TRAFICOM
Liikenne- ja viestintävirasto

Huoltovarmuuskeskus
Försörjningsberedskapscentralen
National Emergency Supply Agency

Tekoälyagenttien kyberturvallisuus

Ohjeistus tekoälyagenttien turvalliseen suunnitteluun ja toteutukseen

Traficom
Huoltovarmuuskeskus

Traficomin julkaisuja
5/2026

Sisällys

Sanasto

Tekoälyyn liittyvä sanasto on suomeksi usein vakiintumatonta, ja valtaosa tekoälyä koskevasta materiaalista on englanninkielistä. Alla on eritelty tässä selvityksessä käytettyjä suomennoksia keskeisistä termeistä ja niiden alkuperäinen englanninkielinen vastine. Taulukon tavoitteena on helpottaa lukijaa, joka haluaa etsiä verkosta lisämateriaalia tietystä asiakysymyksestä.

suomi	englanti
arkaluontoisen tiedon paljastuminen	sensitive information disclosure
datan ja mallin myrkyttäminen	data and model poisoning
järjestelmäkehote	system prompt
järjestelmäkehotteen vuotaminen	system prompt leakage
kehote	prompt
kehoteinjektio	prompt injection
koodin etäsuoritus käyttäjän vahvistus	remote code execution human-in-the-loop
laaja kielimalli	large language model, LLM
laukaisin	trigger
liiallinen toimivalta	excessive agency
lujuus	robustness
misinformaatio	misinformation
moniagenttijärjestelmä	multi-agent system
monikerroksinen neuroverkko	deep neural networks, deep learning
palvelunesto(hyökkäys)	Denial of Service (DoS)
perustamalli	foundation model
rajoittamaton kulutus	unbounded consumption
sovelluspohja	solution accelerator, blueprint
suojaustoimet	guardrails
tekstintunnistus	optical character recognition, OCR
toimintaketjun haavoittuvuus	supply chain vulnerability
upotus	embedding
vektorien ja upotusten heikkoudet	vector and embedding weaknesses
vektorointi	embedding
virheellinen tuotosten käsittely	improper output handling

1 Johdanto: Tekoälyagentit ja turvallisuus

Tekoälyn käyttö on kasvanut nopeasti kaikilla toimialoilla. Monikäyttöisten laajojen kielimallien yleistyminen on tuonut markkinoille myös erilaisia tekoälyagenttiratkaisuja, jotka pystyvät suorittamaan tehtäviä itsenäisesti. Tekoälyagentit auttavat käyttäjiä muun muassa löytämään ja jäsentämään tietoa, käymään nopeasti läpi laajoja tekstiaineistoja, kuten ohjeita ja raportteja, automatisoimaan rutiinitehtäviä, ja myös ketjuttamaan erilaisia toimintoja, joihin ihmisten väliintuloa ei välttämättä aina tarvita. Tekoälyagentit kykenevät teknisiinkin tehtäviin, kuten ohjelmointiin sekä erilaisten komentojen luomiseen ja suorittamiseen.

Tekoälyagenttien hyödyntäminen voi olla hyvinkin suoraviivaista, mutta ne tuovat mukanaan myös uusia turvallisuushaasteita. Esimerkiksi verkossa tarjolla olevat ilmaiset ratkaisut, erilaiset valmiit sovelluspohjat, sekä räätälöityjen agenttien kehittämisen suhteellinen helppous osaltaan nopeuttavat agenttien käyttöönottoa. Näin tekoälyagentteja ja niiden erilaisia toteutuksia voi helposti päätyä organisaatioiden käyttöön myös silloin, kun niiden teknisestä tietoturvasta ja tietoturvallisesta käytöstä ei ole varmuutta.

Monissa organisaatioissa agenttien hallintaa on pyritty helpottamaan rakentamalla itse agentteja, jotka huomioivat tietoturvan valmiita ratkaisuita paremmin. Räätälöityjä ratkaisuja löytyy erityisesti sektoreilta, joissa vaatimukset tietoturvan suhteen ovat tavallista korkeammat ja riskit realisoituessaan vakavammat. Itse rakennettu ratkaisu hyödyntää usein valmiita kielimalleja, mutta käyttää niitä tavalla, joka huomioi toimintaympäristöstä nousevan paikallisen lainsäädännön ja tietoturvan parhaat käytännöt.

Vaikka tekoälyagentit voivat tehostaa organisaation toimintaa, niiden käyttö ei aina ole ainoa tai paras ratkaisu. Erityisesti erilaiset jatkuvan muuttujan analyysit, kuten ennustamistehtävät on usein järkevää ratkaista tarkoitukseen suunnitelluilla menetelmillä, kuten erilaisilla regressioalgoritmeilla. Myös erilaiset luokittelu- ja ryhmittelytehtävät, kohteen tunnistukset kuvista tai tekstintunnistus on usein tehokkainta toteuttaa näitä varten kehitetyillä koneoppimismenetelmillä. Nämä klassisemmat koneoppimistoteutukset voivat myös toimia agenttijärjestelmän osana. Ratkaisutapaa valitessa on harkittava, onko ongelma niin monimutkainen tai vaativa, ettei sitä voi ratkaista yksittäisellä menetelmällä, vaan tekoälyagenttia kannattaa kehittää esimerkiksi muun tyyppisen koneoppimisjärjestelmän sijaan tai sellaisen rinnalla. Tässä ohjeessa oletetaan, että eri toteutusvaihtoehtoja on jo harkittu, ja sen perusteella tekoälyagentti on valittu tehtävään parhaaksi ratkaisuksi.

Tämän ohjeen tavoitteena on auttaa organisaatioita suunnittelemaan, rakentamaan ja ylläpitämään agenttisia tekoälyjärjestelmiä turvallisesti. Selvitys nojaa kirjoitushetkellä tuoreimpaan saatavilla olevaan tietoon tekoälyagentteihin liittyvistä yleisimmistä tietoturvariskeistä ja

uhkamallinnuksen parhaista käytännöistä, joiden avulla jokainen organisaatio voi lähteä liikkeelle suunnittelemaan ja rakentamaan tekoälyagentteja ilman tietoturvan kohtuutonta vaarantamista.

Ohjeistus on suunnattu erityisesti henkilöille, jotka työskentelevät tekoälyn ja tieto- ja kyberturvallisuuden parissa. Kappaleet kaksi ja kolme toimivat yleisenä johdantona siihen, mitä tekoälyagentit ovat, miten ne toimivat ja millaisia uhkia niihin liittyy. Kappaleesta neljä eteenpäin aletaan syventymään tekoälyagenttien suunnitteluun ja ohjeistamme, miten agentteihin liittyviä tietoturvauhkia voidaan mallintaa osana riskien arviointia. Lopuksi tutustumme selkeästi selvityksen teknisimpään osuuteen: millaisia turvallisuuskysymyksiä tulisi huomioida järjestelmien rakennusvaiheessa, miten erilaiset tekniset valinnat vaikuttavat turvallisuuteen ja miten tekoälyagenttijärjestelmien tietoturvallisuutta voi testata.

Ohjeistus on toteutettu Traficomin ja Huoltovarmuuskeskuksen Digitaalinen turvallisuus 2030-ohjelman yhteistyönä ja osana tulevaisuuteen varautumista. Selvitys on kirjoitettu yhteistyössä Solitan kanssa. Selvityksen tekoon on osallistunut myös useita yrityksiä ja muita sidosryhmiä.

2 Tekoälyagentit ja niiden erityispiirteet

Tekoälyagentilla tarkoitetaan tässä oppaassa tietojärjestelmää, joka hyödyntää tekoälymalleja ja erilaisia työkaluja toimintojen automaattiseen suorittamiseen käyttäjän puolesta. Agentit voivat siis toimia ja käynnistää toimintoja itsenäisesti niille määriteltyjen sääntöjen ja toimintaohjeiden pohjalta.

Tekoälyagentteja ei siis tule sekoittaa tekoälyavustajiin. Avustajat ovat tyypillisesti keskustelusovelluksia, jotka vaativat käyttäjän jatkuvaa ohjausta kehotteilla. Agentti pystyy puolestaan ketjuttamaan toimenpiteitä sille määritellyssä ympäristössä reagoimalla edeltävien tehtävien tuotoksiin automaattisesti. Agentteja voidaan siis hyödyntää paremmin monimutkaisissa tehtävissä, joiden tehokas ja turvallinen suorittaminen edellyttää useiden toimintojen automaattista ketjuttamista.

Tekoälyagentti ei siis nimestään huolimatta ole älykäs. Se on tietojärjestelmä, jolla ei ole todellista ymmärrystä käyttäjänsä tai kehittäjänsä aikomuksista, toimintojensa oikeellisuudesta tai niiden seurauksista. Vaikka tekoälyagentille annettavaa taustatietoa kutsutaan kontekstiksi, se ei anna tekoälylle mahdollisuutta päätellä annetun tiedon, kuten koulutusmateriaalin, tausta-aineiston ja agentin ulottuvilla olevien verkkolähteiden, ulkopuolelta asioita. Tekoälyn toiminta riippuu siis täysin sen koulutukseen käytetystä datasta, sen algoritmista, sille annettavasta taustamateriaalista ja käyttäjän antamista toimintaohjeista.

Tässä luvussa kuvailemme agenttien perusrakenteen ja erityispiirteet.

2.1 Agenttien perusrakenne

Tekoälyagentit koostuvat kolmesta pääkomponentista: perustamalleista, kehotteista ja agenttiin liitetyistä työkaluista. Kun organisaatio kehittää tai ottaa käyttöön tekoälyagentin, sen on varmistuttava kaikkien näiden komponenttien turvallisuudesta. Tässä oppaassa kerromme tarkemmin, miten eri komponentit suunnitellaan, rakennetaan ja testataan turvallisesti. Komponentit on kuvattu tarkemmin alla.

2.1.1 Perustamalli

Agenttien toiminta perustuu aina taustalla olevaan tekoälymalliin, eli perustamalliin. Generatiiviset tekoälymallit ovat syväoppimismalleja, eli käytännössä isoja, monikerroksisia neuroverkkoja. Niihin kuuluvat mm. kielimallit ja kuvamallit, jotka on suunniteltu käsittelemään ja imitoimaan ihmisten tuottamaa kielellistä materiaalia, kuten tekstiä, kuvia, ääntä ja videota. Mallit, jotka hyödyntävät näistä useampaa yhtä aikaa ovat multimodaalisia.

Nykyisten tekoälyagenttien rakentamisessa käytetään yleensä **laajoja kielimalleja**, jotka on koulutettu käsittelemään luonnollista kieltä myös silloin, kun data on strukturoimatonta. Laajat kielimallit perustuvat todennäköisyyksiin siitä, mitkä sanat esiintyvät tyypillisesti yhdessä niissä tekstiaineistoissa, joita sen kouluttamiseen on käytetty. Jos kielimallilla toteutettu agentti myös generoi tekstiä, sen tuotokset perustuvat todennäköisyyksiin siitä, mitä käyttäjä syötteen perusteella haluaa järjestelmän tuottavan ja millä sanoilla tuotos tulisi ilmaista, jotta käyttäjä kokee tuotoksen oikeaksi.

Laajojen kielimallien lisäksi tekoälyagenteissa voidaan hyödyntää pieniä kielimalleja, joita voidaan käyttää kuluttajille saatavilla olevilla laitteilla. Pienet kielimallit voidaan esimerkiksi hienosäätää tarkempaa tehtävää varten, jolloin niiden tarkkuus on riittävän hyvä agenttijärjestelmiin[1].

2.1.2 Kehotteet

Kehotteella tarkoitetaan järjestelmälle luonnollisella kielellä annettua ohjetta halutun tehtävän suorittamiseksi. Se voi olla yksinkertainen kysymys tai joukko yksityiskohtaisia ohjeita, joita mallin tulee seurata. Lisäksi kehotteella määritellään mitä työkaluja agentti voi käyttää. **Järjestelmäkehotteella** määritellään, miten agentin tulisi käyttäytyä erilaisissa tilanteissa. Järjestelmäkehote on määritelty ennen kuin loppukäyttäjä ottaa agentin käyttöönsä, eikä se yleensä ole käyttäjälle näkyvissä. Myös käyttäjän tekoälymallille antamaa syötettä kutsutaan kehotteeksi.

2.1.3 Työkalut

Työkalut ovat tekoälyagentille annettavia funktioita tai rajapintoja, joita se voi kutsua tarvittaessa. Ne mahdollistavat kielimallille kyvyn suorittaa tehtäviä, joiden toteuttaminen ei onnistu tekstin tuottamisella.

Työkalut voi jakaa kolmeen kategoriaan:

1. **Data:** Työkalut, joiden avulla agentti pystyy hakemaan vastauksen muodostamiseen tarvittavan aineiston.
2. **Toimenpide:** Agentti kykenee tekemään työkalujen avulla toimenpiteitä erilaisissa järjestelmissä, kuten kirjoittamaan tietokantaan.
3. **Orkestrointi:** Agentti voi toimia avustajana muille agenteille tai se voi kutsua muita agentteja samaan tapaan kuin työkaluja.

Esimerkkejä työkaluista, joiden avulla agentti voi parantaa vastauksen laatua tai ratkaista syötteessä määritelty tehtävä:

[1] Belcak ym (2025). Small language models are the future of agentic AI, https://arxiv.org/pdf/2506.02153

- **Verkkohaku:** Tiedon hakeminen internetistä.
- **Laskin:** Kielimallit epäonnistuvat usein jo yksinkertaisimmissa laskutoimituksissa. Laskimen avulla laskuvirheiden määrä saadaan pienemmäksi.
- **Tiedostojen lukeminen:** Luku- ja kirjoitusoikeus tiedostoihin mahdollistaa erilaisten dokumenttien muokkaamisen kielimallien avulla.
- **Tietokannan lukeminen:** Agentille voidaan määrittää pääsy tietokantaan. Kielimallilla voidaan tietokannasta hakea esimerkiksi edellisen kuukauden myyntiluvut.
- **Tekstin tunnistus:** Tekstin tunnistaminen kuvista ja dokumenteista.
- **Kuvien, taulukoiden ja kuvaajien generointi:** Kuvien, taulukoiden ja kuvaajien generointi mahdollistaa agentin käytön esimerkiksi data-analyyseissä.

Työkalujen lisäksi agentti voi kutsua muita agentteja. Tällöin yksittäinen agentti voidaan määritellä ratkaisemaan tarkemmin rajattuja ongelmia pienemmällä määrällä työkaluja. Samalla osan tehtävistä voidaan suorittaa rinnakkain. Näitä ja muita erityyppisiä agentteja kuvataan tarkemmin seuraavassa alaluvussa.

2.2 Agenttijärjestelmien tyypittelyä

Tässä oppaassa jaamme tekoälyagentit kolmeen kategoriaan niiden toteutustavan monimutkaisuuden perusteella. Monimutkaisuuteen vaikuttavat esimerkiksi agentin autonomia ja sen yhteys muihin järjestelmiin ja toisiin agentteihin. Erilaisia agenttityyppejä ja tapoja kategorisoida on useita, ja listauksia löytyy esimerkiksi IBM[2]:n ja Red Hat[3]:n sivuilta. Tyypillisimpiä agentteja ovat:

2.2.1 Yksinkertainen agentti

Yksinkertaisimmat tekoälyagentit kutsuvat työkaluja selkeiden sääntöjen (esim. järjestelmäkehote) perusteella. Agentti voi esimerkiksi käsitellä vastaanotetun tiedoston ja lisätä siihen ohjeiden mukaiset metatiedot. Hieman monimutkaisempi agentti tunnistaa ympäristön tilan ja reagoi työkalujen kutsujen vaikutuksiin (esimerkiksi agentti, joka käsittelee asiakkaan yhteydenoton ja muodostaa asiakkaalle vastauksen). Jos tavoitetta ei saavuteta,

[2] IBM:n agenttityyppien luokittelu: https://www.ibm.com/think/topics/ai-agent-types
[3] Red Hatin tekoälyagenttien luokittelu: https://www.redhat.com/en/blog/understanding-ai-agent-types-simple-complex

agentti toteuttaa uusia toimenpiteitä, kuten muokkaa vastauksensa sisältöä käyttäjän ohjeistuksen perusteella.

2.2.2 Oppiva agentti

Oppiva tekoälyagentti toimii sille määritellyssä ympäristössä ja suorittaa siinä erilaisia toimenpiteitä. Käyttäjä tai toinen agentti arvioi näiden toimenpiteiden seuraukset ja tallentaa palautteen agentin myöhempää käyttöä varten. Kun agentti kerää aiempien tehtävien palautteet talteen, se pystyy mahdollisesti välttämään aiemmat virheet ja ratkaisemaan uudet, samankaltaiset tehtävät nopeammin ja tehokkaammin.

Oppiva agentti on erityisen hyödyllinen monimutkaisissa käyttötapauksissa, joissa agentin ympäristö muuttuu jatkuvasti. Tällaisia käyttötapauksia voi olla esimerkiksi tuotantolaitoksen lokeja seuraava agentti, joka koostaa virhetilanteesta tiketin, tai poikkeustilanteisiin toimenpiteitä suositteleva agentti.

2.2.3 Moniagenttijärjestelmä

Moniagenttijärjestelmä koostuu useista yhteistyötä tekevistä tekoälyagenteista, joiden tavoitteena on ratkaista vaativia tehtäviä. Menetelmässä rakennetaan useita pienempiä tehtäviä ratkaisevia agentteja, joista jokaiselle määritellään vain tarvittavat työkalut. Montaa agenttia käyttämällä voidaan kompensoida yksittäisen agentin tai kielimallin mahdollisia puutteita, kuten kompleksisten tai monipuolista päättelyä vaativien tehtävien ymmärrys ja suorittaminen. Työkalujen määrän vähentäminen parantaa agenttien tarkkuutta työkalujen kutsumisessa ja tehtävien koon rajaaminen voi parantaa tekoälyagentin mahdollisuuksia suorittaa annettu tehtävä. Lisäksi tehtävän jakaminen osiin saattaa mahdollistaa rinnakkaisen laskennan ja näin nopeuttaa tehtävän ratkaisemista.

Moniagenttijärjestelmän suunnittelu, toteutus, testaus ja turvallisen toiminnan valvonta on usein yhden agentin järjestelmää monimutkaisempaa. Moniagenttijärjestelmää suunnitellessa tulee pohtia, miten agenttien yhteistoimintaa koordinoidaan, miten eri agenttien roolit määritellään, miten agentit jakavat keskenään ohjeita ja tuotoksia, ja millainen ihmisen antamien ohjeiden rooli on agenttien yhteistyössä.

Moniagenttijärjestelmä voidaan toteuttaa esimerkiksi seuraavilla tekniikoilla:

1. **Keskitetty järjestelmä:** Yksi agentti vastaa muiden agenttien kutsumisesta ja päättää, milloin käyttäjän määrittämä tehtävä on valmis.
2. **Hajautettu järjestelmä:** Jokainen agentti toimii itsenäisesti ja jakaa tietoa muiden agenttien kanssa.

3. **Hierarkkinen järjestelmä:** Korkeammalla hierarkiassa olevat agentit määrittelevät tehtäviä ja valvovat matalammalla hierarkiassa olevia agentteja.

Yhteistyötä ja agenttien kommunikaatiota varten on hyvä määritellä joukko sääntöjä sille, miten agentit jakavat tietoja, miten ja missä tilanteista ne pyytävät muilta apua ja miten ongelmista tai epävarmuuksista kommunikoidaan. Agenttien tuotoksia ja niiden laatua on myös hyvä pyrkiä valvomaan joko siihen nimenomaisesti määritellyn agentin, tehtävää keskitetysti tai hierarkkisesti ohjaavien agenttien tai ihmisen toimesta.

Yleensä moniagenttijärjestelmällä on myös yhteisiä, jaettuja vaatimuksia, määräyksiä tai ohjeita. Nämä voi määrittää esimerkiksi yhteisellä järjestelmäkehotteella. Jaettavia vaatimuksia tai määräyksiä ovat esimerkiksi järjestelmää käyttävän organisaation arvot, eettisyys, lakien ja määräysten-mukaisuus sekä tieto- ja yksityisyydensuojan vaatimukset.

2.3 Agenttien autonomia

Tekoälyagentti voi käyttää työkaluja joko käyttäjän luvalla tai täysin automaattisesti. Agenttien autonomian tasoa harkitessa on huomioitava tehtävän virheen aiheuttama vahinko ja kuinka haastava tehtävä on. Esimerkiksi agentti, joka voi käyttää komentokehotetta rajattomasti, voi päällekirjoittaa tai poistaa tiedostoja. Tehtävään sopiva ja turvallinen autonomian taso tulee aina tarkastella tapauskohtaisesti.

Tekoälyagenttien väärinkäytön estämiseksi erityisesti itsenäisesti toimivaa agenttia tulee monitoroida ja sille tulee asettaa rajoja. Jos esimerkiksi halutun ratkaisun löytäminen vaatii liian monta välivaihetta, tulee varmistaa, halutaanko ratkaisun kehittämistä jatkaa.

Jos tekoälyagentti on oppiva ja sen autonomian taso on korkea, monitorointiin on kiinnitettävä erityistä huomiota. Mitä enemmän agentin toiminta perustuu käyttäjän tai toisen agentin aiempiin syötteisiin, sitä tarkemmin niiden ennakoituja ja mahdollisia ennakoimattomia vaikutuksia agentin toimintaan on monitoroitava (ks. 7.3.6).

3 Kielimallipohjaisten agenttien keskeisimmät tietoturvahaavoittuvuudet

Tekoälyagenteilla on autonomiaa hyödyntää eri rajapintojen kautta erilaisia työkaluja ja tietokantoja saavuttaakseen sille asetettuja tavoitteita. Tekoälyagentteihin ja tekoälyavustajiin kohdistuu usein samoja haavoittuvuuksia, mutta tekoälyagentit ovat tyypillisesti niiden tehtävien ja autonomian vuoksi korkeamman riskin käyttötapauksia. Moniagenttijärjestelmissä jo yksi kehoteinjektiolla manipuloitu agentti voi saastuttaa koko järjestelmän toiminnan.

Nojaamme tässä ohjeessa OWASPin tuottamiin listoihin, johon on tunnistettu kymmenen keskeisintä laajoihin kielimalleihin ja tekoälyagentteihin liittyvää riskiä ja keinoja niiden hallintaan sovellusten elinkaaren aikana. Tämä ohje auttaa tunnistamaan yleisimpiä riskejä ja suunnittelemaan tekoälyagentteja hyödyntäviä järjestelmiä niin, että riskit hallitaan asianmukaisesti.

Palaamme näihin oppaan eri osissa tekoälyagentin elinkaaren eri vaiheiden näkökulmista ja annamme esimerkkejä niiden mukanaan tuomista riskeistä. Alkuperäiset ja päivittyvät kuvaukset löytyvät OWASPin verkkosivuilta. OWASP julkaisi joulukuussa 2025 erillisen listauksen erityisesti agenttijärjestelmiä koskevista riskeistä. Tarkat alkuperäiset kuvaukset seuraavaksi käsiteltävistä riskeistä, niiden ennaltaehkäisystä ja hallintakeinoista löytyvät edellä mainituista linkeistä.

3.1 Tekoälyagenttien tyypilliset uhkat

OWASP julkaisi joulukuussa 2025 listauksen kymmenestä keskeisimmästä tekoälyagentteihin kohdistuvasta riskistä ja niiden hallintakeinoista[4], jotka avaamme seuraavaksi lyhyesti.

ASI01: Agentin tavoitteen manipulointi

Luonnollisella kielellä kirjoitettujen ohjeiden ja ohjeiden käsittelyyn liittyvien heikkouksien vuoksi agentit ja niiden taustalla olevat kielimallit eivät pysty luotettavasti erottamaan niille annettuja ohjeita hyökkääjän syöttämästä sisällöstä. Hyökkääjät voivat siis manipuloida agentin tavoitteita, sen tekemiä valintoja ja päätöksentekopolkuja kehoteinjektiolla.

ASI02: Työkalujen väärinkäyttö

Agentti voi käyttää sille sallittuja työkaluja haitallisesti, aiheuttaen tietovuotoja, tuotoksen manipulointia ja työnkulkujen häiriöitä. Agentin muisti, dynaaminen työkalujen valinta ja tehtävien delegointi voivat mahdollistaa väärinkäytön ketjuttamiseen, käyttöoikeuksien laajentamiseen ja tahattomiin toimintoihin.

ASI03: Identiteettien ja oikeuksien väärinkäyttö

Hyökkääjä voi hyödyntää agenttien luottamusta ja tehtävien delegointia oikeuksien korottamiseen ja valvontamekanismien ohittamiseen manipuloimalla tehtävien delegointiketjuja, roolien periytymistä ja agentin kontekstia. Kontekstiin kuuluvat esimerkiksi välimuistiin tallennetut tunnistetiedot tai keskusteluhistoria toisiinsa kytkeytyneiden järjestelmien välillä.

ASI04: Toimitusketjuriskit

Kolmansien osapuolten komponentit (mallit, työkalut, datat, agenttirajapinnat) voivat olla haitallisia tai manipuloituja, ja tuoda järjestelmään turvattomia koodeja ja piilotettuja ohjeita.

ASI05: Koodin etäsuorittaminen

Agenttipohjaiset järjestelmät, ml. vibe coding -työkalut, tuottavat ja suorittavat usein koodia. Hyökkääjät voivat hyödyntää koodinluontiominaisuuksia tai agenttiin upotettua työkalupääsyä laajentaakseen toimintaansa etäkoodin suoritukseksi (RCE).

[4] OWASP TOP 10 For Agentic Applications 2026

ASI06: Muistin ja kontekstin manipulointi

Hyökkääjät voivat myrkyttää agentin muistin tai kontekstin (esim. RAG-tietovarastot), mikä vinouttaa päättelyä ja voi johtaa tietovuotoihin tai vaarallisiin toimintoihin.

ASI07: Turvaton agenttien välinen viestintä

Heikko autentikointi ja salaus agenttien välisessä viestinnässä mahdollistaa viestien sieppauksen, manipuloinnin ja väärentämisen.

ASI08: Ketjureaktiot

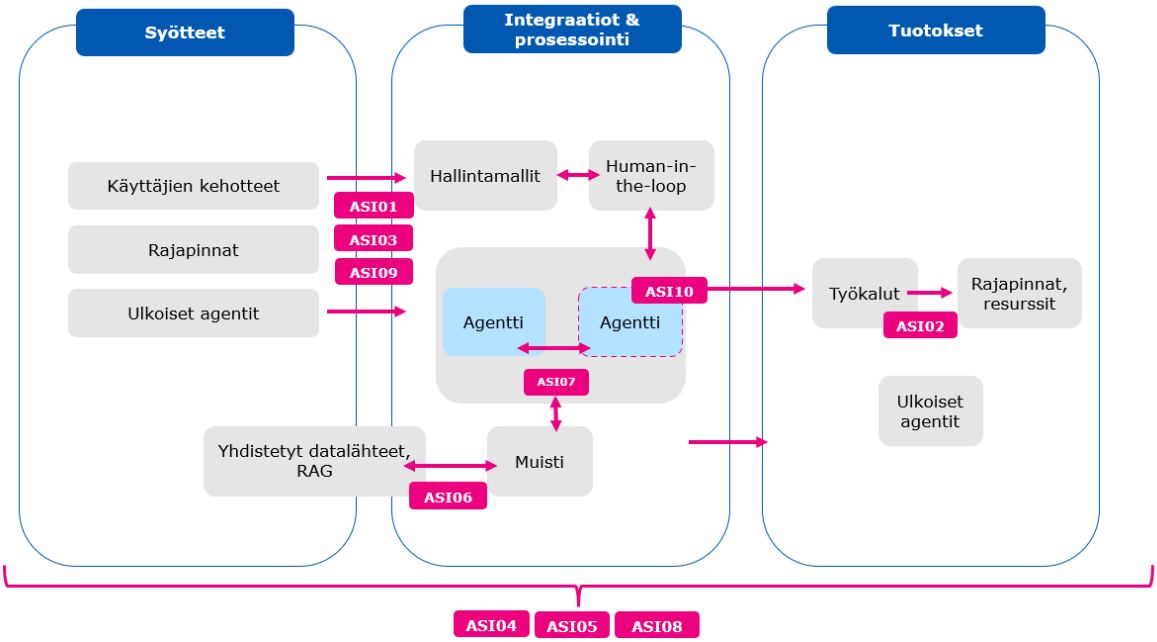
Yksittäinen virhe (hallusinaatio, haitallinen syöte) voi levitä agenttien välillä ja kasvaa laajaksi koko järjestelmää uhkaavaksi riskiksi.

ASI09: Ihmisen ja agentin välisen luottamuksen hyväksikäyttö

Hyökkääjät voivat käyttää hyväksi ihmisen liiallista luottamusta agentin suosituksiin, mikä voi johtaa haitallisiin päätöksiin, taloudellisiin vahinkoihin ja tietovuotoihin.

ASI10: Villiintyneet agentit

Agentit voivat poiketa suunnitellusta toiminnastaan ja sabotoida järjestelmiä, vaikka niiden toiminta vaikuttaa ulospäin oikealta. Agentin villiintyminen voi johtua esimerkiksi kehoteinjektiosta ja toimitusketjujen haavoittuvuuksista, ja aiheuttaa arkaluontoisen tiedon paljastumista, vääristyneen tiedon levittämistä ja operationaalisia ongelmia.



Kuva 1. Tekoälyagentteihin kohdistuvat uhkat. Lähde: OWASP: Top 10 for Agentic Applications for 2026.

3.2 Tekoälyjärjestelmän tyypilliset uhkat

Seuraavaksi käymme läpi OWASPin listaamat kymmenen keskeisintä tekoälyjärjestelmiin kohdistuvaa riskiä[5] ja niiden tyypillisiä ilmenemismuotoja tekoälyagenteissa.

3.2.1 LLM01:2025 Kehoteinjektio

Kehotteen manipulointi eli kehoteinjektio on suurten kielimallien haavoittuvuus, jossa käyttäjä voi haitallisilla kehotteilla muokata mallin toimintaa tai sen tuloksia. Haitalliset kehotteet suunnitellaan siten, että ihmiset eivät välttämättä pysty havaitsemaan niiden sisältöä, vaikka ne ovat tekoälymallille luettavissa. Haavoittuvuus ilmenee, kun malli käsittelee käyttäjän antaman kehotteen, joka sisältää haitallista sisältöä.

Esimerkki haavoittuvuudesta on M365 Copilotin tietojen paljastumisen haavoittuvuus EchoLeak (CVE-2025-32711). Hyökkääjät hyödynsivät injektiota, eli haitallisia komentoja, sähköpostiviesteissä arkaluontoisten tietojen varastamiseen.[6]

Tekoälyagenttiin kohdistuvan hyökkäyksen tavoite voi olla muuttaa agentin toimintaa pitkällä tähtäimellä, mikä vaikeuttaa uhkan havaitsemista. Luonnollisella kielellä toteutettujen ohjeiden ja ohjeiden käsittelyyn liittyvien heikkouksien vuoksi agentit ja niiden taustalla olevat kielimallit eivät pysty luotettavasti erottamaan niille annettuja ohjeita hyökkääjän syöttämästä sisällöstä. Hyökkääjät voivat siis manipuloida agentin tavoitteita (ASI 01), sen tekemiä valintoja ja päätöksentekopolkuja kehoteinjektiolla. Hyökkääjä voi esimerkiksi kehoteinjektiolla pyrkiä manipuloimaan rahaliikennettä hallinnoivaa agenttia, ja pyrkiä ohjaamaan varoja omalle tililleen.

Kehoteinjektiolta voi suojautua mm. rajoittamalla mallin toimintaa, rakentamalla suojaustoimia kehotteelle ja tulosteelle, rajoittamalla agentin oikeuksia ja edellyttämällä ihmisen hyväksyntää kriittisimmissä päätöksissä.

3.2.2 LLM02:2025 Arkaluontoisen tiedon paljastuminen

Arkaluontoisen tiedon paljastaminen tarkoittaa haavoittuvuutta, jossa isot kielimallit voivat paljastaa arkaluontoisia tai luottamuksellisia tietoja vastaustensa kautta. Haavoittuvuutta voi ilmetä tekoälyagenteissa, jos niitä ei ole suunniteltu ja suojattu asianmukaisesti.

Arkaluontoista tietoa voi päätyä mallin saataville koulutusdatan kautta, käyttäjien syötteenä joko tiedostaen tai tahattomasti. Se voi muodostua myös mallin toimintaan vaikuttavan bisneslogiikan kautta, esimerkiksi kun agentilla on pääsy arkaluontoista tietoa sisältäviin tietokantoihin. Tämä

[5] OWASP top 10 for LLM apps & Gen AI agentic Security initiate, v1.
[6] Tapauksen kuvaus saatavilla täältä: https://arxiv.org/html/2509.10540v1.

haavoittuvuus voi johtaa luvattomaan pääsyyn tietoon, yksityisyyden loukkauksiin ja tietovuotoihin

Erityisesti agenttijärjestelmillä arkaluontoisen tiedon paljastuminen johtuu

a) identiteettien ja käyttöoikeuksien väärinkäytön (ASI 03),

b) agenttien välisen turvattoman viestinnän (ASI 07) tai

c) villiintyneiden agenttien seurauksena (ASI 10).

Identiteettien ja käyttöoikeuksien väärinkäytöllä voi ohittaa valvonnan ja laajentaa käyttöoikeuksia. Agentin identiteetti ja autentikointi (API-avaimet, OAuth) ovat kriittisiä, ja ilman selkeitä hallintakeinoja vähimpien oikeuksien periaate ei toteudu. Tässä yhteydessä identiteetti tarkoittaa sekä agentille määriteltyä persoonaa että kaikkea autentikointitunnisteita, jotka edustavat agenttia. Agenttien välinen luottamus tai perityt tunnistetiedot voivat mahdollistaa oikeuksien eskaloinnin, käyttöoikeuksien kaappaamisen tai luvattomien toimien suorittamisen.

Heikko autentikointi ja validointi agenttien välisessä viestinnässä mahdollistaa viestien sieppauksen, manipuloinnin ja väärentämisen, joka voi johtaa arkaluontoisen tiedon paljastumiseen.

Villiintyneet agentit ovat haitallisia tai vaarantuneita agentteja, jotka poikkeavat alkuperäisesti niille tarkoitetuista tehtävistä ja toimivat vahingollisesti tai harhaanjohtavasti (ASI 10). Ne voivat sabotoida järjestelmiä, aiheuttaen tietovuotoja ja työnkulkujen kaappaamista. Uhkaa voidaan hallita erityisesti monitoroimalla tekoälyagentin toimintaa ja minimoimalla sen saatavilla olevaa dataa.

3.2.3 LLM03:2025 Toimitusketjun haavoittuvuus

Laajoja kielimalleja hyödyntävien sovellusten tuotantoketjut ovat alttiita haavoittuvuuksille, jotka voivat vaikuttaa koulutusdatan, mallien ja alustojen eheyteen. Agentit ovatkin usein riippuvaisia kolmannen osapuolen komponenteista, kuten tekoälymalleista, koulutusdatasta, koulutukseen käytettävästä infrastruktuurista ja työvoimasta. Toimitusketjun kautta pahantahtoiset toimijat voivat esimerkiksi jättää järjestelmiin takaportteja ja haavoittuvaisuuksia jo varhaisessa vaiheessa.

Haavoittuvuudet voivat johtaa vääristyneisiin tekoälyagentin vastauksiin, tietoturvaloukkauksiin ja järjestelmävikoihin. Esimerkiksi kolmannelta osapuolelta hankittava koulutusdata voi sisältää ennakkoasenteita tai puolueellista tietoa, haitallista koodia tai muuta haitallista sisältöä, joka vaikuttaa mallin toimintaan tai järjestelmän tuotoksiin.

Avointen kielimallien ja uusien hienosäätömenetelmien, kuten LoRA (*Low-Rank Adaptation*) ja PEFT (*Parameter-Efficient Fine-Tuning*) yleistyminen erityisesti Hugging Facen kaltaisilla alustoilla tuo mukanaan uusia toimitusketjuriskejä, sillä ne ovat kaikkien saatavilla avoimesti, ilman täyttä takuuta tietoturvallisuudesta. Myös laitteessa olevien kielimallien yleistyminen lisää sovellusten hyökkäyspintaa ja toimitusketjuriskejä.

Agenttijärjestelmissä toimitusketjut eivät muodosta vain staattisia riippuvuuksia. Toisin kuin perinteisissä tekoäly- tai ohjelmistotoimitusketjuissa, agenttijärjestelmät hyödyntävät esimerkiksi ulkoisia työkaluja ja käyttöoikeuksia dynaamisesti tehtävän suorittamisen aikana. Toimitusketjuriskejä voi mitigoida esimerkiksi soveltamalla zero-trust -mallia agenttijärjestelmässä.

3.2.4 LLM04:2025 Datan ja mallin myrkyttäminen

Datan myrkyttämisellä tarkoitetaan haavoittuvuutta, jossa tekoälyagentin käyttämää tietoa manipuloidaan haavoittuvuuksien, haitallisen koodin tai esimerkiksi vinoumien tuottamiseksi. Myrkyttäminen vaarantaa mallin suorituskyvyn, tietoturvan tai eettisen toiminnan. Datan myrkyttämisen tarkoituksena on vaikuttaa mallin eheyteen eli käytännössä kykyyn tuottaa agentin omistajan haluamia lopputuloksia.

Myrkyttäminen voi kohdistua tekoälyjärjestelmien eri elinkaaren vaiheisiin, kuten koulutukseen, jolloin malli oppii yleisestä datasta, tai hienosäätöön, jolloin mallia mukautetaan tiettyihin tehtäviin. Näiden lisäksi myrkytys voi tapahtua, kun tekstidataa muunnetaan numeerisiksi vektoreiksi, joita tekoälyagentti hyödyntää vastauksensa generoinnissa.

Myrkyttäminen voi esimerkiksi mahdollistaa takaoven, mikä ei näy mallin toiminnassa ennen kuin tietty laukaisin aktivoi sen. Myrkyttämisen riski on erityisen suuri, kun tekoälyagentin rakentamisessa käytetään avoimen lähdekoodin alustojen kautta jaettuja malleja. Niissä voi esiintyä myrkytetyn datan ja mallin lisäksi myös haittaohjelmia.

Agenttijärjestelmissä hyökkääjä voi erityisesti pyrkiä myrkyttämään agentin muistin ja kontekstin. Hyökkääjä korruptoi tai lisää agentin kontekstiin haitallista tai harhaanjohtavaa dataa, mikä vinouttaa agentin päättelyä, suunnittelua ja työkalujen käyttöä. Esimerkiksi eri syötteiden lähteet, kuten tiedostolataukset, käyttäjäsyötteet ja agenttien välinen tiedonvaihto avaavat mahdollisuuksia muistin ja kontekstin myrkyttämiselle.

3.2.5 LLM05:2025 Virheellinen tuotoksen käsittely

Virheellisellä tuotoksen käsittelyllä tarkoitetaan suurten kielimallien tuottaman sisällön riittämätöntä validointia, puhdistusta ja käsittelyä ennen kuin sitä välitetään eteenpäin muille komponenteille ja järjestelmille. Tekoälyagenttien kohdalla riski syntyy erityisesti silloin, jos agentin tuotosta

käytetään suoraan tehtävien suorittamiseen ilman ihmisen väliintuloa. Tämä on yhteydessä myös liialliseen toimivaltaan, jota esitellään seuraavassa alaluvussa 3.2.6.

Jos suurten kielimallien tuottamaa tekstiä tai koodia ei ole asianmukaisesti validoitu, puhdistettu tai käsitelty ennen sen siirtämistä käyttöliittymiin, tietokantoihin, rajapintoihin tai muille järjestelmille, voi se altistaa virheelliselle tuotoksen käsittelylle. Tämä voi mahdollistaa lisäksi muita haavoittuvuuksia ja hyökkäyksiä, kuten koodin etäsuorituksen (*remote code execution*), XSS (*Cross-Site Scripting*) tai CSRF (*Cross-Site Request Forgery*) -hyökkäyksen tai SQL-injektion.

Agenttipohjaiset järjestelmät – mukaan lukien suositut *vibe coding* -työkalut – tuottavat ja suorittavat usein koodia. Hyökkääjät voivat hyödyntää koodinluontiominaisuuksia tai agenttiin upotettua työkalupääsyä laajentaakseen toimintaansa koodin etäsuorituksella (ASI 05) sisäisiin ja paikallisiin järjestelmiin.

Agentit voivat luoda vahvaa luottamusta ihmiskäyttäjiin luonnollisen kielen sujuvuutensa, tunneälynsä ja koetun asiantuntemuksensa avulla. Hyökkääjät voivat hyödyntää luottamusta ohjatakseen käyttäjän päätöksiä, saadakseen selville arkaluonteista tietoa tai vaikuttaakseen lopputuloksiin haitallisiin tarkoituksiin (ASI 09). Agenttipohjaisissa järjestelmissä riski kasvaa, kun ihmiset luottavat liikaa autonomisen tekoälyn luomiin suosituksiin tai perusteluihin ilman asianmukaista tarkastusta.

3.2.6 LLM06:2025 Liiallinen toimivalta

Kielimallipohjaisille järjestelmille annetaan usein liian paljon itsenäistä toimivaltaa, kuten kyky kutsua funktioita, olla yhteydessä muihin järjestelmiin laajennusten kautta ja suorittaa toimia kehotteiden perusteella. Agentille voidaan määritellä myös päätöksentekovalta kutsuttavan laajennuksen valintaan.

Tekoälyagenttijärjestelmät tekevät tyypillisesti toistuvia kutsuja kielimalleille siten, että edellisten kutsujen tuloksia käytetään seuraavien kutsujen pohjana. Liiallinen toimivalta mahdollistaa vahingollisten toimintojen suorittamisen vastauksena odottamattomiin, epäselviin tai manipuloituihin kielimallin tuloksiin. Agentti toimii siis liian itsenäisesti ja liian suurilla oikeuksilla ilman ihmisen väliintuloa tai vastausten asianmukaista validointia.

Perimmäinen syy haavoittuvuudelle on tyypillisesti joko yksi tai useampi seuraavista:

- liialliset toiminnallisuudet,
- liialliset käyttöoikeudet tai

- liiallinen autonomia.

Tämä haavoittuvuus voi vaarantaa tiedon luottamuksellisuuden, eheyden ja saatavuuden, riippuen siitä, minkä järjestelmien kanssa agentti on vuorovaikutuksessa.

Liiallinen toimivalta on merkittävä riski. Agenttijärjestelmissä hyökkääjä voi hyödyntää hyökkäysvektorina esimerkiksi liiallisia oikeuksia omaavia rajapintoja tai agentille periytyneitä liiallisia käyttöoikeuksia. Liialliset käyttöoikeudet voivat johtaa useisiin eri agenttijärjestelmille ominaisiin uhkiin, kuten ketjureaktioihin, työkalujen väärinkäyttöön tai agentin tavoitteen manipulointiin. Näiden riskien vuoksi käyttöoikeuksien rajaaminen vähimpien oikeuksien periaatteella on keskeinen turvallisuustoimi.

3.2.7 LLM07:2025 Järjestelmäkehotteen vuotaminen

Mallin toimintaa ohjaavat järjestelmäkehotteet voivat sisältää arkaluontoista tietoa, jota käyttäjän ei ole tarkoitus löytää. Järjestelmäkehotteiden tarkoitus on ohjata mallin toimintaa sovelluksen vaatimusten perusteella, mutta ne voivat tahattomasti sisältää myös salaisuuksia. Jos nämä salaisuudet löydetään, niitä voidaan käyttää väärin esimerkiksi osana muita hyökkäyksiä.

Järjestelmäkehotteen paljastuminen ei suoraan aiheuta riskiä. Riski syntyy, jos paljastunut kehote sisältää arkaluontoista tietoa, kuten tietoa järjestelmässä käytetyistä suojaustoimista tai järjestelmän käyttöoikeuksista. Järjestelmäkehotteen paljastuminen voi myös antaa hyökkääjälle tietoa sen suojakeinoista ja tällä tavoin auttaa hyökkääjää ohittamaan ne.

3.2.8 LLM08:2025 Vektorien ja upotusten heikkoudet

Vektorien ja upotusten heikkouksilla tarkoitetaan heikkouksia siinä, miten laajat kielimallit esittävät ja käsittelevät dataa vektoreiden ja upotusten avulla. Upotus tarkoittaa sanojen tai lauseiden muuttamista numeeriseen muotoon vektoreiksi, joiden sijainnit määrittävät niiden keskinäisiä merkityksiä ja suhteita. Upotusten avulla malli ymmärtää tekstin sisältöä, sanojen merkityksiä ja niiden välisiä yhteyksiä. Heikkoudet taas johtuvat siitä, miten teksti muunnetaan vektoreiksi ja miten nämä vektorit tallennetaan ja noudetaan vektoritietokannasta.

Riski on erityisen suuri järjestelmissä, jotka hyödyntävät *Retrieval Augmented Generation* (RAG) -tekniikkaa. RAG-menetelmä rikastaa generatiivisen tekoälyn tuloksia hakemalla tietoa ulkoisista lähteistä, mikä parantaa kielimallipohjaisten sovellusten suorituskykyä ja sisällöllistä relevanssia yhdistämällä esikoulutettuja kielimalleja ulkoisiin tietolähteisiin. RAG-tekniikka onkin yleisesti käytössä tekoälyagenttien rakentamisessa, kun agentin halutaan hyödyntävän organisaation omaa dataa vastausten pohjana.

Näiden heikkouksien avulla voidaan lisätä haitallista sisältöä, manipuloida tekoälyn vastauksia tai etsiä arkaluontoista tietoa. Agenttijärjestelmässä vektorien ja upotusten heikkouksia hyödynnetään erityisesti muistin ja kontekstin myrkyttämiseen (ASI 06).

3.2.9 LLM09:2025 Misinformaatio

Misinformaatio (virheellinen tieto) on yksi keskeisimmistä uhkista suuriin kielimalleihin perustuville sovelluksille. Se voi olla tekoälyn tuottama väärä tai harhaanjohtava tieto, joka vaikuttaa uskottavalta. Yksi väärän tiedon syistä on tekoälyn ns. hallusinaatiot, jotka syntyvät, kun kielimalli pyrkii täyttämään koulutusdatan aukot tilastollisella päättelyllä, ymmärtämättä datan sisältöä tai toimintaympäristöä.

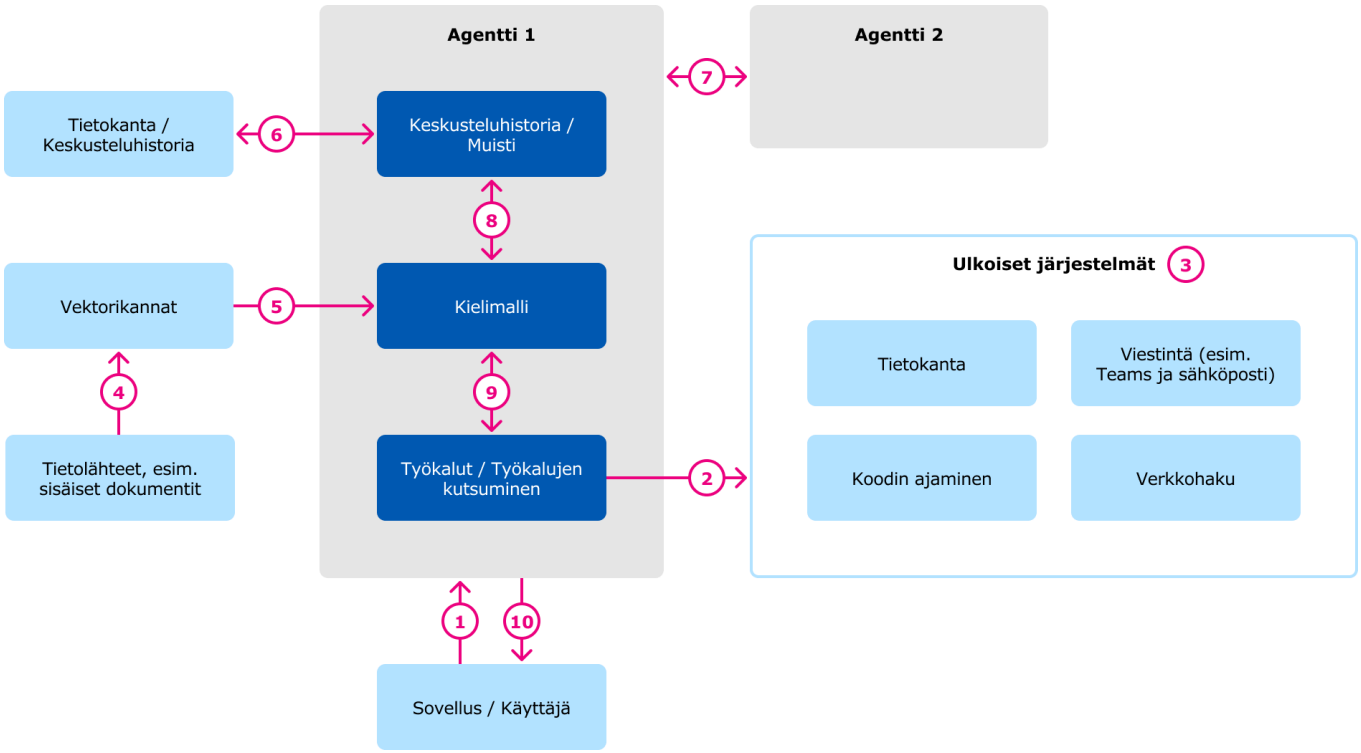
Virheelliseen tietoon liittyvät uhkat realisoituvat usein silloin, jos tekoälyn vastauksiin luotetaan liikaa. Tekoälyn vastausten oikeellisuus on aina varmistettava ihmisen toimesta, etenkin jos niiden perusteella tehdään kriittisiä päätöksiä. Virheellinen tieto voi johtaa moniin ongelmiin, kuten tietoturvaloukkauksiin, mainehaittaan ja oikeudellisin seuraamuksiin.

Agenttijärjestelmässä ihmisen ja agentin välisen luottamuksen hyväksikäyttö (ASI09) ja villiintyneet agentit (ASI10) johtavat usein misinformaatioon.

3.2.10 LLM10:2025 Rajoittamaton kulutus

Rajoittamaton kulutus tarkoittaa tilannetta, jossa laaja kielimalli voi vastata käyttäjän toistuviin kyselyihin ilman rajoitteita.

Tätä haavoittuvuutta käytetään hyökkäyksiin, joiden tarkoituksena on häiritä palvelua, tuhota kohteen taloudellisia resursseja tai varastaa immateriaalioikeuksia kopioimalla mallin käyttäytymistä. Rajoittamatonta kulutusta esiintyy, kun tekoälyagentti sallii käyttäjien tehdä mallille liiallisia ja kuormittavia pyyntöjä, mikä muun muassa kuormittaa järjestelmän kapasiteettia ja tuottaa agentin omistajalle datan prosessoinnista ja generoinnista aiheutuvia kustannuksia. Tämä johtaa riskeihin, kuten palvelunestoon, taloudellisiin tappioihin, mallien varastamiseen ja palvelun laadun heikkenemiseen. Laajojen kielimallien tyypillisesti suuret laskentavaatimukset tekevät niistä alttiita resurssien hyväksikäytölle ja luvattomalle käytölle.



Kuva 2. Tekoälyjärjestelmän haavoittuvuudet. Lähde: OWASP top 10 for LLM apps & Gen AI agentic Security initiate, v1.

1. Kehoteinjektio (LLM01), rajoittamaton kulutus (LLM10)
2. Liiallinen toimivalta (LLM06)
3. Toimitusketjun haavoittuvuus (LLM05)
4. Vektorikannan saastuttaminen varmentamattomilla dokumenteilla (LLM08)
5. Kielimallin käytöksen muutos
6. Muistin saastuttaminen (LLM04)
7. Väärän tiedon jakaminen agenttien kesken, kapinoiva tekoälyagentti
8. Muistin saastuttaminen, tietosuoja
9. Virheellisen/väärän työkalun kutsuminen
10. Hallusinaatio, misinformaatio (LLM09), järjestelmäkehotteen vuotaminen (LLM07), arkaluontoisen tiedon paljastuminen (LLM02), Virheellinen tuotoksen käsittely (LLM05)

4 Tekoälyagentin suunnittelu

Luvussa 3 esiteltyjen riskien tunnistaminen ja hallinta alkaa heti tekoälyagentin suunnitteluvaiheessa.

Suunnitteluvaiheessa tulee miettiä, miksi haluat ottaa käyttöön juuri tekoälyagentin. Miten tekoälyagentti tulisi suunnitella, jotta se on tietoturvallinen? Mitä kannattaa huomioida jo ennen teknisen toteutuksen aloittamista ja ensimmäistä pilottia? Tekoälyjärjestelmien onnistunut käyttöönotto vaatii aina huolellista suunnittelua, jotta lopullinen ratkaisu täyttää sille asetetut odotukset ilman kohtuuttomien tietoturvariskien ottamista.

Tässä osiossa keskitymme tietoturvaan tekoälyagentin suunnitteluvaiheessa. Tutustumme turvallisuuden kannalta keskeisiin kysymyksiin, jotka huomioimalla varmistumme, että suunniteltu tekoälyratkaisu on käyttökelpoinen myös silloin kun sitä käytetään ydinliiketoiminnan kannalta keskeisissä toiminnoissa. Näin etenemme kohti teknologiavalintoja, jotka ratkovat oikeita ongelmia – turvallisesti.

4.1 Turvallisuustietoinen tavoitetilan asettaminen

Tekoälyagentit voivat parhaimmillaan auttaa organisaatioita tehostamaan toimintaansa ja tarjoamaan parempia tuotteita sekä palvelua. Tekoälyagentteihin kohdistuu kuitenkin paljon myös ylimitoitettuja odotuksia, jolloin huolellinen tavoitetilan asettaminen on tärkeää. Tavoitetilan selkeyttäminen auttaa punnitsemaan tietoturvariskejä suhteessa aiottuun käyttötapaukseen ja tuotettavaan arvoon, mikä auttaa priorisoimaan uhkia poistavia toimenpiteitä ja suunnittelemaan alusta asti turvallisia tekoälyjärjestelmiä.

Tekoälyagentti on aina syytä nähdä osana olemassa olevaa prosessia, sillä se ei koskaan toimi erillisenä organisaation muusta toiminnasta. Turvallinen tekoälyn käyttö ja uhkien oikea-aikainen havaitseminen edellyttää ihmisen läsnäoloa, sillä kaikkiin tekoälyratkaisuihin liittyy rajoitteita, joiden vuoksi prosesseja kannattaa vain hyvin harvoin automatisoida täysin (ks. esim. haavoittuvuus luvussa 3.2.6, liiallinen toimivalta). Aloita siis etsimällä vastauksia seuraaviin kysymyksiin:

- **Minkä prosessin osaksi tekoälyagentti tulee? Mikä on prosessin nykytila ja miten haluamme agentin muuttavan sitä?** Kokoa tätä varten yhteen prosessiin normaalisti osallistuvat henkilöt. Jos heitä on paljon (yli ~7 hlöä), valitse mahdollisimman edustava joukko eri näkökulmista. Monipuolinen ryhmä varmistaa, että tavoitetila järjestelmä huomioi eri sidosryhmien näkökulmat kattavasti. Myös tavoitetilaa on syytä pohtia yhdessä prosessiin osallistuvien henkilöiden kanssa, jotta kaikki ovat yhtä mieltä agentin tarpeesta ja tavoitetilasta.

- **Mikä on tekoälyllä haettava positiivinen vaikutus? Kenelle?** Pohtikaa, mikä on se positiivinen vaikutus, jolla agentti vie teitä lähemmäs tavoitetilaa. Onko se tehokkuus, nopeus, laatu, eettisyys, vai jotakin muuta? Kenelle hyödyt kohdistuvat? Mistä voimme tinkiä, mistä emme? Positiivisen vaikutuksen tunnistaminen luo pohjan, jota vasten uhkia tunnistetaan, arvioidaan ja priorisoidaan.
- **Millaisia keskeisiä rajoitteita prosessiin liittyy?** Tietoturvaan liittyvien rajoitteiden tunnistaminen hyvissä ajoin on tärkeää, sillä ne toimivat puitteina ratkaisun suunnittelulle. Onko käsittelemämme data erityisen sensitiivistä? Onko sidosryhmillämme odotuksia, jotka vaikuttavat agentin käyttöönottoon?

Esimerkki: Yritys haluaa tehostaa strukturoimattomien tietovarastojen hyödyntämistä tekoälyagentin avulla. Heillä on useita eri datalähteitä, jotka ovat laadultaan vaihtelevia. Yhteisen keskustelun perusteella tavoitetilaksi määritellään tilanne, jossa asiantuntijat voivat suorittaa tehokkaammin tehtäviä, jotka ovat aikaisemmin vaatineet laajojen tietomassojen manuaalista läpikäyntiä oikeiden toimintojen käynnistämiseksi. Tekoälyagentin avulla manuaalinen tiedonhakuvaihe halutaan ohittaa ja toiminto suorittaa automaattisesti, mikä helpottaa asiantuntijoiden työskentelyä. Muutoksesta hyötyvät sekä organisaatio itse että asiakkaat. Tämä vahvistaa toiminnan vastuullisuutta ja tukee organisaation yhteiskunnallista tehtävää.

Organisaatio tunnistaa kuitenkin myös rajoitteita. Organisaatiota sitovat korkeat tietoturvavaatimukset, minkä vuoksi käytettävissä oleva teknologiavalikoima on rajoittunut. Toiminnallisen turvallisuuden vaatimukset ja muutostenhallinta rajoittavat lisäksi tekoälyn roolia ja korostavat ihmisen harkintaa.

4.2 Alustava tietoturvariskien arviointi osana suunnittelua

Idealle tulee tehdä jo suunnitteluvaiheessa alustava riskiarviointi, jossa tarkastellaan organisaation muiden riskien ohella myös tietoturvariskejä. Alustavaa riskiarviointia täydennetään uhkamallinnuksella (ks. tämän ohjeen luku 5), jossa uhkien todennäköisyyteen ja vaikutusten tunnistamiseen syvennytään tarkemmin. Alustava arviointi antaa kuitenkin pohjan sille, mitä ideoita kannattaa lähteä viemään suunnitteluvaiheesta eteenpäin kohti uhkamallinnusta.

4.2.1 Käyttötapaus

Tekoälyagentin käyttötapaus vaikuttaa merkittävästi sen tietoturvallisuuteen. Kiinnitä suunnittelussa huomiota seuraaviin asioihin ja dokumentoi vastaukset uhkamallinnusta varten.

Käyttötapauksen rajaus. Onko käyttötapaus selkeästi rajattu, vai voiko työkalua käyttää moneen eri työtehtävään? Mitä laajempi käyttötapaus on,

sitä helpompi agentin toimintaan on vaikuttaa pahantahtoisesti. Rajattua käyttötapausta varten agentin toimintaa voidaan rajoittaa mahdollisimman paljon järjestelmäkehotteella sekä muilla teknisillä keinoilla. Jos käyttötapaus on laaja, rajoitteita on vähemmän, jolloin aukkoja syntyy huomattavasti helpommin. Laajassa käyttötapauksessa korostuvat esimerkiksi riskit agentin tuotosten väärinkäyttöön, liialliseen toimivaltaan, järjestelmäkehotteen vuotamiseen ja virheelliseen tietoon.

Käyttötapauksen muuttuminen. Kuinka todennäköisesti käyttötapaus muuttuu matkan varrella? Koska generatiivinen tekoäly on yleiskäyttöistä teknologiaa, käyttäjä voi haluta käyttää agenttia muuhun kuin alkuperäiseen käyttötarkoitukseen. Jos käyttäjillä on paljon moninaisia käyttötapauksia, joihin he haluaisivat käyttää tekoälyä, mutta käytössään vain vähän organisaation hyväksymiä turvallisia työkaluja, voi esiintyä painetta käyttää rajattuunkin käyttöön tarkoitettua agenttia myös muihin tarkoituksiin. Tämä saattaa lisätä riskiä esimerkiksi agentin tuotosten väärinkäyttöön, liialliseen toimivaltaan tai arkaluontoisen tiedon paljastumiseen.

Käyttötapauksen monitorointi. Miten käyttötapauksen ja sen muutosten monitorointi suunnitellaan osaksi ratkaisun arkkitehtuuria? Käyttötapauksen monitorointi ei ole vain tekninen ongelma, sillä se lisää työntekijän tehtäviin kohdistuvaa yksityiskohtaista monitorointia. Siksi jo suunnitteluvaiheessa on syytä miettiä, millaista monitorointia voidaan harjoittaa, jotta käyttötarkoituksen muuttumista ja asianmukaisuutta voidaan mielekkäästi seurata ja haavoittuvuuksia havaita. Monitorointi voi auttaa hallitsemaan riskiä mm. arkaluontoisen tiedon paljastumisesta, järjestelmäkehotteen vuotamisesta, tai rajoittamattomasta kulutuksesta ja havaitsemaan, jos riskit realisoituvat.

Käyttötapauksen alttius väärälle tiedolle. Onko agentti osa prosessia, joka toteutetaan aikapaineessa? Onko agentin toiminnan oikeellisuuden tarkistamiselle riittävästi aikaa ja mahdollisuuksia? Onko agentin käyttötapaus niin suppea ja taustadataa vähän, että agentin toiminnasta on vaikea saada luotettavaa? Isot kielimallit toimivat parhaiten tehtävissä, joihin sopivat mallit toistuvat niiden koulutusdatassa. Näin ollen hyvin erikoistuneet aihealueet ovat agenteille vaikeampia, etenkin jos organisaatiolla ei ole aiheesta riittävän kattavaa tietovaraintoa, jota agentti voisi hyödyntää toimintansa tukena. Virheellisen tiedon levittäminen altistaa organisaation tietoturvariskeille, joten niitä on hyvä pohtia jo käyttötapauksen suunnittelussa.

Käyttäjäryhmän maturiteetti. Tietävätkö agentin käyttäjät, mihin tekoälyn autonominen toiminta perustuu? Osaavatko he arvioida agentin luotettavuutta ja tuotosten oikeellisuutta? Käyttäjän toiminta on yksi tietoturvallisuuden avainkohdista. Siksi jo suunnitteluvaiheessa on tärkeä kerätä

ymmärrystä siitä, kuinka todennäköisesti käyttäjät ymmärtävät, missä tapauksissa agentit ovat luotettavia ja mihin tarkoituksiin agenttia ei kannata käyttää. Tämä auttaa hallitsemaan esimerkiksi tekoälyn tuottaman tiedon väärinkäyttöön, liialliseen toimivaltaan ja virheelliseen tietoon liittyviä riskejä. Tätä ymmärrystä voi syventää esimerkiksi käyttäjähaastatteluilla.

Edellä käsiteltyihin kysymyksiin vastaaminen antaa paremman ymmärryksen siitä, kuinka suuria riskejä agentin käyttöön liittyy suunnitellussa käyttötapauksessa. Jos riskit näyttävät jo alustavasti mittavilta, kannattaa harkita, olisiko jokin toinen käyttötapaus riskeiltään paremmin hallittavissa, tai voisiko ongelmaan soveltaa riskeiltään hallittavampaa teknologiaa. Mittavan riskiluokan käyttötapauksen kanssa etenemistä kannattaa harkita vain silloin, jos tekoälyagentti tuo organisaatiolle mittavaa arvoa ja organisaatiolla on käytössään riittävät resurssit uhkien hallintaan. Edes paljon arvoa tuottavan käyttötapausidean kanssa ei kannata edetä, jos riskit näyttävät ylitsepääsemättömiltä.

4.2.2 Datavarannot

Jos käyttötapauksen riskisyys näyttää hallittavalta, voidaan siirtyä arvioimaan käyttötapauksen vaatiman datan vaikutusta työkalun turvallisuuteen. Kiinnitä suunnittelussa huomiota seuraaviin asioihin ja dokumentoi tiedot uhkamallinnusta varten.

Datan sensitiivisyys ja arkaluontoisuus. Käyttääkö tekoälyagentti hyödykseen arkaluontoista tai sensitiivistä tietoa, kuten liikesalaisuuksia tai henkilötietoja? Onko kaikilla agentin tuotoksia tarkastelevilla käyttäjillä käyttöoikeudet myös lähdedataan? Mitä arkaluontoisempaa dataa agentti hyödyntää, sitä riskialttiimpi ratkaisu on. Yksi kielimallipohjaisten ratkaisujen keskeisistä tietoturvauhkista liittyy juuri arkaluontoisen tiedon paljastumiseen, minkä seurauksena tiedot voivat päätyä vääriin käsiin. Myös datan myrkyttäminen voi vaarantaa tällaisen datan eheyden ja oikeellisuuden.

Datan minimointi. Tekoälyagentin käyttöön haluttaisiin usein antaa laajasti erilaista dataa varmuuden vuoksi. Agentin käytössä oleva data kannattaa kuitenkin rajata mahdollisimman tarkasti, jotta yllä mainittuja riskejä pystytään hallitsemaan. Kun agenttia suunnitellaan, on syytä miettiä, mikä data on oikeasti olennaista käyttötapauksen näkökulmasta ja mikä ylimääräistä. Ylimääräinen data saattaa vaikuttaa negatiivisesti myös agentin toiminnan laatuun ja yhdenmukaisuuteen, joten datan minimointi on myös laatukysymys. Datan minimointi auttaa vähentämään mm. arkaluontoisen tiedon paljastumiseen ja datan myrkyttämiseen liittyviä riskejä.

Käyttöoikeudet. Jo agentin suunnitteluvaiheessa on syytä tunnistaa mahdolliset ristiriidat datan käyttöoikeuksissa. Etenkin jos agentilla on laaja pääsy dataan, voi esiin tulla tilanteita, joissa agentti paljastaa käyttäjilleen tietoja, joihin heillä itsellään ei olisi käyttöoikeuksia. Agentin suodattamasta

tiedosta on myös vaikea nähdä, kenelle käyttäjä voi käyttöoikeuksien näkökulmasta jakaa tietoa, etenkin jos vastaus on yhdistelty useista eri lähteistä. Käyttöoikeuksien hallintaa ja sen tuomia rajoitteita on siksi syytä pohtia jo suunnitteluvaiheessa, jotta suunniteltu käyttötapaus ja ratkaisu on toteutuskelpoinen myös niiden näkökulmasta.

4.2.3 Integraatiot

Tekoälyagentti integroidaan käytännössä aina osaksi organisaation muita järjestelmiä, jos sen halutaan käyttävän toimintansa pohjana organisaation omaa dataa. Suunnitteluvaiheessa on tärkeä tunnistaa, kuinka kriittisiin järjestelmiin tekoälyagentin tulisi päästä käsiksi, jotta ratkaisu saavuttaisi sille asetetut tavoitteet. Erityisesti turvallisuuskriittisten alojen organisaatioiden on tässä vaiheessa harkittava, voiko tekoälyagenttia ottaa käyttöön haluttuun käyttötarkoitukseen, jos siihen vaadittavat integraatiot asettavat esimerkiksi kriittisen infrastruktuurin alttiiksi generatiivisen tekoälyn riskeille.

Jos tekoälyagentti käyttää esimerkiksi samaa resurssia kuin ydinliiketoiminnan kannalta tärkeät järjestelmät, pahantahtoinen käyttäjä voi ylikuormittaa tekoälyagentin kautta koko järjestelmää ja aiheuttaa siten häiriöitä. Se voi lisäksi paljastaa järjestelmäsyötteen kautta salassa pidettävää tietoa organisaation prosesseista ja toimintatavoista.

Jotta kestämättömät riskit voidaan tunnistaa ajoissa, tunnistakaa vastaukset seuraaviin kysymyksiin:

- Mihin kaikkiin järjestelmiin agentti on integroitava, jotta sitä voidaan käyttää haluttuun käyttötarkoitukseen?
- Kuinka kriittisiä järjestelmät ovat tietoturvallisuuden kannalta?
- Kuinka kriittisiä järjestelmät ovat toimintavarmuuden kannalta?
- Jos järjestelmät ovat kriittisiä, voidaanko niihin pääsyä rajata mielekkäästi niin, että tekoälyyn liittyviä riskejä voidaan riittävällä tavalla hallita?
- Voisiko käyttötarkoitus tuottaa arvoa myös ilman yhteyttä kriittisiin järjestelmiin?

Organisaatioiden valmiudet tekoälyagenttien turvalliseen käyttöönottoon ovat tämän oppaan kirjoitushetkellä tyypillisesti matalat, ja parhaat käytännöt agenttijärjestelmien toteuttamiseen ovat vasta kehittymässä. Siksi ydinliiketoiminnan automatisointi tekoälyagenttien avulla on hyvin riskialtista. Jo suunnittelussa on siksi tärkeä tunnistaa, kuinka kriittinen järjestelmäinfrastruktuuri on ja voidaanko agentilla tuottaa tavoiteltua arvoa, vaikka agentin pääsyä kriittisiin järjestelmiin rajoitettaisiin.

4.2.4 Käyttäjien osallistaminen

Käyttäjä on tietoturvan toteutumisen näkökulmasta keskeisessä asemassa. Siksikin on tärkeää, että ratkaisua suunnitellaan yhteistyössä käyttäjien kanssa. Käyttäjien osallistaminen vaikuttaa positiivisesti myös tekoälyagentin laatuun ja käyttöönottoon.

Käyttäjien osallistaminen suunnitteluun varhaisessa vaiheessa auttaa ymmärtämään, miten käyttäjät agenttia tosiasiassa käyttävät, millaista tietoa he sille haluavat syöttää ja millaiset kyvykkyydet heillä on havaita tietoturvariskejä ja toimia havaintojensa pohjalta. Käyttäjät voidaan ottaa mukaan monella tavalla, joista alla listaamme tietoturvallisuuden ymmärryksen näkökulmasta hyväksi havaittuja menetelmiä.

Haastattelut. Haastattelut ovat yksi resurssitehokkaimmista tavoista kerätä ymmärrystä siitä, miten erilaiset käyttäjät haluaisivat tekoälyagenttia käyttää. Haastateltavia kannattaa kerätä erilaisista käyttäjäryhmistä, jotta kerätty tieto kuvaa hyvin agentin todellista käyttöä organisaatiossa. Haastattelut kannattaa toteuttaa suunnitteluvaiheen alussa, jotta niistä kerätystä tiedosta on hyötyä myös uhkamallinnuksessa.

Haastattelujen sisältö riippuu suunnitellusta käyttötapauksesta ja organisaation kontekstista. Seuraavat kysymykset antavat pohjaa siihen, millaisia kysymyksiä haastateltavilta kannattaa kysyä tietoturvahaavoittuvuuksien kartoittamiseksi silloin, kun he eivät ole työkseen tekemisissä tekoälyn tai tietoturvan kanssa:

- Millaisia prosesseja haluaisit automatisoida tekoälyagenttien avulla?
- Millaiseen tietoon tekoälyagentin pitäisi päästä käsiksi, jotta se tarjoaisi hyödyllisiä tuotoksia?
- Onko sinun tiimissäsi/organisaatiossasi/yksikössäsi muita, jotka haluaisivat käyttää tekoälyä johonkin muuhun tarkoitukseen?
- Kuinka todennäköisesti tekoälyagenttia tarvittaisiin muihinkin tarkoituksiin tiimissäsi/organisaatiossasi/yksikössäsi lähitulevaisuudessa? Millaisella aikajänteellä?
- Näetkö, että työssäsi voisi tulla tarve käyttää tekoälyagenttia myös muuhun kuin ensisijaiseen käyttötarkoitukseen? Kuinka todennäköisenä pidät sitä?
- Voisitko antaa esimerkkejä tehtävistä, joita haluaisit nyt antaa tekoälyagentille?
- Miten arvioisit tekoälyagentin tuotosten luotettavuutta kuvailemissasi tilanteissa? Koetko, että sinulla olisi tuotosten tarvittavaan arviointiin ja/tai tarkistamiseen silloin riittävästi aikaa?
- Mihin/miten käyttäisit tekoälyagentin tuotoksia edelleen?

Eri tehtävissä olevilta haastateltavilta kannattaa kysyä erilaisia kysymyksiä heidän työnkuvansa perusteella. Mitä läheisemmin henkilö esimerkiksi työskentelee tietoturvan parissa, sitä suoremmin häneltä voi kysyä tietoturvahaasteisiin liittyviä kysymyksiä.

Yhteissuunnittelu. Ota käyttäjien edustajia mukaan työpajoihin, joissa tekoälyagenttia ja sen käyttötapausta suunnitellaan. Näin käyttäjien suhtautumista agenttiin ja sen käyttötapaukseen voidaan ymmärtää tietoturvan näkökulmasta paremmin jo suunnittelun alkuvaiheessa.

Testaus. Ratkaisun ensimmäisiä versioita kannattaa lisäksi testata yhdessä käyttäjien kanssa hyvin suunnitelluissa testaussessioissa. Testauksesta kerromme lisää ohjeen luvussa 7.

4.3 Tekoälyagentin hankinta

Koska agenttijärjestelmien mahdollisia sovelluskohteita on paljon, myös toteutustapoja on monia erilaisia. Järjestelmän voi hankkia lähes kokonaan palveluna, rakentaa vapaasti saatavilla olevista osasista itse tai yhdistää näitä lähestymistapoja.

Toisinaan tunnistetut tarpeet voidaan ratkaista valmiilla, kolmannen osapuolen kehittämällä tekoälyratkaisulla. Hankintaan päädytään usein etenkin silloin, jos organisaatiolla ei ole resursseja kehittää agenttia alusta alkaen itse tai kumppanin kanssa, tai jos jokin valmiista agenttiratkaisuista sopii hyvin käyttötarkoitukseen.

Agenttia hankkiessa monet turvallisuuteen liittyvät vaatimukset kohdistuvat tekoälyn kehittäjiin. Ostajan on kuitenkin osaltaan varmistettava kilpailutus- ja sopimusvaiheessa, että valittu toimittaja pystyy vastaamaan lainsäädännön vaatimuksiin, ja että vastuut on selkeästi määritelty kilpailutuksen ehdoissa. Näin voidaan välttää epäselvyydet vastuissa, jos tekoälyagentti vaarantaa organisaation tietoturvan.

Oppaan liitteessä 1 on koottuna muistilista asioista, jotka ostajan on hyvä huomioida osana tekoälyagentin hankintaa. Lista kattaa kysymyksiä hallinnasta ja elinkaaresta, toiminnallisuuksista, turvallisuudesta, sääntelystä, läpinäkyvyydestä ja muista keskeisistä näkökulmista. Lista auttaa kilpailutuksen ja hankinnan valmistelussa ja sopimusten laatimisessa, jotta ulkopuolisen kehittämän agentin turvallisuudesta voidaan varmistua mahdollisimman hyvin.

4.4 Kohti toteutusta: riskiarviointi ja EU:n tekoälyasetus

Yllä kerätyt pohjatiedot ovat antaneet kattavan kuvan siitä, miten riskialtis kaavailtu käyttötapaus on tietoturvan näkökulmasta. Näiden tietojen pohjalta tehdään päätös siitä, ovatko riskit hyväksyttävällä tasolla. Kokonaisarvioinnin tukena voi käyttää oheista riskimatriisia (Kuva 3).

	Matala	**Kohtalainen**	**Korkea**	**Erittäin korkea**
Vaikutus ydinliiketoimintaan	Järjestelmä ei suorita ydinliiketoimintaan liittyviä tehtäviä.	Järjestelmä suorittaa helposti hallittavia ja rajoitettuja osia työnkuluista, jotka ovat osa ydinliiketoimintaa.	Järjestelmä suorittaa osia organisaation ydinliiketoiminnasta.	Järjestelmä suorittaa keskeisiä osia organisaation ydinliiketoiminnasta.
Autonomia ja monimutkaisuus	Järjestelmä ei ole integroitu muihin järjestelmiin tai toisiin agentteihin.	Järjestelmä on integroitu vähäiseen määrään ulkoisia järjestelmiä, jotka eivät ole ydinliiketoiminnan kannalta kriittisiä.	Järjestelmä on integroitu useisiin ulkoisiin työkaluihin ja/tai on moniagenttijärjestelmä. Osa integraatioista kohdistuu ydinliiketoiminnan kannalta kriittisiin järjestelmiin.	Järjestelmä on moniagenttijärjestelmä, jossa on useita rajapintoja ja integraatioita ydinliiketoiminnan kannalta kriittisiin järjestelmiin.
Datan arkaluontoisuus	Järjestelmä käsittelee vain julkista tietoa.	Järjestelmä käsittelee organisaation sisäistä tietoa.	Järjestelmä käsittelee luottamuksellista tietoa.	Järjestelmä käsittelee salaista ja arkaluontoista tietoa.

Kuva 3. Matriisi riskien arvioinnin tueksi

Riskiarvioinnissa tulee huomioida ainakin järjestelmän kriittisyys ydinliiketoiminnalle, ratkaisun vaatima autonomian taso ja monimutkaisuus, sekä agentin käsittelemän datan arkaluontoisuus.

Yllä kuvattu alustava riskiarviointi helpottaa arvioimaan tekoälyn riskejä suhteessa suunniteltuun käyttötapaukseen. Sen lisäksi on kuitenkin huomioitava, että myös **EU:n tekoälyasetus** ((EU) 2024/1689) luo tekoälyn käytölle yhteiset säännöt tekoälyn käyttämiseksi reilusti, asianmukaisesti ja läpinäkyvästi. Tekoälyasetus säätelee tekoälyjärjestelmiä niiden riskin perusteella ja asettaa järjestelmän riskiperusteen perusteella niille velvollisuuksia. EU:n tekoälyasetuksen mukaan tekoälyjärjestelmät jaetaan seuraaviin kategorioihin niiden riskin perusteella

- **Kielletyt käytännöt tekoälyjärjestelmissä**: Kielletyt tekoälyjärjestelmien käytännöt ovat käytäntöjä, joiden riskit ovat vastoin EU:n perusoikeuksia, esimerkiksi terveyden ja turvallisuuden näkökulmista. Niitä ei saa tarjota, kehittää, asettaa saataville markkinoille eikä käyttää EU:n alueella, ellei tälle ole lainvalvonnallista perustetta. Tällaisia ovat esimerkiksi kasvojentunnistuksen käyttö massavalvontaan ja sosiaalinen pisteytys.

- **Suuririskiset tekoälyjärjestelmät**: Tekoälyjärjestelmät, jotka aiheuttavat merkittävän haitallisen vaikutuksen kansalaisten terveydelle, turvallisuudelle tai EU:n perusoikeuksille. Tällaisia ovat esimerkiksi kriittisen infrastruktuurin turvallisuuskomponentit, koulutus ja työn ohjaaminen, demokratia ja oikeusprosessit, biometriikka ja kasvojentunnistus. Näiden tarjoajiin kohdistuu suurin osa tekoälyasetuksen vaatimuksista ja velvollisuuksista.

- **Avoimuusvelvoitteiden alaiset tekoälyjärjestelmät**: Tietyille niin sanotuille matalan riskin tekoälyjärjestelmille tekoälyasetus asettaa avoimuuteen liittyviä vaatimuksia ja velvollisuuksia, erityisesti näiden järjestelmien tarjoajille asettamalla muun muassa vaatimuksen merkitä tekoälyn tuottama sisältö (teksti, kuva, video) niin, että se erottuu ihmisen tuottamasta sisällöstä. Tällaisia tekoälyjärjestelmiä voivat olla järjestelmät, jotka on suunniteltu vuorovaikutukseen henkilöiden kanssa, kuten esimerkiksi chatbotit.

- **Muut järjestelmät**: Tekoälyjärjestelmät, jotka eivät kuulu mihinkään muuhun yllä mainittuun kategoriaan. Tekoälyasetus ei aseta näille järjestelmille vaatimuksia.

Jokaisen tahon, joka tarjoaa tekoälyagentin, on siis varmistuttava tekoälyjärjestelmänsä lainmukaisuudesta. Organisaation on tunnistettava oma **roolinsa** suhteessa tekoälyyn (tarjoaja, käyttöönottaja, joku muu), käyttötapaus ja tekoälyjärjestelmän **riskiluokka**, ja riskiluokan mukaiset vaatimukset, jotka omalle roolille kohdistuvat. Myös kriittisen infrastruktuurin ja muun Unionin yhdenmukaistamislainsäädäntöjen parissa olevien alojen on huomioitava heihin kohdistuvat erityisvaatimukset esim. tuoteturvallisuutta koskien. Tekoälyasetuksen soveltuvuuden arvioinnissa voi käyttää esimerkiksi **EU:n omaa tarkistuslistaa**[7], joka auttaa pääsemään alkuun tekoälyasetuksen soveltamisessa.

Tekoälyasetus ja siihen liittyvät ohjeet löytyvät kokonaisuudessaan Euroopan komission tekoälytoimiston verkkosivuilta[8].

[7] Tekoälyasetuksen tarkistuslista EU-komission verkkosivuilla: https://ai-act-service-desk.ec.europa.eu/en/eu-ai-act-compliance-checker.
[8] Euroopan komission tekoälytoimiston verkkosivut: https://digital-strategy.ec.europa.eu/fi/policies/ai-office.

Kun suunnitteluvaiheesta siirrytään kohti teknisiä valintoja, nämä tiedot antavat ymmärrystä teknisten valintojen riskisyydestä valitussa käyttötapauksessa. Seuraavissa luvuissa kuvaillaan tarkemmin, millaisia tietoturvariskejä teknisiin valintoihin liittyy ja millaiset valinnat vahvistavat tekoälyagentin tietoturvallisuutta.

Ennen kuin suunnittelusta siirrytään toteutukseen, ota vielä kerran katse ympäröivään organisaatioon ja olemassa oleviin hallintakeinoihin.

- Onko organisaatiossasi tekoälyn hallintamalli ja sen mukaisia prosesseja, joissa tekoälyagentti pitäisi käyttää?
- Onko organisaatiossasi tekoälyrekisteri tai -katalogi, johon tekoälyagentti tulisi kirjata?
- Onko tietoturvan tai tiedonhallinnan osalta olemassa vielä muita prosesseja tai toimenpiteitä, jotka koskevat kaikkia järjestelmiä ja joita myös tekoälyagentille pitäisi siten tehdä (esim. DPIA)?
- Jos tekoälyagentti tai sen osia hankitaan ulkopuolelta, onko organisaation hankintaohjeessa jotain, mitä meidän pitäisi vielä tietää?
- Olemmeko tutustuneet myös EU:n tekoälyasetukseen ja muuun sääntelyyn ja varautuneet niiden mukanaan tuomiin vaatimuksiin?

Kun nämä asiat huomioidaan ennen toteutukseen siirtymistä, tekoäly tulee osaksi organisaatio kokonaisvaltaista riskienhallintaa, mikä tukee tietoturvallisuuden toteutumista koko tekoälyn elinkaaren aikana.

Jos näyttää siltä, että suunnitteluvaiheessa tunnistetut riskit voidaan kantaa ja agentin kanssa halutaan edetä kohti toteutusta, kannattaa tietoturvauhkat mallintaa tarkemmin, jotta niitä voidaan rajata mahdollisimman tehokkaasti. Syvennymme uhkamallinnukseen tarkemmin seuraavassa luvussa.

5 Uhkamallinnus

Uhkamallinnus on jäsennelty ja järjestelmällinen lähestymistapa sovellusten turvallisuusriskien tunnistamiseen. Se on tärkeä vaihe kehitystyössä, sillä se auttaa tunnistamaan mahdolliset uhkaskenaariot sekä niiden todennäköisyydet ja vaikutukset.

Uhkamallinnus tuo tekoälyagentteihin mahdollisesti kohdistuvat uhkat näkyviksi. Kun uhkat on tunnistettu, voidaan hallintakeinot kohdentaa asianmukaisesti ja kustannustehokkaasti. Näin resursseja käytetään järkevästi ja kohdennetusti, ja organisaatio voi varmistua agentin tietoturvallisuuden riittävästä tasosta oman riskinottokykynsä mukaisesti.

Uhkamallintaminen on hyvä aloittaa projektin alkuvaiheilla, jolloin siitä syntyvät vaatimukset ja havainnot osataan huomioida riittävän ajoissa. Lisäksi uhkamallintamista olisi hyvä tehdä aina, kun projektissa tapahtuu suuria muutoksia esimerkiksi arkkitehtuurin tai henkilöstön osalta. Uhkamallinnus ei ole kertaluonteinen prosessi, vaan sen pitäisi olla jatkuvaa toimintaa. Tässä oppaan luvussa tarkastelemme, miten tekoälylle tyypilliset uhkat voidaan mallintaa.

5.1 Agenttipohjaisten tekoälyratkaisujen uhkamallintaminen

Vaikka tekoälyn turvallisuus pohjautuu vahvasti perinteiseen tietoturvallisuuteen, vaaditaan tekoälyn uhkamallintamisessa perinteisten viitekehysten laajentamista. Vakiintuneet uhkamallinnusmenetelmät, kuten STRIDE [9] ja LINDDUN[10], keskittyvät perinteisten kyberturvallisuusuhkien tunnistamiseen, eivätkä huomioi tekoälyn näkökulmia. Tekoälyagenttien erityispiirteitä on kuvattu tarkemmin luvussa 2.

Yksi tapa jäsentää agenttiratkaisujen uhkia on MAESTRO[11], Multi-Agent Environment, Security, Threat, Risk and Outcome -menetelmä. Se pyrkii täydentämään perinteisiä menetelmiä kerroksittaisella lähestymistavalla, kun uhkamallinnetaan tekoälyratkaisuja. Jokainen uhka yhdistetään aluksi viitearkkitehtuurin kerrokseen, jossa se on ensisijaisesti läsnä tai vakavin. Sen jälkeen se yhdistetään muihin olennaisiin kerroksiin, joiden kanssa uhka on vuorovaikutuksessa tai joihin se voi levitä. Näin monimutkaiset tekoälyagenttiratkaisut voidaan purkaa erillisiin toiminnallisiin kerroksiin, joiden avulla riskejä voidaan ymmärtää ja käsitellä yksityiskohtaisemmin huomioiden myös kerroksia läpileikkaavat uhkat.

5.2 Uhkamallinnusohje

Uhkamallinnus koostuu yleensä kolmesta vaiheesta, joiden sisältöä tarkastellaan seuraavaksi. Itse käytännön järjestelyt voi toteuttaa organisaatiolle sopivien käytäntöjen mukaisesti. Tärkeintä ei ole tietty työskentelytapa tai käytettävä viitekehys, vaan se, että uhkamallinnettavan kohteen kanssa työskentelevät henkilöt pysähtyvät yhdessä pohtimaan mahdollisia uhkia.

Uhkamallintaminen on usein parasta toteuttaa työpajoissa, joko yhdessä pidemmässä tai useammassa lyhyessä. Jokaiseen vaiheeseen on hyvä varata kahdesta kolmeen tuntia aikaa. Jos organisaatiolla ei vielä ole vakiintuneita uhkamallinnuskäytäntöjä, voi työpajoissa käyttää tukena Uhkamallinnuspohjaa liitteessä 2.

5.2.1 Vaihe 1: Uhkamallinnettavan kohteen määrittely

Ensimmäisessä työpajassa määritellään selkeästi, mitä ollaan rakentamassa ja mitä on suojattava. Vaiheen myötä muodostuu selkeä käsitys uhkamallinnettavasta agentista ja siitä, millaisista komponenteista se koostuu, mitä tietoa siellä liikkuu ja miten. Lisäksi tunnistetaan liityntäpisteet muihin järjestelmiin. Uhkamallinnuksessa voi hyödyntää käyttötapaukseen

[9] STRIDE -malli: https://learn.microsoft.com/en-us/azure/security/develop/threat-modeling-tool-threats
[10] LINDDUN viitekehys: https://linddun.org/linddun/whyuselinddun/
[11] Agenttiratkaisujen uhkamallinnuksen viitekehys: https://cloudsecurityalliance.org/blog/2025/02/06/agentic-ai-threat-modeling-framework-maestro

soveltuvaa menetelmää, agenttiratkaisun kohdalla esimerkiksi MAESTROa. Tässä ohjeessa kuvattu tapa soveltuu monenlaisiin uhkamallinnettaviin kohteisiin.

Osallistujien tulee tuntea hyvin agentille suunnitellut ominaisuudet ja sen toimintaympäristö. Mukana on oltava henkilöitä, jotka ovat vastuussa liiketoiminnasta ja sen riskien hallinnasta. Tärkeää on huomioida eri näkökulmat kehitettävään agenttiratkaisuun, jotta uhkamallista saadaan kattava. Työpajaan suositellaan osallistuvan esimerkiksi liiketoimintaomistaja, tuoteomistaja, projektipäällikkö ja arkkitehti.

Esimerkki: Organisaatio suunnittelee tekoälyagentin rakentamista asiakaspalvelun automatisointiin ja tehostamiseen. Ensimmäisessä työpajassa tunnistetaan ja kirjataan tekoälyratkaisun keskeisin suojattava omaisuus kuten asiakaspalvelun prosessikuvaukset, organisaation asiakastietokanta, tekoälyagentin tekniset ratkaisut, agenttia ohjaava dokumentaatio ja käytettävät kielimallit.

Toteutettavaan ratkaisuun liittyvät käyttäjät ja käyttöoikeudet, kuten sisäiset ja ulkopuoliset kehittäjät, ylläpitotiimi ja loppukäyttäjät listataan. Tekoälyagentilla on lisäksi riippuvuuksia mm. rajapintoihin, kolmansien osapuolten kirjastoihin, sekä käytettäviin kielimalleihin. Toteutettavasta ratkaisusta muodostetaan kuvaus, josta nähdään keskeiset komponentit ja tietovirta tavallisessa asiakaspalvelutilanteessa.

5.2.1.1 Suojattavat tiedot ja toiminnallisuudet

Tekoälyagenttien suojattavia kohteita ovat agentin käsittelemä data, ja agentin tärkeimmät toiminnallisuudet sekä muut järjestelmät, joihin agentti kytkeytyy. Kaikki tieto ei ole yhtä kriittistä tai arkaluontoista, joten tässä vaiheessa tunnistetaan ja priorisoidaan mahdollista hyökkääjää kiinnostavat kohteet. Olennaista on tunnistaa, missä tietoa käsitellään tai tallennetaan ja kenellä on siihen pääsy.

Tekoälyjärjestelmistä[12] on hyvä huomioida seuraavat tekijät:

- **Mallit** – koulutetut mallit, mallien parametrit, hyperparametrit
- **Toimijat** – tiedon omistaja, tekoälykehittäjä, mallin kehittäjä/tuottaja
- **Prosessit** – mallin hienosäätö, esikäsittely, tiedon kerääminen
- **Työkalut** – visualisointi, data-analyysi

[12] Stephen Tete (2024), Threat Modelling and Risk Analysis for Large Language Model (LLM)-Powered Applications: https://arxiv.org/abs/2406.11007v1

- **Artefaktit** – mallien käyttötapaukset, metadataskeemat, malliarkkitehtuuri
- **Data** – raakadata, luokiteltu data, validoitu data

5.2.1.2 Käyttäjät ja käyttöoikeuden tasot

Käyttäjät ja kunkin käyttöoikeudet on listattava, jotta eri käyttäjäryhmiin ja niihin sidottuihin rooleihin kohdistuvat uhkat voidaan tunnistaa. Tämä helpottaa myös käyttöoikeuksien hallintaa, mahdollisten hyökkäysreittien mallintamista ja luvattoman pääsyn estämistä. Näin voidaan myös ehkäistä liian laajat käyttöoikeudet omaavien käyttäjäroolien syntymistä.

Ymmärrys käyttäjien oikeuksista auttaa luomaan todenmukaisia hyökkäys- ja uhkaskenaarioita. Keskeisessä roolissa on agentti ja sen käyttöoikeuksien huomioiminen. Agentin identiteetin perusteella se voidaan tunnistaa, sille voidaan myöntää käyttöoikeuksia ja seurata sen toimintaa.

Yleisiä käyttäjäryhmiä ovat ylläpitäjät, sisäiset ja ulkopuoliset kehittäjät, palvelu- ja konetilit (service account / machine-to-machine APIs), sekä eritasoiset palvelun loppukäyttäjät.

5.2.1.3 Riippuvuudet

Riippuvuuksien kautta pyritään tunnistamaan, mihin sisäisiin ja ulkoisiin resursseihin järjestelmän toiminta tukeutuu. Näin voidaan paremmin ymmärtää koko hyökkäyspinta-alaa ja valvoa todentamisen, oikeuksien ja haavoittuvuuksien hallintaa. Mitä enemmän järjestelmällä on integraatioita, sitä vaikeammaksi kokonaisriskiprofiilin hallinta muuttuu.

Seuraavat ovat yleisiä riippuvuuksia agenttipohjaisissa tekoälyratkaisuissa:

- Julkipilviympäristö
- Yksityinen konesali
- Yksityiset verkot ja virtuaaliverkot
- Integraatiot ja niiden rajapinnat (REST, SOAP, GraphQL, WebSocket)
- Kolmannen osapuolen kirjastot ja mallikirjastot
- Koneoppimismallit ja laajat kielimallit, joita agentit käyttävät päättelyyn ja generointiin
- Kolmannen osapuolen tekoälyagentit
- Orkestrointijärjestelmät
- Identiteetin- ja pääsynhallintaratkaisut (IAM)

- Valvonta- ja lokijärjestelmät
- Todennus- ja oikeutuspalvelut
- Ulkoiset datalähteet
- SaaS-palvelut
- Kehitys- ja testityökalut (CI/CD-järjestelmät, simulaattorit, versionhallinta)

Riippuvuudet voivat synnyttää järjestelmälle esimerkiksi seuraavia uhkia:

- Järjestelmän saatavuus voi vaarantua, jos jokin kolmannen osapuolen sovelluspalvelu, rajapinta tai kielimallipalvelu ei ole käytettävissä.
- Hyökkääjä voi saada pääsyn järjestelmään vuotaneiden tunnisteiden, kuten API-avaimien, kautta.
- Luottamuksellista tietoa voi vuotaa mallin palveluntarjoajalle.
- Orkestrointijärjestelmä hallitsee, mitä työkaluja agentti saa käyttää, jolloin orkestraattorin vaarantuessa hyökkääjä voi ohjata agentin tekemään haitallisia kutsuja.

5.2.1.4 Tietovirtakaavio

Tietovirtakaavion tehtävä on havainnollistaa, miten tieto liikkuu järjestelmän osien välillä. Tämä helpottaa tunnistamaan kriittiset kohdat, joissa uhkat voivat realisoitua tai olla mahdollisia hyökkäyspintoja. Tällaisia ovat esimerkiksi salaamaton liikenne, puutteellinen pääsynhallinta, epäluotettava tietolähde tai suojaamaton rajapinta.

Tietovirtakaaviosta voidaan nähdä, mihin dataan tekoälyagentilla on pääsy, minne se voi itse lähettää dataa, ja millaisia komentoja se voi muihin järjestelmiin suorittaa. Kaaviossa tulisi kuvata tiedon keräämisen, prosessoinnin ja säilytyksen menetelmät ja sijainnit. Lisäksi kaaviossa tulee aina kuvata luottamusrajat, joissa todentamisen tulee tapahtua. Tietovirtakaavion päivittäminen varmistaa, että järjestelmän dokumentaatio on ajan tasalla.

5.2.2 Vaihe 2: Uhkien tunnistaminen

Toisessa työpajassa kartoitetaan mahdollisia uhkatoimijoita, heidän motivaatioitaan ja hyökkäysvektoreitaan. Hyökkäysvektorilla tarkoitetaan polkua, menetelmää tai skenaariota, jota hyödyntäen päästään murtautumaan järjestelmään.

Vaiheen tavoitteena on tunnistaa mahdollisimman monta potentiaalista uhkaa riippumatta siitä, kuinka todennäköisiltä ne vaikuttavat. Edellisen työpajan tuloksia hyödyntäen tunnistetaan missä uhkat voivat ilmetä.

Tunnistamisen tukena on hyvä käyttää tunnettuja viitekehyksiä ja uhkalistoja, esimerkiksi:

- OWASP Top 10 listaus merkittävimmistä tekoälyjärjestelmiin ja tekoälyagentteihin liittyvistä haavoittuvuuksista luvussa 3.
- MITRE ATLAS (Adversarial Threat Landscape for Artificial-Intelligence Systems) on maailmanlaajuinen ja jatkuvasti elävä tietokanta taktiikoista ja tekniikoista tekoälyjärjestelmiä vastaan: MITRE ATLAS™
- OWASP ohje agenttipohjaisten tekoälyratkaisujen uhkamallintamiseen: Multi-Agentic system Threat Modelling Guide v1.0 - OWASP Gen AI Security Project

Lisäksi on hyvä huomioida myös perinteiset kyberturvallisuusuhkat, joiden läpikäynnissä voi käyttää esimerkiksi OWASP Top 10 -listausta[13]. Myös tietosuojanäkökulma on tärkeä ottaa huomioon, sillä tieto on jokaisen tekoälyjärjestelmän ytimessä.

Uhkien tunnistamiseen osallistuvilla on hyvä olla ymmärrys siitä, mikä tyypillisesti voi mennä pieleen teknisestä ja liiketoiminnallisesta näkökulmasta. Osallistujien valinnassa painotus riippuu jonkin verran tarkastelun kohteesta ja halutusta laajuudesta. Uhkia voidaan tarkastella koko kehitettävän kohteen laajuudelta tai työpajoissa voidaan keskittyä tiettyyn osa-alueeseen tai agentin elinkaaren vaiheeseen. Osallistujia voivat olla esimerkiksi arkkitehti, tietoturvavastaava, tuote- tai järjestelmäomistaja ja muut soveltuvat asiantuntijat.

Uhkien tunnistamisen myötä rakennettavan tekoälyagentin uhkamalli alkaa muodostumaan. On tärkeä kirjata myös ne uhkat, joille on jo määritelty ja toteutettu hallintakeinot. Uhkamalli elää kehityksen mukana ja kattava uhkamalli onkin olennainen uhkaympäristön seurannan työkalu.

Esimerkki: Organisaation asiakaspalvelua automatisoivan agentin tapauksessa huomiota kiinnitetään erityisesti agentin toimintavarmuuteen liittyviin uhkiin, kuten resurssien ylikuormittamiseen, haitallisiin kehotteisiin, mallin manipulaatioon ja työkalujen väärinkäyttöön. Näiden uhkien tarkoituksena on haitata agentin toimintaa, mikä voi johtaa häiriöihin toiminnassa ja pahimmillaan estää sen kokonaan.

Uhkamallinnuksen yhteydessä havaitaan, että ala ja käytettävät teknologiat kehittyvät ja muuttuvat nopeasti, joten uhkamallinnusta päätetään tehdä kvartaaleittain.

[13] OWASP Top 10: https://owasp.org/Top10/

5.2.3 Vaihe 3: Uhkien arviointi ja hallintakeinojen määrittäminen

Kolmannessa työpajassa uhkien tärkeysjärjestys määritetään niiden todennäköisyyksien ja vaikutusten perusteella. Näin voidaan määrittää ja toteuttaa lieventämistoimia riskien vähentämiseksi ja järjestelmän turvallisuuden parantamiseksi resurssitehokkaasti tärkeysjärjestyksessä.

Hallintakeinoja pohtiessa voidaan hyödyntää esimerkiksi NIST[14] Cybersecurity Frameworkin määrittelemiä toimintoja: suojautuminen, havainnointi, reagointi ja palautuminen. Päätökset uusien hallintakeinojen toteuttamisesta ja niiden aikataulutus kirjataan projektin käytäntöjen mukaisesti. Konkreettisille hallintakeinoille tulee sopia vastuuhenkilöt, jotka huolehtivat hallintakeinojen viemisestä käytäntöön esimerkiksi kirjaamalla ja vastuuttamalla toimet tiketöintijärjestelmään.

Viimeisen työpajan aikana keskustellaan iteraatio- ja testauspäätöksistä ja seuraavan uhkamallinnuksen aikataulutuksesta. Työpajan osallistujilla tulee olla ymmärrys erityyppisten uhkien vaikutuksesta liiketoimintaan ja riskienhallintaan. Teknisissä uhkissa voidaan tarvita ymmärrystä ympäristön erityispiirteistä, kuten olemassa olevien hallintakeinojen vaikutuksista.

Uhkien arvioinnin myötä uhkamalli tarkentuu, ja muodostuu käsitys siitä, kuinka todennäköisiä uhkat ovat ja millaisia vaikutuksia niillä voi olla. Uhkamalli on dokumentaatio siitä, että uhkat on tunnistettu ja riskit otettu hallintaan.

Esimerkki: Uhkamallinnuksen lopputuloksena organisaatiolle muodostuu uhkamalli, joka sisältää tunnistetut uhkat, sekä niiden todennäköisyyksien ja vaikutusten arviot. Uhkille on määritelty asianmukaiset hallintakeinot sekä suunniteltu niiden toteuttaminen ja aikataulu. Agenttiratkaisun kehittämisessä otetaan huomioon sille tyypilliset uhkat. Uhkille toteutetaan hallintakeinot, joita voivat olla esimerkiksi pyyntöjen määrän rajoittaminen ja syötteen sekä tulosten suodattaminen.

Vastuuhenkilö vie hallintakeinojen toteutuksen kehitystyön työjonoihin ja huolehtii siitä, että uhkamallia päivitetään, kun hallintakeinot on otettu käyttöön. Uhkamallin läpikäynti aikataulutetaan seuraaville kvartaaleille sovitusti, ellei kehitystyössä ilmene tarvetta käydä uhkamallia läpi aiemmin.

On suositeltavaa määrittää, minkälaiset muutokset ovat niin merkittäviä, että ne vaativat uhkamallin uudelleen käsittelyn. Tällaisia muutoksia voisivat olla esimerkiksi mallin päivitys, uusi malli, uusi integraatio, laajennus uuteen käyttötapaukseen tai muutokset järjestelmän infrastruktuurissa.

[14] National Institute of Standards and Technology (NIST) Cybersecurity Framework (CSF): https://www.nist.gov/cyberframework

6 Tekoälyagentin rakentaminen

Kun uhkat on mallinnettu, hallintakeinot määritetty ja vastuutettu, voidaan edetä tekoälyagentin rakentamiseen. Tässä luvussa kuvataan, miten erilaiset rakennusvaiheen valinnat vaikuttavat agentin turvallisuuteen yleisesti tunnettujen riskien pohjalta. Luvussa kuvaillaan datan valintaan, käsittelyyn ja säilyttämiseen liittyvät valinnat, laajoihin kielimalleihin liittyvät huomiot ja agentin operointiin liittyvät kysymykset. Näiden ohjeiden avulla organisaatio voi rakentaa itselleen agentin tämän hetken parhaiden turvallisuuskäytäntöjen mukaisesti.

6.1 Data ja sen turvallinen hallinta

Datan on tekoälyagenttien kehittämisessä keskeisessä roolissa. Jos data on heikkoa tai vääristynyttä, agentilla ei voida saavuttaa hyviä tuloksia. Lähdedatan laatua, turvallisuutta ja arvoa on hyvä arvioida jo dataa hankittaessa ja ennen sen hyödyntämistä agenttijärjestelmien kouluttamisessa, testaamisessa ja varsinaisessa käytössä. Näitä toimia kuvataan tarkemmin alla.

6.1.1 Datan ja lähteiden arviointi

Datan arvioinnin tavoitteena on muodostaa käsitys siitä, miltä osin data on turvallista ja käyttökelpoista, tuleeko sitä esimerkiksi hankkia lisää ja miten sitä täytyy esikäsitellä.

Taulukkomuotoista dataa voidaan arvioida esimerkiksi listaamalla muuttujat, laskemalla näytemäärät ja puuttuvien arvojen osuudet sekä tarkastelemalla jakaumia, trendejä ja poikkeavia arvoja. Poikkeamat tai puuttuvat tiedot voivat viitata datan valikointiin tai manipulointiin. Tekstidataa analysoitaessa tunnistetaan kielet, vertaillaan sanaston yleisyyttä yleiskieleen, arvioidaan ammattitermien osuutta ja tarvittaessa tehdään sentimentti-, entiteetti- ja aiheanalyysiä. Samalla datasta etsitään haitallista sisältöä, jonka tavoitteena on manipuloida kielimallia. Kuvien, videoiden ja äänen laadun ja turvallisuuden arvioinnissa voidaan hyödyntää tunnistamista, luokittelua, segmentointia ja sisällön kuvailua.

6.1.2 Generatiivisen tekoälyn mallit ja data

Perustamallia valitessa on huomioitava, että mallin valmistaja on jo valinnut ja käsitellyt datan, jolla malli on koulutettu. Mallin valinnassa onkin hyvä kiinnittää huomioita suorituskyvyn lisäksi siihen, mitä ja miten valmistaja kertoo koulutusdatasta. Jos koulutusdatan lähteet ovat erikseen ja uskottavasti kuvattu, on mallin valinta paremmin perusteltavissa.

Koulutuksessa käytettyyn dataan tulee kiinnittää huomiota, sillä tutkimuksissa on huomattu, että jo pieni määrä haitallista dataa voi saastuttaa mallin[15].

Täyttä varmuutta esimerkiksi laajojen kielimallien koulutusdatan mahdollisesti sisältämiin ongelmiin ei yleensä voida saada, ellei data ole täysin julkista. Mallin koulutusdatan laatua ja turvallisuutta voi yrittää arvioida etsimällä tieteellisestä tai teknisestä kirjallisuudesta tutkimuksia ainakin niiden datajoukkojen ja lähteiden osalta, jotka valmistaja on julkistanut.

Tekoälyagenttiin voi kohdistua vaatimuksia, jotka on hyvä tarkistaa myös sen käyttämien mallien koulutusdatan näkökulmasta. Näitä ovat esimerkiksi eettisyys, kestävyys, mahdolliset mallin valmistajan asettamat lisenssi- tai käyttöehdot, käytön rajoitukset sekä koulutusdatan tekijänoikeudet. Koulutusdatan mahdolliset harhat ja vinoumat erityisesti koskien ihmisten ikää, sukupuolta, kansallisuutta tai muita erityisiä ominaisuuksia kannattaa pyrkiä selvittämään, jotta niihin voidaan soveltaa tarvittavia hallintakeinoja (esim. vinoumien hallinta, *bias mitigation*).

6.1.3 Metatiedot

Metatietoja voi mahdollisesti käyttää datan luokitteluun sekä luotettavuuden ja hyödynnettävyyden arviointiin. Esimerkiksi internet-sisällön arvioinnissa ja suodatuksessa metatiedoilla (esim. laatijan käyttäjätunnus, IP-osoitetiedot, IP-osoitteen yhdistettävyys operaattoriin, maantieteelliseen alueeseen tai VPN-palveluun) voi olla merkitystä. Esimerkiksi IP-osoite on usein tietosuojaperiaatteiden tarkoittama henkilötieto, joten sitä tulee myös käsitellä sellaisena. Metatietojen olemassaolo on hyvä tiedostaa, jotta riski suojattavien tietojen vuodolle voidaan hallita.

6.1.4 Datan versiointi ja jäljitettävyys

Agenttijärjestelmien testauksessa käytetty data on hyvä säilyttää erillään. Mikäli järjestelmä ei toimi halutulla tavalla, voi syytä etsiä testidatasta. Mikäli järjestelmässä käytettyjä malleja on tarkoitus kouluttaa, on koulutukseen käytetty data myös hyvä versioida. Näin mahdolliset ajan kuluessa ilmenevät ongelmat voidaan yhdistää oikeaan koulutusdatan versioon. Versioimalla voidaan myös testata datan esikäsittelyn tai täydentämisen vaikutusta mallin suorituskykyyn. Jos agenttijärjestelmää päivitetään, voidaan versioidulla datalla verrata tuloksia ennen ja jälkeen päivityksen.

6.1.5 Datan nouto ja integraatiot

Agenttijärjestelmän käyttämä data kerätään usein organisaation järjestelmistä tai julkisista lähteistä erilaisilla integraatioilla. Integraatioiden tehtävä

[15] Souly ym. (2025). Poisoning attacks on LLMs require a near-constant number of poison samples. https://arxiv.org/abs/2510.07192

on noutaa ja muuntaa data järjestelmälle käyttökelpoiseen muotoon. Integraatiot voivat toimittaa datan järjestelmälle suoraan tai tallentaa sen organisaation tietovarastoon tai -alustalle. Luotettavat integraatiot ovatkin sovelluksen käyttöarvon ja luotettavuuden kannalta erittäin tärkeitä.

Datan hankinnan luotettavuutta ja turvallisuutta voidaan parantaa hyvin määritellyillä rajapinnoilla. Rajapintojen käyttöoikeudet tulee määritellä huolellisesti tiedon minimoinnin, anonymisoinnin ja nollaluottamuksen (*zero trust*) periaatteet huomioiden. Rajapintojen suunnittelussa on hyvä pyrkiä selkeyteen ja yksinkertaisuuteen. Esimerkiksi yhdelle tiedolle on hyvä olla vain yksi rajapintakutsu.

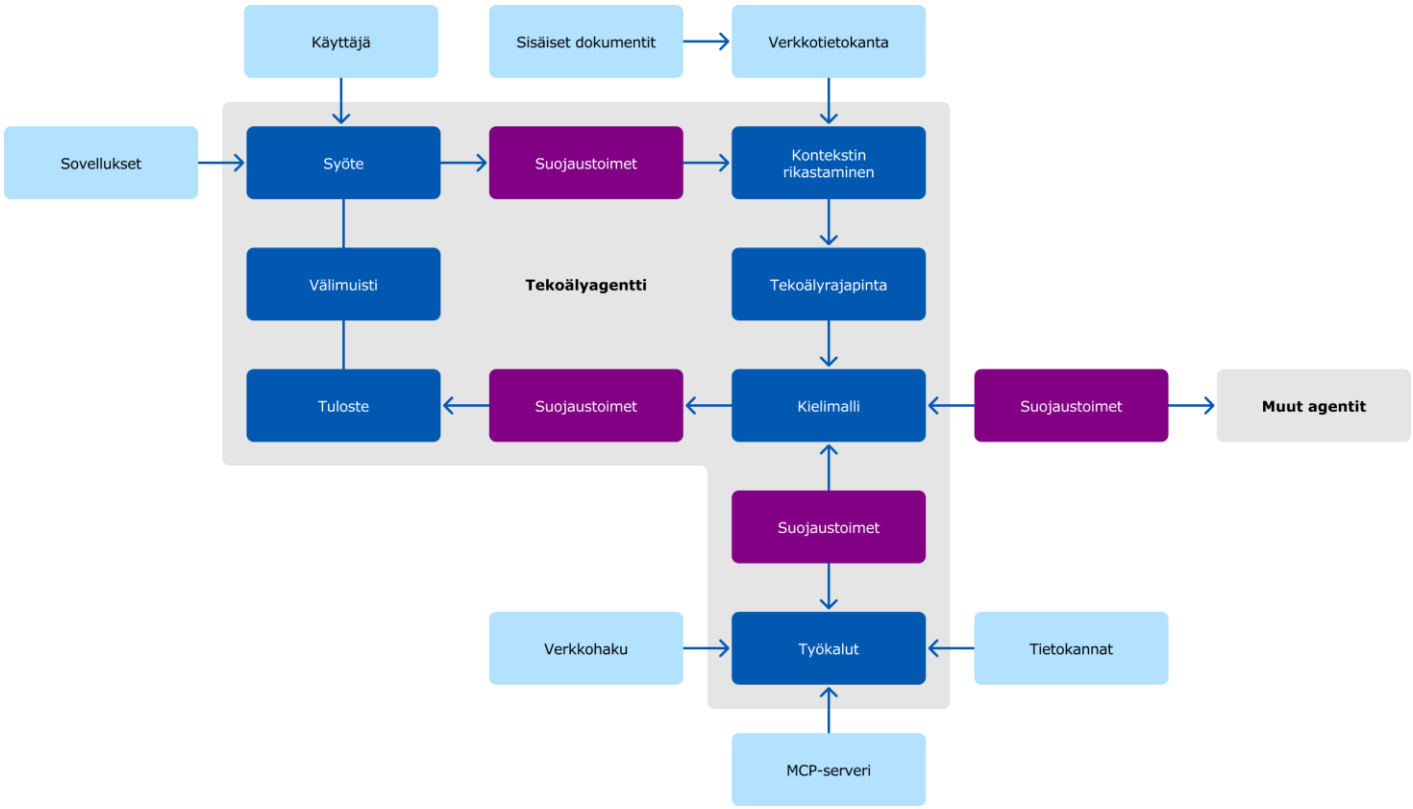
Datan haun osalta tulee määritellä integraation, tietovaraston ja sovelluksen roolit. On hyvä määritellä, mikä osa datasta tulee säilyttää itse sovelluksen yhteydessä esimerkiksi tilapäisesti ja mikä osa on haettavissa integraatiolla tai kyselyllä tietovarastosta aina tarvittaessa. Datan säilytys hallinnointi sovelluksen yhteydessä voi olla jossain tilanteessa järkevää, erityisesti jos käyttötapaus vaatii nopeaa vasteaikaa tai datan toistuvan noutamisen kustannukset ovat suuret. Datan yhdistäminen, ajantasaisuus ja luotettava toimitus on yleensä helpoin ratkaista näihin tarpeisiin suunnitelluilla järjestelmillä. Näin sovelluksen oma datan hallinnointitaakka voidaan pitää mahdollisimman pienenä.

6.1.6 Datan ajantasaisuus

Dataa tulee hankkia vähintään vastaavalla syklillä kuin millä tuloksia odotetaan toimitettavan (esim. viikko-, kuukausi-, kvartaali- tai vuosisyklillä). Datan pitäminen ajantasaisena on erityinen haaste esimerkiksi laajoja dokumenttiaineistoja hyödyntävissä sovelluksissa. Tällaisia ovat esimerkiksi agenttijärjestelmien osana usein käytettävät RAG-arkkitehtuuria hyödyntävät työnkulut. Sovelluksen käyttämään dokumenttivarantoon tulee pystyä lisäämään tuoreimmat uudet dokumentit ja uusimmat versiot olemassa olevista dokumenteista. Vanhentunutta tietoa sisältävät dokumentit tulee poistaa tai leimata vanhentuneiksi. Datan ajantasaisuus ja sisällön tehokas indeksointi auttavat tehostamaan järjestelmän tiedonhakua.

6.2 Tekoälyagenttien komponentit

Tekoälyagentit koostuvat useista pienistä komponenteista, joita yhdistelemällä voidaan rakentaa monimutkaisia ongelmia ratkaisevia järjestelmiä. Tässä kappaleessa käsitellään agentin rakentamisessa käytettäviä komponentteja ja kerrotaan, kuinka ne toteutetaan turvallisesti. Kuvassa 4 havainnollistetaan agenttijärjestelmän eri osia.



Kuva 4. Tekoälyagentin komponentit

6.2.1 Perustamalli: Kielimallit agentin mahdollistajana

Kielimallit mahdollistavat tekoälyagenttien rakentamisen. Ne prosessoivat strukturoimattomasta tekstidatasta muodostuvat käyttäjän kehotteet ja valitsevat tarvittavat toimenpiteet kehotteen sisällön ratkaisemiseksi. Kielimallit eivät itsessään kykene kutsumaan erilaisia työkaluja, tai hakemaan vastauksen muodostamiseen vaadittavaa aineistoa. Tästä syystä kielimallin ympärille tulee rakentaa sovellus, joka suorittaa toimenpiteitä kielimallin tuloksien perusteella. Kehittämistä helpottavat erilaiset sovelluskehykset, jotka automatisoivat tämän työkalujen käytön, jotta sovelluskehittäjän ei tarvitse itse luoda ja ylläpitää tarvittavia prosesseja.

6.2.2 Sopivan kielimallin valitseminen

Toimiva agentti vaatii pohjakseen aina edistyneen kielimallin, joka on koulutettu käyttämään työkaluja. Lisäksi mallin tarkkuus tulee arvioida, jotta työkalujen kutsuminen voidaan toteuttaa turvallisesti. Heikko tarkkuus heikentää mallin turvallisuutta erityisesti silloin, kun agentilla on pääsy myös ulkoisiin järjestelmiin.

Mallien vertailu kannattaa aloittaa tehokkaista malleista. Näin saadaan selville, kuinka hyvin malli parhaimmillaan toimii kyseisessä käyttötarkoituksessa. Seuraavaksi voidaan kokeilla, onnistuuko pienempi kielimalli

suorittamaan saman tehtävän yhtä laadukkaasti. Samalla voidaan arvioida, missä kohdissa pienempi malli epäonnistuu. Pienempien mallien käyttö säästää aina resursseja, joten niitä kannattaa mahdollisuuksien mukaan suosia[16].

Varmista, että malli täyttää oman organisaatiosi vaatimukset datan käsittelystä. Tarkista muun muassa, missä kielimallin laskenta suoritetaan ja miten kielimallille lähetetty tieto tallennetaan. Osa kielimalleista käyttää käyttäjien tietoja mallien uudelleen kouluttamiseen, mikä voi rikkoa organisaation datapolitiikkaa.

6.2.3 Turvallisen kehotteen laatiminen

Kielimallien tuotos perustuu aina kehotteeseen. Kehote koostuu useasta eri komponentista ja useiden kielimallien tarjoajien rajapinnat mahdollistavat kehotteen jakamisen loogisiin komponentteihin:

- **Järjestelmäkehote**: Määrittää agentin roolin, kontekstin ja kielimallin toimintaperiaatteet erilaisissa tilanteissa.
- **Käyttäjän syöte**: Kertoo, mitä kielimallin tulee tehdä. Syöte voi olla esimerkiksi kysymys keskustelusovelluksessa tai prosessiautomaation käynnistävän tehtävän määrittely.
- **Kehotteen rikastaminen**: Kielimallin tuloksia voidaan usein parantaa tarjoamalla mallille kontekstia ulkoisilla dokumenteilla. Kielimallin vastausta voi esimerkiksi parantaa antamalla käyttäjän tietoja kielimallille.
- **Agentin tuotokset**: Lopputuotokset voivat olla esimerkiksi ulkoisen järjestelmän palauttama sanoma, laskimen palauttama tulos tai kielimallin rakentaman sovelluksen tuloste. Näitä tuotoksia voidaan käyttää uusien tehtävien automaattiseen käynnistämiseen.
- **Historiatiedot**: Kielimallille voidaan syöttää historiaa aiemmista syötteistä ja tuotoksista, joiden avulla agentin suorittamia toimintoja voidaan johdonmukaistaa.

Kehotetta laatiessa on tärkeä huomioida, että jokainen komponentti on kielimallille yhdenvertainen. Tämä tarkoittaa, että mikä tahansa kehotteen osista voi muokata mallin käytöstä ja mahdollistaa esimerkiksi hyökkääjälle kehoteinjektion suorittamisen.

Kaikilla malleilla optimaalisesti toimivaa järjestelmäkehotetta ei ole, vaan agentti vaatii käyttötarkoitukseen sopivan kehotteen. Kehotteen muotoilu on iteratiivinen prosessi, jossa kokeillaan, millä kehotteella agentti toimii

[16] OpenAI. A practical guide to building agents. https://cdn.openai.com/business-guides-and-resources/a-practical-guide-to-building-agents.pdf

käyttötarkoitukseensa parhaiten. Kielimallille kirjoitettavissa ohjeissa tulee kiinnittää huomioita seuraaviin kohtiin: Kirjoita selkeät ja tarkat ohjeet, määritä kielimallille persoona, erottele syötteen osat selkeästi. anna kielimallille esimerkkejä, määrittele tuotokselle formaatti sekä ohjeista mallia vaiheittaiseen päättelyyn.

Hyvä kehote on laadun lisäksi myös turvallisuustekijä. Kehotteen osalta keskeisimmät haavoittuvuudet ovat kehotteen manipulointi ja järjestelmäkehotteen paljastuminen. Kehotteen manipuloinnin avulla hyökkääjä voi piilottaa kehotteeseen haitallista sisältöä. Hyökkääjä voi myös pyrkiä paljastamaan järjestelmäkehotteen päästäkseen käsiksi siihen mahdollisesti sisällytettyyn sensitiiviseen tietoon. Kehoteinjektio voidaan toteuttaa joko suoraan kielimallin kehotteen kautta tai epäsuorasti esimerkiksi sisällyttämällä haitallisia komentoja agentin työkaluun tai ulkoisista lähteistä haettavaan tietoon.

Kehote on siis tärkeä suojata uhkilta. Yllä mainittuja ohjeita voidaan täydentää OWASPin laatimilla ohjeilla[17] kehotteen suojaukseen:

1. **Rajoita mallin toimintaa.** Ohjeista agenttia olemaan välittämättä muista ohjeista. Muistuta mallia ulkoisten dokumenttien lukemisen jälkeen agentin tehtävästä, jotta mahdolliset hyökkääjän ohjeet sivuutetaan. Lisäksi kielimallin voi ohjeistaa reagoimaan vain rajatun tyyppisiin syötteisiin.
2. **Validoi kielimallin tuotoksen formaatti.** Kielimallin tuotoksille määritelty formaatti voidaan validoida toisella ohjelmalla.
3. **Rakenna suojaustoimet kehotteelle ja tulosteelle.** Tarkista esimerkiksi toisella kielimallilla, että kehotteessa ei ole kiellettyjä aiheita ja että kehote on agentin tehtävän mukainen (ks. suojaustoimet, 6.2.5). On tärkeää huomata, että toisen kielimallin käyttö tarkistamiseen ei ole täydellinen suojaustoimi.[18] Muiden keinojen käyttö on suositeltavaa, kun se on mahdollista.
4. **Anna kielimallille pienimmät mahdolliset oikeudet.** Rajaa kielimallin oikeudet siten, että vain tarvittavat tehtävät voidaan toteuttaa. Rajaamalla tiedostojen lukuoikeuksia vältetään esimerkiksi arkaluontoisen tiedon vuotaminen.
5. **Edellytä ihmisen hyväksyntää kriittisissä päätöksissä.** Edellytä kriittisissä operaatioissa käyttäjän vahvistus. Vahvistus pienentää luvattomien operaatioiden riskiä.
6. **Erottele tuntematon sisältö selkeästi muusta kehotteesta.** Tuntemattomalla sisällöllä tarkoitetaan ulkoisia dokumentteja, internet-

[17] OWASPin 2025 Top 10 Risk & Mitigations for LLMs and Gen AI Apps: https://genai.owasp.org/llmrisk/llm01-prompt-injection/
[18] Ks. Esim. Quanta Magazine: Cryptographers Show That AI Protections Will Always Have Holes: Cryptographers Show That AI Protections Will Always Have Holes

hakujen tuloksia sekä käyttäjän kehotetta. Se tulee erotella muusta kehotteesta selkeästi, jotta dokumenttien sisältö ei vaikuta kielimallin toimintaa. Tuntemattoman sisällön voi eritellä esimerkiksi hyvän kehotteen ohjeissa mainituilla erottimilla.

7. **Testaa mallin toimintaa aktiivisesti.** Rakenna testit tai simulaatiot, joilla pyritään murtamaan mallin suojaus. Testien tarkoituksena on selvittää mallin rajat sekä pääsyoikeuksien todellinen laajuus.

6.2.4 Muisti ja ulkoiset dokumentit

Kielimallien mustilla tarkoitetaan tekniikkaa, jonka avulla järjestelmään tallennetaan tietoa, jota hyödynnetään myöhemmin mallin toiminnassa. Esimerkiksi tekoälyagentti hyödyntää muistia tehtävien suorittamisen nopeuttamiseen. Kielimallin muistin voi jakaa kolmeen osaan: 1) sisäinen tieto, 2) lyhytaikainen muisti, ja 3) pitkäaikainen muisti. Sisäinen tieto kattaa tiedon, joka mallille on syötetty koulutuksen aikana. Lyhytaikainen muisti voi kattaa esimerkiksi toimintahistoriaa. Pitkäaikaisella muistilla voidaan tallentaa ja jakaa agenttien tehtäviä eri sessioiden aikana.

Tekoälyagentin kanssakäymisen lisäksi agentin muistiin voidaan tallentaa ulkoisia dokumentteja. RAG-menetelmällä agenttia ei tarvitse kouluttaa uudestaan, vaan tarvittava tieto syötetään agentin kehotteeseen.

Suunnittele rakennusvaiheessa, miten kielimallin lyhyt- ja pitkäaikainen muisti saadaan suojattua haavoittuvuuksilta. Varmista käyttöoikeuksien määrittelyllä, että vain rajattu joukko käyttäjiä pääsee lukemaan tietoja muistista. Kun agentin pitkäaikaiseen muistiin lisätään dataa, siitä tulee poistaa mahdollisesti agentin toimintaa tai turvallisuutta haittaavat tekstit. Tällaisia ovat esimerkiksi aikasidonnaiset ilmaukset, kuten ”tästä eteenpäin”, tai muiden ohjeiden käyttöä rajaavat ilmaukset, kuten ”älä välitä edellisistä ohjeista”. Muistiin tallennettavien dokumenttien lähteet tulee varmistaa, jotta tietokantaan ei tallenneta saastunutta dataa. Varmista lisäksi, että henkilötietoja käsitellään agentin muistissa säädösten mukaisesti.

6.2.5 Suojaustoimet

Suojaustoimet muodostavat joukon sääntöjä ja ohjeistuksia, jotka varmistavat, että kielimalli toimii kuten on suunniteltu. Jokaiselle käyttötapaukselle tulee luoda omat säännöt, kuinka mallin tulee käyttäytyä eri tilanteessa. Suojaustoimet voidaan toteuttaa sekä syötteelle että kielimallin palauttamalle tiedolle esimerkiksi seuraavilla menetelmillä:

1. **Merkkijonojen tunnistaminen tekstistä**: Tunnistetaan tekstistä sanoja tai merkkijonoja, joita kielimallin ei tule käsitellä tai generoida käyttäjälle. Tällaisia ovat esimeriksi henkilötiedot, kuten henkilötunnukset tai sähköpostiosoitteet.

2. **Syötteen vertaaminen kiellettyihin sanoihin**: Tekoälyagentille voidaan etukäteen määritellä sanoja/teemoja, joita se ei saa käsitellä.
3. **Syötteen luokittelu erillisellä mallilla sallittuihin kategorioihin**: Teksti voidaan luokitella erillisellä mallilla eri kategorioihin joihin agentti saa antaa vastauksen.

Syötteen turvallisuutta voidaan parantaa tunnistamalla esimerkiksi seuraavia tietoja:

4. **Vaarallinen tai haitallinen sisältö:** Vaarallisen tai haitallisen syötteen tunnistaminen, jolla yritetään esimerkiksi löytää kielimallin haavoittuvuuksia, varastaa järjestelmäkehote tai luoda haitallista sisältöä.
5. **Henkilötiedot:** Ennen kielimallin kutsumista syötteestä voidaan esimerkiksi piilottaa henkilötiedot. Jos henkilötietoja tarvitaan lopullisessa vastauksessa, ne voidaan lisätä mallin luomaan tekstiin.

Harmittoman oloinen syöte voi mahdollisesti saada mallin palauttamaan tietoja, joita käyttäjälle ei haluta näyttää. Palautetun arvon tarkastelussa voidaan huomioida esimerkiksi:

6. **Sisältö, henkilötiedot tai haitallinen tieto:** Suojaustoimet tulee implementoida siten, että sisältö vastaa käyttäjän kyselyä ja että siinä ei ole kielimallin kehittäjän määrittelemää haitallista sisältöä, kuten disinformaatiota.
7. **Hallusinointi:** Hallusinointi on yleinen kielimallien ongelma. Lopputulosta voidaan esimerkiksi verrata syötteeseen ja ulkoisiin dokumentteihin tiedon oikeellisuuden varmentamiseksi.
8. **Syntaksi:** Kielimallin lopputulos voidaan validoida vastaamaan esimerkiksi tiettyä ohjelmointikieltä. Tuotoksen voidaan tarkistaa olevan esimerkiksi JSON-syntaksin mukaista.

Kielimallien suojaamista varten on myös koulutettu erikoistuneita malleja, kuten Llama Guard[19] ja Constitutional Classifiers[20].

On tärkeä huomata, että kielimallit eivät ole täysin suojattuja, vaikka niiden ympärille rakennetaan erilaisia suojia. Tekoälyjärjestelmät edellyttävät monitorointia, jotta haitallinen käyttö huomataan mahdollisimman nopeasti.

[19] Inan ym. (2023), Llama Guard: LLM-based input-output safeguard for human-AI conversations, https://ai.meta.com/research/publications/llama-guard-llm-based-input-output-safeguard-for-human-ai-conversations/
[20] Sharma ym. (2025), Constitutional Classifiers: Defending against universal jailbreaks across thousands of hours of red teaming, https://arxiv.org/pdf/2501.18837

6.2.6 Työkalut: data, toimenpiteet ja orkestrointi

Kielimallille annetaan kehotteessa lista työkaluista, joita se voi käyttää annetun tehtävän ratkaisemiseen. Työkalujen käyttö voi parantaa mallin tarkkuutta mallille määritellyssä ympäristössä. Oikean työkalun valitseminen ongelman ratkaisuun on haastavaa etenkin, jos työkalujen määrä on suuri. Työkalun tiedoista pitää selkeästi tunnistaa, mihin sitä voi käyttää ja millaista tietoa sille tulee syöttää.

Jotta agentti voi käyttää työkaluja turvallisesti, valinnassa on varmistettuttava siitä, mitä työkalu tekee, mitä tietoa sen käyttäminen vaatii ja missä työkalun laskenta suoritetaan. Alla on listattu esimerkkejä toimista, joilla tekoälyagenttien käyttämiin työkaluihin kohdistuvia riskejä voidaan hallita.

1. **Ohjelmien suorittaminen**: Tekoälyagentin generoiman sovelluksen sisältö tulee varmistaa ja sovellus suorittaa rajatussa ympäristössä.
2. **Käyttöoikeuksien rajaaminen:** Työkalujen käyttöoikeudet tulee rajata mahdollisimman pieniksi. Esimerkiksi tietokantaan tulee antaa vain rajalliset oikeudet, jotta agentti voi hakea tai kirjoittaa vain tarpeellisen tiedon tietokannasta.
3. **Varmennetun työkalun korvaaminen haitallisella**: Osa työkaluista voi toimia ulkoisilla palvelimilla (ks. esim. MCP-palvelin, luku 6.2.8). Tällöin aikaisemmin turvallisen työkalun sisältöä voidaan muokata haitalliseksi. Työkaluun voidaan esimerkiksi lisätä uusia toiminnallisuuksia, jotka lähettävät työkalun käsittelemät tiedot sähköpostilla työkalun muokkaajalle. Työkalua käytettäessä voi esimerkiksi tarkistaa työkalun hajautusarvon.

6.2.7 Työkalut ja agenttien autonomia

Itsenäinen agentti on haavoittuvainen tavoitteen manipuloinnille. Hyökkääjä voi kehoteinjektion avulla pyrkiä muokkaamaan agentin suunnittelua tai päättelyä suorittaakseen haitallisia toimenpiteitä, kuten järjestelmän asiakastietojen kalastelua.

Seuraavat toimet ovat keskeisiä agenttijärjestelmän tehtäväketjujen suojaamiseksi:

- Rajoita käyttöoikeudet mahdollisimman suppeiksi, jotta agentti ei voi tehdä toimenpiteitä annetun tehtävän ulkopuolella (ks. 3.2.6.)
- Rajoita työkalujen saatavuutta vain niihin, joita agentti tarvitsee tehtävän suorittamiseksi
- Monitoroi tekoälyagentteja tavoitteiden ja käyttäytymisen muutosten tunnistamiseksi.
- Aseta rajat sille, kuinka paljon eri agentteja voi kutsua (ks. 3.2.10)

- Edellytä ihmisen hyväksyntää kriittisissä toimenpiteissä.

6.2.8 Työkalujen ja agenttien hallinta: MCP-protokolla

Omien työkalujen ylläpitäminen jokaiselle tekoälyagentille erikseen voi olla pitkällä aikavälillä työlästä. Ratkaisuksi on esitetty erilaisia protokollia, joista tunnetuimpia ovat Anthropicin *Model Context Protocol* (MCP) ja Googlen *Agent2Agent* (A2A). MCP-protokolla on jo vakiintunut käyttöön, joten tässä ohjeessa tarkastellaan siihen liittyviä turvallisuuskysymyksiä tarkemmin.

MCP on avoimen lähdekoodin standardi, jonka tarkoituksena on luoda selkeä yhteinen toimintamalli työkalujen kutsumiselle. Se mahdollistaa tekoälysovellusten yhdistämisen ulkoisiin järjestelmiin. Sen voi ajatella toimivan kuin USB-C-kaapeli: yhtenäinen ja standardoitu tapa yhdistää useita erilaisia palveluita, mikä nopeuttaa tekoälyagenttien kehittämistä ja helpottaa työkalujen jakamista. MCP:n toiminnot voidaan suorittaa joko paikallisesti tai ulkoisella palvelimella.

MCP:n arkkitehtuuri koostuu kolmesta pääosasta:

- **MCP-isäntä** (*host*) on tekoälysovellus, joka käyttää työkaluja kontekstin rikastamiseen.
- **MCP-asiakas** (*client*) ylläpitää yhteyttä palvelimeen ja välittää tietoa osapuolten välillä. Jokainen MCP-asiakas kommunikoi vain yhden MCP-palvelimen kanssa.
- **MCP-palvelin** (*server*) tarjoaa asiakkaalle kontekstin ja toiminnallisuudet. Palvelin koostuu kolmesta osasta: työkaluista, resursseista ja kehotteista.

MCP-protokolla luo uusia mahdollisuuksia työkalujen rakentamiseen. Samalla se tuo uuden haavoittuvuuden tekoälyagenttiin. Jos käytät MCP-protokollaa agenttiratkaisussasi, huomioi ainakin seuraavat haasteet:

- **Harhautetun sovelluksen ongelma** (*confused deputy problem*)**:** Hyökkääjä voi hyödyntää MCP-palvelimia välityspalvelimina, jos MCP-asiakastunniste on määritelty staattiseksi.
- **Tunnisteen läpivienti** (*token passthrough*)**:** MCP-palvelin hyväksyy tunnisteen (*token*) MCP-asiakkaalta ilman varmistusta, ja välittää tunnisteen käytössä oleville rajapinnoille.
- **Istunnon kaappaus** (*session hijacking*)**:** Hyökkääjä voi saada MCP-palvelimen käyttöönsä viemällä staattiseksi määritellyn tunnisteen, jonka MCP-palvelin on määritellyt MCP-asiakkaalle.

- **Paikallisen MCP-palvelimen murto** (*local MCP server compromise*): Kolmannen osapuolen tarjoama MCP-palvelin voi sisältää komentoja, jotka varastavat käyttäjän tietoja tai tuhoaa niitä. Uutta palvelinta asentaessa tulee varmistaa sen toiminnallisuudet sekä rajata käyttöoikeudet.

- **Käyttöoikeuksien rajaaminen** (*scope minimization*): Liian laajojen tunnisteiden vuotaminen voi johtaa tiedon väärinkäyttöön tai käyttöoikeuksien ketjuttamiseen. On suositeltavaa välttää jokerimerkkien (*) käyttöä.

MCP on vielä uusi teknologia ja kehittyy nopeasti. MCP:n uhkista ja niiltä suojautumisesta voi lukea esimerkiksi MCP:n omilta sivuilta[21].

6.2.9 Moniagenttijärjestelmä

Moniagenttijärjestelmillä voi olla oikeuksia useisiin ulkoisiin järjestelmiin ja eri tasoisiin salattuihin tietoihin. Tästä syystä on tärkeää huolehtia agenttien välisen kommunikoinnin suojaamisesta. Mikäli järjestelmällä on pääsy salassa pidettävään tietoon, agentit voivat jakaa tätä tietoa keskenään. Mahdollisten vuotojen riski tuleekin arvioida agenttien keskinäisen kommunikaation osalta ja tehdä tarvittavat suojaukset, kuten agenttien välisten viestien autentikointi sekä salaus.

Tiedon minimoinnin, lokeroinnin (*compartmentalisation*) ja nollaluottamuksen periaatteita on hyvä noudattaa luottamuksellista tietoa käsittelevissä moniagenttijärjestelmissä. Agenteille tulee antaa pääsy vain niihin lähteisiin, joita niille määritelty rooli edellyttää. Erityisesti tilanteissa, joissa vaaditaan sekä luottamuksellisten että julkisten materiaalien käsittelyä, tulee pääsyoikeuksiin ja tiedon minimointiin kiinnittää huomioita. Esimerkiksi julkisen tiedon noutaminen voidaan antaa tehtävään määritellylle agentille, jolla ei ole pääsyä luottamukselliseen aineistoon. Tällainen agentti tulee myös määritellä toimittamaan tuloksensa rajattuun ympäristöön, jonne luottamuksellista materiaalia käsitteleville agenteille annetaan vain lukuoikeudet. Välitettävää tietoa voi myös monitoroida, sanitoida ja anonymisoida tarpeen mukaan.

Agenttien toimintaa tulee myös seurata lokein. Lokeja puolestaan tulee monitoroida tehokkuuden ja tietoturvan näkökulmasta auditoimalla. Lokeihin tulee tallentaa tietoa sellaiseen muotoon, että virheellisesti toimivat agentit voidaan tunnistaa sen avulla tehokkaasti. Jos agentti alkaa esimerkiksi yllättäen kutsumaan uusia työkaluja tai määrittelee itselleen tuntemattoman tavoitteen, agentti tulee tunnistaa ja eristää verkosta ja järjestelmistä.

[21] MCP security best practices: https://modelcontextprotocol.io/specification/draft/basic/security_best_practices

Ydinliiketoiminnan kannalta riittisissä tehtävissä toimivien agenttien toimenpiteitä tulee valvoa erityisen tarkasti. Valvonnan voi hoitaa ihminen tai toiset agentit (ks. Monitorointi ja poikkeamien hallinta, 7.3.6). Toimenpiteisiin voi esimerkiksi edellyttää usean agentin yhteisymmärrystä ennen kuin ne suoritetaan. Näillä keinoilla varmistetaan, että agenttijärjestelmä ei ole haavoittuvainen yksittäisiin agentteihin kohdistuvissa hyökkäyksissä tai vikatilanteissa.

6.2.10 Skaalautuvuus

Moniagenttijärjestelmään voidaan lisätä uusia agentteja ja niitä voidaan poistaa suhteellisen helposti. Järjestelmää voidaan skaalata joko monistamalla jo määriteltyjä agenttirooleja tai pilkkomalla yhden agentin tehtävä osiin. Skaalautuvuudesta saa suurimman edun, jos tehtävänkulku on mahdollista määritellä niin, että tehtävien suorittaminen myös rinnakkain onnistuu. Skaalautuvuudessa on riskinä liiallinen resurssien käyttö, jos agenttijärjestelmän agentit pystyvät lisäämään uusia agentteja järjestelmään ilman valvontaa. Riskinä on valtavat kustannukset tai muiden käyttäjien tehtävien hidastuminen.

6.2.11 Vikojen sieto

Agenttijärjestelmän suorittamien tehtävien laadulle ja suoritusajalle on hyvä määritellä selkeät tavoitteet. Poikkeustilanteista toipuminen tulee mahdollistaa mahdollisimman hyvin. Jos toipuminen ei kuitenkaan onnistu, tilanne on tärkeää tunnistaa ajoissa. Tehtävä on voitava tarvittaessa keskeyttää ja vapauttaa agenttien käyttämät resurssit. Toimenpiteistä on voitava myös kirjoittaa virheloki.

Agenteille on mahdollista määrittää tehtäväkohtaisia reaaliaikavaatimuksia ja rajoituksia esimerkiksi uudelleenyritysten määrälle. Mikäli agentti ei suoriudu tehtävästään, tulee valvontaa suorittavan tahon päättää, onko tehtävää mahdollista jatkaa vai tuleeko se keskeyttää. Valvonta voidaan toteuttaa ihmisen toimesta erilaisten lokien kautta tai hyödyntämällä riippumattomia kielimalleja lokien arviointiin (ks. *LLM-as-a-judge*, 7.3.1). Määritellystä tehtävästä riippuen työnkulkua voi olla mahdollista jatkaa yksittäisen agentin vikaantumisesta huolimatta. Moniagenttijärjestelmä voi esimerkiksi jatkaa muiden agenttien tuottamien tuotosten turvin toimintaansa myös silloin, kun yhden agentin tiedonhaku ei tuota tuloksia. Tällöin tieto yhden lähteen puuttumisestä sisällytetään mukaan tulokseen.

Järjestelmälle voidaan myös määritellä vaihtoehtoinen työnkulku. Esimerkiksi tuloksen jalostuskierroksia voi jättää välistä ja hyväksyä myös vähemmän valmiit tulokset. Turvallisuustarkistuksen tai haitallisen sisällön tunnistuksen ohitusta ei kuitenkaan kannata mahdollistaa. Moniagenttijärjestelmän työnkulkua suunnitellessa on hyvä tunnistaa kohdat, missä jatkaminen on mahdollista ja milloin se tulee keskeyttää. Toiminta vikatilanteissa on

mahdollista määritellä joko agenttijärjestelmän staattisiin sääntöihin, valvovan agentin järjestelmäkehotteeseen tai molempiin.

Jos agenttijärjestelmä suorittaa iteratiivista tehtävää (esim. raportin laatiminen), tarvitaan yleensä useita laatimis- ja katselmointikierroksia. Iterointikierroksille on hyvä asettaa lukumäärälliset rajat viansiedon, suoritusajan ja kustannusten hallinnan takia.

6.2.12 Tietoturva osana agentin operointia (MLOps)

Machine Learning Operations on kokonaisvaltainen lähestymistapa, joka tuo koneoppimis- ja tekoälymallit osaksi yritysten tuotantoympäristöjä hallitusti ja tehokkaasti. Mallit, kehotteet ja koodi versioidaan systemaattisesti, mikä helpottaa toistettavuutta ja mahdollistaa palaamisen aiempaan, toimivaan versioon. Auditointi ja katselmoinnit varmistavat laadun ja luotettavuuden. Kaikki muutokset dokumentoidaan ja arvioidaan ennen käyttöönottoa, mikä lisää läpinäkyvyyttä ja turvallisuutta. Lisäksi sovelluksessa käytettävät ulkoiset komponentit tulee lukita turvalliseksi validoituun versioon, jotta komponenttien automaattinen päivitys ei aiheuta tietoturvahaavoittuvuuksia.

Automaatio on MLOps:in ydin. Kehityksen eri vaiheet (esim. datan lataus ja prosessointi, mallien koulutus ja julkaisu) automatisoidaan. Tämä parantaa johdonmukaisuutta, toistettavuutta ja skaalautuvuutta. Mallien koulutus ja julkaisu voidaan käynnistää esimerkiksi data- tai koodimuutosten ja ajastusten perusteella. Automaattiset testit paljastavat virheet aikaisessa vaiheessa. Infrastruktuurin hallinta koodina nopeuttaa julkaisuja sekä helpottaa ylläpitoa.

MLOps tulee viedä osaksi organisaation toimintakulttuuria. Tiivis yhteistyö datatieteilijöiden, datainsinöörien ja liiketoiminnan välillä on edellytys onnistuneelle projektille. Selkeä dokumentaatio, toimivat kommunikaatiokanavat ja systemaattinen palautteen kerääminen mallien toiminnasta luovat perustan hallitulle kehitykselle. Erityistä huomiota kiinnitetään arkaluonteisen ja sensitiivisen datan hallintaan ja pääsyoikeuksien rajoittamiseen vain niille, joilla on siihen todellinen tarve. Lisäksi MLOps:in avulla varmistetaan sääntelyn ja tietoturvavaatimusten noudattamisen.

Koska tekoälyagentit toimivat itsenäisesti, on monitorointi keskeisessä roolissa läpinäkyvyyden kannalta. Riskinä itsenäisessä päätöksenteossa ovat vaatimuksenmukaisuuden rikkomukset, kun säännösten noudattamista ei voida todentaa, toiminnalliset häiriöt, kun päätöksentekoon tai virheisiin ei ole näkyvyyttä, ja luottamuksen rapautuminen, kun agenttien toimintaa ei erityisesti kriittisessä päätöksenteossa voida selittää. Monitoroinnin avulla organisaatiot saavuttavat hallintaa agenttien toimintaan ja suorituskykyyn. Monitoroinnista kerrotaan lisää testausohjeen yhteydessä (ks. luku 6.3.6).

7 Testaus

Tekoälyagenttien teknologia, mallit ja data muuttuvat ja kehittyvät nopeaa vauhtia. Tekoälyagentit toimivat myös usein dynaamisesti. Näidenkin seikkojen vuoksi pelkkä kertaluontoinen hyväksymistestaus ei riitä, vaan tekoälyagentteja on testattava säännöllisesti.

Tekoälyn testaamisessa on huomioitava perinteisen tietoturvallisuuden lisäksi tekoälyn ominaispiirteet. Agenttijärjestelmien komponentit myös kehittyvät jatkuvasti, mikä nostaa testaamisen tärkeyttä. Haavoittuvuuksien ja turvattomien konfiguraatioiden löytämisen lisäksi tekoälyratkaisuissa on tarkasteltava myös mallin käytöstä ja tulosten luotettavuutta. Tässä luvussa kuvataan tekoälyagenttien testaus tietoturvan näkökulmasta. Opas nojaa OWASPin *AI Testing Guide* -projektin[22] dokumentaatioon.

7.1 Tekoälytestaamisen periaatteet

Tekoälyjärjestelmien pysyminen turvallisena ja luotettavana voidaan varmistaa testauksella, joka ulottuu koko järjestelmän elinkaaren eri vaiheisiin. Näin varmistetaan agentin toiminnan turvallisuus, tulosten luotettavuus ja mallin tarkoituksenmukainen toiminta. Tämän lisäksi agentin toimintaa on monitoroitava jatkuvasti sen luotettavuuden ja suorituskyvyn varmistamiseksi.

Agenttien itsenäinen päätöksenteko, yhteistyö muiden agenttien, käyttäjien ja työkalujen kanssa, sekä jatkuva oppiminen tuovat haasteita myös testaamiseen. Testitapausten valinnassa hyödynnetään uhkamallinnuksessa tunnistettuja uhkia ja hyökkäysvektoreita, jotta tunnistetut riskit ja niihin suunnitellut hallintakeinot voidaan testata ennakoivasti. Testausta tulee tehdä lisäksi aina, kun järjestelmässä tapahtuu muutoksia, esimerkiksi malli tai rajapinta muuttuu, tai agenttiin tehdään päivityksiä.

OWASP on kehittänyt tekoälypohjaisten sovellusten turvallisuuden sekä eettisten näkökulmien arviointiin sekä varmistamiseen *Artificial Intelligence Security Verification Standardin* (AISVS)[23]. Se on tarkastuslista, joka on mallinnettu olemassa olevien standardien pohjalta, kuten OWASP *Application Security Verification Standard* (ASVS), joka on laajalti käytetty verkkosovellusten turvalliseen kehitykseen ja testaamiseen soveltuva työkalu. AISVS tarjoaa järjestelmällisen lähestymistavan tekoälysovellusten testaamiseen.

[22] OWASP AI Testing GUIDE: https://owasp.org/www-project-ai-testing-guide/
[23] OWASP Artificial Intelligence Security Verification Standard (AISVS): https://github.com/OWASP/AISVS/tree/main

OWASPin testausohjeen mukaan tehokas testaus perustuu neljään osa-alueeseen:

1. **Turvallisuus**
2. **Yksityisyys**
3. **Vastuullisuus**
4. **Luotettavuus**

Näitä yhdessä tarkastelemalla tekoälyn käyttöönottoon liittyviä riskejä voidaan hallita kattavasti. Aloitamme tarkastelemalla testauksen keskeisimpiä tekijöitä kunkin osa-alueen osalta.

7.1.1 Turvallisuus

Tietoturvatestauksella varmistetaan, että tekoälyagentti on suojattu luvattomalta käytöltä ja erilaisilta hyökkäyksiltä mahdollisimman hyvin. Testauksessa hyödynnetään uhkamallinnuksessa tunnistettuja riskejä ja hyödynnetään tunnettuja viitekehyksiä ja testausohjeita, kuten edellä mainittua OWASP AISVS:tä. AISVS sisältää luokitellut vaatimukset muun muassa käyttäjän syötteiden validointiin ja mallin turvallisuuden arviointiin. Työkalu keskittyy erityisesti sovelluskehityksen osa-alueisiin.

Tekoälyagenttien kehityksessä testataan erityisesti järjestelmäkehotteiden, ohjeiden ja käyttäjän syötteiden suojaamista myrkyttämiseltä ja manipulaatiolta. Lisäksi testataan tekoälyagentin infrastruktuuria, komponentteja ja liityntäpisteitä, jotta niiden muodostaman kokonaisuuden turvallisuus voidaan varmistaa. Toimitusketjun riskejä voivat aiheuttaa kolmansien osapuolten tuottamat komponentit, toiminnot tai muut riippuvuudet. OWASP AISVS käsittelee myös näitä osa-alueita tarkemmin.

7.1.2 Yksityisyys

Yksityisuuteen liittyy kaksi keskeistä osa-aluetta: tietosuoja ja pääsynhallinta. Tekoälyn toiminta perustuu tyypillisesti moninaiseen tietoon, jolloin henkilötietojen käsittelyn turvallisuudesta on varmistuttava riittävällä testauksella. Testitapauksissa testataan erityisesti tiedon vuotamista ja mallin toimintaa, jotta henkilötietoja käsitellään lainsäädännön mukaisesti.

Pääsynhallinnan testauksella varmistetaan, ettei agentille ole annettu liiallisia oikeuksia tai oikeuksia sellaisiin järjestelmiin, mihin sillä ei ole tehtäviensä perusteella tarvetta. Uhkatoimijoiden tavoitteena voi olla väärinkäyttää agenttiratkaisua paljastamaan arkaluontoista tietoa. Mallin toimintaa testatessa tarkastellaan, paljastuuko mallin toiminnasta tietoja, jotka voivat vaarantaa mallin toiminnan tai aiheuttaa tietovuodon.

7.1.3 Vastuullisuus

Agenttien testaamisessa tulee huomioida eettisyys ja oikeudenmukaisuus. Testaus pyrkii paljastamaan haitalliset vinoumat tuloksissa, mikä vaatii

mallin käyttäytymisen testaamista riittävän kattavilla aineistoilla. Väärän tai virheellisen tiedon (misinformaatio) havaitsemiseksi on tarkasteltava, miten malli käsittelee haitallista sisältöä, kuten vihapuhetta. Testauksessa on arvioitava lisäksi suojaustoimien kattavuus testaamalla turvallisuutta ohjaavia ja ydinliiketoiminnan kannalta keskeisiä mekanismeja sekä sitä, voidaanko ne jollain tavoin ohittaa. Tämä voidaan toteuttaa *red teaming* -testauksella[24], jossa simuloidaan vihamielistä toimintaa testattavaan kohteeseen tarkoituksena paljastaa haavoittuvuuksia.

7.1.4 Luotettavuus

Tekoälyjärjestelmien luotettavuus perustuu päätöksenteon selitettävyyteen erityisesti kriittisten tulosten osalta. Mallien vastauksia on testattava vastausten hajonnan ja odottamattoman käytöksen havaitsemiseksi. Lisäksi käytön aikaista monitorointia on hyvä tehdä poikkeamien tunnistamiseksi ja harhailun havaitsemiseksi.

Puhuttaessa suurten yritysten suljetun lähdekoodin ratkaisuista (esim. OpenAI:n GPT-mallit, Googlen Gemini), yksi riskeistä on, ettei mallin sisälle voida nähdä eikä sen toimintaa muokata. Näissä on etuna suurten kehitysresurssien mahdollistama korkea suorituskyky ja sisäänrakennettu turvallisuus. Avoimen lähdekoodin ratkaisuissa (esim. Mistralin Large, Langchain) mallin sisältöä voidaan tarkastella ja muokata paremmin. Tämä kuitenkin edellyttää teknistä tekoälyasiantuntemusta ja kehitysresursseja.[25]

Luotettavuuteen liittyy olennaisesti myös vaatimuksenmukaisuus. Tekoälyjärjestelmien vaatimuksenmukaisuus on varmistettava noudattamalla lakeja, määräyksiä, ja alan standardeja. OWASPin testausohje ohjeistaa noudattamaan testauksessakin tunnettuja viitekehyksiä, kuten NIST AI RMF (Risk Management Framework)[26] tai ISO 42001 Artificial intelligence – management system-standardia[27]. Lisäksi on syytä varmistaa, koskevatko EU:n tekoälyasetuksen vaatimukset kyseistä agenttijärjestelmää ja huolehtia tarvittavista testauksista.

Kaikkiin neljään osa-alueeseen – turvallisuuteen, yksityisyyteen, vastuullisuuteen ja luotettavuuteen – liittyvät toimet ulottuvat kaikille

[24] OWASP GenAI Red Teaming Guide: https://genai.owasp.org/resource/genai-red-teaming-guide/
[25] Saumya Patel (2025), The LLM Divide: Open Source vs. Closed Source — Choosing Your AI Champion: https://medium.com/@psaumya567/the-llm-divide-open-source-vs-closed-source-choosing-your-ai-champion-ad44c9893316
[26] The NIST AI Risk Management Framework (AI RMF): https://www.nist.gov/itl/ai-risk-management-framework
[27] ISO/IEC 42001:2023 Information technology — Artificial intelligence — Management system: https://www.iso.org/standard/42001

tekoälyarkkitehtuurin kerroksille. Seuraavassa alaluvussa tarkastelemme, miten kerroksellisuus tulee huomioida osana testausta.

7.2 Testausta läpi tekoälyarkkitehtuurin kerrosten

OWASP AI Testing Guide -projekti määrittelee kattavia, jäsenneltyjä menetelmiä ja parhaita käytäntöjä tekoälyjärjestelmien testaamiseen arkkitehtuurin eri kerroksilla. Projekti kokoaa perinteisen kyberturvallisuuden, MLOps-testauksen ja vastuullisen tekoälyn testitapaukset jäsenneltyyn viitekehykseen. Testitapaukset luokitellaan neljään tasoon, jotka pohjautuvat tyypilliseen tekoälyarkkitehtuuriin: sovellus, malli, infrastruktuuri ja data.

Sovellustasolla testataan sovelluksen käyttöön liittyvät ja vuorovaikutuksen kautta ilmenevät riskit. Sovellustason testitapauksiin lukeutuvat esimerkiksi kehotteen suorat ja epäsuorat injektiot sekä sensitiivisen datan, syötteen tai järjestelmäkehotteiden vuotaminen. Jokainen testitapaus edistää tekoälyjärjestelmien tietoturvaa kokonaisvaltaisesti puuttumalla sovellustason riskeihin. Lisää testitapauksia löydät projektin dokumentaatiosta: www-project-ai-testing-guide/Document/content/3.1_AI_Application_Testing.md at main · OWASP/www-project-ai-testing-guide · GitHub.

Mallin tasolla testataan mallin kestävyyttä, haavoittuvuuksia, mallin ominaisuuksia ja käyttäytymistä. Lisäksi varmistetaan, että malli toimii tavoitteiden mukaisesti. Mallitasolla testaus kohdistuu erityisesti tekoälymallien luontaisiin ominaisuuksiin. Testitapaukset liittyvät erityisesti myrkyttämiseen (esim. mallin käytönaikainen myrkyttäminen) tai yksityisyyteen (esim. mallin käänteishyökkäykset, joissa rekonstruoidaan mallin tuloksista arkaluontoista koulutustietoa). Lisää mallitason testitapauksia: www-project-ai-testing-guide/Document/content/3.2_AI_Model_Testing.md at main · OWASP/www-project-ai-testing-guide · GitHub.

Infrastruktuurin testaus keskittyy tekoälyagentin toimintaa tukevan teknisen infrastruktuurin ja komponenttien haavoittuvuuksiin sekä riskeihin. Testauksessa huomioidaan mallin toimitusketjut, resurssien hallinta, turvalliset konfiguraatiot sekä suojaaminen väärinkäytöksiltä ja hyväksikäytöltä. Infrastruktuuritasolla testataan toimitusketjun peukalointiin, resurssien ylikuormittamiseen ja kehitysaikaisen mallin varkauksiin liittyviä tapauksia. Lisää infrastruktuuriin liittyviä testitapauksia: www-project-ai-testing-guide/Document/content/3.3_AI_Infrastructure_Testing.md at main · OWASP/www-project-ai-testing-guide · GitHub

Datan testauksessa keskitytään tekoälyagentin elinkaaren aikana käytetyn datan validointiin ja suojaamiseen. Datan testaamisessa pyritään varmistumaan datan laadusta ja yksityisuuden suojasta, datan riittävästä kattavuudesta sekä haitallisen ja sopimattoman sisällön vaikuttamisen

estämisestä. Dataan liittyvät haavoittuvuudet ovat vaikutuksiltaan usein laaja-alaisia. Testitapauksissa tarkastellaan muun muassa koulutusdatan paljastumista sekä tietojoukkojen haitallisia sisältöjä. Lisää testitapauksia: www-project-ai-testing-guide/Document/content/3.4_AI_Data_Testing.md at main · OWASP/www-project-ai-testing-guide · GitHub.

Jokaisella arkkitehtuurin tasolla testitapaukset linkittyvät aiemmin luvussa 7.1 käsiteltyihin tekoälytestaamisen osa-alueisiin (turvallisuus, yksityisyys, vastuullisuus ja luotettavuus). Jossain tapauksissa voi olla perusteltua keskittää testitapaukset tiettyyn osa-alueeseen, esimerkiksi turvallisuuteen tai tekoälyn luotettavuuteen.

7.3 Tekoälyagenttien testaaminen

Tekoälyagenttien testauksessa noudatetaan perinteisiä tietoturvatestaamisen vaiheita. Alussa on tärkeä määritellä tavoitteet, laajuus ja kriteerit, jotka vaikuttavat sovellettavien testitapausten valintaan, jotta testattava kokonaisuus ymmärretään hyvin. Uhkamallinnuksessa tunnistettuja riskejä käytetään hyökkäysskenaarioiden ja testitapausten määrittämisessä. Testien suorittamisen jälkeen havainnot ja niiden muodostamat riskit arvioidaan ja priorisoidaan, jotta tarvittavat korjaustoimenpiteet voidaan suorittaa kohdennetusti.

Testauksessa kiinnitetään huomiota mm. suorituskykyyn, muutoksiin sopeutumiseen, luotettavuuteen ja turvallisuuteen. Tehokkaassa testausprosessissa asiantuntijoiden kokonaiskäsitys testattavasta kohteesta yhdistyy automatisoituun testaukseen (ks. luku 7.3.1). Automatisoituja työkaluja voidaan käyttää apuna generoimaan massana useita erilaisia testitapauksia. Asiantuntijoiden näkemystä ja ohjausta tarvitaan erityisesti monimutkaisissa käyttötapauksissa, jotta testiskenaariot saadaan huomioimaan myös käyttötapaukseen mukautetut testidatat ja -tapaukset.

Agenttien epädeterministinen käytös tuo haasteita testaamiseen, sillä ei ole olemassa yhtä ainoaa oikeaa tavoiteltavaa vastausta. Toisin kuin perinteisissä sovelluksissa, tekoälyagenttien tulokset voivat vaihdella riippuen syötteen muotoilusta, historiasta tai päivityksistä itse malliin. Testaamisessa on otettava huomioon joukko hyväksyttyjä tuloksia ja arvioitava sekä oikeellisuutta että johdonmukaisuutta. Tekoälyä voi hyödyntää luomaan automatisoituja ja todellisiin skenaarioihin perustuvia testitapauksia, joita voidaan täydentää käsin valituin syöttein. Poikkeamien havaitseminen voi vaatia useiden testauskierrosten tulosten vertailua mahdollisten ongelmien löytämiseksi.

Koska vuorovaikutteisuus on agenteille ominaista, staattiset testitapaukset eivät yksin riitä. Testausstrategioiden on edustettava todellisia käyttötapoja tarkemmin kuin perinteisen laadunvarmistuksen. Testitapauksiksi on

simuloitava todellisen maailman skenaarioita, mukaan lukien rutiinitapauksia ja ääritilanteita. Lisäksi on arvioitava tekoälyn kyvykkyyttä käsitellä myös epäselviä tai kehittyviä skenaarioita ja agenttien vastuullisuutta ennakkoluulojen, väärän tiedon tai muun haitallisen sisällön osalta. [28]

7.3.1 Hybriditestaus

Tehokas testaaminen edellyttää usein hybridimallia, jossa yhdistetään automaatiota ja asiantuntijoiden näkemystä. Automatisoitu testaus nopeuttaa arviointiprosessia, kun arvioitavana on suuria määriä tekoälyn tuloksia. Asiantuntijoiden käyttö puolestaan auttaa kokonaiskuvan hahmottamisessa erityisesti monimutkaisissa käyttötapauksissa. Asiantuntijat huolehtivat, että tulokset ovat linjassa käyttötapauksen kanssa. Tekoälyagentin toimintaa on hyvä arvioida myös suorien käyttäjäkokeiluiden avulla. Monipuolinen käyttäjäpalaute auttaa käytettävyyteen liittyvien ongelmien selvittämisessä.

Myös kielimalleja voidaan hyödyntää toisten tekoälyjärjestelmien arvioinnissa (*LLM-as-a-judge*-menetelmä). Tätä tapaa voidaan käyttää täydentämään asiantuntijoiden analyysia ja samalla hyödyntää kielimallien skaalautumiskyvykkyyttä. Menetelmässä tekoälyagentista riippumaton kielimalli arvioi, kuinka hyvin agentti on onnistunut vastaamaan käyttäjän syötteeseen. Arviota tekevälle kielimallille annetaan kehote, joka kuvailee käyttäjän antaman tehtävän ja kielimallin tuotoksen. Näiden lisäksi kehotteessa tulee antaa kriteerit, joiden valossa kielimalli arvioi agentin tuotosta. Lopputuloksesta voidaan arvioida esimerkiksi tarkkuutta, hyödyllisyyttä tai sitä, vastaako tekstin sävy alkuperäisiä ohjeita. Arvioivan kielimallin lopputulos voi olla esimerkiksi useampi mitattava metriikka sekä perustelut sille, miksi näihin numeroihin on päädytty. Vertaamalla syötettä, lopputulosta ja ulkoisia lähteitä keskenään voidaan esimerkiksi arvioida, onko agentti hallusinoinut. Valmiita malleja sisällön tunnistamiseen ovat esimerkiksi Llama Guard[29] ja Constitutional Classifiers[30].

Laajoja kielimalleja voidaan vastaavasti hyödyntää myös testihyökkäysten generointiin osana *red teaming* -testausta. Tällaisia työkaluja ovat mm. Garak atkgen: Attack Generation[31] ja LLAMATOR Core[32]. Garak, Generative AI Red-teaming & Assessment Kit, on laajojen kielimallien haavoittuvuusskanneri, jota voidaan käyttää haavoittuvuuksien löytämiseen ja luokitteluun. Garak atkgen on automatisoitu hyökkäyssimulaattori, jonka anturi sisältää erillisen mallin, joka luo dynaamisesti hyökkäyskehotteita tarkoituksena

[28] Rashi Chandra (2025), Testing Your AI Agent: 6 Strategies That Definitely Work: https://insights.daffodilsw.com/blog/testing-your-ai-agent-6-strategies-that-definitely-work
[29] Inan ym., 2023, https://doi.org/10.48550/arXiv.2312.06674
[30] Sharma ym. 2025, https://doi.org/10.48550/arXiv.2501.18837
[31] atkgen: Attack Generation: https://reference.garak.ai/en/latest/garak.probes.atkgen.html
[32] LLAMATOR: https://github.com/LLAMATOR-Core/llamator

saattaa kielimalli vikatilaan. LLAMATOR Core on vastaavasti python-viitekehys chatbottien ja generatiivisten tekoälyjärjestelmien testaamiseen. Se sisältää laajan valikoiman laajoihin kielimalleihin, RAG-arkkitehtuuriin ja agentteihin kohdistuvia hyökkäyksiä. Lisäksi se mahdollistaa omien räätälöityjen hyökkäysten ja tietojoukkojen käytön.

7.3.2 Tietojoukot

Testaamisessa tulee käyttää monipuolisia tietojoukkoja käyttötapauksen kannalta olennaisista skenaarioista. Benchmark-tietojoukkojen etuna on suorituskyvyn mittaamisen vertailukyvykkyys suhteessa muihin agenttiratkaisuihin. Benchmark-tietojoukot voivat sisältää esimerkkikoodia ja standardoituja arviointiskriptejä, joiden tarkoitus on tehdä testauksesta toistettavaa ja läpinäkyvää. Tällaisten tietojoukkojen etuna on myös se, että ne ovat laajemman yhteisön validoimia. Valmiiden tietojoukkojen riittävyyttä on hyvä arvioida tekoälyagentin käyttötarkoituksen ja toimintaolosuhteiden perusteella.

Itse luodut tietojoukot soveltuvat parhaiten käyttötarkoitukseen ja antavat siten tarkempia vastauksia kuin julkiset vertailuarvot. Mukautettujen tietojoukkojen luominen edellyttää, että tietojoukon käyttötarkoitus on tarkkaan määritelty. Raakadata voi olla joko ihmisten luomaa tai automaattisesti kerättyä esimerkiksi julkisesta verkosta tai eri alustojen tarjoamien rajapintojen kautta. Myös kielimalleja tai generatiivista tekoälyä voidaan hyödyntää raakadatan luomiseen. Raakadata tulee käsitellä ja puhdistaa ennen käyttöä. Tämän lisäksi tietoaineistoon tulee lisätä metatiedot, jotta malli voi oppia siitä. Lopuksi tieto pitää jäsentää ja tallentaa siten, että se on helposti käytettävissä.

Yhdistämällä vakioituja ja käyttötarkoitukseen mukautettuja tietojoukkoja saadaan kattava kuva agentin suorituskyvystä[33].

Tietojoukoista lisää esimerkiksi seuraavista lähteistä:

- 25 AI benchmarks: examples of AI models evaluation[34]
- Google: Academic benchmarks to evaluate responsibility metrics[35]
- Hugging face: The AI community building the future[36]

[33] Conor Bronsdon (2025), How to Test AI Agents Effectively: https://galileo.ai/learn/test-ai-agents
[34] 25 AI benchmarks: examples of AI models evaluation (2025): https://www.evidentlyai.com/blog/ai-benchmarks
[35] Google AI for Developers, Academic benchmarks to evaluate responsibility metrics: https://ai.google.dev/responsible/docs/evaluation#benchmarks
[36] The AI community building the future: https://huggingface.co/

7.3.3 Adversariaalitestaus

Adversariaalitestaamisella pyritään löytämään tavat agentin tai mallin toiminnan rikkomiseen ja suojaustoimien kiertämiseen. Testiskenaarioiden tarkoitus on vahvistaa agentin puolustusta todellisen maailman hyökkäyksiä ja haavoittuvuuksia vastaan. Tekoälyagenttia tulee testata haitallisella sisällöllä ja lisäksi on huomioitava, että näennäisesti vaarattomat syötteet voivat paljastaa haavoittuvuuksia tai heikkouksia agentin toiminnassa. Tavoite on löytää keinot, joilla agentti voidaan turvata tunnistetuilta uhkilta.

Testiskenaarioiden on edustettava todellisia käyttötapauksia, mukaan lukien ääritilanteet. Agentteja testatessa tulee toiminnallisuuksien toimivuuden lisäksi arvioida myös toiminnan laatua. Perinteisen testauksen ja laadunvarmistuksen tulee mukautua tekoälyagenttien luonteen mukaisiin vaatimuksiin. On arvioitava eettisyyttä, tarkkuutta, turvallisuutta ja stressinsietokykyä.

Stressitestauksella mitataan kestävyyttä, arvioidaan agentin virheenkäsittelykykyä ja testataan kyvykkyyttä toimia arvaamattomissa tilanteissa, jossa esimerkiksi lähdeaineistot ovat ristiriidassa keskenään. Tekoälyn tietoturvatestaukseen soveltuvia työkaluja on esitelty luvussa 7.3.1. Testauksen työkalut ja liitteessä 3. Näistä työkaluista on apua hyökkäysskenaarioiden suunnittelussa, *red teaming* -testauksen suorittamisessa ja riskiarvioinnissa.

Tekoälyagenttien testaamiseen keskittyvä *Agentic AI Red Teaming* -opas [37] on kehitetty Cloud Security Alliancen (CSA) ja OWASP AI Exchangen yhteistyössä. Opas käy läpi agentillisen tekoälyn tietoturvahaasteet, testausvaatimukset, toiminnalliset vaiheet ja esimerkkikehotteet. Myös OWASP GenAI Red Teaming -opasta[38] voi hyödyntää testauksen suunnittelussa, sillä se kokoaa yhteen *red teaming* -testauksen periaatteet, tarkastuslistat sekä erilaisia tekniikoita.

7.3.4 Simulaatiotestaus

Simuloitujen testien avulla saadaan käsitys tekoälyagentin toiminnasta eri olosuhteissa. Lisäksi on arvioitava käytöstä todellisissa tilanteissa esimerkiksi pienimuotoisen käyttöönottotestauksen avulla. Simuloituja ja todellisia testejä yhdistämällä lisätään todennäköisyyttä sille, että tekoälyagentti toimii kaikissa tilanteissa. Tekoälyagenttien vuorovaikutuksen testaamiseen on olemassa esimerkiksi viitekehys τ-*bench*[39]. Sen avulla voidaan jäljitellä

[37] Cloud Security Alliance (CSA), Agentic AI Red Teaming Guide (2025): https://cloudsecurityalliance.org/artifacts/agentic-ai-red-teaming-guide
[38] OWASP GenAI Red Teaming Guide: https://genai.owasp.org/resource/genai-red-teaming-guide/
[39] Yao ym. (2024), τ -bench: A Benchmark for Tool-Agent-User Interaction in Real-World Domains: https://arxiv.org/abs/2406.12045

dynaamisia keskusteluja käyttäjän ja agentin välillä, mikä auttaa arvioimaan agentin suorituskykyä ja luotettavuutta.

7.3.5 Regressiotestaus

Etenkin päivitykset ja hienosäätö voivat muuttaa agenttien käyttäytymistä. Tekoälyagenttien vakaan toiminnan ylläpitämiseksi tulisi regressiotestausta suorittaa automatisoidusti etenkin mallin päivitysten yhteydessä. Regressiotestaukselle olisi hyvä määrittää säännölliset aikataulut, jolloin voidaan tunnistaa pienetkin poikkeamat suorituskyvyssä tai tarkkuudessa.[40]

7.3.6 Monitorointi ja poikkeamien hallinta

Tekoälyagenttien monitorointi on tärkeää, sillä tekoälyagentit oppivat ja voivat mukautua tiedoissa ja käyttäjissä tapahtuvien muutosten myötä. Jatkuva monitorointi on yksi tapa varmistaa, että agenttien toiminta pysyy käyttötarkoituksen mukaisena. Toimintaa on arvioitava vertaamalla tuloksia validoinnin perustasoja vasten poikkeamien havaitsemiseksi. Testien suorituskykyä on seurattava ja tuloksia arvioitava virheellisten toimintamallien havaitsemiseksi.

Agenttien luonteen takia on kiinnitettävä huomiota myös siihen, että testitapaukset päivitetään vastaamaan agentin nykytilaa. Agentin tila ei ole pysyvä, vaan sitä parannellaan ja hienosäädetään, jolloin myös testitapausten on pysyttävä muutoksen mukana. Agentin toiminnan ja tulosten nykytila tulee dokumentoida, jotta voidaan havaita myös pidemmän aikavälin muutokset.

Luotettavuuden varmistamiseksi tulee valvoa tuotannossa olevan agentin toimintaa, päätöksentekoa, suorituskykyä sekä vuorovaikutusta ulkoisten toimijoiden sekä käyttäjien kanssa. Lisäksi erityistä huomiota tulee kiinnittää lukuihin, joista voidaan havaita tekoälysovelluksen väärinkäyttö. Esimerkiksi luvut sovelluksen käyttömääristä, kutsutuista työkaluista, agenttien tavoitteista/tehtävistä tai suojaustoimista *(ks. 6.2.5)* voivat paljastaa agentin väärinkäytön. Monitoroinnin avulla voidaan myös seurata agenttien päätösten yhdenmukaisuutta, sillä agenttijärjestelmän hyökkäys voi paljastua, jos aikaisemmin hylätyt toimenpiteet yllättäen hyväksytään.

Poikkeamat agentin toiminnassa voivat olla järjestelmässä ilmeneviä vikoja tai harhoja, jotka voi aiheutua algoritmien vääristymisestä, tai haitallista toimintaa kuten kehotehyökkäys. Kummassakin tapauksessa olennaista on havainnoida poikkeamat mahdollisimman nopeasti, jotta niihin voidaan reagoida ja estää agentin virheellinen tai haitallinen toiminta. Havainnoinnissa

[40] Rashi Chandra (2025), Testing Your AI Agent: 6 Strategies That Definitely Work: https://insights.daffodilsw.com/blog/testing-your-ai-agent-6-strategies-that-definitely-work

voidaan käyttää automaattisia valvontaa, poikkeamien havaitsemiseen soveltuvia algoritmeja sekä käyttäjäraportointia.

Erityisesti moniagenttijärjestelmissä monitoroinnin avulla voidaan monimutkaisista kokonaisuuksista jäljittää viat tiettyyn komponenttiin tai toimintoon. Vaikka poikkeamien selvittäminen on monesti edelleen manuaalista työtä, voidaan automaatiota hyödyntää tietojen keräämisessä ja analysoinnissa.[41]

Joissain poikkeamatilanteissa voi olla tarpeen tehdä välittömiä rajoittavia toimenpiteitä (esim. järjestelmän irrottaminen verkosta tai sensitiivisten tietojen käytön rajoittaminen). Toimenpiteet poikkeaman korjaamiseksi voivat sisältää ohjelmiston tai algoritmien päivityksiä, käytön väliaikaisen keskeyttämisen tai kiertoteiden käyttöönottamista. Keskeistä tekoälypoikkeamien hallinnassa on ennakoida ja tunnistaa mahdollisia poikkeamatilanteita, laatia näihin varautumistoimenpiteet ja huolehtia järjestelmän jatkuvasta monitoroinnista, jotta poikkeavat tilanteet voidaan havaita. Poikkeamienhallinta tulee aina räätälöidä organisaation tekoälyjärjestelmien ja mahdollisten uhkaskenaarioiden mukaan.

7.4 Testauksen työkalut

Tekoälyn testaustyökaluja löytyy laajalti eri käyttötarkoituksiin soveltuen. OWASP AI Exchangen listaamia ja kuratoimia työkaluja on koottu käyttötapauksittain liitteelle 3. Listaus edustaa tämän hetken tilannetta, ja uusia työkaluja ja menetelmiä syntyy jatkuvasti lisää. Ajantasaisin lista kannattaa tarkastaa AI Exchangen dokumentaatiosta[42].

[41] Lindemulder ym., Why observability is essential for AI agents: https://www.ibm.com/think/insights/ai-agent-observability
[42] OWASP AI Exchange 5. AI security testing: https://owaspai.org/goto/testing/

Liite 1: Hankintaohje agenttijärjestelmille

Tässä liitteessä tarjoamme tekoälyagentin hankintaan muistilistan asioista, joita ostajan on syytä varmistaa agentin toimittajalta. Kriteereitä on paljon, eikä aina ole tarpeen olettaa, että jokainen järjestelmä täyttää niistä kaikki. Listasta kannattaa siis poimia ne kohdat, jotka ovat sovelluskohteen, tehtävän, ydinliiketoiminnan turvaamisen ja organisaation tavoitteiden kannalta tärkeimmät.

Ohjeistuksen tarkoituksena on täydentää organisaatioiden jo olemassa olevia turvallisuusohjeistuksia ja -politiikkoja. On tärkeää, että hankittavia tekoälyagentteja arvioidaan myös olemassa olevien ohjelmistoturvallisuutta koskevien ohjeistusten mukaisesti.

Hallinta ja elinkaaren ylläpito

☐ Tarkista, mitä kielimalleja toimittaja tarjoaa ja että ne sopivat käyttötapaukseen.

- Tarjolla on kielimalleja useilta eri toimittajilta. Miten laaja tarjonta palvelun tarjoajalla on? Millainen mallien julkaisuviive tarjoajalla on?

☐ Tarkista, kuinka avoimesti tarjoaja mahdollistaa agenttien viemisen toisiin järjestelmiin.

- Kehote, oma data ja määritykset ovat tärkeä osa tekoälyagenttien kehitystä. Varmistamalla, miten tiedot haetaan sovelluksesta, mahdollistetaan helpompi siirtyminen tarvittaessa uuteen palveluntarjoajaan.

☐ Varmista, että hankittava järjestelmä mahdollistaa elinkaaren hallinnan, kun kielimallit päivittyvät.

- Voidaanko samaa kielimallia käyttää vähintään oman julkaisusyklin tai useamman yli? Jääkö aikaa uuden testaamiseen nykyisen vanhetessa?
- Millaisen päivityspolun toimittaja tarjoaa uuteen kielimalliin? Mitkä uuden mallin toiminnallisuudet tai kyvykkyydet voi ottaa käyttöön suoraan, ja mitkä vaativat muutoksia malleja käyttävään sovellukseen, järjestelmään tai agenttiin?
- Voidaanko palata aikaisempaan malliin tai julkaisuun tarvittaessa?
- Kuinka taataan, että kielimalli on käytettävissä halutulla maantieteellisellä alueella?

☐ Varmista, myös että toimitukseen sisältyy

- ModelOps-/agenttiversiohallinta.
- CI/CD-putket turvalliseen käyttöönottoon ja palautukseen.
- hallintapaneelit järjestelmän tilan ja sääntelyn seurantaan.
- mahdollisuus hallintasääntöjen pakottamiseen (esim. eskalointirajat).
- toimittajan päivitys- ja korjauspolitiikan dokumentaatio.
- näyttöä jatkuvasta tutkimus-, kehitys ja standardointiosallistumisesta.

Toiminnallisuus ja integraatiot

☐ Tarkista, että sovellus tarjoaa mahdollisuuden tallentaa omia dokumentteja.

☐ Tarkista, että sovellus tarjoaa sopivan arkkitehtuurin tietojen prosessointiin.

- Tukeeko sovellus RAG-arkkitehtuuria?
- Mitä upotusmalleja on käytössä?

☐ Varmista, että agenttia voidaan kustomoida tarpeen vaatiessa.

- Miten agentteja voi muokata?
- Onnistuuko kehotteen, työkalujen ja tehtävien määrittely helposti?

☐ Tarkista, kuuluuko toteutukseen valmis palvelu agenttien testaamiseen useissa eri ympäristöissä.

☐ Varmista agentin asianmukainen monitorointi.

- Kuinka palveluntarjoaja hoitaa monitoroinnin?
- Voiko monitoroitavia komponentteja muokata?
- Voiko monitoroinnin lisäksi tehdä hälytyksiä/ilmoituksia esimerkiksi sähköpostiin?

☐ Tarkista, tukeeko palvelu moniagenttijärjestelmiä etenkin, jos käyttötapaus vaatii sitä nyt tai mahdollisesti tulevaisuudessa.

☐ Varmista, että agenttiin on mahdollista liittää tarvittavat työkalut.

- Onko omien työkalujen lisääminen tekoälyagenteille järjestelmään mahdollista?
- Tukeeko toimittaja uusimpia protokollia (esim. MCP, A2A)?
- Kuinka uudet työkalut lisätään järjestelmään?
- Kuka “omistaa” uudet työkalut?

☐ Tarkista, että tarjoajalla on valmiit tai muokattavat rajapinnat (API:t) myös yrityksen järjestelmäintegraatioita varten.

- Voiko järjestelmää integroida organisaatiosi käytössä oleviin kyberturvallisuusratkaisuihin (esim. SIEM)?

☐ Tarkista, että tarjoaja on varmistanut standardien tuen (esim. FIPA ACL, RESTful API:t).

☐ Varmista, että agentti on tarvittaessa integroitavissa myös vanhojen järjestelmien kanssa.

Turvallisuus

☐ Varmista turvallinen käyttäjien hallinta.

- Mahdollistaako sovellus käyttäjien hallinnan?
- Kuinka pääsyoikeudet erilaiseen dataan varmistetaan?
- Voiko järjestelmän käyttöä rajoittaa IP-osoitteella?
- Agenttien identiteetin vahvistus (IAM-integraatio, RBAC).

☐ Varmista, että tekoälyagenttien työkalujen käytössä on huomioitu työkalujen ja integraatioiden hallinta ja turvallisuus. Ilman työkalujen tarkkaa validointia agentit voivat tehdä epätoivottuja tehtäviä, kuten lähettää yrityksen dataa ulkoisiin järjestelmiin.

- Missä työkalut suoritetaan?
- Miten työkalujen tietoturva on varmistettu?
- Voiko jokaisen työkalukutsun väliin vaatia ihmisen varmistuksen?
- Onko työkalujen kuvaukset ja toiminnallisuudet käyttäjien saatavilla?
- Voiko tekoälyagentin yhdistää sovelluksen ulkopuolisiin työkaluihin (esim. omat MCP-palvelimet).

☐ Varmista, että agenttiin sisältyy riittävät suojaustoimet syötteen ja palautetun arvon validointiin.

- Onko sovelluksessa tarkistukset syötteelle ja palautetulle arvolle?
- Voiko järjestelmään rakentaa omat validoinnit?

☐ Varmista lisäksi seuraavat suojatoimet:

- Zero trust -kommunikaatio (molemminpuolinen tunnistautuminen, TLS-salaus).
- Tietoturvaväylä/sandbox ulkoisia työkalu- ja API-kutsuja varten.
- Toimittajalla on riittävät uhkamallinnuskäytännöt esimerkiksi kehoteinjektioiden, identiteettiväärennösten ja datan myrkytyksen varalta.
- Monitorointi ja poikkeamien havaitsemista agenttien epäilyttävälle käytökselle.

☐ Halutessasi voit myös tarkistaa, tukeeko toimittaja luottamus-/mainepisteytystä agenteille.

Tietosuoja ja sääntely

☐ Varmista, että tarjoaja on ottanut huomioon tietosuojan osalta ainakin seuraavat asiat:

- Datan minimointi ja luokittelu agenttien välillä
- Salaus levossa ja siirrossa.

- GDPR/HIPAA (ja muu organisaatiota ja käyttötarkoitusta vastaava sääntely) mahdollisimman pitkälle sisäänrakennettuna.
- Säilytys ja poistopolitiikat määriteltävissä
- Toimittaja noudattaa vaadittuja viitekehyksiä (esim. NIST AI RMF / ISO 42001 / AI TRiSM)

Läpinäkyvyys ja auditoitavuus

☐ Varmista, että agenttiin sisältyy globaali auditointiloki kaikista agenttien toimista ja vuorovaikutuksista.

☐ Tarkista, että agenttiratkaisu tarjoaa riittävät työkalut selitettävyyden takaamiseksi (päätösketjut, perustelulokit).

☐ Varmista, että järjestelmä tukee ihmisen valvontaa ja ohjausta reaaliaikaisessa toiminnassa.

☐ Varmista, että toimittaja pystyy osoittamaan yhteistyömittarit (esim. Component Synergy Score).

Muut

☐ Varmista laskutuksen osalta seuraavat asiat:

- Kuinka sovelluksen laskutus toimii? Tekoälyagentit vievät paljon laskentatehoa, joten varmista, mistä laskutetaan ja miten.
- Kuinka ennakoitavia kustannukset ovat? Muutokset ja epätasaisuus agentin käytössä vaikuttaa kustannusten jakautumiseen ja ennakoitavuuteen.

☐ Tarkista myös, onko organisaatiossasi muita (tekoälyn) hallintaprosesseja, jotka juuri sinun organisaatiosi on otettava huomioon, jotta hankittu agenttiratkaisu ei vaaranna ydinliiketoimintaa. Esimerkkejä alakohtaisista huomioitavista asioista:

- Kriittisen infrastruktuurin turvallisuussääntely
- Hoitotyön eettiset periaatteet ja lääkinnällisten laitteiden sääntely
- Demokraattisilta prosesseilta vaadittu läpinäkyvyys ja lainmukaisuus (ml. hallintolain vaatimukset automaattisesta päätöksenteosta)

Liite 2: Uhkamallinnuspohja

Uhkamallinnus toteutettiin <ajankohtana> <palvelun nimelle>. Uhkamallinnuksen tarkoituksena on tunnistaa ja dokumentoida järjestelmän kriittiset tiedot, rajapinnat, ulkoiset riippuvuudet ja tietovirrat. Järjestelmästä saadun ymmärryksen perusteella voidaan arvioida, millaisia potentiaalisia uhkia järjestelmään kohdistuu, kuinka todennäköisiä ne ovat sekä mitä vaikutuksia on uhkien realisoitumisella. Uhkamallinnuksessa tunnistetuille uhkille ehdotetut mahdolliset hallintakeinot toteutetaan aloittaen korkeamman priorisoinnin saaneista uhkista. Toteuttamisen jälkeen, on suotavaa varmistaa teknisten kontrollien toimivuus käytännön testauksella.

Taulukko 1. Työpajat

Työpaja xx.xx.20xx	
<Omistajayritys>	<nimi>
<Toimittajayritys>	<nimi>
<Konsulttiyritys>	<nimi>
Työpaja xx.xx.20xx	
<Omistajayritys>	<nimi>
<Toimittajayritys>	<nimi>
<Konsulttiyritys>	<nimi>
Työpaja xx.xx.20xx	
<Omistajayritys>	<nimi>
<Toimittajayritys>	<nimi>
<Konsulttiyritys>	<nimi>

Kohde

Kohteen kuvaus

- Käyttötarkoitus
- Korkean tason toiminnan kuvaus
- Käytetyt teknologiat

Suojattavat tiedot, kohteet ja toiminnot

Taulukko 2. Suojattavat kohteet

Nimi	Kuvaus

Käyttäjätasot ja oikeudet

Taulukko 3. Käyttäjätasot

Käyttäjätaso	Oikeudet

Riippuvuudet

Taulukko 4. Riippuvuudet

Riippuvuus	Kuvaus	Ylläpitävä taho

Tietovirtakaavio

Kuva tietovirroista, josta ilmenee järjestelmäkokonaisuuden luottamusrajat sekä ne tietovirrat, jotka kuvaavat, mitkä komponentit kommunikoivat keskenään.

Uhkat

Taulukko 5. Uhkat

Uhka	Kuvaus

Uhkien hallinta

Taulukko 6. Todennäköisyys

Todennäköisyys	
Lähes varma	Uhka voi realisoitua toistuvasti
Todennäköinen	Toteutuminen on kohtalaisen yleistä.
Mahdollinen	Konkreettisesti toteutunut meillä tai muualla.
Epätodennäköinen	Toteutuminen on teoreettisesti mahdollista, mutta käytännössä tapahtuminen on erittäin harvinaista.

Taulukko 7. Vaikutus

Vaikutus	
Kriittinen	Toteutuminen johtaa koko toimintaprosessin pysähtymiseen ja järjestelmän sisältämä data saattaa olla palauttamattomissa. Palautuminen ja tavallisiin toimintaprosesseihin palaaminen voi viedä päiviä tai jopa viikkoja. Usein kriittisen uhkan toteutuminen voi johtaa operatiiviseen kriisitilanteeseen. Ihmishenkiin vaikuttavat uhkat ovat aina kriittisiä.
Merkittävä	Toteutuminen voi johtaa kriisitilanteeseen tai olla huomattavia taloudellisia ja/tai yrityksen imagoon kohdistuvia vaikutuksia. Mahdollisuus kohdistua suureen määrään käyttäjiä/dataa.
Kohtalainen	Toteutuminen voi johtaa rajattuun datan tai yrityksen toiminnan vaarantumiseen. Kohdejärjestelmä saattaa olla hetkellisesti saavuttamattomissa. Vaikuttaa kohtuulliseen määrään käyttäjiä/dataa, voi johtaa turvallisuuden heikkenemiseen epäsuorasti.
Vähäinen	Toteutuminen saattaa aiheuttaa käytettävyyden vähäistä heikkenemistä pienelle joukolle, mutta sovelluksen toiminta ei vaarannu.

Uhkien suuruusluokan laskentakaava:

todennäköisyys x vaikutus = riskiluokka

Taulukko 8. Riskiluokat

Riskiluokat				
Kriittinen	Merkittävä	Sietämätön	Sietämätön	Sietämätön
Merkittävä	Huomioitava	Merkittävä	Merkittävä	Sietämätön
Kohtalainen	Huomioitava	Huomioitava	Merkittävä	Merkittävä
Vähäinen	Vähäinen	Huomioitava	Huomioitava	Merkittävä
Vaikutus / Toden-näköisyys	Epätodennä-köinen	Mahdollinen	Todennäköi-nen	Lähes varma

Sietämätön riski – Vältä riskialttiin toiminnan aloittamista. Käynnistä välittömästi toimenpiteet riskin poistamiseksi.

Merkittävä riski – Suunnittele ja käynnistä toimenpiteet riskin pienentämiseksi nopeasti.

Huomioitava riski – Suunnittele ja aikatauluta toimenpiteet riskin alentamiseksi. Harkitse toimenpiteiden kannattavuus.

Vähäinen riski – Toimenpiteitä ei yleensä tarvita. Valvo, että tilanne pysyy hallinnassa.

Uhkamalli

Taulukko 9. Uhkamalli

Uhka	Toden-näköi-syys	Vaikutus	Riskiluokka	Hallintakeinot
				Suojautuminen: Havainnointi: Reagointi: Palautuminen: Vastuuhenkilö: Status:
				Suojautuminen: Havainnointi: Reagointi: Palautuminen: Vastuuhenkilö: Status:
				Suojautuminen: Havainnointi: Reagointi: Palautuminen: Vastuuhenkilö: Status:
				Suojautuminen: Havainnointi: Reagointi: Palautuminen: Vastuuhenkilö: Status:
				Suojautuminen: Havainnointi: Reagointi: Palautuminen: Vastuuhenkilö: Status:

Versiohistoria

Taulukko 10. Versiohistoria

Versio	Päivämäärä	Tekijä	Muutokset

Liite 3: Testaamisen työkalut

Tässä liitteessä listataan avoimen lähdekoodin *red teaming* -testauksen työkaluja käyttötapauksittain. Ajantasaisin listaus työkaluista kannattaa tarkastaa OWASP AI Exchangen sivuilta: https://owaspai.org/goto/testing/

Kehotteen injektointi

Kehotteen injektoinnissa hyökkääjä antaa mallille manipuloivia ohjeita, joiden tarkoituksena on saavuttaa haitallisia tuloksia tai tavoitteita.

Taulukko 11. Soveltuvat työkalut

Nimi	Toiminta	APIn käyttö	Kieli	Github
PyRIT, Python Risk Identification Tool for generative AI	Havainnointi	✔	Python	https://github.com/Azure/PyRIT
Garak	Havainnointi	✔	Python	https://github.com/NVIDIA/garak
Prompt Fuzzer	Havainnointi	✔	Python	https://github.com/prompt-security/ps-fuzz
Guardrails	Havainnointi ja suojaaminen	✔	Python	https://github.com/guardrails-ai/guardrails
Promptfoo	Havainnointi ja suojaaminen	✔	Python, NodeJS	https://github.com/promptfoo/promptfoo

Taulukko 12. Työkalujen soveltuvuus viitekehyksiin ja alustoihin

Viitekehys / alusta	Kategoria	Työkalun soveltuvuus
Tensorflow	DL, GenAI	PyRIT
PyTorch	DL, GenAI	PyRIT, Garak, Guardrails
Azure OpenAI	GenAI	PyRIT, Guardrails, Promptfoo
Huggingface	ML, GenAI	PyRIT, Garak, Guardrails, Promptfoo
Azure managed endpoints	Machine Learning Deployment	PyRIT
Cohere	GenAI	PyRIT, Garak, Guardrails, Promptfoo
Replicate Text Models	GenAI	PyRIT, Garak, Promptfoo
OpenAI API	GenAI	PyRIT, Garak, Prompt Fuzzer, Guardrails, Promptfoo

GGUF (Llama.cpp)	GenAI, Light-weight Inference	PyRIT, Garak, Promptfoo
OctoAI	GenAI	Garak

Taulukko 13. Tietomuotojen tuki

Tyyppi	Työkalu
Teksti	PyRIT, Garak, Prompt Fuzzer, Guardrails, Promptfoo
Kuva	-
Audio	-
Video	-
Taulukoitu data	-

Mallin päätelmien hyödyntäminen

Näiden hyökkäysten tarkoituksena on hyödyntää päättelyprosessia luvattoman tiedon hankkimiseksi tai mallin vaarantamiseksi.

Taulukko 14. Soveltuvat työkalut

Nimi	Toiminta	APIn käyttö	Kieli	Github
ART, The Adversarial Robustness Toolbox	Havainnointi	✔	Python	https://github.com/Trusted-AI/adversarial-robustness-toolbox
Armory	Havainnointi	✔	Python	https://github.com/twosixlabs/armory

Taulukko 15. Työkalujen soveltuvuus viitekehyksiin ja alustoihin

Viitekehys / alusta	Kategoria	Työkalun soveltuvuus
Tensorflow	DL, GenAI	ART, Armory
Keras	DL, GenAI	ART
PyTorch	DL, GenAI	ART, Armory
MxNet	DL	ART
Scikit-learn	ML	ART
XGBoost	ML	ART
LightGBM	ML	ART
CatBoost	ML	ART
GPy	ML	ART

Taulukko 16. Tietomuotojen tuki

Tyyppi	Työkalu
Teksti	ART, Armory
Kuva	ART, Armory
Audio	ART, Armory
Video	ART, Armory
Taulukoitu data	ART, Armory

Mallin varastaminen käytön aikana

Hyökkääjät kohdistavat hyökkäyksensä mallin osiin tai kriittisiin komponentteihin, kuten järjestelmäkehotteisiin. Näin he saavat mahdollisuuden luoda kehittyneitä syötteitä, jotka ohittavat suojaustoimet.

Taulukko 17. Soveltuvat työkalut

Nimi	Toiminta	APIn käyttö	Kieli	Github
ART, The Adversarial Robustness Toolbox	Havainnointi	✔	Python	https://github.com/Trusted-AI/adversarial-robustness-toolbox
Armory	Havainnointi	✔	Python	https://github.com/twosixlabs/armory
Promptfoo	Havainnointi ja suojaaminen	✔	Python, NodeJS	https://github.com/promptfoo/promptfoo

Taulukko 18. Työkalujen soveltuvuus viitekehyksiin ja alustoihin

Viitekehys / alusta	Kategoria	Työkalun soveltuvuus
Tensorflow	DL, GenAI	ART, Armory
Keras	DL, GenAI	ART
PyTorch	DL, GenAI	ART, Armory
MxNet	DL	ART
Scikit-learn	ML	ART
XGBoost	ML	ART
LightGBM	ML	ART
CatBoost	ML	ART
GPy	ML	ART
Azure OpenAI	GenAI	Promptfoo
Huggingface	ML, GenAI	Promptfoo
Cohere	GenAI	Promptfoo
Replicate Text Models	GenAI	Promptfoo
OpenAI API	GenAI	Promptfoo
GGUF (Llama.cpp)	GenAI, Lightweight Inference	Promptfoo

Taulukko 19. Tietomuotojen tuki

Tyyppi	Työkalu
Teksti	ART, Armory, Promptfoo
Kuva	ART, Armory
Audio	ART, Armory
Video	ART, Armory
Taulukoitu data	ART, Armory

Suora mallin varastaminen

Tässä hyökkäyksessä mallin parametrit tai toiminnallisuus varastetaan. Tämä mahdollistaa hyökkääjän luomaan kopion mallista, jota voidaan sitten käyttää oraakkelina vihamielisten hyökkäysten ja muiden yhdistettyjen uhkien luomiseen.

Taulukko 20. Soveltuvat työkalut

Nimi	Toiminta	APIn käyttö	Kieli	Github
Prompt Fuzzer	Havainnointi	✔	Python	https://github.com/prompt-security/ps-fuzz

Taulukko 21. Työkalujen soveltuvuus viitekehyksiin ja alustoihin

Viitekehys / alusta	Kategoria	Työkalun soveltuvuus
OpenAI API	GenAI	Prompt fuzzer

Taulukko 22. Tietomuotojen tuki

Tyyppi	Työkalu
Teksti	Prompt Fuzzer
Kuva	-
Audio	-
Video	-
Taulukoitu data	-

Mallin myrkyttäminen kehityksen aikana

Tämä tarkoittaa mallin parametrien, tietojen tai toimitusketjun manipulointia kehitysvaiheen aikana mallin toiminnan muuttamiseksi. Hyökkääjän tavoitteena on muuttaa mallin käyttäytymistä, mikä voi johtaa ei-toivottuun toimintaan.

Taulukko 23. Soveltuvat työkalut

Nimi	Toiminta	APIn käyttö	Kieli	Github
ART, The Adversarial Robustness Toolbox	Havainnointi	✔	Python	https://github.com/Trusted-AI/adversarial-robustness-toolbox
Armory	Havainnointi	✔	Python	https://github.com/twosixlabs/armory
TextAttack	Havainnointi	✔	Python	https://github.com/QData/TextAttack

Taulukko 24. Työkalujen soveltuvuus viitekehyksiin ja alustoihin

Viitekehys / alusta	Kategoria	Työkalun soveltuvuus

Tensorflow	DL, GenAI	ART, Armory, TextAttack
Keras	DL, GenAI	ART
PyTorch	DL, GenAI	ART, Armory, TextAttack
MxNet	DL	ART
Scikit-learn	ML	ART
XGBoost	ML	ART
LightGBM	ML	ART
CatBoost	ML	ART
GPy	ML	ART

Taulukko 25. Tietomuotojen tuki

Tyyppi	Työkalu
Teksti	ART, Armory, TextAttack
Kuva	ART, Armory
Audio	ART, Armory
Video	ART, Armory
Taulukoitu data	ART, Armory

Suojausten kiertäminen

Suojausten kiertäminen tarkoittaa sitä, että vastustaja manipuloi syötettä päättelyhetkellä niin, että malli tuottaa väärän tuloksen, vaikka manipuloitu syöte näyttää edelleen normaalilta ihmisille tai jatkokäsittelyjärjestelmille. Mallia ei kouluteta uudelleen tai myrkytetä — sen sijaan hyökkääjä luo syötteitä, jotka ”liukuvat ohi” mallin päätösrajan.

Taulukko 26. Soveltuvat työkalut

Nimi	Toiminta	APIn käyttö	Kieli	Github
ART, The Adversarial Robustness Toolbox	Havainnointi	✔	Python	https://github.com/Trusted-AI/adversarial-robustness-toolbox
Armory	Havainnointi	✔	Python	https://github.com/twosixlabs/armory
Foolbox	Havainnointi	✔	Python	https://github.com/bethgelab/foolbox
DeepSec	Havainnointi	✔	Python	https://github.com/ryderling/DEEPSEC
TextAttack	Havainnointi	✔	Python	https://github.com/QData/TextAttack
PyRIT, Python Risk Identification Tool for generative AI	Havainnointi	✔	Python	https://github.com/Azure/PyRIT

Garak	Havain-nointi	✔	Python	https://github.com/NVI-DIA/garak

Taulukko 27. Työkalujen soveltuvuus viitekehyksiin ja alustoihin

Viitekehys / alusta	Kategoria	Työkalun soveltuvuus
Tensorflow	DL, GenAI	ART, Armory, Foolbox, Deepsec, TextAttack, PyRIT
Keras	DL, GenAI	ART, Foolbox
PyTorch	DL, GenAI	ART, Armory, Foolbox, Deepsec, TextAttack, PyRIT, Garak
MxNet	DL	ART
Scikit-learn	ML	ART
XGBoost	ML	ART
LightGBM	ML	ART
CatBoost	ML	ART
GPy	ML	ART
Azure OpenAI	GenAI	PyRIT
Huggingface	ML, GenAI	PyRIT, Garak
Azure managed endpoints	Machine Learning Deployment	PyRIT
Cohere	GenAI	PyRIT, Garak
Replicate Text Models	GenAI	PyRIT, Garak
OpenAI API	GenAI	PyRIT, Garak
GGUF (Llama.cpp)	GenAI, Light-weight Inference	PyRIT, Garak
OctoAI	GenAI	Garak

Taulukko 28. Tietomuotojen tuki

Tyyppi	Työkalu
Teksti	ART, Armory, Foolbox, DeepSec, TextAttack, PyRIT, Garak
Kuva	ART, Armory, Foolbox, DeepSec
Audio	ART, Armory
Video	ART, Armory
Taulukoitu data	ART, Armory

Liikenne- ja viestintävirasto Traficom
PL 320, 00059 TRAFICOM
p. 029 534 5000

traficom.fi

ISBN 987-952-425-008-5
ISSN 2669-8787 (verkkojulkaisu)

TRAFICOM
Liikenne- ja viestintävirasto