor, vilket försvårade klassificeringen.

Filmuppspelning visade sig vara särskilt svår att urskilja. Trafikmönstren varierade kraftigt mellan olika mätningar och kunde därför förväxlas med både lågintensiva aktiviteter, som läsning av artiklar, och mer datakrävande aktiviteter, som videosamtal eller TV-tittande. Att aktiviteterna grupperades i relativt övergripande kategorier förbättrade identifieringen till viss del, men även dessa grupper var svåra att särskilja konsekvent. Det är även sannolikt att andra, inte testade, aktiviteter kan ge upphov till liknande trafikmönster, vilket utgör ytterligare en osäkerhet.

Flera faktorer bidrog till svårigheterna med att identifiera användaraktiviteter genom trafikmönster. Nätverkskvaliteten påverkar exempelvis trafikflödet, där sämre uppkoppling kan skapa variation även för aktiviteter som normalt uppvisar stabila trafikmönster. Även enhetstyp och bakgrundsaktivitet påverkar resultaten, eftersom olika enheter genererar olika kommunikationsmönster och olika mängder bakgrundstrafik. I de genomförda testerna isolerades den studerade aktiviteten, men i verkliga scenarier förekommer ofta parallella aktiviteter och processer, vilket sannolikt gör identifieringen ännu svårare.

Användning av maskininlärningsmodell för att klassificera användaraktiviteter baserat på trafikflöde skulle potentiellt kunna förbättra identifieringen. Detta hade kunnat möjliggöra upptäckten av mönster som annars är svåra att hitta. Samtidigt kvarstår utmaningar som överlappande trafikmönster och variation mellan olika användarscenarier. Eftersom endast ett begränsat antal aktiviteter undersöktes i projektet hade en mer generell modell dessutom krävt betydligt större och mer representativa datamängder. Att detta begränsade urval av användaraktiviteter redan är svåra att särskilja indikerar att problemet är komplext.

Sammantaget tyder resultaten på att användaraktiviteter i vissa fall kan uppskattas genom analys av observerbar nätverkstrafik, men att en tillförlitlig och generell identifiering är svår att uppnå. Även med mer avancerade metoder är det sannolikt att variationer mellan användare, enheter och nätverksförhållanden fortsatt skulle begränsa träffsäkerheten.

5.5 Identifiering av beteendemönster

Resultaten för identifiering av beteendemönster visar att förändringar i nätverkstrafik kan användas för att uppskatta övergripande tidsscheman. Återkommande förändringar i trafikvolym kan exempelvis indikera när flera användare ansluter till eller lämnar nätverket under liknande tidsperioder. Om trafikflödet på ett kontor är lågt fram till en viss tidpunkt då många enheter ansluter, kan detta indikera början på en arbetsdag. Liknande mönster skulle även kunna observeras i hushållsmiljöer där nätverksaktiviteten förändras markant vid vissa tider på dygnet. Informationen skulle potentiellt kunna användas för att uppskatta när användare befinner sig hemma eller på en arbetsplats. Återkommande trafikmönster skulle därmed ge insikt i användares rutiner och närvaro över tid, vilket kan leda till profilering och intrång i användarnas personliga integritet.

En viktig aspekt att beakta är att olika nätverk kan dela samma trådlösa kanal. Den observerade trafiken behöver därför inte enbart tillhöra det nätverk som analyseras, utan kan även innehålla trafik från närliggande nätverk. Detta försvårar möjligheten att koppla observerade trafikmönster till ett specifikt nätverk eller en specifik grupp av användare. Stora mängder överlappande trafik kan dessutom göra generella mönster svårare att urskilja och i vissa fall dölja dem helt. Följden blir att denna typ av analys är starkt beroende av gynnsamma miljöförhållanden i miljö,

och är därför svår att generalisera till andra, mindre isolerade nätverksmiljöer.

5.6 Svårigheter med trafikavlyssning

Trafikavlyssning garanterar inte att alla paket fångas upp. Antalet paket som registreras kan bland annat påverkas av nätverkskortets egenskaper, dess fysiska placering och störningar i den omgivande miljön. Den insamlade datan representerar därför endast den del av nätverkstrafiken som kunde registreras under de givna testförhållandena, vilket innebär att den inte nödvändigtvis motsvarar den fullständiga trafiken i nätverket.

Även stabiliteten i nätverksuppkopplingen kan påverka analysresultaten. En svag eller instabil uppkoppling kan leda till fördröjda eller förlorade paket, vilket i sin tur påverkar både tidsmässiga och volymbaserade egenskaper i den observerade trafiken. Detta innebär att de extraherade trafikmönstren kan avvika från de faktiska kommunikationsmönstren som genereras.

Tillförlitligheten i resultaten är därmed starkt kopplad till den kontrollerade testmiljön där systemet utvärderades. Inom denna avgränsade miljö kunde metoden ge en representativ bild av observerbara mönster och skillnader mellan enheter. Resultaten bör dock tolkas med försiktighet vid generalisering till mer komplexa eller störningspåverkade miljöer.

5.7 Etiska implikationer

Arbetet belyser flera etiska frågor kopplade till analys av observerad nätverkstrafik. De metoder som undersökts kan, under de givna testförhållandena, användas för att identifiera applikationer, operativsystem, användaraktiviteter och beteendemönster utan åtkomst till trafikens innehåll. Alla analyser har genomförts passivt, vilket innebär att analysen kan göras utan den observerades medvetenhet. Detta medför stora etiska risker.

De etiska riskerna skiljer sig mellan projektets olika delområden. Identifiering av operativsystem och enhetstyp kan framstå som relativt begränsad information, men kan användas för att urskilja specifika enheter eller anpassa riktade angrepp. Identifiering av applikationer och användaraktiviteter är mer känslig, eftersom sådan information kan ge insikt i användares vanor, intressen och kommunikationsmönster. Analys av beteendemönster över tid kan dessutom avslöja när användare är aktiva, när flera enheter ansluter samtidigt och potentiellt när personer befinner sig på en viss plats.

Arbetet visar därmed att kryptering av trafikens innehåll inte nödvändigtvis innebär att användaren är skyddad mot all form av insyn. Detta motiverar studien ur ett säkerhets- och medvetenhetsperspektiv, eftersom resultaten synliggör vilka risker som kvarstår även när nyttolasten inte analyseras. Samtidigt innebär metoderna

en risk för felanvändning, eftersom samma kunskap kan användas för övervakning, profilering eller kartläggning av användare.

Det finns även en avvägning mellan realistiska testförhållanden och etisk försiktighet. En mer realistisk testmiljö, med fler användare och spontana aktivitetsmönster, hade kunnat ge mer generaliserbara resultat. Samtidigt hade en sådan insamling ökat risken för att trafik från utomstående användare eller integritetskänsliga beteendemönster exponerades. Den valda metoden innebär därför en kompromiss, där naturliga användningsförhållanden delvis har begränsats för att uppnå bättre kontroll över datainsamlingen och skapa en mer etiskt försvarbar testmiljö.

5.8 Vidareutveckling

En möjlig vidareutveckling är att förbättra identifieringen av applikationer. Den nuvarande implementationen fungerar främst som en prototyp och kan identifiera vissa trafikmönster, men med begränsad precision. För att förbättra resultaten skulle en ny och mer aktuell datamängd behöva samlas in, så att den trafik maskininlärningsmodellen tränas på bättre representerar applikationers nätverkstrafik. Ytterligare förbättringar skulle även kunna uppnås genom användning av mer avancerade modeller som tar hänsyn till fler egenskaper hos nätverkstrafiken samt inkluderar ett större antal applikationer. Ett långsiktigt mål skulle därmed kunna vara att med högre tillförlitlighet identifiera fler av de mest använda applikationerna baserat på observerad nätverkstrafik.

Även analysen av enhetstyp hade kunnat göras betydligt mer omfattande. I nuläget används en väldigt primitiv metod som ger mycket begränsad information om vilken typ av enhet som faktiskt observeras. Resultaten ger i praktiken endast en grov indikation, exempelvis om det sannolikt rör sig om en enhetstyp som tenderar att vara i rörelse, som en mobiltelefon, eller enhetstyp som oftare är stationär. Metoden erbjuder därmed inga säkra klassificeringar, utan endast osäkra och vaga uppskattningar. Detta innebär att området har stor utvecklingspotential och en möjlig förbättring skulle kunna vara att utnyttja mer avancerade mönster i nätverkstrafiken. Maskininlärning skulle exempelvis kunna användas för att identifiera gemensamma egenskaper i trafiken för specifika enhetstyper. Detta skulle kunna möjliggöra en mer tillförlitlig klassificering, där subtila skillnader mellan enheter fångas upp på ett mer effektivt sätt och därmed förbättrar resultatens noggrannhet.

I samband med enhetstyp är positionering ett område med utvecklingspotential. I nuläget används variationer i signalstyrka för att indikera rörelse, vilket endast ger en grov uppskattning av om en enhet förflyttar sig. Genom att istället uppskatta enheters position i den fysiska miljön hade mer avancerad analys av rörelsemönster och användarbeteenden kunnat genomföras.

En betydande begränsning i analysverktyget är MAC-adresslumpning, vilket försvårar spårning av enheter över tid. Vid analys av användaraktivitet från samma enhet över olika nätverksuppkopplingar blir det exempelvis svårt att koppla observa-

tionerna till samma enhet, eftersom MAC-adressen inte är konstant. Att undersöka metoder för att kringgå eller delvis hantera denna begränsning, exempelvis genom att analysera andra typer av identifierande metadata, utgör därför ett intressant område för framtida forskning. Samtidigt är detta en funktion som är avsedd att skydda användares integritet, vilket innebär att sådana analyser måste genomföras med beaktande av etiska aspekter.

6
Slutsats

I detta projekt undersöktes vilken information om enheter och användarbeteenden som kan erhållas genom passiv analys av krypterad WiFi-trafik. Resultaten visade att det i vissa fall var möjligt att identifiera vilken applikation som genererade observerad nätverkstrafik med hjälp av en maskininlärningsmodell. Modellen uppnådde då en träffsäkerhet på omkring 50%. För de fall där identifieringen var mindre träffsäker bedömdes resultaten kunna förbättras genom användning av en mer aktuell datamängd. Vidare visade analysen att operativsystem kunde identifieras med hög träffsäkerhet genom mönstermatchning baserad på IE-sekvenser i nätverkets sökförfrågningar, även när MAC-adresslumpning var aktiverad.

Identifiering av enhetstyp utifrån variationer i signalstyrka visade sig vara mer problematisk. Stationära enheter kunde identifieras med relativt hög träffsäkerhet, medan enheter i rörelse endast klassificerades korrekt i ungefär hälften av fallen. Resultaten bedöms dessutom vara beroende av den kontrollerade testmiljön, vilket innebär att träffsäkerheten sannolikt skulle vara lägre i mer varierande verkliga miljöer. Analysen av användaraktiviteter visade att aktiviteter i viss utsträckning kunde kategoriseras, men med begränsad träffsäkerhet. Mer avancerade analysmetoder skulle sannolikt kunna förbättra resultaten, även om noggrannheten fortsatt kan förväntas vara begränsad. Samtidigt visade resultaten att det i högre grad var möjligt att identifiera återkommande beteendemönster och tidpunkter för aktivitet.

Sammanfattningsvis indikerar projektet att passiv analys av krypterad WiFi-trafik kan ge viss insikt i både enheter och användarbeteenden, men att tillförlitlig identifiering av mer detaljerad eller känslig information ofta kräver mer avancerade metoder och bättre datagrund.

Källförteckning

[1] Mauro Conti m. fl. "The Dark Side(-Channel) of Mobile Devices: A Survey on Network Traffic Analysis". I: *IEEE Communications Surveys Tutorials* 20.4 (2018), s. 2658–2713. DOI: 10.1109/COMST.2018.2843533.

[2] Manish Bhurtel och Danda B Rawat. "Unveiling the Landscape of Operating System Vulnerabilities". I: *Future Internet* 15 (juli 2023), s. 248. DOI: 10.3390/fi15070248.

[3] Agencia Española de Protección de Datos (AEPD). *Survey on Device Fingerprinting*. 2019. URL: https://www.aepd.es/guides/survey-on-device-fingerprinting.pdf.

[4] Noah Apthorpe, Dillon Reisman och Nick Feamster. *A Smart Home is No Castle: Privacy Vulnerabilities of Encrypted IoT Traffic*. 2017. DOI: 10.48550/arXiv.1705.06805.

[5] *Lag (2022:482) om elektronisk kommunikation*. Riksdagen. 26 maj 2022. URL: https://www.riksdagen.se/sv/dokument-och-lagar/dokument/svensk-forfattningssamling/lag-2022482-om-elektronisk-kommunikation_sfs-2022-482/ (hämtad 2026-02-10).

[6] Europaparlamentet och Europeiska unionens råd. *Förordning (EU) 2016/679 av den 27 april 2016 om skydd för fysiska personer med avseende på behandling av personuppgifter och om det fria flödet av sådana uppgifter och om upphävande av direktiv 95/46/EG (allmän dataskyddsförordning)*. 27 april 2016. URL: https://eur-lex.europa.eu/eli/reg/2016/679/oj?locale=SV (hämtad 2026-02-13).

[7] "IEEE Standard for Information Technology–Telecommunications and Information Exchange between Systems Local and Metropolitan Area Networks–Specific Requirements Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications". I: *IEEE Std 802.11-2024 (Revision of IEEE Std 802.11-2020)* (2025), s. 1–5956. DOI: 10.1109/IEEESTD.2025.10979691.

[8] IEEE. *IEEE – The World's Largest Technical Professional Organization*. https://www.ieee.org/. Hämtad: 2026-05-12.

[9] DIGINTO. *Trådlöst nätverk*. u.å. URL: https://dator-natverksteknik.diginto.se/natverks-grunder/tradlost-natverk/ (hämtad 2026-03-30).

[10] HowStuffWorks. *How Wireless Networking Works*. 2024. URL: https://computer.howstuffworks.com/wireless-network.htm (hämtad 2026-03-30).

[11] ISO/IEC. *ISO/IEC 7498-1: Information technology – Open Systems Interconnection – Basic Reference Model*. https://www.iso.org/standard/20269.html. Hämtad: 2026-05-10. 1994.

[12] *Internet Protocol*. RFC 791. Sept. 1981. DOI: 10.17487/RFC0791. URL: https://www.rfc-editor.org/info/rfc791.

[13] Robert Moskowitz m. fl. *Address Allocation for Private Internets*. RFC 1918. Febr. 1996. DOI: 10.17487/RFC1918. URL: https://www.rfc-editor.org/info/rfc1918.

[14] Kjeld Borch Egevang och Pyda Srisuresh. *Traditional IP Network Address Translator (Traditional NAT)*. RFC 3022. Jan. 2001. DOI: 10.17487/RFC3022. URL: https://www.rfc-editor.org/info/rfc3022.

[15] "IEEE Standard for Local and Metropolitan Area Networks: Overview and Architecture". I: *IEEE Std 802-2024 (Revision of IEEE Std 802-2014 as amended by IEEE Std 802d-2017, IEEE Std 802c-2017, and IEEE Std 802f-2023)* (2025), s. 1–99. DOI: 10.1109/IEEESTD.2025.10935844.

[16] Android Open Source Project. *MAC Randomization Behavior*. 2025. URL: https://source.android.com/docs/core/connect/wifi-mac-randomization-behavior (hämtad 2026-02-09).

[17] Adrienne Porter Felt m. fl. "Measuring HTTPS Adoption on the Web". I: *26th USENIX Security Symposium (USENIX Security 17)*. Vancouver, BC, Kanada: USENIX Association, aug. 2017, s. 1323–1338. ISBN: 978-1-931971-40-9. URL: https://www.usenix.org/conference/usenixsecurity17/technical-sessions/presentation/felt.

[18] Eric Rescorla. *The Transport Layer Security (TLS) Protocol Version 1.3*. RFC 8446. Aug. 2018. DOI: 10.17487/RFC8446.

[19] Charles Wright, Fabian Monrose och Gerald Masson. "On Inferring Application Protocol Behaviors in Encrypted Network Traffic." I: *Journal of Machine Learning Research* 6 (dec. 2006), s. 2745–2769. URL: https://dl.acm.org/doi/10.5555/1248547.1248647.

[20] Lucia Pintor och Luigi Atzori. "Analysis of Wi-Fi Probe Requests Towards Information Element Fingerprinting". I: *GLOBECOM 2022 - 2022 IEEE Global Communications Conference*. 2022, s. 3857–3862. DOI: 10.1109/GLOBECOM48099.2022.10001618.

[21] Luiz Oliveira m. fl. "Mobile Device Detection Through WiFi Probe Request Analysis". I: *IEEE Access* 7 (2019), s. 98579–98588. DOI: 10.1109/ACCESS.2019.2925406.

[22] Robert C. Martin. *Clean Architecture: A Craftsman's Guide to Software Structure and Design*. 1st. USA: Prentice Hall Press, 2017. ISBN: 0134494164.

[23] Erich Gamma m. fl. *Design Patterns: Elements of Reusable Object-Oriented Software*. Reading, MA: Addison-Wesley Professional, 1994, s. 293–304. ISBN: 0-201-63361-2.

[24] Matthias Schulz, Daniel Wegemer och Matthias Hollick. "NexMon: A Cookbook for Firmware Modifications on Smartphones to Enable Monitor Mode". I: *arXiv preprint arXiv:1601.07077* (2016). URL: https://arxiv.org/abs/1601.07077.

[25] Aircrack-ng. *Airmon-ng*. 2026. URL: https://www.aircrack-ng.org/doku.php?id=airmon-ng (hämtad 2026-05-18).

[26] Scapy Project. *Scapy*. 2026. URL: https://scapy.net/ (hämtad 2026-02-13).

[27] The Wireshark Team. *Wireshark*. 2026. URL: https://www.wireshark.org/ (hämtad 2026-02-13).
[28] Python Package Index. *mac-vendor-lookup – PyPI*. 2026. URL: https://pypi.org/project/mac-vendor-lookup/ (hämtad 2026-04-30).
[29] Pallets Projects. *Flask — web development, one drop at a time*. 2026. URL: https://flask.palletsprojects.com/en/stable/ (hämtad 2026-04-01).
[30] WHATWG. *WebSockets Standard*. u.å. URL: https://websockets.spec.whatwg.org/ (hämtad 2026-04-03).
[31] Socket.IO. *Socket.IO — Real-time application framework*. u.å. URL: https://socket.io/ (hämtad 2026-04-03).
[32] Chart.js. *Chart.js — Open source HTML5 charts for your website*. u.å. URL: https://www.chartjs.org/ (hämtad 2026-04-03).
[33] *Git — Distributed Version Control System*. Git SCM, 2026. URL: https://git-scm.com/ (hämtad 2026-02-13).
[34] GitLab Inc. *GitLab*. Hämtad: 2026-05-12. 2026. URL: https://gitlab.com.
[35] Issam El Naqa och Martin J. Murphy. "What Is Machine Learning?" I: *Machine Learning in Radiation Oncology*. Springer, 2015, s. 3–11. DOI: 10.1007/978-3-319-18305-3_1.
[36] Samina Khalid, Tehmina Khalil och Shamila Nasreen. "A survey of feature selection and feature extraction techniques in machine learning". I: *2014 Science and Information Conference*. 2014, s. 372–378. DOI: 10.1109/SAI.2014.6918213.
[37] Christopher M. Bishop. *Pattern Recognition and Machine Learning*. New York: Springer, 2006. ISBN: 978-0-387-31073-2.
[38] scikit-learn developers. *scikit-learn: Machine Learning in Python*. u.å. URL: https://scikit-learn.org/stable/ (hämtad 2026-04-29).
[39] Gérard Biau och Erwan Scornet. "A random forest guided tour". I: *TEST* 25.2 (juni 2016), s. 197–227. DOI: 10.1007/s11749-016-0481-7.
[40] Kevin P. Balanda och H. L. MacGillivray. "Kurtosis: A Critical Review". I: *The American Statistician* 42.2 (1988), s. 111–119. DOI: 10.1080/00031305.1988.10475539.
[41] Jon Kleinberg. "Bursty and hierarchical structure in streams". I: *Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. KDD '02. Edmonton, Alberta, Canada: Association for Computing Machinery, 2002, s. 91–101. ISBN: 158113567X. DOI: 10.1145/775047.775061. URL: https://doi.org/10.1145/775047.775061.
[42] PyPI. *burst-detection: Jon Kleinberg's Burst Detection Algorithm*. https://pypi.org/project/burst-detection/. Hämtad: 2026-05-18. 2024.
[43] José Armando Tesén Marañón och Romaric Duvignau. "Classifying 5G Encrypted Packet Traces". I: *2024 International Conference on Electrical, Communication and Computer Engineering (ICECCE)*. 2024, s. 1–6. DOI: 10.1109/ICECCE63537.2024.10823417.
[44] Pieter Robyns m. fl. "Noncooperative 802.11 mac layer fingerprinting and tracking of mobile devices". I: *Security and Communication Networks* 2017.1 (2017), s. 6235484.

A

Inledning

A.1 Enkät till förstudien

Länk till formuläret och dess svar: *Formuläret*

B

Resultat för Metod A

Fullständig resultattabell för Metod A inklusive modell på samtliga testenheter visas i tabell B.1.

Tabell B.1: Resultaten från testerna för enheter som använder iOS och Android som operativsystem presenteras i tabellen. Tabellen visar information om enheternas namn, operativsystem, systemversion samt om identifieringen av operativsystemet är korrekt. Samtliga testade enheter har MAC-adressslumpning aktiverad.

Enhet	**OS**	**Version**	**OS korrekt**
OnePlus 9 Pro	Android	14	Ja
iPhone 13	iOS	26.3.1	Ja
OnePlus Open	Android	16	Ja
iPhone 13	iOS	26.3.1	Ja
iPhone 15	iOS	26.3.1	Ja
iPhone 15 Plus	iOS	26.3.1	Ja
iPhone 13	iOS	26.2.1	Ja
iPhone 12	iOS	26.3.1	Ja
iPhone 14	iOS	26.3.1	Ja
iPhone 13	iOS	26.3	Ja
Samsung Galaxy Z Flip 3	Android	15	Ja
Samsung Galaxy Tab S9 FE Plus	Android	15	Ja
iPhone 16 Pro	iOS	26.3.1	Ja
Google Pixel 8	Android	16	Ja
iPhone 15	iOS	26.4	Ja
iPhone 13 Pro Max	iOS	26.2.1	Ja
iPhone 11	iOS	16.2	Ja
Samsung Galaxy Note 9	Android	10	Ja
Honor 200 Pro	Android	16	Ja
Samsung Galaxy A33	Android	16	Ja
iPhone 12	iOS	26.3.1	Ja
iPhone 15 Pro	iOS	18.3	Ja
iPhone 12 Pro	iOS	18.6.2	Ja
Samsung Galaxy S23 Plus	Android	16	Ja
iPhone SE	iOS	26.4.1	Ja
Samsung Galaxy Tab S7 FE	Android	14	Ja
Samsung Galaxy Tab S9 FE Plus	Android	15	Ja

Testresultatet för att se om MAC-slumpning påverkar Metod A finns i figur B.1.

26:[illegible]	-	iOS
e0:[illegible]	Apple, Inc.	iOS

(a) iOS

64:[illegible]	Samsung Electronics Co.,Ltd	Android
fa:[illegible]	-	Android

(b) Android

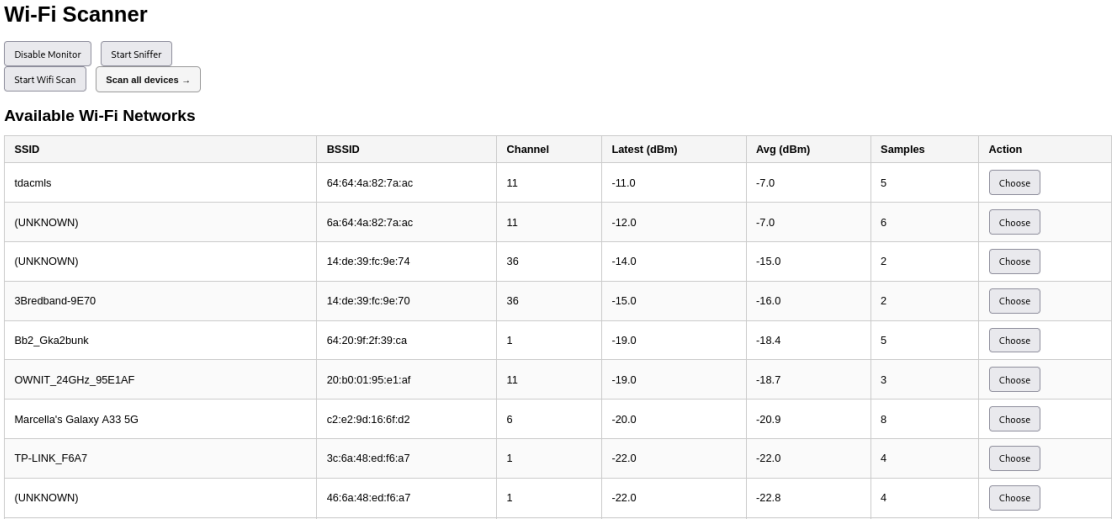
Figur B.1: Resultaten från testerna för enheter som använder iOS respektive Android visas i figuren. Endast den första tecknen av MAC-adressen visas, vilket används för att avgöra om MAC-adressslumpning är aktiverad. Resterande delar av MAC-adressen är dolda av integritetsskäl.

C

Analysverktyget

C.1 WiFi-skanner

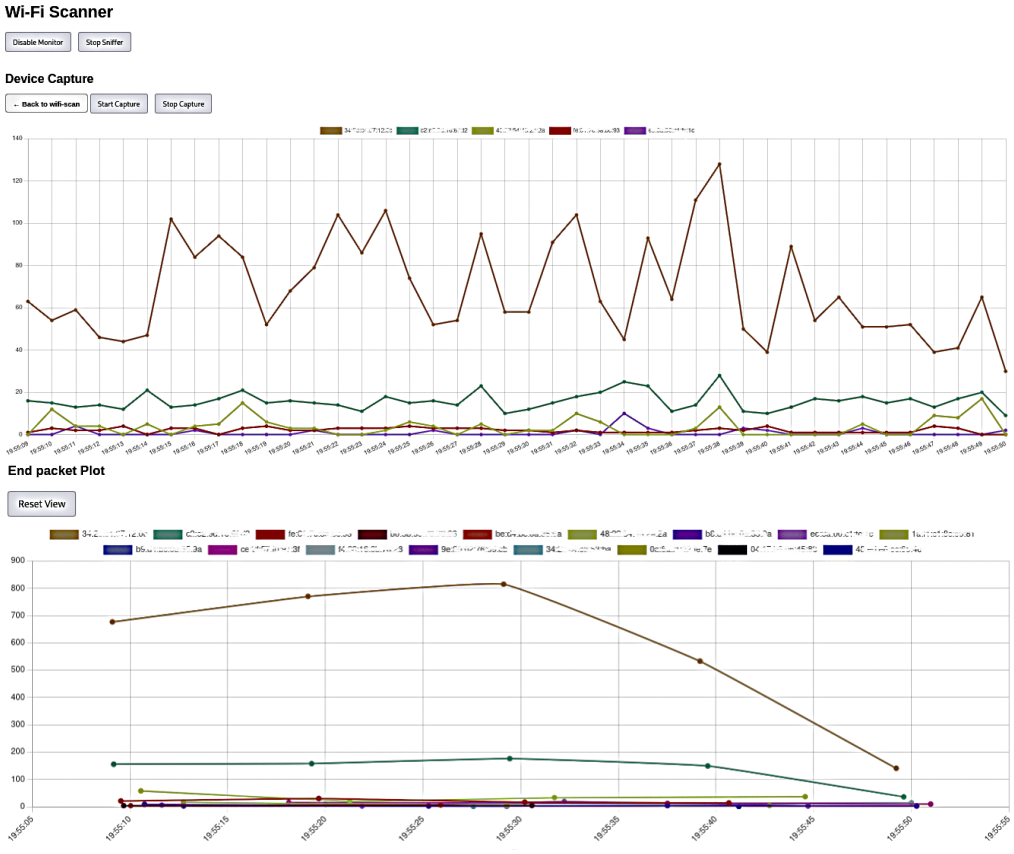
Klientsidans utseende på sidan *WiFi-skanner* redovisas i figur C.1.



Figur C.1: Sidan *WiFi-skanner* i analysverktyget.

C.2 Alla enheter

Klientsidans utseende på sidan *Alla enheter* redovisas i figur C.2.



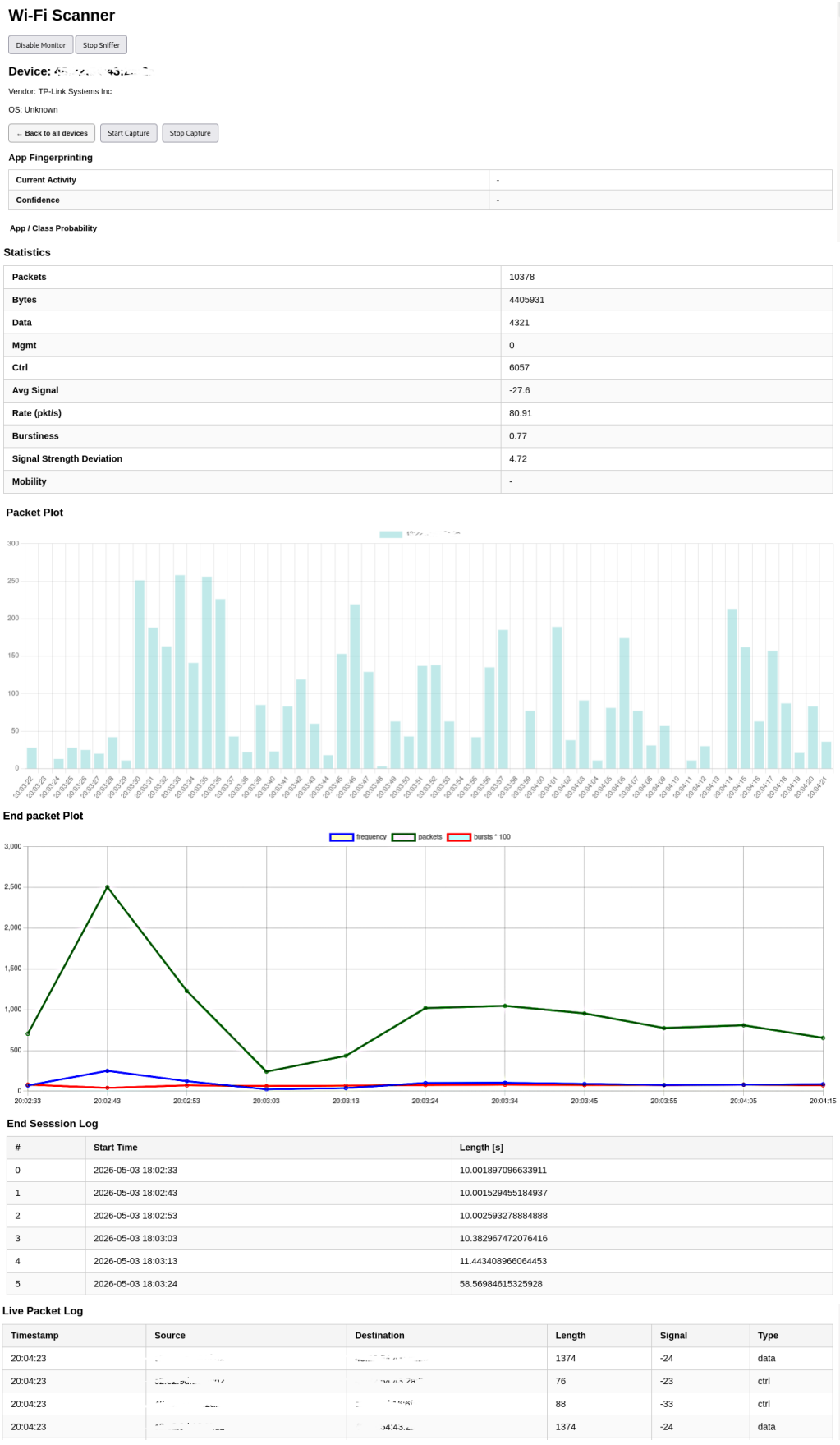
Mac	#	Start Time	Length [s]
[redacted]	0	19:55:09	42.44204878807068
[redacted]	0	19:55:09	42.292455196380615
[redacted]	0	19:55:09	39.015422105789185
[redacted]	0	19:55:09	38.71364378929138
[redacted]	0	19:55:10	37.5773739814758
[redacted]	0	19:55:10	10.69985294342041
[redacted]	1	19:55:21	10.473223209381104
[redacted]	2	19:55:31	18.106133937835693
[redacted]	0	19:55:10	10.661369800567627
[redacted]	1	19:55:25	25.71091866493225
[redacted]	0	19:55:11	10.240191459655762
[redacted]	1	19:55:21	10.325589895248413
[redacted]	2	19:55:32	18.22986888885498
[redacted]	0	19:55:12	16.54578423500061

MAC	Vendor	OS	Packets	Bytes	Data	Mgmt	Ctrl	Avg Signal	Strength Var	Rate (pkt/s)	Burstiness	Mobility	Action
[redacted]	Compal Broadband Networks, Inc.	Unknown	2941	239020	0	0	2941	-83.1	0.79	50.77	0.03	stationary	View Device
[redacted]	-	Unknown	680	151030	81	461	138	-21.3	1.59	17.03	-0.12	-	View Device
[redacted]	TP-Link Systems Inc	Unknown	153	16783	40	0	113	-30.8	0.83	3.90	0.51	stationary	View Device
[redacted]	-	Unknown	86	41366	0	86	0	-85.7	0.61	2.20	-0.08	stationary	View Device
[redacted]	-	Unknown	42	5001	21	5	16	-56.8	0.97	1.28	0.51	stationary	View Device
[redacted]	-	Unknown	35	3462	33	0	2	-25.0	0.00	0.90	0.28	stationary	View Device
[redacted]	-	Unknown	29	3497	13	3	13	-52.1	-	0.97	0.50	-	View Device
[redacted]	Apple, Inc.	macOS/iOS	26	2192	11	0	15	-75.7	0.31	0.86	0.65	stationary	View Device
[redacted]	zte corporation	Unknown	23	5021	3	7	13	-75.0	0.32	0.57	0.06	stationary	View Device

Figur C.2: Sidan *Alla enheter* i analysverktyget.

C.3 Specifik enhet

Klientsidans utseende på sidan *Specifik enhet* redovisas i figur C.3.



Figur C.3: Sidan *Specifik enhet* i analysverktyget.