

Mikko Luopajarvi

TEKOÄLYN HYÖDYNTÄMINEN TIETOVERKOISSA

Koneoppiminen verkonhallinnassa ja tietoturvassa

Kandidaattitutkielma
Informaatioteknologian ja viestinnän tiedekunta
Tarkastaja: Juha Vihervaara
Maaliskuu 2023

TIIVISTELMÄ

Mikko Luopajarvi: Tekoälyn hyödyntäminen tietoverkoissa
Kandidaatintutkielma
Tampereen yliopisto
Tietoliikennetekniikka
Maaliskuu 2023

Koneoppiminen on yksi tietotekniikan ajankohtaisimmista ja nopeiten kehittyvimmistä aloista. Sen avulla kehitettävät uudet tekoälysovellukset luovat valtavasti uusia mahdollisuuksia automaation toteuttamiselle eri käyttökohteissa. Samaan aikaan tietoliikennetekniikan alalla jatkuvasti kasvavat käyttäjämäärät ja uudet teknologiat, kuten esineiden internet (Internet of Things, IoT), ovat luoneet uusia haasteita internet protokollaan (Internet Protocol, IP) pohjautuvien tietoverkkojen resursseille, tietoturvalle ja verkonhallinnalle. Tässä työssä tutkitaan uusia tekoälysovelluksia, joilla pyritään ratkaisemaan tietoverkkojen nykyisiä haasteita ja yleisesti parantamaan verkonhallintaa, sekä tietoturvaa.

Työn kahdessa ensimmäisessä luvussa perehdytään ensin tietoverkkojen ja koneoppimisen perusteisiin. Ennen tietoverkoissa hyödynnettävän tekoälyn tarkempaa tutkimista, on tärkeää tutustua tietoverkkojen ympäristöön ja ymmärtää sen perus tekniikat ja nykyiset haasteet, sekä koneoppimisen tarjoamat uudet mahdollisuudet ja rajoitukset. Työn kolmannessa luvussa tarkastellaan kirjallisuuskatsauksen muodossa mahdollisia tekoälyn avulla toteutettavia ratkaisuja eri tietoliikenteen osa-alueilla verkonhallinnan automaation ja tietoturvan parantamiseksi.

Katsauksessa löydettiin useita ratkaisuja sille, miten tekoälyä voidaan hyödyntää ratkaisemaan tietoverkkojen nykyisiä haasteita. Konkreettisin tekoälyn käyttökohde tietoverkoissa on tällä hetkellä verkkojen tietoturvan parantaminen älykkäiden itseoppivien palomuurien avulla. Tekoälyn avulla voidaan myös automatisoida verkon valvontaa ja parantaa luotettavuutta ennakoivan ylläpidon avulla.

Kunnianhimoisin verkonhallinnan tekoälysovellus on tällä hetkellä tahtopohjainen verkonhallinta (Intent Based Networking, IBN). IBN-konseptin tavoitteena on moniosaisen tekoälyn avulla abstrahoida ja automatisoida verkonhallinnan tehtävät tasolle, jossa yksittäinen verkon ylläpitäjä voi yleiskielisillä käskyillä vastata koko verkon hallinnasta. Kompleksisemmilla tekoälysovelluksilla on paljon potentiaalia, mutta kokonaisuudessaan tekoälyn avulla toteutettavan verkonhallinnan vahvuus on kuitenkin edelleen selkeästi rajatuissa käyttökohteissa, joissa on helpompi hallinnoida tekoälyn koulutukseen käytettävää opetusdataa ja valvoa sovelluksen tuottamia tuloksia.

Avainsanat: Tekoäly, verkonhallinta, tietoturva, koneoppiminen, tietoverkot

Tämän julkaisun alkuperäisyys on tarkastettu Turnitin OriginalityCheck -ohjelmalla.

SISÄLLYSLUETTELO

1.	Johdanto	1
2.	IP-tietoverkot	2
2.1	Verkonhallinta	3
2.2	Ohjelmisto-ohjatut verkot.	3
2.3	Esineiden internet	5
2.4	Verkkojen tietoturva	5
2.4.1	Tunnelointi.	5
2.4.2	Palomuurit.	6
2.4.3	Haittaohjelmat	6
2.4.4	Tunkeilijan tunnistus ja estäminen	7
2.4.5	Palvelunestohyökkäykset	7
3.	Tekoäly ja koneoppiminen	9
3.1	Koneoppiminen	9
3.2	Oppimistavat	9
3.3	Mallit ja algoritmit.	10
4.	Tekoäly tietokoneverkoissa.	12
4.1	Käyttökohteita	12
4.2	Intelligent Intent Based Networking.	13
4.3	Tekoäly ja palomuurit	16
4.4	Tekoälyn hyödyntäminen IoT-ympäristössä.	19
5.	Yhteenveto	23
	Lähteet	25

LYHENTEET JA MERKINNÄT

A-tietue	DNS:n tapa pitää kirjaa doimainin IP-osoitteista
ABD	poikkeaman havainnointi (engl. Anomaly Based Detection)
API	ohjelmointirajapinta (engl. Application Programming Interface)
ASAP	Deep Q-oppimista hyödyntävä penetraatiotestausviitekehys (engl. Autonomus Security Analysis and Penetration)
CL	suljettu takaisinkytkentä (engl. closed-loop)
DDoS	hajautettu palvelunestohyökkäys (engl. Distributed Denial of Service)
Deep Q-oppiminen	Hermoverkkopohjainen tapa tehdä ohjattua Q-oppimista
DNS	domainin nimipalvelu (engl. Domain Name Service)
DoS	palvelunestohyökkäys (engl. Denial of Service)
DQN	Deep Q-verkko (engl. Deep Q network)
ENI	tekoälyn soveltamista tietoverkoissa tutkiva projekti (engl. Experimental Networked Intelligence)
ETSI	European Telecommunications Standard Institute
FCAPS	yleinen verkohallinnan viitekehys
FTP	File Transfer Protocol
HTTP	Hypertext Transfer Protocol
I ² BN	uusi verkonhallinnan paradigma (engl. Intelligent Intent Based Networking)
IDS/IPS	tunkeutumisen tunnistus- ja estojärjestelmä (engl. Intrusion Detection System/Intrusion Prevention System)
IMAP	Internet Message Access Protocol
Intent Manager	I ² BN-mallissa toimiva tahtoa toteuttava moduuli
IP	Internet protokolla
IPSec	IP Security Architecture
ISO	kansainvälinen standardointiorganisaatio
L2TP	Layer 2 Tunneling Protocol

MuIVAL	työkalu, jolla voidaan luoda penetraatiotestauksessa käytettäviä hyökkäysgraafeja (engl. Multi-host, Multi-stage Vulnerability Analysis Language)
Nessus	verkon haavoittuvuuksien tutkimiseen tarkoitettu avoimen lähdekoodin työkalu
NETCONF	Network Configuration Protocol
NGFW	seuraavan sukupolven palomuri (engl. Next Generation Firewall)
OpenVAS	avoin haavoittuvuuksien arviointijärjestelmä (engl. Open Vulnerability Assessment System)
OSI	tiedonsiirron referenssimalli (engl. Open Systems Interconnecton Reference Model)
PyTorch	Python-ohjelmointikielelle luotu koneoppimiskirjasto
QoS	palvelunlaatu (engl. Quality of Service)
REST	yleinen ohjelmointirajapintatyyppi (engl. Representational state transfer)
SBD	tunnistepohjainen havainnointi(engl. Transport Layer Security)
SBDAR	tunnistepohjainen havainnointi poikkeavalle liikenteelle (engl. Signature Based Detection of Anomaly Requests)
SBDKIT	tunnistepohjainen havainnointi haitalliselle liikenteelle (engl. Signature Based Detection of Intrusion Types)
SBDNR	tunnistepohjainen havainnointi tavalliselle liikenteelle (engl. Signature Based Detection of Normal Requests)
SDN	ohjelmisto-ohjattu verkko (engl. Software Defined Network)
SNMP	Simple Network Management Protocol
SSH	Secure Shell
TCP	Transmission Control Protocol
TLS	Transport Layer Security
UDP	User Datagram Protocol
VPN	Virtual Private Network
VXLAN	Virtual Extensible Local Area Network
WAF	verkkosovelluspalomuri (engl. Web Application Firewall)
XML	merkitäkilistandardi (engl. Extensible Markup Language)

1. JOHDANTO

Internet on noussut korvaamattomaksi osaksi yhteiskuntaamme. Sen suosion mukana nopeasti nousseet käyttäjämäärät ja niiden luomat vaatimukset ovat johtaneet yhä monimutkaisempiin ratkaisuihin internetin tiedonsiirtoverkkojen rakentamisessa. Kompleksisuuden ja laajuuden kasvaessa monet alun perin vuosikymmeniä sitten pieniä kokonaisuuksia varten kehitetyt peruskomponentit, joiden varaan modernit IP-verkot edelleen rakentuvat, eivät pysty enää vastaamaan verkon ylläpidollisiin haasteisiin. Perinteisten verkkolaitteiden kuten kytkimien ja reitittimien varaan rakentuvien IP-verkkojen skaalautuvuus on siis haasteellinen ja verkonhallintaan liittyvä työmäärä kasvaa kohtuuttomaksi. Tämä puolestaan johtaa verkon ylläpitokustannusten merkittävään kasvuun.

Yhdeksi ratkaisuksi verkonhallinnan skaalautuvuuteen ja tietoturva- haasteisiin on noussut tekoälyn hyödyntäminen. Koneoppimisen mukana kehittyneet tekoälyratkaisut luovat paljon uusia mahdollisuuksia myös verkonhallinnan helpottamiseksi. Aikaisemmin manuaalisena työnä pidettäviä verkon konfigurointitehtäviä voidaan esimerkiksi automatisoida ja verkon dynaamista suojaamista parantaa tekoälyn avulla.

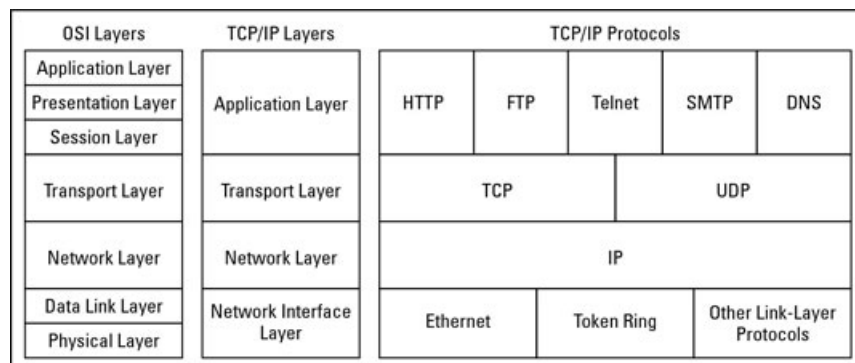
Tässä kirjallisuuskatsauksessa pyritään tuoreiden tieteellisten artikkeleiden avulla tutki- maan tekoälyä hyödyntäviä uusia ratkaisuja modernien IP-verkkojen hallinnan ja tietotur- van parantamiseksi. Työn tavoitteena on vertailla erilaisten toteutuksien hyötyjä ja haas- teita sekä luoda kokonaiskuva tekoälyn hyödyllisyydestä sekä verkonhallinnan että verkon tietoturvan osalta.

Työn kahdessa ensimmäisessä luvussa perehdytään ensin IP-verkkojen, sitten koneop- pimisen ja tekoälyn teknisiin periaatteisiin ja ratkaisuihin. Kolmannessa luvussa esitel- lään erilaisia uusia tietoverkkoihin liittyviä tekoälysovelluksia ja perehdytään niiden luo- miin mahdollisuuksiin ja haasteisiin.

2. IP-TIETOVERKOT

IP-tietoverkkojen tiedonsiirto perustuu ISO:n standardisoiman OSI-mallin tyyliseen protokollapinoon, jossa monimutkainen tiedon siirto on jaettu hallittaviin kerroksiin. Kuvan 2.1 mukaisesti TCP/IP-protokollan arkkitehtuurissa pino on jaettu neljään tasoon. Pinon alimalla tasolla on yhdistettynä siirtoyhteys sekä fyysinen kerros. Fyysinen kerros tarkoittaa fyysistä kanavaa, jossa siirrettävä tieto liikkuu bitteinä. Siirtoyhteyskerros puolestaan vastaa biteistä koostuvien kehyksien lähettämisestä fyysisen kerroksen yli. [2].

Pinon toinen kerros vastaa IP-kehyksistä koostuvien pakettien reitittämisestä verkossa IP-osoitteiden perusteella reitittimien yli lähettäjältä vastaanottajalle. Tätä kerrosta kutsutaan verkkokerrokseksi. Pinon kolmas kerros on päästä-päähän-tyyppinen, ja sitä kutsutaan kuljetuskerrokseksi. Yleisesti kuljetuskerros käyttää joko yhteydellistä ja tiedonsiirrollisesti luotettavaa TCP:tä tai yhteydetöntä ja epäluotettavaa UDP:tä. Pinon ylimmällä tasolla on sovelluskerros, joka sisältää sovelluksille spesifiset toteutukset. Tällaisia sovelluksia voivat olla esimerkiksi tiedostonsiirrossa käytetty FTP, selaimen ja palvelimen tiedonsiirrossa käytettävä HTTP ja sähköpostien hakemiseen käytettävä IMAP. Yhdessä nämä kerrokset luovat toimivan kokonaisuudet, johon internetin tiedonsiirto perustuu.



Kuva 2.1. TCP/IP Protokollapino [24].

Perinteisesti tietoverkot rakentuvat fyysisten laitteiden kuten kytkimien ja reitittimien sekä niihin liittyvien lisätoimintojen varaan. TCP/IP-protokollapinin mukaiset paketit ohjataan näiden laitteiden läpi. Abstraktilla tasolla verkkojen tiedonsiirtoon voidaan katsoa liittyvän kaksi käsitettä: dataliikenteen ohjaus eli datataso (data plane) ja verkkolaitteiden hallinta eli ohjaustaso (control plane). Yksinkertaistettuna datataso vastaa pakettien siirrossa linkiltä toiselle ohjaustason logiikan mukaisesti. Perinteisessä toteutuksessa verk-

kolaitteet sisältävät molemmat tasot. Laitteella datatasosta vastaa kolmannen kerroksen IP-protokolla ja ohjaustasosta toisen kerroksen käyttötarkoitukselliset protokollat kuten reititysprotokollat tai vaikkapa palomuri.

2.1 Verkonhallinta

ISO on määritellyt verkonhallinnalle viisi päätehtävää: vianhallinta (Fault management), konfiguraation hallinta (Configuration management), asiakkuuden hallinta (Accounting management), suorituskyvyn hallinta (Performance management) ja turvallisuuden hallinta (Security management). Yhdessä tämä kokonaisuus tunnetaan FCAPS-viitekehyksenä, joka toimii pohjana verkonhallinnan suunnittelussa. [4] Verkonhallinnan tehtävä on siis käytännössä hallita ohjaustason toimintaa.

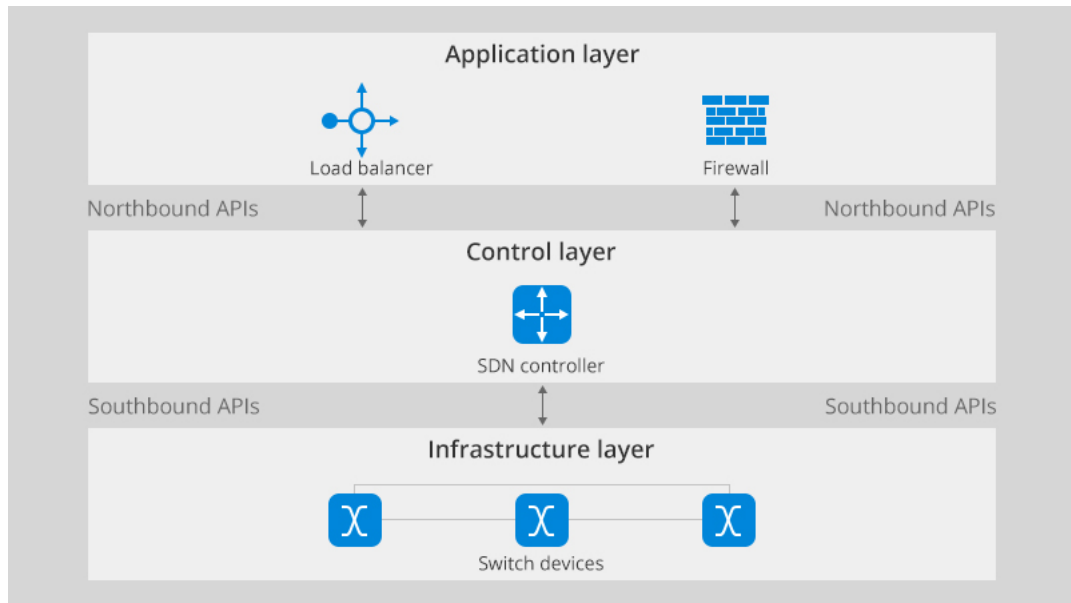
Käytännössä verkonhallinta tapahtuu sovelluserroksessa, josta verkkoa valvotaan sekä sen laitteita hallinnoidaan. Perinteisissä TCP/IP-verkoissa on valvontaprotokollana edelleen laajasti käytössä SNMP (Simple Network Management Protocol). SNMP:tä käytetään kuitenkin vain verkon valvomiseen, verkkolaitteiden konfigurointi tapahtuu ottamalla esimerkiksi SSH-yhteys niiden komentoliittymään. [37] Alun perin 1980-luvulla kehitetty SNMP on kuitenkin jäämässä kehityksen jalkoihin, jatkuvasti monimutkaistuvien verkkojen tarpeisiin on kehitetty uusia hallintaprotokollia. Yksi näistä kilpailevista protokollista on NETCONF (Network Configuration Protocol), joka yhdistää valvonnan ja konfiguroinnin toiminnallisuudet ja pyrkii parantamaan verkonvalvonnan tehokkuutta kompleksisissa verkoissa. [13] Kokonaisuudessaan IP-verkkoteknologiat ovat kehittyneet jo jonkin aikaa suuntaan, jossa koko verkon arkkitehtuuri itsessään muuttuu tehokkaammaksi.

2.2 Ohjelmisto-ohjatut verkot

Ohjelmaohjatut verkot (Software Defined Networking, SDN) on kehittyvä verkkoarkkitehtuuri, joka on kehitetty moderniksi vaihtoehdoksi perinteisten fyysisten verkkolaitteiden varaan rakentuville verkoille. SDN-verkossa vanhat hajautetut verkkolaitteet, jotka vastaavat sekä data- että ohjaustasosta, korvataan kokonaisuudella, jossa itse verkkolaitteet ovat niin kutsuttuja pakettikytkimiä (packet switch) ja vastaavat vain datatasosta. SDN-verkossa ohjaustasosta vastaa keskitetysti yksi älykkäämpi ohjainlaite, eli käytännössä palvelin. [23] Keskitetty ohjainlaite voidaan ohjelmoida hoitamaan useita eri tehtäviä, joita laitekohtaiset protokollat ovat ennen hoitaneet. Kun ohjaustasoa voidaan hallita yhdeltä keskitetyltä laitteelta, monimutkaisten verkkojen hallinta ja konfigurointi muuttuvat huomattavasti tehokkaammaksi.

SDN-verkkojen tärkein etu on niiden joustavuus ja skaalautuvuus. Keskitetty verkonhallinta ja konfigurointi mahdollistaa verkon laajentamisen ja uusien laitteiden lisäämisen nopeasti ja tehokkaasti. Kuvan 2.2 tyyliin SDN-verkkojen ohjainpalvelimelle voidaan ohjel-

mointirajapintojen (Application Programming Interface, API) avulla helposti sulauttaa muita verkoissa käytettävien laitteiden toiminnallisuuksia, kuten palomuri, kuormantasaus ja reitityksenhallinta. [12]



Kuva 2.2. SDN-verkon esimerkkirakenne [14].

Käytännön tapauksissa SDN-verkoille on olemassa muutamia erilaisia toteutusmenetelmiä. Avoimen lähdekoodin SDN-protokollat kuten OpenFlow olivat kehityksen keskipisteessä vielä 2010-luvulla, mutta eivät muun muassa skaalautuvuusongelmien vuoksi ikinä saavuttaneet suuria käyttäjämääriä. [5] Yleisemmin SDN-verkkojen toiminta perustuu tällä hetkellä REST-tyyppisiin ohjelmointirajapintoihin [39] ja fyysisten laitteiden päälle rakentuvan VXLAN (Virtual eXtensible Local Area Network) topologian varaan. [38]

SDN-verkkojen konkreettisin haaste on tällä hetkellä niiden vaatimat suuret investoinnit. Käytännössä missä tahansa sovelluskohteessa koko olemassa olevan verkon korvaaminen uudella järjestelmällä on kallista ja vie paljon aikaa. Keskitetty ohjainpalvelin luo verkoon myös selkeän kohteen mahdollisille hyökkääjille. Keskitetty verkonhallinta tarkoittaa myös sitä, että hyökkääjän tarvitsee murtautua vain yhdelle laitteelle saadakseen koko verkon haltuunsa. Toisaalta mahdollinen hyökkääjä ei välttämättä saa yhtä suurta etua murtautuessaan pakettikytkimelle kuin esimerkiksi perinteisen verkon reitittimelle. SDN-verkoissa, kuten kaikissa muissakin verkoissa, on joka tapauksessa ensisijaisen tärkeää varmistaa ensin riittävä tietoturva. Ohjelmoitavuutensa ansiosta SDN-verkot luovat hyvän sovellusalustan jatkokehityksellä ja toimivat mahdollistajana esimerkiksi potentiaalisissa verkon ylläpidon tekoälysovelluksissa.

2.3 Esineiden internet

Esineiden internet (Internet of Things, IoT) on nopeasti kehittyvä trendi, jossa yhä useammat fyysiset laitteet yhdistetään joko lähi- tai mobiiliverkon kautta internettiin ja niille annetaan älykkäitä toiminnallisuuksia. IoT-laitteita voivat esimerkiksi olla erilaiset sulautettuja järjestelmiä sisältävät laitteet, kuten sensorit, kodinkoneet tai teollisuudessa käytettävät valmistuskoneet. Esineiden internet on koko yhteiskunnan tasolla luonut valtavasti uusia mahdollisuuksia, IoT on jo mahdollistanut esimerkiksi ihmisten hyvinvoinnin seuraamisen älykellojen ja -sormusten avulla. Tärkeimpiä kohteita IoT-sovelluksille ovat esimerkiksi tehtaiden tuotantolinjat, kaupungit, toimistot ja ihmisten kodit. [31]

Nopeasti kehittyvä ja kasvava esineiden internet on kuitenkin luonut uusia merkittäviä vaatimuksia sen laitteiden käyttämille verkoille sekä suuren määrän uusia tietoturva-avoittuvuuksia ja hyökkäyspintoja. IoT-laitteiden ympäristö tarkoittaa usein sitä, että niiden laskentateho ja laiteresurssit ovat hyvin rajallisia, mikä johtaa helposti kyseenalaisiin tietoturvaratkaisuihin. Esineiden internet on siis luonut selkeän tarpeen uusille tietoturvaa sekä verkkojen ylläpitoa parantaville ratkaisuille.

2.4 Verkkojen tietoturva

Yksi tärkeimmistä tietoverkkoihin liittyvistä käsitteistä on niiden tietoturva. Verkon tietoturvan tehtävä on hallita riskejä ja tällä tavalla suojata ja varmistaa verkon haluttu toiminta. Verkon tietoturvan toteutuksessa täytyy siis implementoida erilaisia tietoturvakäytäntöjä ja prosesseja. Verkon tietoturva on todella laaja kokonaisuus, mutta yleisesti tietoturva-prosessien tulisi ainakin hallita verkkoon pääsyä, salata verkon liikenne, varmistaa verkon käyttäjien oikeuden käsitellä verkon eri resursseja, suojata verkkoa haittaohjelmilta sekä estää ja tunnistaa verkkoon luvaton tunkeutuminen tai liikenteen estäminen. [8]

2.4.1 Tunnelointi

Tietoturvan peruselementteihin kuuluu liikenteen luottamuksellisuuden suojaus salauksen avulla. Etenkin kaupallisessa käytössä verkon yhteyksien salaamiseen käytetään Virtual Private Network (VPN) tunnelointia. VPN-tunnelin avulla käyttäjät pystyvät luomaan suojatun yhteyden esimerkiksi lähiverkkoon sen ulkopuolelta tai verkosta toiseen. Itse tunnelointi voi tapahtua esimerkiksi toisen kerroksen tunnelointiprotokollan (Layer 2 Tunneling Protocol, L2TP) ja IPsec-protokollan avulla. Tunneloinnissa normaalisti verkossa kulkevat paketit salataan, uudelleenpaketoitetaan ja ohjataan L2TP:n avulla tunnelin päästä päähän suojattuna muulta liikenteeltä. L2TP/IPsec on kuitenkin vanhaa tekniikkaa ja pystyy käsittelemään vain UDP-paketteja. [26] Tunneloinnin perusidea on kuitenkin hyvin pitkälti sama millä tahansa protokollalla. VPN-palvelun salaustapa riippuu usein käytettävästä protokollasta ja saattaa olla konfiguroitavissa.

Relevantein tunnelointiprotokolla on tällä hetkellä OpenVPN. OpenVPN on avoimen lähdekoodin tunnelointiprotokolla, joka antaa käyttäjälle eniten vaihtoehtoja. OpenVPN kykenee esimerkiksi käsittelemään sekä TCP- että UDP-paketteja. Vapaasti tarkasteltavissa oleva lähdekoodi ja toimintatapa tarkoittavat myös, että protokollaa on tutkittu paljon, mikä edistää sen tietoturvaa ja vähentää aukkojen riskiä. [19] OpenVPN-protokolla käyttää oletusarvona TLS-suojausta, mutta sallii myös salaustyyppin vaihtamisen. [25]

2.4.2 Palomuurit

Verkoissa konkreettisin tietoturvan perusosa on palomuuuri. Perinteisesti palomuuuri toimii kuvan 2.1 mukaisen IP-protokollapinon toisella tasolla eli verkkokerroksessa, sekä kolmannessa eli kuljetuskerroksessa. Palomuurin pääasiallinen tehtävä on suojata verkkoon liitettyjä laitteita luvattomalta liikenteeltä. Parametrit sille, miten palomuuuri suodattaa sen läpi kulkevaa liikennettä on ylläpitäjän konfiguroitavissa. Perustoimintona palomuuuri kuitenkin käy läpi sille tulevia IP-paketteja ja analysoi niiden otsakkeiden sisällön ja tarkastaa esimerkiksi kohteen IP-osoitteen ja portin. Konfigurointinsa perusteella palomuuuri voi tällöin joko päästää paketin läpi tai pudottaa sen pois liikenteestä. [1] Palomuurin hallinta ja konfigurointi kuuluu siis verkonhallinnan päätehtäviin.

Nykyisin palomuurit ovat levinneet myös protokollapinon ylemmille kerroksille. Yksi sovel-luskerroksessa toimiva palomuurityyppi on verkkosovelluspalomuuuri. Verkkosovelluspalo-muurin toiminta periaate on sama kuin verkkokerroksen palomuurilla, mutta se suodattaa IP-liikenteen sijasta verkkosivuille tulevaa HTTP-liikennettä.

SDN-verkkojen ja jossain määrin myös tekoälyn kehittymisen myötä verkkolaitevalmis-tajat ovat alkaneet markkinoimaan uuden sukupolven palomuuureja (Next Generation Fi-rewall). Varsinaista standardisointia sille, mitä seuraavan sukupolven palomuurit pitävät sisällään ei ole, mutta pääpiirteittäin ne ovat ohjelmisto-ohjattuja laitteita ja kykenevät pe-rinteisen suodatintyyppisen palomuurin lisäksi myös monimutkaisempiin valvonta ja suo-jaustoimintoihin.

2.4.3 Haittaohjelmat

Verkon suojaaminen haittaohjelmilta on luonnollisesti elintärkeää siihen kytkettyjen laittei-den ja datan kannalta. Lähtökohtaisesti verkkoa voidaan ennakoivasti suojata asettamal-la esimerkiksi sähköpostifilttereitä sekä ohjelmia, jotka estävät epäilyttävien tiedostojen ja linkkien avaamisen. Lisäksi verkon käyttäjille voidaan antaa tietoturvakoulutusta.

On kuitenkin varsin realistista, että verkkoon saattaa silti käyttäjävirheen tai muun vuok-si päästä haittaohjelma, mikä tarkoittaa, että verkossa täytyy myös olla kyky vastata ja sopeutua tilanteeseen. Helpoin tapa suojata verkkoa tässä tapauksessa on pitää sen lait-teilla ajantasaiset antivirukset ja haittaohjelmien suojaukset. Verkko voidaan myös sege-

mentoida esimerkiksi verkon virtualisoinnin avulla. [7] Tällä voidaan rajoittaa vahinkoja tilanteessa, jossa haittaohjelma pääsee leviämään verkon sisällä. Verkon tärkeimmät dataresurssit on myös syytä pitää varmuuskopioituna verkon ulkopuolella vahingon rajoittamiseksi.

2.4.4 Tunkeilijan tunnistus ja estäminen

Verkon uhkana ei ole pelkästään haittaohjelmat tai virukset. Mahdollinen hyökkääjä voi koittaa hyödyntää verkon haavoittuvuuksia ja koittaa murtautua verkkoon haitallisen liikenteen avulla. Haitallisen liikenteen avulla tehtäviä hyökkäyksiä voidaan torjua tunkeilijan tunnistus- ja estojärjestelmillä (Intrusion Detection/Intrusion Prevention system, IDS/IPS). Yleisesti IDS/IPS-järjestelmät eroavat palomuurista siinä, että ne analysoivat kokonaisia paketteja, kun taas palomuurit analysoivat vain pakettien otsakkeita. [3]

Pelkät IDS-järjestelmät ovat jo jossain määrin vanhentuneita, sillä ne vain valvovat verkkoa ja ilmoittavat haitallisesta liikenteestä valvojalle, mutta eivät automaattisesti estä haitallista liikennettä. Etenkin suurissa teollisissa tai kaupallisissa verkoissa automaattisen liikenteen eston puuttuminen on selkeä heikkous. IDS/IPS-järjestelmät puolestaan pystyvät automaattisesti myös reaaliaikaisesti estämään haitallisen liikenteen verkosta. [22]

Perinteisesti kummatkin järjestelmät käyttävät tunnistepohjaista havainnointia (Signature Based Detection, SBD) liikenteen analysointiin. [22] Järjestelmillä on siis tietokanta, joka sisältää haitallisen liikenteen tunnisteita, eli tunnetun mallisia haitallista liikennettä sisältäviä paketteja. IDS/IPS on kuitenkin jälleen uusia konfiguroitava ja ylläpidettävä laite verkossa. Siksi uuden sukupolven palomuuressa on usein sisäänrakennettu IDS/IPS-järjestelmä. [10]

2.4.5 Palvelunestohyökkäykset

Verkkoa vastaan voidaan hyökätä myös kohdistamalla verkkoon massiivinen määrä liikennettä, jolloin verkko ruuhkaantuu pahasti ja sen toiminta käytännössä lamaantuu. Tätä kutsutaan palvelunestohyökkäykseksi (Denial of Service, DoS). DoS-hyökkäykset ovat selkeä uhka esimerkiksi SDN-verkoille. Koska keskitetty hallintaohjain keskustelee pakettikytkimien kanssa verkon yli, verkko voidaan lamauttaa ohjaamalla verkkoon paljon ohjaustason liikennettä. [9] Ruuhkauttavan liikenteen luomiseen käytetään yleensä useiden kaapattujen laitteiden muodostamia bottiverkkoja (botnet). Palvelunestohyökkäystä, jossa käytetään bottiverkkoja kutsutaan hajautetuksi palvelunestohyökkäykseksi (Distributed Denial of Service, DDoS). [27] Nopeasti lisääntyvät IoT-järjestelmät ovat DDoS-hyökkäysten kannalta suuri riski ja antavat mehkkaan ja suhteellisen helpon maalin hyökkääjille, jotka pyrkivät luomaan bottiverkkoja. Palvelunestohyökkäyksiltä voidaan suojautua jo suhteellisen helposti ja tehokkaasti seuraamalla domainin nimipalvelun (Do-

main Name Service, DNS) A-tietueita ja ohjaamalla DDoS-hyökkäyksen liikenne erilliselle DDoS-hyökkäyksiltä suojaavalle palvelulle. [20] Ongelmaan tulisi kuitenkin kehittää ratkaisuja myös juuritasolla, IoT-laitteet olisi ensisijaisen tärkeää suojata kaappaukselta.

3. TEKOÄLY JA KONEOPPIMINEN

Tekoälyllä tarkoitetaan nykyisin käytännössä aina koneoppimiseen perustuvaa sovellusta. Mitä sitten on koneoppiminen ja miten sitä hyödynnetään tekoälyn kehittämisessä? Mitä ovat koneoppimisen vahvuudet ja heikkoudet? Minkälaisissa sovelluksissa koneoppimista voidaan käyttää?

3.1 Koneoppiminen

Koneoppimisella tarkoitetaan yleensä tietokoneohjelmaa joka oppii tekemään jonkin asian sille annetun opetusdatan perusteella. Nopeasti kasvaneet datamäärät ja prosessointiteho ovat toimineet 2010-luvulla mahdollistajina koneoppimisen kehittämiselle, sen suosio on noussut viime vuosina nopeasti. [16] Nopean yleistymisen myötä koneoppiminen ja siihen usein suoraan yhdistettävä termi tekoäly ovat muuttuneet yleisesti hiukan väärin ymmärretyiksi. Koneoppiminen itsessään on vain kattotermi syötteestä oppivalle tietokoneohjelmalle, mutta toteutustapoja tällaiselle ohjelmalle on lukuisia, jotka eroavat toisistaan huomattavasti matemaattisella tasolla. Tekoäly puolestaan voi olla koneoppimisen avulla toteutettu ohjelma, joka kykenee suorittamaan niin sanottuja älykkäitä toimintoja. Tekoäly ei kuitenkaan aina suoraan tarkoita, että ohjelman toiminnoissa on käytetty koneoppimista. Esimerkiksi videopelien kontekstissa ei-pelattavien hahmojen niin kutsuttu tekoäly on usein toteutettu perinteisen logiikkaohjelmoinnin avulla ilman koneoppimista.

3.2 Oppimistavat

Korkeammalla tasolla koneoppiminen voidaan jakaa karkeasti kolmeen osaan perustuen siihen, millä tavoin ohjelma käsittelee ja oppii sille annettavista syötteistä:

- ohjattu oppiminen
- ohjaamaton eli itseoppiminen
- vahvistusoppiminen

Kullakin näistä oppimismalleista on omat käyttökohteensa johon ne soveltuvat parhaiten. Tekoälysovelluksia kehitettäessä onkin tärkeää pystyä tunnistamaan ja ymmärtämään näiden oppimismallien erot ja miksi jokin tietty niihin perustuva algoritmi valitaan

sovelluksen toteuttamiseen.

Ohjattu oppiminen on koneoppimisen tavoista perinteisin ja helpoin käsittää. Ohjatussa oppimisessa opetettavalle ohjelmalle annetaan syöte ja lopputulospareja, joiden välille ohjelma pyrkii muodostamaan yhteyden. Ohjelma siis ensin opetetaan niin kutsutun opetusdatan perusteella toimimaan halutulla tavalla. Sen jälkeen mallin toiminta ja tarkkuus tarkastetaan erillisellä varmistusdatalla. Tämän jälkeen ohjelmalle voidaan antaa uusia syötteitä, jotka se käsittelee opetetulla tavalla ja tarkkuudella. Ohjattu oppiminen soveltuu parhaiten hahmontunnistukseen, ja sitä sovelletaan usein datan luokitteluun.[18] Datan luokittelussa käytettävä malli tai algoritmi opetetaan siis jakamaan sille annettava data ennalta määritettyihin kategorioihin opetusdatan avulla, jonka jälkeen se osaa luokitella näihin kategorioihin sille annettavia uusia syötteitä.

Ohjattua oppimista käytetään hyvin samaan tapaan myös regressioanalyysissä tutkimaan ja todentamaan muuttujien välisiä riippuvuuksia. Opetettua algoritmia voidaan sen jälkeen käyttää esimerkiksi ennustavan mallin luomiseen datasta. [28] Ohjatussa opetuksessa datan tyyppi ja luokiteltavat kategoriat joudutaan kuitenkin määrittelemään ennalta, mikä selvästi rajaa sen käyttökohteita. Ohjatussa oppimisessa käytettävä opetusdata vaatii siis ihmisen käsittelyä ennen kuin sitä voidaan soveltaa algoritmin kouluttamiseen.

Ohjaamaton oppiminen, jota kutsutaan myös itseoppimiseksi, ei puolestaan tarvitse käsiteltä dataa algoritmin opettamiseen. Itseoppimisen sovelluskohteet ovat ohjattua oppimista abstraktimpia ja keskittyvät pääosin statistiikkaan ja data-analyysiin. Ohjaamatonta oppimista voidaan soveltaa tällä hetkellä parhaiten analysoitavien datajoukkojen ominaisuuksien tutkimiseen joko datan ryhmittelyn (clustering) samanlaisuuksien perusteella tai datassa esiintyvien parametrien välisten riippuvuuksien tutkimisella (association). [29]

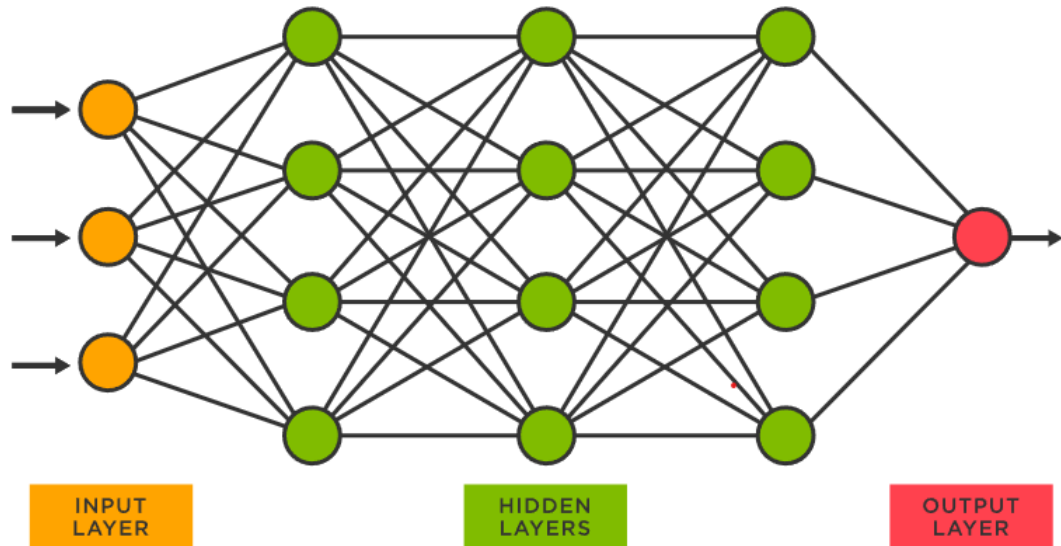
Kolmas merkittävä tapa toteuttaa koneoppimista on vahvistusoppiminen. Vahvistusoppimisessa opetettava algoritmi oppii sille annettavan palautteen perusteella. Ohjelma siis oppii suorittamaan jonkin asian palautteesta johdetun kokemuksen perusteella. [34] Palaute annetaan ohjelmalle pisteiden muodossa. Yksinkertaistettuna oikeasta liikkeestä tai suorituksesta ohjelmaa palkitaan antamalla sille pisteitä ja vääristä suorituksista ohjelmaa rangaistaan ottamalla siltä pisteitä pois. Todellisuudessa vahvistusoppimismallit ovat hie- man monimutkaisempia, mutta periaate oppimistavan taustalla on melko yksinkertainen.

3.3 Mallit ja algoritmit

On useita erilaisia matemaattisia malleja ja niihin pohjautuvia algoritmitoteutuksia sille, miten näitä eri koneoppimisen muotoja voidaan toteuttaa. Koneoppimista sovellettaessa on tärkeää ensin tunnistaa, millaista sovelluskohteen data on, ja sen perusteella valita sopiva oppimisen tyyppi. Kun sovellukselle on valittu koneoppimisen tyyppi, voidaan lähteä arvioimaan, kokeilemaan ja säätämään erilaisia valittuun oppimistapaan perustuvia

algoritmeja parhaan tuloksen saavuttamiseksi.

Neuroverkkoja käytetään laajasti monilla eri koneoppimisen aloilla. Ne eivät ole aina lähtökohtaisesti paras malli koneoppimissovelluksen toteuttamiseen, mutta niillä on suuri potentiaali monissa eri käyttökohteissa. Neuroverkot perustuvat matemaattisiin malleihin, joilla imitoidaan ihmisaivojen toimintaa. Kuvan 3.1 tapaan neuroverkot muodostuvat hermosoluja eli neuroneita sisältävistä kerroksista, jotka ovat vuorovaikutuksessa keskenään ja muodostavat verkoston, joka pystyy oppimaan ja tekemään päätöksiä. [21]



Kuva 3.1. Havainnollistava kuva kolmikerroksisen neuroverkon rakenteesta [35].

Neuroverkot ovat erittäin tehokkaita. Niillä on todella vahva suorituskky ohjatun oppimisen sovelluksissa, kuten esimerkiksi kuvantunnistuksessa, äänentunnistuksessa ja luonnollisen kielen käsittelyssä. Neuroverkkojen toiminta perustuu painojen säätämiseen, joiden avulla verkko optimoi suorituskynsä. Neuroverkot ovat myös skaalautuvia ja niiden tehokkuutta voidaan parantaa lisäämällä neuroneita tai verkon kerroksia. Useammista kerroksista koostuvia hermoverkkoja kutsutaan syväneuroverkoiksi (Deep Neural Networks). [21] Erikoistuneita neuroverkkoja on kehitetty erilaisiin tarkoituksiin. Konvoluutioneuroverkot sisältävät esimerkiksi konvoluutiokerroksia, jotka jakavat tunnistettavan hahmon aina pienempiin osiin ja suorittavat sille uuden hahmontunnistuksen. Konvoluutiohermoverkoilla on esimerkiksi kuvantunnistuksessa ylivoimainen suorituskky. [17]

4. TEKOÄLY TIETOKONEVERKOISSA

European Telecommunications Standards Institute (ETSI) käynnisti vuonna 2017 Experimental Networked Intelligence (ENI) -projektin, jonka tavoitteena oli kartoittaa käyttötapaukset, vaatimukset ja tulevaisuuden referenssinä käytettävä referenssiarkkitehtuuri tekoälyn hyödyntämiseksi IP-verkkojen hallinnassa ja ylläpidossa. [36] Vuonna 2018 julkaistussa ENI-projektin raportissa "Network Management and Orchestration Using Artificial Intelligence" Wang et al. arvioivat, että merkittävästi kasvaneet ja monimutkaistuneet tietoverkot ovat nopeasti nostaneet verkonhallinnan työkuormaa ja sen myötä verkkojen operointikustannukset ovat nousseet erittäin korkeiksi. [36] ENI-projektin päätavoitteena oli tunnistaa tekoälyn mahdolliset käyttökohteet ja tuottaa arkkitehtuurikuvaus tekoälysovelluksesta, jolla pystytään lisäämään verkon ylläpidon automaatiota ja laskemaan operointikustannuksia.[36]

4.1 Käyttökohteita

Wang et al. mukaan datakeskusten käytössä suurin yksittäinen osa verkon operointikustannuksista on keskusten tehonkulutus. [36] Ratkaisuksi ryhmä ehdottaa palvelinkapasiteetin mukauttamista automatisoidusti tekoälyn avulla keskukselle kohdistuvan liikenteen mukaan. Tekoälyä käyttävä automaatio voisi siis ohjata vähäisen liikenteen aikana vain tarvittavan osan palvelimista hoitamaan tehtäviä ja loput palvelimista tyhjäkäynnille, jolloin datakeskuksen tehonkulutusta saadaan laskettua tarvittaessa. [36] Tällaiseen sovellukseen ei kuitenkaan tarvita välttämättä tarvita suoraan koneoppimista ja tekoälyä, vaan datakeskuksen kuorman hallinnan automatisointi olisi varmasti mahdollista toteuttaa myös parametripohjaisesti logiikkaohjelmoinnilla. Kuormanhallinnassa voitaisiin hyödyntää myös ohjattua syväoppimista ja sen avulla pyrkiä ennakoimaan ruuhka-aikoja ja kapasiteetin tarvetta, jolloin vaste kapasiteetin lisäämiseen nopeasti tarvittaessa olisi parempi. [36] Tästä voi olla konkreettista hyötyä kuormanhallinnalle ja palvelun laadulle (Quality of Service, QoS) verkon ruuhkautuessa tiettyinä aikoina nopeasti.

Raportti keskittyy osittain tekoälyn soveltamiseen mobiiliverkoissa, mikä on tämän työn laajuuden ulkopuolella. Tietokoneverkkojen kannalta muita esitettyjä potentiaalisia käyttökohteita ENI-tekoälymallille ovat DDoS-hyökkäyksiltä suojautuminen ja verkon ennakoiva ylläpito. ENI-järjestelmän tulisi myös kyetä ennustamaan verkon vikoja, ennen kuin ne

tapahtuvat ja verkko kaatuu. [36] Tekoäly voidaan kouluttaa ennustamaan verkosta vikoja liikenteen ja muiden verkosta kerättävien parametrien, kuten esimerkiksi laitteiden lämpötilojen ja tehonkulutusten avulla. Jos verkkojen käyttökatkoja pystytään vähentämään, niin ylläpitäjä pystyy välttämään niistä koituvat ylimääräiset kulut.

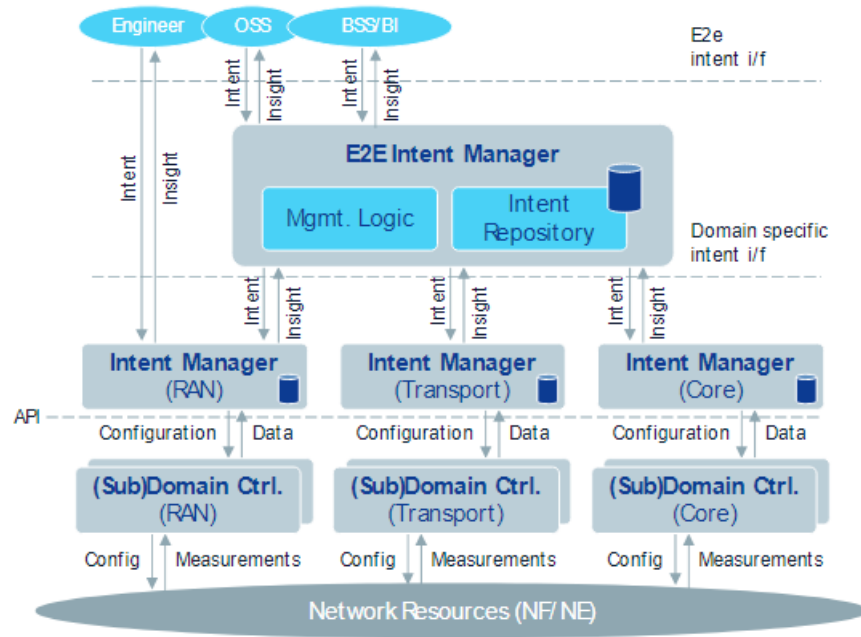
Tekoälyn potentiaalinen vahvuus DDoS-suojauksissa on kyetä hahmontunnistuksen avulla havaitsemaan protokollasta riippumatta alkava DDoS-hyökkäys. [36] Kyky tunnistaa alkava DDoS-hyökkäys eri verkkokerroksen tasoilla protokollasta riippumatta on selkeä vahvuus, mutta käyttökohteen kuvaus on raportissa melko ympäröivä. Kuten kappaleessa 2.4.5 mainittua DDoS-hyökkäyksille on olemassa jo suojautumistapoja, eli tekoälyn hyödyntäminen tällaisessa tapauksessa kannattaisi harkita ja suunnitella tarkkaan.

Lähtökohtaisesti vaatimukset sille, mitä ENI-järjestelmä tarvitsee verkolta, johon se implementoidaan, riippuu paljon käyttökohteesta ja ratkaistavista haasteista. Jos esimerkiksi tavoitteena on ennakoiva verkon ylläpito, verkosta täytyy pystyä keräämään erilaisten antureiden avulla tehokkaasti laitteiden tilatietoa. Implementoinnissa täytyy myös pysyä tietoisena käyttökohteen ja ympäristön aiheuttamista vaatimuksista, kuten esimerkiksi reaaliaikaisuudesta. [36] ENI-järjestelmän implementoinnissa on siis tärkeä pysyä päämäärätietoisena siitä, mitä järjestelmällä halutaan ratkaista ja mitä järjestelmä vaatii ratkaisun saavuttamiseksi, sekä mitä rajoitteita sovellusympäristö aiheuttaa järjestelmälle. Tekoäly tarjoaa paljon mahdollisuuksia ja potentiaalia, mutta sen todellinen tarve ja rajoitteet hyödyntämiselle on tärkeä kyetä tunnistamaan.

4.2 Intelligent Intent Based Networking

Intelligent Intent Based Networking (I^2BN) on uusi Intent Based Networking-konseptista jatkokehitetty verkonhallinnan paradigma, jossa käytäntöjen ja laitteiden manuaalisen ohjelmoinnin sijaan verkon ylläpitokomentoja on tekoälyn avulla abstrahoitu ja ylläpitäjät hallitsevat verkkoa ilmoittamalla verkolle mitä sen toiminnalta halutaan. [32] I^2BN perusideaa voi verrata perinteiseen verkonhallintaan samalla tavalla kuin deklaratiivista ohjelmointia voi verrata imperatiiviseen. Deklaratiivisessa ohjelmoinnissa kielen komennot ilmoittavat *mitä* ohjelman halutaan tekevän, kun taas imperatiivisessa ohjelmoinnissa ohjelmalle kerrotaan *miten* ohjelman halutaan tekevän. I^2BN :ssä samalla tavoin ylläpitäjät kertovat korkeammalla abstraktiotasolla *mitä* he haluavat verkon toiminnalta, kun taas perinteisesti ylläpitäjät ovat joutuneet matalammalla abstraktiotasolla kertomaan verkon laitteille *miten* he haluavat verkon toimivan konfiguraation muodossa. Tavoitteena on siis järjestelmä, jossa sen ylläpitäjän ei tarvitse tuntea laitekohtaisia toteutuksia ja komentoja, vaan hän pystyy antamaan yleistettyjä käskyjä verkkoa hallittaessa.

Toimiakseen I^2BN tarvitsee huomattavan määrän automaatiota luomaan halutun abstraktiotason ja hoitamaan tehtäviä, joita verkon ylläpitäjät ovat aiemmin hoitaneet. Szilagy [\[32\]](#) määrittelee kolme teknistä vaatimusta I^2BN :n mahdollistamiseksi:



Kuva 4.1. I²BN esimerkkiarkkitehtuuri [32].

- Kyky tehokkaalle verkon datan keräämiselle ja analysoinnille. Kerätty data täytyy olla tietokoneen ymmärrettävissä ja sen konteksti tunnistettavissa.
- Analysoidun datan käsittelyn täytyy tapahtua käytännössä reaaliajassa ja kontekstista tietoisena.
- Verkon täytyy olla ohjelmisto-ohjattu eli SDN-verkko

Käytännön toteutus vaatii kuitenkin edelleen korkeamman tason hallintaliittymän ja SDN-verkon ohjaustason väliin näitä ominaisuuksia hyödyntävän komplekseihin hallintatehtäviin kykenevän automaation. Kuvan 4.1 mukaisessa rakenteessa I²BN-toimintamalli koostuu Intent Managereista (IM) ja verkon hallintalaitteista. Hierarkisesti tasoihin jaotellut IM:t toimivat rakennuspalikoina abstraktiolla. Niiden tehtävä on tekoälyn avulla tulkitä ylemmältä tulevia käskyjen tahtoa ja välittää sen mukaisia uusia käskyjä eteenpäin joko alemman tason IM:lle tai verkon hallintalaitteille. Samalla hallintalaitteilta tuleva data kulkee rakenteessa ylöspäin IM:en tuottamina oivalluksina (insights), eli datasta kontekstin mukaan purettuna tietona. Kun IM:lle syötetään tahto, se pyrkii täyttämään tahdon omalla tasollaan takaisinkytkentäsilmukoiden (closed-loop, CL) avulla. Yksi takaisinkytkentäsilmukka saa ylemmän tason silmukalta jonkin käyttäjän tahdon perusteella luodun ohjeen, käsittelee sen oman tasonsa toteutuksen perusteella ja ohjaa sen eteenpäin. Kunkin CL:n toteutus tulisi olla mikropalveluna toteutettu ohjelma, jolla on oma automaatiologiikkansa. Intent managerit pyrkivät siis luomaan ketjun takaisinkytkentäsilmukoita, jotka hoitavat automaatiokokonaisuuden toteuttamisen.[32] Lähtökohtaisesti rakenne vaikuttaa mutkikkaalta ja luo valtavan työkuorman eri mikroarkkitehtuuriohjelmien toteuttamiselle.

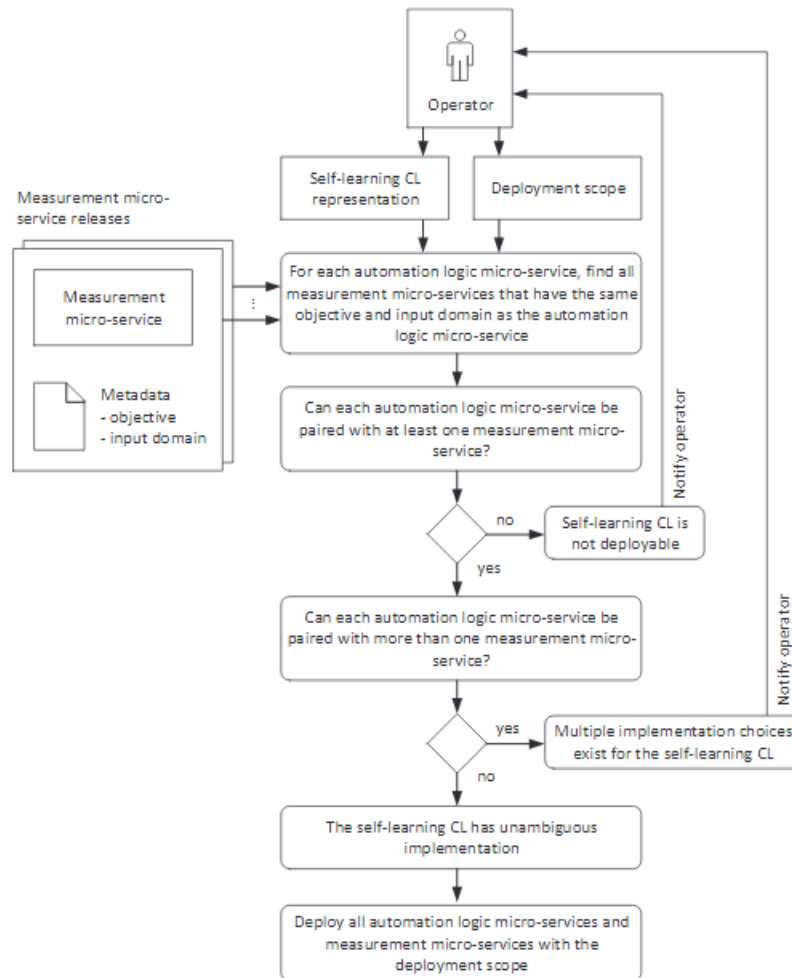
Koska tahtopohjaiset järjestelmät eivät ole eksplisiittisesti tietoisia käytännöistä ja toimintatavoista kussakin tilanteessa, toimiakseen itsenäisesti muuttuvissa tilanteissa niiden täytyy siis kyetä johtamaan ne itse. [32] Tapaukseen, jossa ihmisen käsittelemää dataa on hankala soveltaa, voidaankin hyödyntää ohjaamatonta eli itseoppimista. Yksittäisen CL:n tulisi itseoppimisen avulla kyetä tunnistamaan tehokkain tapa toimia ylläpitäjän asettaman tahdon mukaisesti. Eli CL:n täytyy pystyä johtamaan omalla tasollaan paras tapa toimia, jotta haluttu tila säilyy verkossa tilanteiden muuttuessa.

Tekoälyn fasilitoimiseksi, useista kerroksittaisista ja rinnakkaisista CL:stä koostuvan järjestelmän tulisi siis kyetä keräämään tarvittava data ja siirtämään se järkevästi sopivalle itseoppimismallille. Ei ole kuitenkaan olemassa minkäänlaista standardia sille, miten dataa kerättäisiin CL:ssä. [32] Tämä vaatisi jokaiselle CL:lle oman datankeruutoteutuksen, mikä ei ole kannattavaa. Ratkaisu tälle voisivat olla monikäyttöisemmät mikroarkkitehtuurina toteutettavat erilliset mittausohjelmat, jotka vastaavat datan keräämisestä. Sopivat datankeruu- ja automaatiologiikkaohjelmat voitaisiin yhdistää siihen erikoistuneella funktiolla. [32]

Sopivan mallin valitsemisen lisäksi tekoälyn toteutuksessa pitäisi ainakin pystyä tunnistamaan mallista johtumattomat huonot oppimistulokset sekä mahdollinen yhteensopimattomuus koulutusdatan ja sovellettavan datan välillä. Mallin pitäisi olla valmis myös uudelleen koulutettavaksi käytöstä kerättävällä datalla. [32] Sovelluskohteen laajan ulottuvuuden vuoksi tekoälymallille esitetään melko monimutkaisia ja haastavia vaatimuksia.

Kuvassa 4.2 on esitetty yleispätevä esimerkki siitä, kuinka I²BN-järjestelmää voidaan läheteä toteuttamaan. Kuten kappaleessa 4.1 todettua, on tekoälysovelluksessa ensin tärkeää tunnistaa sovelluksen käyttökohteen laajuus. Tämän jälkeen on määritettävä jokaiselle automaatiologiikalle sopiva datankeruuohjelma. Jos sopivia datankeruuohjelmia löytyy jokaiselle automaatiologiikalle tasan yksi, pitäisi itseoppimismallin määrittäminen olla yksiselitteistä ja järjestelmä pitäisi olla toteuttamiskelpoinen. Jos sopivia datankeruuohjelmia löytyy useampi yhtä automaatiologiikkaa kohden, täytyy järjestelmää uudelleen määritellä tarkemmin.

Toimiessaan I²BN on selkeästi keino vähentää verkonhallinnan luomaa taakkaa ja poistaa tarpeen laitekohtaiselle tekniselle tiedolle ja kokemukselle. I²BN:llä on potentiaalia mullistaa verkonhallintaa ja tehdä FCAPS:n ylläpitämisestä yksinkertaista ja paljon pienemmällä henkilöresursseilla toteutettavaa. Kuitenkin tämä ehdotettu toimintamalli on todella tuore ja tässä vaiheessa tämä massiivinen kokonaisuus on varsinaista käytännön toteutusta vailla. Toistaiseksi on epäselvää esimerkiksi kuinka mahdollista on saada tästä koneoppimisen kannalta toimintavarma ja tehokas. On myös haasteellista, ettei ole olemassa mitään standardia sille, miten CL:t käsittelevät ja keräisivät dataa. Useiden mikroarkkitehtuurina toteutettujen palikoiden hyödyntäminen tämän paikkaamiseksi tekee helposti järjestelmästä kohtuuttoman monimutkaisen ja huonosti optimoidun. Artikkelin onkin pää-



Kuva 4.2. Esimerkki I²BN-järjestelmän suunnittelun työnkulusta. [32]

asiassa arkkitehtuurikuvaus, eikä ota syvempää kantaa käytännön tasolla teknologisiin ratkaisuihin ja valintoihin. Artikkelissa tuodaan kuitenkin hyvin esille järjestelmän mahdolliset edut ja käydään hyvin läpi teoreettiset asiat, jotka mahdollistavat tällaisen arkkitehtuurin mukaisen sovelluksen kehittämisen.

4.3 Tekoäly ja palomuurit

Tekoälyn hyödyntäminen palomuuriratkaisuissa voidaan katsoa kuuluvan Next Generation Firewall-käsitteen sisälle. Koneoppiminen ja tekoäly mahdollistavat uusia ratkaisuja monimutkaisilta uhilta suojautumiseen tarjoamalla mahdollisuuden dynaamiseen järjestelmään, joka kykenee oppimaan ja mukautumaan näihin uhkiin. Tunnetun mallisten uhkien tunnistamisen lisäksi palomuriin voidaan yhdistää tekoäly, jonka avulla palomuri saadaan mukautumaan muuttuviin hyökkäyksiin ja pystyy tällä tavoin dynaamisesti parantamaan verkon tietoturva.

Varsinaista standardia sille, mitä Next Generation Firewall tai tekoälyllä toimiva palomuri tarkoittaa tai pitää sisällään ei ole, mutta teoreettisia sekä käytännön malleja on jo ke-

hitetty. Käytännön toteutukset ovat kuitenkin pääasiassa kaupallisia. Esimerkiksi Huawei toi markkinoille omat ensimmäiset tekoälypalomuurituotteensa jo vuonna 2018. [15] Kaupallisista tuotteista ei kuitenkaan ole julkisesti saatavilla tarkempaa teknistä tietoa tai dokumentaatiota. Niiden lähdekoodi ei luonnollisesti ole myöskään avointa. Akateemisista lähteistä löytyy onneksi kuitenkin joitain hyviä esimerkkejä siitä, millä perusidealla tekoälypalomuuuri voidaan toteuttaa.

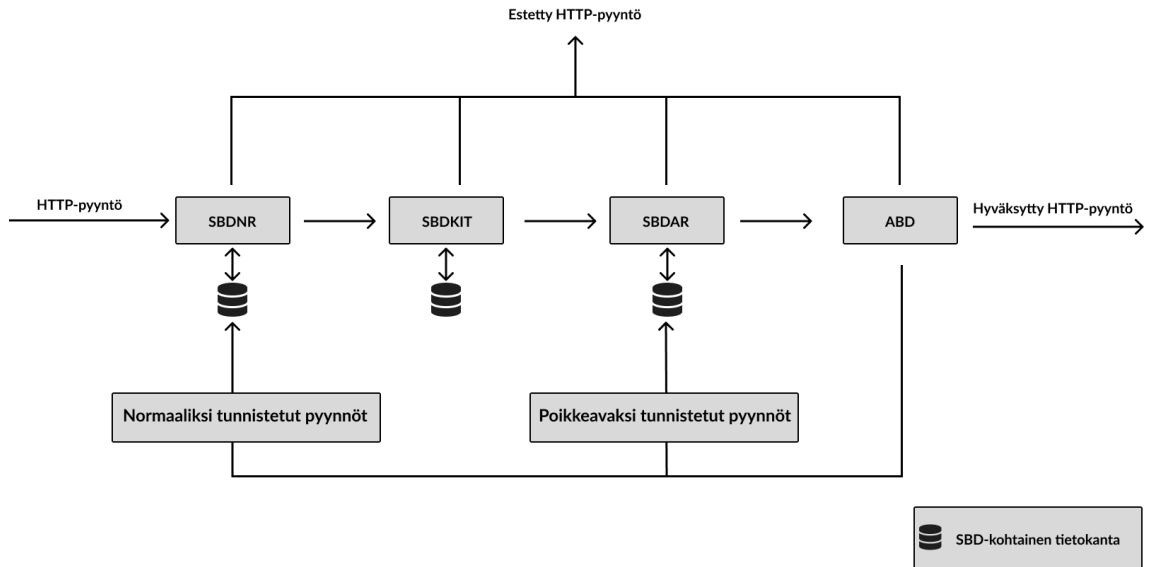
Esimerkiksi verkkosovellusten palomuurit (Web Application Firewall, WAF) ovat hyvä potentiaalinen sovelluskohde tekoälylle. Verkkosivut ovat tietoturvan kannalta haasteellisia sovelluksia. Verkkosivujen kommunikoinnissa käytettävän sovelluserroksessa toimivan HTTP:n dynaamisuus antaa hyökkääjille paljon erilaisia lähestymistapoja verkkosivua vastaan hyökätessä. Tarve verkkosivujen omalle sovelluserroksessa toimivalle palomuurille on tiedostettu jo ainakin 2010-luvun alussa.[30] Vuonna 2019 julkaistu artikkeli "Development of a hybrid web application firewall to prevent web based attacks" on hyvä konkreettinen esimerkki siitä, miten tekoäly pystyy parantamaan verkkosovelluspalomuurin suorituskykyä.

Artikkelissa Tekerek ja Bay [33] esittelevät yhdistetyn WAF-mallin, jossa tunnistepohjainen havainnointi (Signature Based Detection, SBD) on yhdistetty tekoälyn avulla toteutettuun poikkeamien havainnointiin (Anomaly Based Detection, ABD). Tässä mallissa tunnistepohjainen havainnointi koostuu kolmesta osasta:

- Signature based detection of normal requests (SBDNR)
- Signature based detection of known intrusion types (SBDKIT)
- Signature based detection of anomaly requests (SBDAR).

Kuvan 4.3 mukaisesti näillä osilla on omat tietokantansa, jotka sisältävät tunnetun mallista liikennettä. Ne joko estävät pyyntöjä tai päästävät niitä läpi tämän tietokannan perusteella. SBDNR-tietokanta sisältää aikaisemmin normaaliksi havaittua liikennettä. SBDKIT-tietokanta puolestaan sisältää aikaisemmin haitalliseksi havaittua liikennettä. Lopuksi läpi päästetyt viestit ohjataan poikkeaman havainnoinnille, jonka avulla järjestelmä pystyy mukautumaan ja suojaamaan verkkosovellusta uuden tyyppisiltä haitallisilta HTTP-pyyntöiltä. ABD:n tehtävä on toimia yhdessä SBDNR- ja SBDAR-osien kanssa ja päivittää niiden tietokantoja tehtyjen tunnistusten perusteella. SBDNR-kantaan siis jatkuvasti päivitetään normaaliksi tunnistettuja HTTP-pyyntöjä, kun taas vastaavasti SBDAR-kantaan päivitetään haitalliseksi tunnistettuja pyyntöjä. [33] Tällä tavalla järjestelmästä saadaan dynaamisesti haitallisiin pyyntöihin mukautuva ilman tietokantojen ulkopuolista päivittämistä.

ABD:n tekoälyn pohjaksi on valittu yhden piilotetun kerroksen sisältävä hermoverkkotoetus, joka käsittelee tunnistettavia pyyntöjä aakkosnumeerisen esityksen, kirjainmäärien ja pyynnön pituuden perusteella. Hermoverkon koulutusdatana ratkaisussa on käytet-

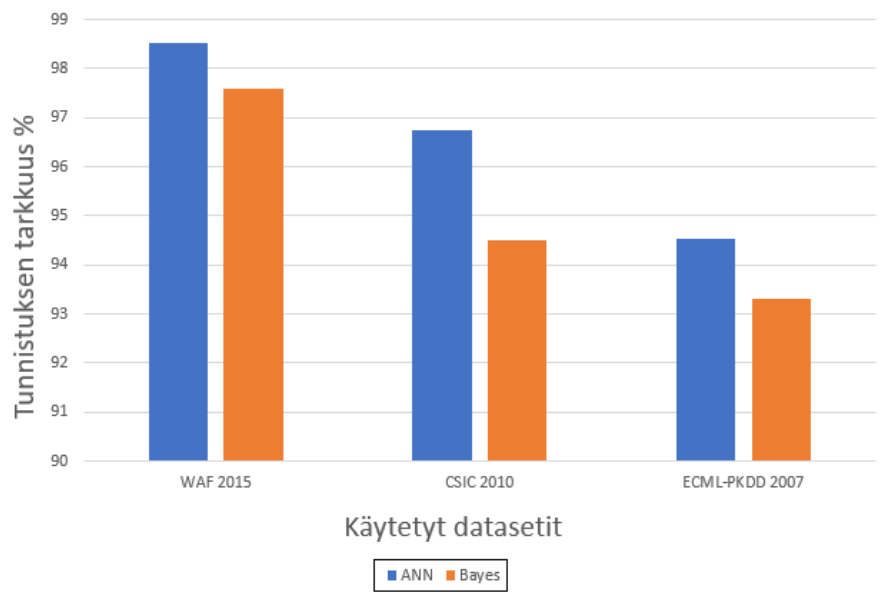


Kuva 4.3. Esitetyn verkkosovelluspalomuurin toiminta.

ty useampaa normaalia HTTP-liikennettä sisältävää aineistoa. [33] Tämä tekoäly on melko yksinkertainen versio hermoverkosta, mutta selkeästi rajatun käyttötapauksen ja analysoitavan datan suhteellisen yksinkertaisuuden puolesta sen pitäisi pystyä tehokkaasti erottelemaan haitalliset HTTP-pyyntöt.

Käytännöntestauksessa Tekerek ja Bay päätyivät siihen lopputulokseen, että pidemmästä haitallisen HTTP-pyyntönnön tunnistusajasta huolimatta poikkeamantunnistus pystyi tunnistepohjaiseen havainnointiin yhdistettynä minimoimaan SBD:n heikkouksia. Kuvaajasta 4.4 voidaan todeta, että tämä hermoverkkoa hyödyntävä malli pystyi tunnistamaan HTTP-pyyntöjä korkeammalla tarkkuudella kuin esimerkiksi aikaisemmin esitetty bayesilaiseen lineaariregressioon perustuva ohjatun oppimisen malli. Artikkelissa on vertailtu hermoverkkototeutuksen tarkkuutta myös muihin kuin bayesilaiseen malliin, mutta bayesilainen malli oli ainut, jonka testauksessa oli käytetty samoja datasettejä kuin hermoverkkomallissa. Hermoverkon tunnistustarkkuus oli kuitenkin näistä kaikista vertailuista korkein. [33] Pidemmän tunnistusajan takia ratkaisun arkkitehtuurin merkitys korostuu. Toiminnan tehokkuuden ylläpitämiseksi on tärkeää, että saapuvat HTTP-viestit ajetaan ensin SBD-osien läpi. Vasta kun ne eivät ole löytäneet siitä haitallista sisältöä, ne luovutetaan hermoverkolle tunnistettavaksi. Näin jo aikaisemmin tunnistetut haitalliset viestit saadaan suodatettua pois ja käsittelyajat pysyvät matalampina.

Tämä esitetty WAF-malli on hyvä esimerkki siitä, miten selkeästi rajatussa käyttötapauksessa koneoppimisen avulla toteutetulla tekoälyllä pystytään automatisoimaan verkkohallintaa tietoturvan näkökulmasta ja helpottamaan FCAPS-viitekehityksen toteuttamista suurilla liikennemäärillä. Verkkosovelluspalomuri on hyvä käyttökohde juurikin sen takia, että se käsittelee vain HTTP-liikennettä, jolloin koulutettavan hermoverkon täytyy oppia



Kuva 4.4. Hermoverkon tunnistustarkkuus verrattuna bayesilaiseen malliin.

tunnistamaan vain tietyn muotoista liikennettä. Tämä puolestaan nostaa edellytyksiä sille, että tunnistus on tarkkaa ja toimii tehokkaasti.

4.4 Tekoälyn hyödyntäminen IoT-ympäristössä

Samaan tapaan kuin muissakin verkoissa, tekoälyä voidaan hyödyntää myös IoT-verkkojen hallinnan ja tietoturvan parantamiseen. IoT-verkonhallintaan käytettävien tekoälysovellusten peruseriaatteet ovatkin siis yleispäteviä muissakin verkko-ympäristöissä. IoT-verkot voivat koostua laajasta määrästä erilaisia laitteita, joilla on rajalliset laiteresurssit. Tästä syystä tekoälyn implementoiminen suoraan näille laitteille ei ole aina mahdollista. Siksi tekoälyn käyttöönotto IoT-verkossa on usein helpointa toteuttaa verkkoon liitetyllä palvelimella tai muulla vastaavalla järjestelmällä, johon kaikki laitteet ovat kytkettyinä. Tekoälyä voidaan siis esimerkiksi soveltaa IoT-ympäristössä yhtäläillä ennakoivaan ylläpitoon, kuin muissakin verkoissa.

Tekoälyn avulla pyritään siis paikkaamaan IoT-ympäristön luomia useita uusia haasteita, kuten tarpeen lisääntyneelle verkon hallinnalle ja tietoturvalle. IoT-laitteiden ja -järjestelmien lisääntyessä tietoturva-aukot ja uhkien määrä kasvaa, mikä edellyttää tehokkaita ja luotettavia tietoturvajärjestelmiä. IoT:n ollessa vielä tuore teknologia sen haasteisiin onneksi kehitetään tällä hetkellä jatkuvasti uusia älykkäitä ratkaisuja.

IoT-markkinoiden räjähdysmäinen kasvu on luonut mahdottoman määrän erilaisia laitteita, jotka verkkoon liitettynä luovat valtavan määrän haavoittuvuuksia, joita on vaikea ennakoita. Etenkin kriittisissä kohteissa, kuten teollisuusautomaatioissa ja sotilaskäytössä oleville laitteille, olisi ensisijaisen tärkeää suorittaa tietoturvaheikkouksien ja aukkojen analysointia penetraatiotestauksella, eli eettisellä hakkeroinnilla. [11] Useiden eri laittei-

den penetraatiotestaaminen käsin on kuitenkin varsin työlästä ja vaatii paljon ammattitaitoa, joten penetraatiotestaus pitäisi pystyä automatisoimaan. Penetraatiotestaus vaatii kuitenkin testaajalta usein oveluutta ja mukautumiskykyä, jota on vaikea simuloida ohjelmallisesti.

IoT-laitteiden penetraatiotestaukseen soveltuvia viitekehyksiä ja työkaluja on ehditty kehittää jo huomattava määrä. [11] Tekoälyratkaisut IoT-laitteiden penetraatiotestauksessa keskittyvät kuitenkin Deep Q-verkkoihin (DQN). Yksinkertaistettuna DQN on vahvistusoppimiseen käytettävä neuroverkko. DQN:t käyttävät Q-tilalukkoja, joilla pidetään kirjaa eri toimista ja tiloista, sekä niiden odotetun suorittamisen palkkiosta osana oppimisen vahvistusta. [34] DQN:llä pystytään rakentamaan tekoäly, joka pystyy toimimaan monimutkaisessakin ympäristössä, jossa on paljon ulottuvuuksia. Kuten esimerkiksi verkossa, jossa on useita eri laitteita ja haavoittuvuuksia. [6] Vuonna 2020 julkaistussa artikkelissa "Autonomous Security Analysis and Penetration Testing" Chowdhary et al. [6] esittelevät Deep Q-verkkoon perustuvan järjestelmän, joka pyrkii analysoimaan kohteensa ja luo penetraatiotestaajalle optimaalisen hyökkäyssuunnitelman testin toteuttamiselle. Lähtökohtana projektissa oli luoda DQN:ään perustuva autonomous security analysis and penetration (ASAP) -algoritmi, joka kartoittaa optimaalisen hyökkäyssuunnitelman penetraatiotestauksen toteuttamiselle.

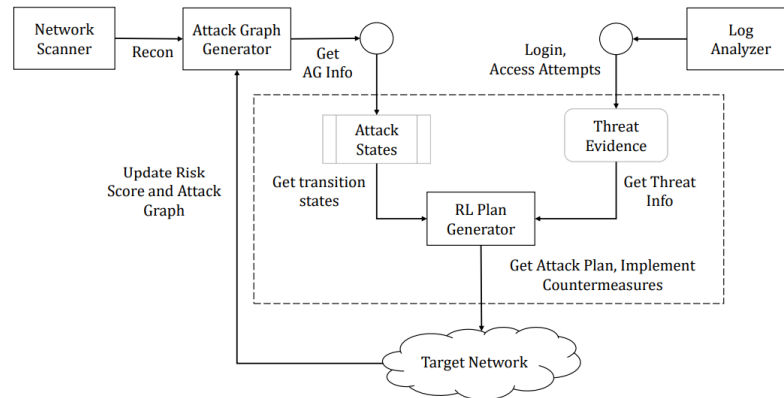
Tavallisesti penetraatiotestaus sisältää seuraavat vaiheet:

- Laajuuden kartoitus ja tiedustelu. Tässä vaiheessa määritellään turvallisuusanalyysin laajuus ja kerätään verkosta saatavilla oleva informaatio, kuten esimerkiksi mahdollinen topologia Nmap-verkkoskannerin avulla.
- Haavoittuvuusanalyysi. Tässä vaiheessa selvitetään haavoittuvuuden skannaus työkaluilla mahdollisia haavoittuvuuksia verkon sisällä.
- Hyökkäyksen suoritus ja raportointi. Tässä vaiheessa suunniteltu hyökkäys toteutetaan ja dokumentoidaan.

Penetraatiotestauksesta saatava lopullinen hyöty ja löytyneet heikkoudet riippuvat luonnollisesti paljon testaajan taidoista ja kokemuksesta. Tekoälyn avulla voidaan kuitenkin paikata mahdollisia testaajan sokeita kohtia. Tekoälyllä ei siis ainoastaan pystytä nopeuttamaan, vaan myös parantamaan penetraatiotestauksen laatua.

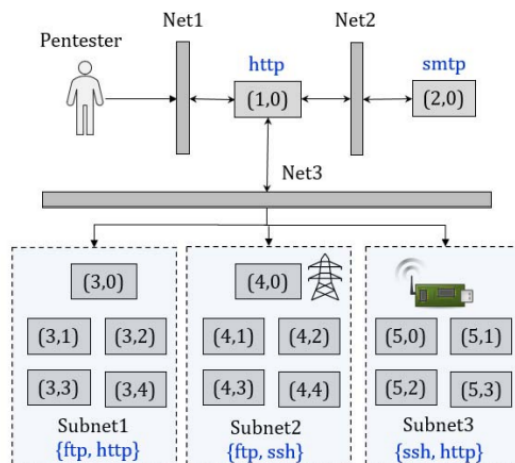
Q-oppimisessa käytännön π Q-funktio mittaa odotettua kokonaispalkkiota eli Q-arvoa tilassa s suoritettavalle toiminnolle a . Näistä toimintoista ja tiloista pidetään kirjaa Q-tilalukkoilla. Deep Q-oppimisessa suorat Q-arvot ja taulukko korvataan funktion approksimaattorilla, kuten hermoverkolla käytännöllisemmän rakenteen saavuttamiseksi. [34] Q-oppimisessa tavoitteena on siis etsiä tilalle s optimaalisin toimi a . ASAP:n tapauksessa Q-arvojen funktioiden estimointia varten on valittu konvoluutioneuroverkko parametreilla $Q(s, a, \theta_i)$, jossa θ_i on neuroverkon mallin parametrit, kuten neuronien painoarvot iteraa-

tiolla *i*.



Kuva 4.5. ASAP-viitekehysten arkkitehtuuri. [6]

Kuvassa 4.5 on esitetty korkeammalla tasolla ASAP:n toimintamalli. Yksinkertaistettuna työjärjestys alkaa ihmisen tekemällä tiedustelulla, jonka perusteella luodaan hyökkäysgraafi, josta saadaan parsittua DQN:lle parametrit s ja a . Chowdhary et al. [6] luomassa esimerkissä on käytetty haavoittuvuuksien tiedusteluun Nessus ja OpenVAS työkaluja. Niistä saatujen tietojen avulla luotiin hyökkäysgraafi MulVAL työkalulla. Hyökkäysgraafi on sen jälkeen tallennettu XML-tiedostona, josta saatiin parsittua tilat ja toimet DQN:lle. Kuvassa 4.5 lohko "RL Plan Generator" tarkoittaa nyt DQN:n avulla tehtyä sovellusta. DQN:n toteutukseen oli tässä esimerkissä käytetty Pythonin PyTorch-kirjastoja. [6]



Kuva 4.6. ASAP-viitekehysten testaukseen käytetyn kohdeverkon rakenne. [6]

ASAP:n toimintaa simuloitiin kuvan 4.6 mukaisessa kohdeverkossa. Testaajien tarkoituksena oli päästä käyttämään hyväkseen verkon SMTP-palvelun haavoittuvuutta ja päästä käsiksi kuvassa esitetyn aliverkon kolme IoT-ympäristöön sen gateway-koneella olevan haavoittuvuuden kautta. Tässä yksinkertaisessa simulaatioverkossa ASAP:n DQN pystyi löytämään optimaalisen hyökkäyksen etenemissuunnitelman vain muutamassa sekun-

nissa. [6] ASAP:n suorituskyky on hämmästyttävän hyvä ja lupaa hyvää tosielämän sovelluksille.

Todistaakseen ASAP:n skaalatuuskyvyn Chowdhary et al. [6] kokeili ASAP:ia myös 300 host-laitteen verkossa, joka sisälsi tasaisesti jakautuneen satunnaisen määrän tunnettuja haavoittuvuuksia. Optimaalisilla asetuksilla ASAP pystyi luomaan hyökkäyssuunnitelman noin 70 sekunnissa. Tämä on massiivinen parannus aikaisemmin kehitettyyn Markovin päätösprosessiin perustuvaan autonomiseen penetraatiotestaukseen, joka kykeni löytämään 7 host-laitteen verkossa hyökkäyssuunnitelman noin 300 sekunnissa. [6] ASAP kykenee siis skaalautumaan loistavasti ja luo todellisen hyödyn penetraatiotestauksen työmäärän pienentämiseksi. ASAP:ia voidaan siis tulevaisuudessa hyödyntää monimutkaisten ympäristöjen kuten IoT:n tietoturvan parantamiseen.

5. YHTEENVETO

Tässä työssä perehdyttiin ensin IP-tietoverkkojen perustekniikoihin ja uudempiin verkko-teknologioihin. On selvää, että perinteisten verkkolaitteiden, kuten kytkimien ja reitittimien varaan rakentuvat verkot eivät enää pysty skaalautumaan halutulla tavalla, ja tekoälylle on tilaa verkkojen ylläpidon parantamisessa. Etenkin SDN-verkot luovat myös pohjan ja mahdollistavat tekoälyn implementoimisen verkonhallinnassa ja tietoturvas-
sa. Työssä tutustuttiin myös lyhyesti koneoppimisen perusteisiin, eri oppimistapoihin ja ongelmiin, joita voidaan ratkaista näillä oppimistavoilla. Kolmannessa luvussa tarkasteltiin erilaisia mahdollisia verkoissa käytettäviä tekoälysovelluksia ja tunnistettiin kaikille kolmelle yleisimmälle oppimistavalle käyttökohteet. Verkkosovelluspalomuurissa oli järkevää käyttää ohjattua oppimista, ohjaamatonta oppimista voidaan soveltaa I²BN-järjestelmässä ja vahvistusoppimista voidaan soveltaa Deep Q-verkkojen avulla IoT-laitteiden penetraatiotestaukseen.

Konkreettisimmat tulokset on näistä kolmesta kuitenkin saavutettu ohjatulla oppimisella verkkosovelluspalomureissa. Verkkosovelluspalomuurien käyttökohde on näistä kolmesta kaikista selkeimmin rajattu, mikä tekee oppimiseen tarvittavan datan määrittelemisestä ja mallintamisesta helpompaa. Myös opetettava yhden kerroksen neuroverkon on näiden kolmen kohteen malleista kaikkein yksinkertaisin, mikä helpottaa mallin parametrien optimointia ja lisää edellytyksiä saada siitä helpommin tarkka ja toimiva.

Vahvistusoppimisen avulla tehtävällä penetraatiotestauksella on selkeä kyky tehostaa kaikenlaisten verkkolaitteiden tietoturvan auditointia merkittävästi. ASAP-viitekehityksessä sovellettava Deep Q-malli antaa realistisen mahdollisuuden tekoälyn avulla tapahtuvalle heikkouksien analysoinnille. I²BN-järjestelmillä on valtava potentiaali mullistaa verkonhallintaa. I²BN-tekniikka vaatii kuitenkin vielä paljon tutkimus- ja kehitystyötä, ennen kuin sen todellista tehokkuutta ja hyötyjä päästään todentamaan.

Kokonaisuudessaan voidaan todeta, että tekoälyn soveltaminen IP-tietoverkoissa on todellakin mahdollista ja jopa kannattavaa. Tekoälyllä on selkeä kyky parantaa verkkojen tietoturvaa sekä hallintaa. Esimerkiksi ASAP:in tapauksessa tekoäly kykenee hämmästyttävän monimutkaiseen autonomiseen toimintaan. Tällä hetkellä kuitenkin tekoälyn implementointi on tehokkainta selkeästi rajatuissa käyttökohteissa, joiden kokonaisvaikutus alalle on ehkä pienempi. Suuremman potentiaalin omaavat ja kompleksisemmat tekoä-

lyjärjestelmät vaativat vielä kehitystä, ennen kuin niistä saadaan käytännön tapauksissa todellinen hyöty irti.

LÄHTEET

- [1] Habtamu Abie. "An overview of firewall technologies". *Teletronikk* 96.3 (2000), s. 47–52.
- [2] Mohammed M. Alani. "OSI Model". eng. Teoksessa: *Guide to OSI and TCP/IP Models*. SpringerBriefs in Computer Science 9783319051512. Cham: Springer International Publishing, 2014, s. 5–17. ISBN: 3319051512.
- [3] Asmaa Shaker Ashoor ja Sharad Gore. "Difference between Intrusion Detection System (IDS) and Intrusion Prevention System (IPS)". Teoksessa: *Advances in Network Security and Applications*. Toim. David C. Wyld et al. Berlin, Heidelberg: Springer Berlin Heidelberg, 2011, s. 497–501. ISBN: 978-3-642-22540-6.
- [4] R. Boutaba ja A. Polyraakis. "Projecting FCAPS to Active Networks". Teoksessa: *2001 Enterprise Networking, Applications and Services Conference Proceedings.. EntNet@SUPERC0MM2001 (Cat. No.01EX543)*. 2001, s. 97–104. DOI: 10.1109/ENTNET.2001.981995.
- [5] Flavio Castro. *Has OpenFlow failed? – Challenges and implementations*. URL: <https://sdn-lab.com/2017/07/11/has-openflow-failed-openflow-challenges-and-implementations/>.
- [6] Ankur Chowdhary et al. "Autonomous Security Analysis and Penetration Testing". Teoksessa: *2020 16th International Conference on Mobility, Sensing and Networking (MSN)*. 2020, s. 508–515. DOI: 10.1109/MSN50589.2020.00086.
- [7] Cisco. *What is network segmentation?* Helmikuu 2023. URL: <https://www.vmware.com/topics/glossary/content/network-segmentation.html>.
- [8] Eric Cole. *Network security bible*. John Wiley & Sons, 2011.
- [9] Lubna Fayez Eliyan ja Roberto Di Pietro. "DoS and DDoS attacks in Software Defined Networks: A survey of existing solutions and research challenges". *Future Generation Computer Systems* 122 (2021), s. 149–171.
- [10] Jazib Frahim, Omar Santos ja Andrew Ossipov. *Cisco ASA: All-in-one Next-Generation Firewall, IPS, and VPN Services*. Cisco Press, 2014.
- [11] Claudia Greco et al. "AI-enabled IoT penetration testing: state-of-the-art and research challenges". *Enterprise Information Systems* 0.0 (2022), s. 2130014. DOI: 10.1080/17517575.2022.2130014. eprint: <https://doi.org/10.1080/17517575.2022.2130014>. URL: <https://doi.org/10.1080/17517575.2022.2130014>.
- [12] E. Haleplidis et al. *Software-Defined Networking (SDN): Layers and Architecture Terminology*. RFC 7462. RFC Editor, tammikuu 2015. URL: <https://www.rfc-editor.org/info/rfc7426>.

- [13] Brian Hedstrom, Akshay Watwe ja Siddharth Sakthidharan. "Protocol efficiencies of NETCONF versus SNMP for configuration management functions". *University of Colorado, Master Thesis* (2011).
- [14] Howard. *What Is Software-Defined Networking (SDN)?* Kesäkuu 2022. URL: <https://community.fs.com/blog/what-is-software-defined-networking-sdn.html>.
- [15] Huawei. *What Is an AI Firewall?* Syyskuu 2021. URL: <https://info.support.huawei.com/info-finder/encyclopedia/en/AI+Firewall.html>.
- [16] Tim Hwang. "Computational power and the social impact of artificial intelligence". *arXiv preprint arXiv:1803.08971* (2018).
- [17] IBM. *Convolutional Neural Networks*. Helmikuu 2023. URL: <https://www.ibm.com/topics/convolutional-neural-networks>.
- [18] Laura Igual ja Santi Seguí. "Supervised Learning". Teoksessa: *Introduction to Data Science: A Python Approach to Concepts, Techniques and Applications*. Cham: Springer International Publishing, 2017, s. 67–96. ISBN: 978-3-319-50017-1. DOI: 10.1007/978-3-319-50017-1_5. URL: https://doi.org/10.1007/978-3-319-50017-1_5.
- [19] Muhammad Iqbal ja Imam Riadi. "Analysis of security virtual private network (VPN) using openVPN". *International Journal of Cyber-Security and Digital Forensics* 8.1 (2019), s. 58–65.
- [20] Mattijs Jonker et al. "Measuring the adoption of DDoS protection services". Teoksessa: *Proceedings of the 2016 Internet Measurement Conference*. 2016, s. 279–285.
- [21] Eda Kavlakoglu. *AI vs. Machine Learning vs. Deep Learning vs. Neural Networks: What's the Difference?* URL: <https://www.ibm.com/cloud/blog/ai-vs-machine-learning-vs-deep-learning-vs-neural-networks>.
- [22] Poonam Sinai Kenkre, Anusha Pai ja Louella Colaco. "Real time intrusion detection and prevention system". Teoksessa: *Proceedings of the 3rd international conference on Frontiers of intelligent computing: theory and applications (FICTA) 2014: volume 1*. Springer. 2015, s. 405–411.
- [23] Hyojoon Kim ja Nick Feamster. "Improving network management with software defined networking". *IEEE Communications Magazine* 51.2 (2013), s. 114–119. DOI: 10.1109/MCOM.2013.6461195.
- [24] Candace Leiden ja Marshall Wilensky. *TCP/IP For Dummies*. John Wiley & Sons, 2009.
- [25] OpenVPN. *Change encryption cipher in Access Server*. Helmikuu 2023. URL: <https://openvpn.net/vpn-server-resources/change-encryption-cipher-in-access-server/>.
- [26] Baiju Patel et al. *Securing L2TP using IPsec*. Tekninen raportti. 2001.
- [27] Tao Peng, Christopher Leckie ja Kotagiri Ramamohanarao. "Survey of network-based defense mechanisms countering the DoS and DDoS problems". *ACM Computing Surveys (CSUR)* 39.1 (2007), 3–es.

- [28] Java T Point. *Regression Analysis in Machine learning*. Helmikuu 2023. URL: <https://www.javatpoint.com/regression-analysis-in-machine-learning>.
- [29] Java T Point. *Unsupervised Machine Learning*. Helmikuu 2023. URL: <https://www.javatpoint.com/unsupervised-machine-learning>.
- [30] Abdul Razzaq et al. "Critical analysis on web application firewall solutions". Teoksessa: *2013 IEEE Eleventh International Symposium on Autonomous Decentralized Systems (ISADS)*. 2013, s. 1–6. DOI: 10.1109/ISADS.2013.6513431.
- [31] Karen Rose, Scott Eldridge ja Lyman Chapin. "The internet of things: An overview". *The internet society (ISOC)* 80 (2015), s. 1–50.
- [32] Péter Szilágyi. "I2bn: Intelligent intent based networks". *Journal of ICT Standardization* (2021), s. 159–200.
- [33] ADEM Tekerek ja OF Bay. "Design and implementation of an artificial intelligence-based web application firewall model". *Neural Network World* 29.4 (2019), s. 189–206.
- [34] TensorFlow. *Introduction to RL and Deep Q Networks*. Joulukuu 2022. URL: https://www.tensorflow.org/agents/tutorials/0_intro_rl.
- [35] Tibco. *What is a Neural Network?* Helmikuu 2023. URL: <https://www.tibco.com/reference-center/what-is-a-neural-network>.
- [36] Yue Wang et al. "Network Management and Orchestration Using Artificial Intelligence: Overview of ETSI ENI". *IEEE Communications Standards Magazine* 2.4 (2018), s. 58–65. DOI: 10.1109/MCOMSTD.2018.1800033.
- [37] James Yu ja Imad Al Ajarmeh. "An Empirical Study of the NETCONF Protocol". Teoksessa: *2010 Sixth International Conference on Networking and Services*. 2010, s. 253–258. DOI: 10.1109/ICNS.2010.41.
- [38] Zhifeng Zhao, Feng Hong ja Rongpeng Li. "SDN based VxLAN optimization in cloud computing networks". *IEEE Access* 5 (2017), s. 23312–23319.
- [39] Wei Zhou et al. "REST API Design Patterns for SDN Northbound API". Teoksessa: *2014 28th International Conference on Advanced Information Networking and Applications Workshops*. 2014, s. 358–365. DOI: 10.1109/WAINA.2014.153.