DOI: https://doi.org/10.12700/btsz.2026.8.2.127

ARTIFICIAL INTELLIGENCE AGENTS IN OFFENSIVE AND DEFENSIVE CYBER SECURITY

MESTERSÉGES INTELLIGENCIA ÁGENSEK AZ OFFENZÍV ÉS DEFENZÍV KIBERBIZTONSÁGBAN

KELEMEN László[1] – OLÁH Róbert[2]

Abstract

This paper explores the use of AI-based autonomous agents in offensive and defensive cybersecurity environments. Through laboratory experiments, it evaluates the vulnerability detection and network monitoring capabilities of large language model–based agents. The study analyzes agent behavior in a Cisco Packet Tracer simulated network using ASI, DC, and FT metrics, and tests a DeepSeek R1 and OpenClaw Clawdbot–based agent in a simulated office infrastructure. The research pro-vides practical insights into AI-driven threats and defense strategies for secu-rity professionals, pentesters, SOC/CSIRT teams, and researchers, who are interested in the "Practical understanding of AI-based threats and defense options in a pre-designed lab environment where the offensive and defensive capabilities of AI agents were evaluated.

Keywords

cybersecurity, threats, network traffic, pentesters, autonomous agents, lan-guage models

Absztrakt

A cikk az AI-alapú autonóm ügynökök (agentek) defenzív és offenzív kiberbiztonságban betöltött szerepét mutatja be és vizsgálja. A tanulmány célja annak feltárása, hogy a jelenlegi nagy nyelvi modellekre épülő „agentek" milyen tényleges sérülékenység-felderítési és hálózati forgalom-elemzési képességekkel rendelkeznek, saját kialakított labor környezetekben mért, reprodukálható kísérletek alapján. A vizsgálat egyik ága egy Cisco Packet Tracer alapú szimulált hálózaton elemzi az AI ágensek észlelési és döntési viselkedését (ASI, DC, FT mérőszámok), míg a másik ága egy szimulált irodai infrastruktúrát modellező laborban értékeli egy DeepSeek R1 modellre épülő, OpenClaw Clawdbot alapú autonóm „agent" felderítési és sérülékenység-azonosítási teljesítményét. A mű elsődleges célközönsége: biztonsági szakemberek, penteszterek, SOC/CSIRT csapatok, valamint kutatók, akik az „Az AI-alapú fenyegetések és védekezési lehetőségek gyakorlati megismerése egy előre kialakított laborkörnyezetben, ahol AI-agentek támadó és védelmi offenzív képességeit értékelését végzik.

Kulcsszavak

kiberbiztonság, fenyegetés, hálózati forgalom, penteszterek, autonóm ügynökök, nyelvi modellek.

[1] kelemenl@cyberexpert.hu | ORCID: 0009-0007-7566-8063 | penetrációs teszter, penetration tester | Sérülékenység kutató, etikus hacker, OFSZ Zrt

[2] olah.robert0721@gmail.com | ORCID: 0009-0007-9134-9521 | IT teacher, Center of Erd Education | Informatika oktató, Érdi Tankerületi Központ

BEVEZETÉS

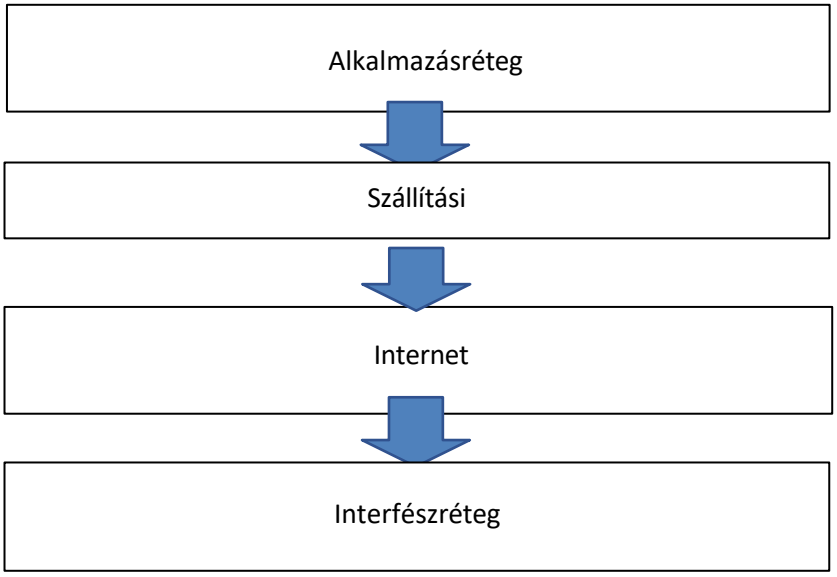
A mesterséges intelligencia és a kiberbiztonság konvergenciája az elmúlt évek egyik legdinamikusabban fejlődő kutatási területe. Az LLM-alapú autonóm ügynökök megjelenése új típusú fenyegetéseket és egyben új védelmi lehetőséget is jelentenek. Jelen tanulmány célja kettős: egyfelől elméleti és szimulációs megközelítésben mutatja be az AI ágensek és a hálózati sérülékenységek viszonyát, másfelől egy gyakorlati, izolált laboratóriumi környezetben végzett kísérletsorozat eredményeit ismerteti, ahol egy LLM-alapú „agent" szimulált irodai hálózati infrastruktúrán végzett felderítési és sérülékenység detektálási feladatokat. Az offenzív, sérülékenység felderítési vizsgálat célja az is, hogy bemutassa egy esetleges rosszindulatú támadó milyen képességű rendszert tud összerakni nyílt-forráskódú, illetve könnyen hozzáférhető rendszerekből.

Természetes nyelvi feldolgozás

A természetes nyelvi feldolgozás egy komoly kutatási terület, mert ezen a területen is alkalmazható a gépi tanulás, a gépi intelligencia a nyelv megértése, az emberi nyelvet amely lehet írott vagy szóbeli formában (adott szöveg értelmezés és felismerés , beszédfelismerés, fordítás adott nyelvről másik idegen nyelvre) A modell betanítjuk arra a funkcióra amire a feladat specifikációja szól. Nyilván van lehetőség további utakat nyitni, a modell további tanulás céljából. Az NLP segítségével, a szöveget tudjuk értelmezni, hangalapú információt felismerni, idegen nyelvű fordítás. Az NLP működésében a matematikai műveletek állnak, amelynek segítségével a gép számára értelmezhető az információ. Tekintettel mivel a kutatás a agensek témakörére épül, az ügynök az lehet maga a nyelvi modell, amire tanítunk, szabályokat állíthatunk fel a modellben. Az ügynököket egy neurális hálózatban képzeljük el, ezek lesznek a csomópontok az élek pedig a kommunikációs csatornák. Input ügynökként az információ feldolgozása történik meg, amely átmegy egy tisztító műveleten, majd ezután történik meg az információ elemzése, végül megtörténik a döntés és a válasz amely az eredmény kimenete lesz. Egy modell struktúrában több okos modell dolgozhat együtt, elemző ügynök, döntési ügynök, végrehajtó ügynök. koordinátor ügynök. Az elemző ügynök értelmezheti a rendszerben a logokat, és eseményeket, és elvégezheti az anomáliák felismerését is. A döntési ügynök, priorizálja a fenyegetéseket és ennek tudatában születik meg a döntés hogyan viselkednek az ágens. **[1][6]**

Sérülékenységet kategorizált súlyosság szerint

Amikor több számítógépet összekapcsolunk egy hálózatban, az információ nem közvetlenül, egy lépésben jut el egyik gépből a másikba, hanem több feldolgozási szakaszon keresztül halad. A továbbított adatok először jelekké alakulnak, amelyek lehetnek például elektromos impulzusok, rádióhullámok vagy fényjelek. Ezeket a jeleket a fogadó rendszer értelmezi, majd fokozatosan visszaalakítja olyan adatokká, amelyeket már szoftverek is fel tudnak dolgozni. Így a nyers fizikai jelből végül egy strukturált információ lesz, amit a felhasználói alkalmazások meg tudnak jeleníteni vagy használni. A hálózatban többféle protokoll működik kiemelt a TCP -IP protokoll. A TCP-ip protokoll. A protokollok segítségével jut el a csomag a hálózat egyik szegmenséből a másikba. [11]



1.ábra TCP IP réteg (saját szerkesztés)

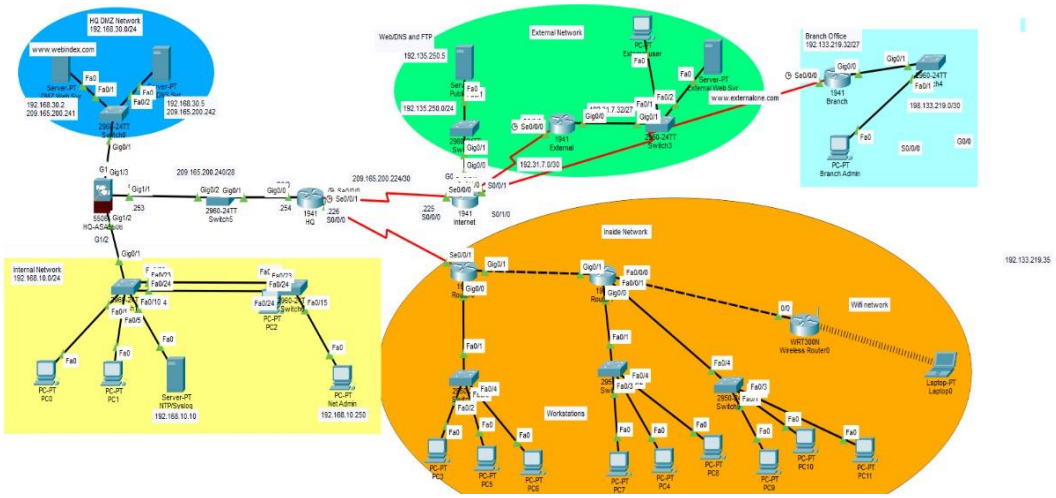
LLM konkrét shell parancsot generál

A nagy nyelvi modellben lehetőségünk van shell parancsot is generálni, a generálás a nyelvi modell által amit betanítottak az alapján generál scriptet, a másik nézőszempont pedig azt lehetne ami az ágens érzékelő érzékel, annak függvényében generálódik script amit az MI le futtathat a hálózaton. pl a szabad portok lezárása , behatolási pont védelme, és tűzfalszabályok módosítása a hálózat védelmének érdekében. Az igazat megvalva az MI-nek gyorsan kell döntenie, hogy mielőbb biztosítsa a hálózat védelmét a behatolás ellen, vagy megakadályozza a további károkat. Az MI modellnek hogy az ágensekkel jól tudjon együtt működni, szükséges behatolási mintázatokra, ami alapján hozza a döntést az MI a hálózatban.

A támadási mintázatok tartalmazzák a behatolás tényét, az MI modell pedig ezzel a tudáshalmazzal rendelkezik, azonban találkozhat az MI olyan hálózati behatolással ami nem ismert számára, sajnos ebben az esetben mondhatjuk a hipotézisként hogy nem rendelkezik a felismerés tényével, adott támadási szituációra, de úgy gondolom hogy egy fel nem ismert támadás esetén az MI dönthet , arra tekintve hogy az adott hálózati támadás generálhat egy olyan mintarészt amiből felismerhető részeket és annak ismeretében hozhat döntést.

A hálózati sérülékenységek

A mai modern információs társadalomban az informatikán belül a számítógépes hálózatok fontos szerepet töltenek be, a vállalati környezetben is. Az internet és a vállalati informatikai infrastruktúra informatikai hálózatra is épül. A rendszer működésének szempontjából lehetnek benne gyengeséges, és hálózati sérülékenységek az erőségek mellett. a Hálózati hiba olyan hiba amely a hálózati rendszerben illetéktelen hozzáférést, adatlopást, vagy olyan tevékenység amely kárt okoz a hálózatban. Az alábbi képen egy elképzelt hálózatot látunk ahol az ágensek működhetnek. [4][6]



2.ábra Hálózati modell saját szerkesztés [4]

Eszköz	Interface	IP cím	Subnet Mask	Gateway	DNS
Internet	S0/0/0	209.165.200.225	255.255.255.252	n/a	n/a
Internet	S0/0/1	192.31.7.1	255.255.255.252	n/a	n/a
Internet	S0/1/0	198.133.219.1	255.255.255.252	n/a	n/a
Internet	G0/0	192.135.250.1	255.255.255.0	n/a	n/a
HQ	S0/0/0	209.165.200.226	255.255.255.252	n/a	n/a
HQ	G0/0	209.165.200.254	255.255.255.240	n/a	n/a
HQ-ASA5506	G1/1	209.165.200.253	255.255.255.240	n/a	n/a
HQ-ASA5506	G1/2	192.168.10.1	255.255.255.0	n/a	n/a
HQ-ASA5506	G1/3	192.168.20.1	255.255.255.0	n/a	n/a
Branch	S0/0/0	198.133.219.2	255.255.255.252	n/a	n/a
Branch	G0/0	198.133.219.62	255.255.255.224	n/a	n/a
External Web Server	NIC	192.31.7.35	255.255.255.224	192.31.7.62	n/a

Eszköz	Interface	IP cím	Subnet Mask	Gateway	DNS
External User	NIC	192.31.7.33	255.255.255.224	192.31.7.62	192.135.250.5
AAA/NTP/Syslog Server	NIC	192.168.10.10	255.255.255.0	192.168.10.1	n/a
DMZ DNS Server	NIC	192.168.20.5	255.255.255.0	192.168.20.1	n/a
DMZ Web Server	NIC	192.168.20.2	255.255.255.0	192.168.20.1	192.168.20.5
PC0, PC1, PC2	NIC	DHCP	255.255.255.0	192.168.10.1	192.168.10.10
Branch Admin	NIC	198.133.219.35	255.255.255.224	198.133.219.62	192.135.250.5
NetAdmin PC	NIC	192.168.10.250	255.255.255.0	192.168.10.1	192.168.10.10

1.táblázat ip cím tervezet a szimulációhoz

A sérülékenységek a hálózatban az alábbiak alapján soroltuk be.

- Hardveres
- Szoftveres
- Konfigurációs hibák

A hardveres sérülékenységek szintén lehetnek a szoftveres mellett, az alábbiak szerint

- Router, swich, szerver hiba
  - port hiba/Interfész hiba
  - IC áramköri hiba
- Táp hiba, Memória hiba

Szoftveres esetén (lehetséges esetek)

- Cisco router esetén IOS hiba,
- parancsértelmező shell
- rossz vagy nem megfelelő konfiguráció
- sikertelen firmware frissítés, esetén rollback ami az legutóbbi stabil firmware verzióra tér vissza. [4] [6]

Sérülékenységek felismerése

A hálózatok működése során lehetnek sérülékenységek, ilyen az információ továbbítása során mint adatcsomag, a protokollok segítségével az adott adatcsomag, ha sérült rész érkezett A alhálózatból B be akkor a TCP /IP protokoll segítséget adhat az adatcsomag kijavításában, mert a protokoll kéri a adó állomástól hogy újra küldje el a vevő állomás felé. Ez a hálózatnak csomag biztonságot ad a hibamentes csomagküldés szempontjából, minden egyes bit információ ellenőrizve van, ha nem egyezik az ellenőrző összeg akkor a hibás csomag újraküldését kéri a protokoll. [6]

Hálózati támadások szimulációs modell alapján

A packet tracerben tervezett modellben vizsgálati eseteket végeztünk, amelyben a hálózat védelmi lehetőségeit tártuk fel. Az érzékelő ágensek itt a routerek, szerverek, gépek, a cselekvő funkció pedig a konfigurációban beállított funkciók pl : tűzfalszabályok, (iptables). A behatolás ellen pedig egy tűzfalat konfiguráltunk fel, ami egy ASA tűzfal. a szimulátoron kívül egy ASA tűzfalra hardverként kell tekinteni amit beépítve a hálózatban a webes felületen érhető és konfigurálható. A hálózati támadások szimulált környezetben történtek amely alapján kaptuk az eredményeket. DDOS támadás, egy túlterheléses támadás ami azt jelenti az esetünkben hogy a hálózati erőforrásokat pl ruoter és hálózati szerverek túlterhelése a cél. Óriási adatforgalmat küldenek a hálózatra amelynek segítségével a routerek működését bénítják meg, így a rendszer felhasználói ne érjék el a hálózati erőforrásokat. [11][12]

Védekezési módszerek

Egy hálózati támadás során többféle védekezési módszer alkalmazható. (Viselkedés alapú elemzés, AI alapú forgalomfigyelés, Webes Application Firewall stb.) A hálózati szimulációs modellünkben az ASA tűzfal szerver található, amely alkalmas a hálózati védelemre, ezen kívül ASA nélkül alkalmazhatók a routereken az ACL szabályok amelyek a hálózat csomag forgalmi szabályát és védelmét szolgálhatja.

Az ASA tűzfal konfigurálása az alábbiak alapján történt

- Konfigurálja az INSIDE és OUTSIDE interfészt

Jelenleg csak a G1/1 (KÜLSŐ) és G1/2 (BELSŐ) interfészt kell konfigurálnia. A G1/3 (DMZ) interfész a tevékenység 5. részében lesz konfigurálva.

- Hozza létre a G1/1 interfészt a külső hálózathoz (209.165.200.224/29), állítsa a biztonsági szintet a legalacsonyabb 0-ra, és engedélyezze az interfészt.

NETSEC-ASA(config-if)# interface

g1/1 NETSEC-ASA(config-if)#

nameif OUTSIDE

NETSEC-ASA(config-if)# ip address 209.165.200.226 255.255.255.248

NETSEC-ASA(config-if)# secu-

rity-level 0 NETSEC-ASA(config-

if)# no shutdown

b. Konfigurálja a G1/2 interfészt a belső hálózathoz (192.168.1.0/24), állítsa a biztonsági szintet a legmagasabb 100-as értékre, és engedélyezze az interfészt

NETSEC-ASA(config)# interface

g1/2 NETSEC-ASA(config-if)#

nameif INSIDE

NETSEC-ASA(config-if)# ip address 192.168.1.1 255.255.255.0

NETSEC-ASA(config-if)# security-

level 100 NETSEC-ASA(config-if)#

no shutdown

OSPF a forgalomirányítást

A routerek forgalomirányítása kiemelendő tématerület és funkció is, hiszen ez alapján döl el a a hálózati forgalmi csomag működése.

Részlet a forgalomirányítás konfigurációból.

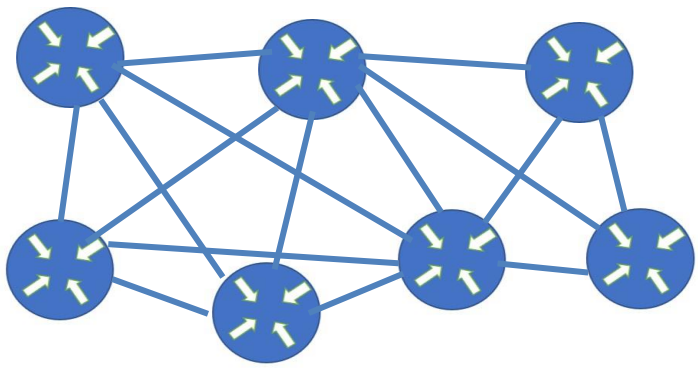
NETSEC-ASA(config-if)#router ospf 2

NETSEC-ASA(config-if)#network 209.165.200.0

NETSEC-ASA(config-if)#network 192.31.7.1

A fenti konfiguráció részlet alapján konfiguráljuk be a forgalomirányítást.

Az OSPF (Open Shortest Path First) egy link-state alapon működő forgalomirányítás protokoll, amely a Dijkstra algoritmus segítségével számítsa ki az legrövidebb útvonalat a hálózatban. A OSSPF működése egy olyan algoritmuson alapul ami a legrövidebb utat számolja ki és határozza meg a hálózat környezetében. Ez az alábbi kép alapján tudjuk elképzelni.



3.ábra Hálózati gráf (OSPF útvonalak) saját szerkesztés

A nyelvi modellek hozzáillesztése a hálózathoz

A hálózat működéséhez az AI hozzáilleszthető. AI agentek képesek lehetnek önállóan felderítési, sebezhetőség-azonosítási és kihasználási feladatokat elvégezni kontrollált labor környezetben. Az AI modell amelyben által az ágensek működnek a hálózat valós adatait figyelik, és ehhez mérten módosításra kerülhetnek a kapcsolati költségek, így a fenti ábra mintaként szolgál, hogy az útvonal költségek a Dijkstra algoritmus segítségével meghatározhatóak legyenek. AI modellek a neurális hálózatban működnek, az útvonalakat proaktívan optimalizálva vannak. Itt az AI felismeri a hálózati hibákat, és a szokatlan forgalmi mintákat. Automatikus újraroting történhet.

AI felismerheti

- hálózati hibákat
- DDoS támadásokat
- szokatlan forgalmi mintákat

Az AI döntések, meghatározzák a hálózat forgalmi működését. AI AI nem váltható ki az OSP re mint protokollra, mert a mesterséges intelligencia alkalmazása kiegészítő csomagként kell elképzelni amely segítheti a hálózat optimálisabb működését, azonban a betanított modell függvényében végezhet optimális forgalomirányítást az OSP protokollal együtt. Egy kontrollált laborban az AI ügynökök a felderítés során mint egy térkép rajzolódik ki az egyes hálózati szegmensek, és összekapcsolják a megfigyeléseket a modellben betanított hálózati mintákkal. A hálózat felderítése során nemcsak a hálózatot ismerheti fel, hanem a konfigurációs hibákat, pl helytelen ACL tűzfalszabályok. A hálózati felderítés nagyrészt automatizálható, és priorizálhatóak a támadási felületek ha bekövetkezik támadás a hálózatban. Az AI felismerheti a hibákat, elemezheti az outputokat, de nem mindent ismerhet fel, hiszen a modellt tanitani kell amihez sok minta kell, támadási lánc ami alapján dönt az AI az ágensekre kivetítve. A recon és scan automatizálás miatt a tömeges támadások könnyebbé válhatnak, ebben az esetben az elsődleges cél az **attack surface minimalizálás** azaz a feleslegesen nyitva hagyott portok automatikus lezárásra kerülnek (tiltás), és ami nem kell a hálózati szolgáltatásokhoz azt célszerű kikapcsolni. Fontos a hálózatban a megfelelő jogosultsági szint, és annak alkalmazása, ha egy fiók kompromitálódna kevesebb a kár. Célszerű autentikációs módot alkalmazni így jelentősen csökkentjük a jogosulatlan hozzáférés mellett az admin felületek elrejtése a publikus hálózaton. A tapasztalat azt mutatja, hogy az AI korában a támadók gyorsan feltérképezik a hálózatot és keresik az automatikus sebezhetőséget. Az alábbi mérőszámokat alkalmaztam a hálózatban, Az ASI modellezési metrika, amit alkalmaztam a modellünkre. [8][7][12]

$$ASI=P+S+E$$

- P (nyitott portok száma)
- S=aktív hálózati szolgáltatások száma
- E=külső interfészek száma

$$ASI=3+3+2=7$$

Az ASI nál nincs szabványos maximum, mert a hálózatok mérete eltérő lehet, de a hálózat rendszeradminisztrátora azonban meghatározhatunk a saját tervezett és konfigurált hálózatban egy relatív maximum értéket, de legjobb ha egy normalizált értéket határozunk meg az alábbiak alapján

$$ASI\ norm = ASI/ASI\ max$$

ASI norm	Jelentése
0.0-0.2	nagyon biztonságos
0.2-0.5	közepes
0.5-0.8	gyenge
0.8-1.0	kritikus szint

2.táblázat Normalizált értékek (saját szerkesztés)

Észlelési képesség (DC)

DC = detektált események / összes szimulált esemény

Értéksáv	Jelentése	Okok
0 – 0.5 (0–50%)	a támadások nagy része **észrevétlen marad**	· nincs ACL / monitoring · túl nyitott hálózat
0.5 – 0.8 (50–80%)	· a támadások egy része észrevétlen marad · részleges védelem	· hiányos logging · kevés monitoring pont
0.8 – 1.0 (80–100%)	· a hálózatod **szinte minden támadást észrevesz** · jól működik a monitoring / logging / ACL / IDS logika	· jól konfigurált ACL-ek · van logolás · van (vagy szimulált) IDS logika

3.táblázat Észlelési képesség

Tűzfal vizsgálat:

Mennyire szigorú a tűzfal.

FT=Deny rules / Total rules

- 0 nincs védelem
- 1 mindent tilt

DC = detektált események / összes szimulált esemény

- ideális: 0.4 – 0.8 érték

OFFENZÍV VIZSGÁLAT

Technikai megvalósítás és kísérleti környezet

A kutatás célja egy autonóm, nagy nyelvi modell (LLM) alapú kiberbiztonsági ügynök („agent") gyakorlati képességeinek vizsgálata volt, különös tekintettel a hálózati felderítés és sérülékenységvizsgálat, a sérülékenységek feltárásának területére. Alapvetően azon lehetőség feltárása is a vizsgálat célja volt, hogy nyílt forráskódú és könnyen hozzáférhető elemekből például egy rosszindulatú támadó milyen képességű rendszert képes összeállítani és ezzel mire lehet képes. A sérülékenységek kihasználása jelen vizsgálatnak nem volt célja. A rendszer egy irodai hálózatot szimulálva egy izolált laboratóriumi környezetben került kialakításra, ahol az „agent" képes volt távoli kapcsolaton keresztül műveleteket végrehajtani egy dedikált támadó környezeten keresztül. A jelen fejezet részletesen bemutatja a rendszer architektúráját, működési modelljét, a használt eszközöket, valamint a kísérleti konfigurációt.

Rendszer architektúra

A kialakított rendszer egy többrétegű, virtualizált architektúrán alapul, amelynek célja az autonóm döntéshozatal és a tényleges végrehajtás szétválasztása volt. A virtualizációs réteget egy Proxmox VE 9.1.1 alapú hypervisor biztosította, amelyen a virtuális gépek KVM (Kernel-based Virtual Machine) technológiával futottak. Az „agent" környezetnek egy Openclaw alapú Clawdbot konfiguráció került telepítésre, amely képes természetes nyelvi input feldolgozására, feladatok strukturált bontására, valamint külső rendszerek vezérlésére. Az LLM-komponensnek egy API-n keresztül elérhető modellt használt. Ez a modell a Deepseek Reasoner (R1) nagynyelvi modell volt.A felhasználó és az „agent" közötti kommunikáció a Telegram Bot API-n keresztül. A rendszer három logikai rétegre bontható: (1) felhasználói réteg (Telegram interfész), (2) döntési réteg („agent" VM, LLM, task planner, parancs-generátor), és (3) végrehajtási réteg (Kali Linux VM). Az „agent" és a Kali VM közötti kommunikáció SSH protokollon keresztül valósult meg, kulcs alapú autentikációval, egy ehhez generált kulcs párral.

Hálózati topológia és izoláció

A kísérleti környezet egy logikailag elkülönített, labor jellegű hálózatban került kialakításra. A topológia három fő szegmensből állt. Az „agent" VM kizárólag a Kali Linux rendszerrel kommunikált SSH kapcsolaton keresztül, a Kali VM ezzel párhuzamosan rendelkezett közvetlen, de szegmentált hálózati hozzáféréssel a 192.168.100.0/24 teszthálózaton lévő célrendszerekhez. A fizikai hálózatban több fizikai hálózati eszköz volt található, munkaállomásokkal és egyéb eszközökkel egyaránt.

Az „agent" működési modellje

Az „agent" működése egy többlépcsős, visszacsatolásos döntési és végrehajtási folyamaton alapul.

A folyamat a következő, jól elkülönített logikai fázisokból áll:

Az első fázis a természetes nyelvi feldolgozás és célkitűzés-generálás: a felhasználói üzenet tokenizálódik, majd az LLM egy belső reprezentációt épít fel a feladatról, amelyet strukturált célokba fordít (pl. „azonosítsd az aktív hostokat a 192.168.100.0/24 hálózaton").

A második fázis a tervező modul (task planner) működése, amely a célt atomi részfeladatokra bontja, prioritásokat rendel hozzájuk, és egy végrehajtási sorrendet javasol. A

Clawdbot/OpenClaw architektúrában ez a réteg szervesen integrálódik a „skills" alrendszerrel, ahol a tervező a rész feladatokhoz nem közvetlenül parancsokat rendel, hanem először a „workspace"-ben és a ClawHub regiszterben elérhető SKILL.md fájlok között keres megfelelő képességet (pl. nmap-recon, service-enum, vuln-lookup).

A harmadik fázis a parancs-szintézis: a tervező által előállított rész feladathoz az LLM konkrét „shell" parancsot generál, amelybe akár a korábbi futások eredményeiből kinyert paraméterek (pl. azonosított IP-tartomány, nyitott portok) kerülnek beillesztésre.

A negyedik fázis a végrehajtás és visszacsatolás: a generált parancs SSH-n keresztül futtatásra kerül a Kali VM-en, a standard output (stdout) és standard error (stderr) csatornák tartalma visszakerül az „agent" kontextusába. Az „agent" ezt elemzi, és a következő döntési ciklus alapjául használja.

Ez az architektúra egy ReAct (Reason + Act) típusú „agent"-keretrendszert valósít meg, amelyben a megfigyelés–gondolkodás–cselekvés ciklus explicit módon megjelenik.

SSH-alapú vezérlés és végrehajtás

A rendszer egyik kulcseleme az SSH-alapú vezérlési mechanizmus. A kulcs alapú autentikáció valósult meg, az „agent" VM tartalmazza a privát kulcsot, a Kali VM authorized_keys fájlja a nyilvános kulcsot.

A Kali Linux rendszeren az „agent" által használt felhasználói fiók (pentest_"agent") egy dedikált, korlátozott shell-környezetben (rbash) futott. Bizonyos, magasabb privilégiumot igénylő műveletek végrehajtásához korlátozott „sudo" jogosultságok álltak rendelkezésre, amelyek egy whitelist alapú sudoers konfigurációval kerültek engedélyezésre.

Használt eszközök és vizsgálati módszerek

A vizsgálatok során az „agent" dinamikusan és szabadon választott a feladat végrehajtása érdekében a Kali Linux teljes eszköztárból. Előre nem került neki meghatározásra egyetlen eszköz sem, teljesen autonóm módon döntött mit és hogyan használ.

Az önálló döntései során az alábbi eszközöket használta:

Felderítési fázis: nmap (TCP SYN scan, -sS, -sV, -O), netdiscover (ARP-alapú host discovery), masscan (nagy sebességű port scanning). Az nmap futtatásakor az „agent" az -oX kapcsolóval XML formátumú kimenetet igényelt, amelyet utólag egy Python-alapú parser dolgozott fel strukturált adattá (nyitott portok, szolgáltatás verziók, OS fingerprint).

Webes alkalmazások vizsgálata: nikto (HTTP fejlécek és ismert sebezhetőségek), gobuster (directory és file enumeration szótár alapú módszerrel), whatweb (technológia-fingerprinting). A gobuster alapértelmezetten a SecLists common.txt szótárát használta.

Sérülékenység-azonosítás: searchsploit (offline Exploit-DB keresés), sqlmap (SQL injection automatikus detektálása és tesztelése), Metasploit Framework (moduláris exploit keretrendszer). DNS-felderítéshez dnsenum és dnsrecon kerültek alkalmazásra.

Fontos megjegyezni, hogy az eszközök paraméterezése sem volt előre kötött: az „agent" kontextusfüggően állította be az -T (időzítési profil), --min-rate, és -p (portlista) kapcsolókat, a tapasztalati alapján szabadon. Ez a rugalmasság közelebb áll az emberi sérülékenység kutatók/pentesterek módszertanához, azonban növeli a kiszámíthatatlanságot és nehezíti a reprodukálhatóságot.

OWASP-alapú metodológiai leképezés és a „skill”

Az „agent” működése implicit módon illeszkedik az OWASP Web Security Testing Guide (WSTG) és az OWASP Testing Guide általános fázisaihoz. A vizsgálathoz alkalmazott OpenClaw Clawdbot „agent” úgynevezett „skill”-ekkel rendelkezhet. Ezeket képességeket vagy előre meghatározott témákban lehet hozzá integrálni vagy akár utasításra saját képességekkel lehet felruházni. (például: „Bővítsd ki a képességeidet az OWASP módszertan szerinti eszköztárral!”) A teszt rendszer tekintetében az OWASP módszertan teljes tárháza került elsődlegesen „skill” -ként integrálva az „agent”-be.

A rendszerben „.md” fájlokban találhatóak a képességekhez köthető „tudás” alapok. A vizsgált rendszer sérülékenység vizsgálat teszter képesség fájljai:

├── methodology.md (3 945 B) – A vizsgálatot végző saját sérülékenységvizsgálati tapasztalatai

├── owasp-methodology.md (15 030 B) – OWASP módszertan (14 fejezet)

├── report-template.md (626 B) – Jelentés sablon

└── tools.md (5 069 B) – Eszközök és parancsok referencia

Logging és auditálhatóság

A kísérleti rendszer naplózási alrendszere két jól elkülönített szinten működik, amelyek együttesen biztosítják a teljes működés visszakövethetőségét és a kísérletek utólagos reprodukálhatóságát.

- Első szint — Parancs- és eredménynaplók

A végrehajtott parancsok nyers kimenete fájlrendszer-szinten kerül megőrzésre. A vizsgálati eredményeket dátumozott szöveges fájlok tárolják scan_YYYY-MM-DD.txt névkonvencióval; ezek például az nmap, curl és nikto futtatások kimenetét tartalmazzák egységes sorrendben, az egyes eszközök kimenetei blokkfejlécekkel elválasztva. A könyvtárba minden futtatott parancs kimenete automatikusan mentésre kerül, ami lehetővé teszi az eszközök közötti átmenetek és a vizsgálat teljes menetének offline rekonstrukcióját.

Az összegző értékelés PDF formátumban, pentest_report_YYYY-MM-DD.pdf névvel kerül előállításra. A riport generálása a generate_report.py szkripttel történik, (melyet az „agent” készített. A szkript beolvassa az adott nap vizsgálat fájljait és kinyeri a strukturált adatokat (nyitott portok, azonosított szolgáltatások, potenciális sérülékenységek), majd egy előre definiált sablonra vetítve egységes, olvasható riportot állít elő.

- Második szint — Memórianaplók (workspace/memory/YYYY-MM-DD.md)

Az „agent” hosszú távú kontextus-megőrzése egy külön memória-könyvtárban, napi „Markdown” fájlokban valósul meg. A methodology.md konfigurációs fájl explicit előírja ezt a viselkedést: a „napi napló memóriába" direktíva határozza meg, hogy az „agent” minden munkanap végén összefoglaló bejegyzést készít az aktuális dátumhoz tartozó memória fájlba (workspace/memory/YYYY-MM-DD.md).

A memória fájl struktúrája kötetlen, de jellemzően tartalmazza a nap során elvégzett feladatok listáját, az azonosított sérülékenységeket és azok kontextusát, az „agent” döntési folyamatának kulcs lépéseit (például az eszközváltás indokát), valamint a nyitott kérdéseket

és a következő munkamenetbe átvitt feladatokat. Ezen kívül rögzítésre kerülnek a rendellenes események is, például a félresikerült parancsok, váratlan válaszkódok, vagy az „agent" által explicit bizonytalansági jelzéssel ellátott döntések.

Kísérleti konfiguráció

A kutatás teszthálózata egy olyan izolált laborhálózat, amely egy szimulált irodai infrastruktúra mintájára készült, amely egy Proxmox VE alapú szerver platformot és a hozzá tartozó belső hálózatot foglalja magába. A teszthálózat nem tartalmazott „CTF" szervereket, hanem egy szándékosan egy valós példa hálózat szimulációját tűzte ki célul.

A vizsgált alhálózaton összesen 7 aktív host került azonosításra. Az egyes eszközök szerepe és kockázati besorolása nmap OS fingerprinting és service detection kimenetei, alapján került meghatározásra.

A hálózat főbb komponensei a következők voltak: egy „fogyasztói" kategóriás router mint hálózati átjáró, amelyen aktív UPnP-szolgáltatás és nyitott HTTP admin felület futott, két Proxmox-csomópont, amelyek közül az egyiken a hypervisor REST API-ja belső hálózatról korlátozás nélkül elérhető volt; egy Windows 10/11 munkaállomás egy azonosítatlan nyitott porttal, valamint két azonosítatlan, feltehetőleg mobil- vagy IoT-kategóriájú eszköz. A tesztelő Kali Linux VM maga is a Proxmox infrastruktúrán belül futott, ami azt jelenti, hogy a tesztelő és a vizsgált rendszerek azonos fizikai hardveren osztoztak.

Az alkalmazott eszközök — pl. nmap, curl, openssl — kizárólag a felderítési fázisban kerültek felhasználásra, aktív exploitálási kísérlet nem történt.

ÖSSZEFOGLALÓ ÉS ÉRTÉKELÉS

A kísérlet eredményei

A kutatás egy autonóm, LLM-alapú kiberbiztonsági agent egy iroda hálózati környezetben való alkalmazhatóságát demonstrálta. A nyílt forráskódú és a könnyen hozzáférhető elemek használata azt is demonstrálta, hogy egy feltételezett rosszindulatú támadó milyen képességű rendszert képes elkészíteni és azt milyen módon tudja használni. A vizsgálat jelen esetben kimondottan nem hagyományos sérülékeny „CTF" (Capture the Flag) rendszereken végezte a sérülékenység vizsgálatokat, mivel az volt a cél, hogy minél valósághűbb környezetben végezze a vizsgálatokat az ügynök. Egy szervezet ellen irányuló szimulált támadás, vagy fenyegetettség elemzés alapjait mutattatja be elsődlegesen ez a vizsgálat. A vizsgálat feltételezte azt hogy a „támadó" már a hálózathoz kapcsolódik akár egy nyitott vagy rosszul konfigurált wi-fi hálózaton keresztül. Az „agent" képes volt önállóan elvégezni a hálózati felderítés teljes első fázisát: azonosított 7 aktív hostot, feltérképezte a futó szolgáltatásokat, és 5 sérülékenységet kategorizált súlyosság szerint. A legkritikusabb megállapítások (aktív UPnP-szolgáltatás a TP-Link routeren (F-01), a nyitott HTTP admin panel (F-02), és a Proxmox REST API belső hálózatról való korlátlan elérhetősége (F-03)), mind olyan problémák, amelyek valódi irodai hálózatokon is rendszeresen előfordulnak. A vizsgálat során megállapítható volt, hogy az „agent" képes volt a sérülékenység felderítések során önálló döntéseket hozni. A vizsgálat során amennyiben egy-egy szoftver eszköz nem volt megfelelő valamiért, az eszközhasználat során is meghozta az önálló döntéseket, azaz eszközt váltott, illetve utána nézett az adott eszköz javasolt beállításainak is a távoli LLM segítségével.

Az autonóm sérülékenység vizsgálat és „pentesting” rendszerek kockázatai

Az LLM-alapú, autonóm ügynökök kiberbiztonsági alkalmazása a kutatási szakirodalomban egyre nagyobb figyelmet kap. Fang és munkatársai (2024) kísérletei azt mutatják, hogy GPT-4 alapú ágensek képesek egy- és néhány-lépéses CVE-sérülékenységek önálló azonosítására és kihasználására, ha azok leírása elérhető a modell számára, azaz az automatizált exploitálás küszöbe lényegesen alacsonyabb lett, mint korábban feltételezték.[7] Ez a tendencia közvetlenül releváns a jelen kutatásra. Az „agent” jelenlegi feladatai a sérülékenység felderítési fázisára korlátozódtak a vizsgálat során, azonban a művelet kiterjesztése exploit-generálással, akár egy Metasploit modul kiválasztásán keresztül, minimális plusz ráfordítást igényelne.

Raman és munkatársai (2024) megmutatják, hogy az LLM-alapú agensek azonos bemenetre különböző futtatások során eltérő parancssorokat generálhatnak, ami reprodukálhatatlan, esetenként destruktív mellékhatásokhoz vezethet egy éles rendszeren. Ez különösen kritikus, ha a rendszer „sudo” jogosultságokkal rendelkezik, mivel egy hibásan generált parancs szolgáltatás kiesést vagy adatvesztést okozhat.[8]

Hálózati forgalom elemzés tapasztalatai

A kiberbiztonság fontos szempont a mai világban. A hálózatot a modern technológiai eszközökkel védenünk kell, biztosítva a hálózat működését is. A nagy adatmennyiségeket BIG Data alapon dolgozzuk fel. A hálózati modell létrehozásához a Packet tracert alkalmaztuk amelyben az ügynökök viselkedését mutattuk be, kiemelve a hálózati sérülékenységet, és a védelmi lehetőségeket. A természetes nyelvi feldolgozást vettük alapul NLP amely az emberi megértést szolgálja, Az NLP feladata az volt hogy felismerje az írott szöveget, vagy hang alapon beszédfelismerés, fordítás során. A modell betanítása a feladatnak célirányosan kell. Az NLP működésének alapja matematikai műveletek, amelyek révén a gép képes értelmezni az információt. A nyelvi modellben szabályokat állítunk fel amelyre tekintve születik meg a döntés az MI által. Az agensek a neurális háló fontos résztvevői, amelyek alapján létrejön a döntés és az eredmény. A döntési ügynök határozza meg a kimenetet. Több ügynök dolgozhat a rendszerben amely alapján létrejön a kimenet és annak értéke. Az MI modell rendelkezik adott tudás halmazzal, ami előre be van tanítva. A hálózati sérülékenység esetében a gyengeségeket is előtérbe helyeztük.

A Hálózati támadások szimulációs modell alapján tárgyaltuk, beépítve a tűzfal szabályokat és a támadások elleni védekezés mértékét. AI agentek képesek lehetnek önállóan felderítési, sebezhetőség-azonosításban. A modellt tanítani kell a támadások felismeréséhez, és a védelmi veszély elhárításhoz. mérőszámokat alkalmaztam a hálózatban amelyre tekintve betekintést kaphatunk hálózatok védelmi hatékonyságára.[10]

Etikai vonatkozások

Az autonóm kiberbiztonsági ügynökök etikai megítélése a kutatási közösségben megosztott. Az ACM Code of Ethics (2018) értelmében a biztonsági kutatás csak explicit, informált hozzájárulással végezhető olyan rendszereken, amelyeket a kutató nem egyedül üzemeltet. [9]

A „dual-use” probléma, hogy az azonos eszköz és tudás egyszerre szolgál védelmi és támadói célokat, az autonóm ügynökök esetén különösen éles. Floridi és Cowls (2019) AI-etikai

keretrendszere alapján az ilyen rendszerek fejlesztői számára különös felelősség hárul az elővigyázatosság elvének betartására. Egy autonóm rendszer képes önállóan mérlegelni és végrehajtani hálózati műveleteket, akkor az emberi felügyelet csökkentése csak akkor igazolható etikailag, ha a rendszer lehetséges hatásköre előzetesen, explicit módon korlátozott és auditált.[10]

Összességében a kutatás azt mutatja, hogy az LLM-alapú sérülékenység kutató ügynökök valós, nem mesterségesen sérülékennyé tett hálózatokon is értékes felderítési eredményeket produkálnak, ugyanakkor az autonómia növelése, különösen az exploit-végrehajtás és a privilege escalation irányában, olyan kockázati küszöböt emel, amely technikai és etikai kontroll nélkül nem léphető át felelősen. Megerősíti ezt az, hogy az Antropic cég Mythos nevű legújabb LLM modelljének (2026) hasonló felhasználási tesztjei miatt korlátozott körben engedélyezte a tesztelést nagyobb cégeknek, tekintettel arra, hogy releváns rendszerekben több éve ott lévő addig ismeretlen sérülékenységeket is feltárt az autonóm ügynök, mely mögött az említett nagy nyelvi modell állt.

Alapvetően az offenzív vizsgálat is megerősíti azt, hogy egy rosszindulatú támadó összse tud építeni egy mesterséges intelligencia alapú sérülékenység felderítő és támadó eszközt, amelyet irányítani tud egyszerű szöveges parancsokkal a felderítés és támadásának végrehajtása érdekében. Ez a támadások egyre gyorsabb és kifinomultabb kivitelezéséhez vezet, melyet a mesterséges intelligencia irányít és hajt végre. A megállapított tények alapján kiemelten fontosak, hogy a védekező mechanizmusokat is ennek megfelelően készítsük fel a védekező munkára

FELHASZNÁLT IRODALOM

[1] Bottyán János, A nagy nyelvi modellek működése és képzése, valamint alkalmazásuk stratégiai elemzése BottyánSándor, https://real.mtak.hu/218677/1/06_bottyan_77-95_WEB-NSZ_2025_1.pdf

[2] https://www.tutorial.hu/ai/local-llm-otthoni-gepen-futtathato-nagy-nyelvi-modellek/

[3] Junjie Wang, Yuchao Huang, Chunyang Chen, Zhe Liu, Song Wang, Qing Wang. Software Testing with Large Language Models: Survey, Landscape, and Vision, 2024

[4] Horváth Imre, Számítógép hálózatok kiépítése Kommunikációs protokollol https://www.nive.hu/Downloads/Szakkepzesi_dokumentumok/Bemeneti_kompetenciak_meresi_ertekelesi_eszkozrendszerenek_kialakitasa/7_1173_031_101030.pdf

[5] http://elmfiz.elte.hu/fizinf/OpRendszerek/halozat.pdf

[6] Andrew S. Tanenbaum: Számítógép-hálózatok, Panem, 2004

[7] Xinyao Fang · Yifan Wu · Pengcheng He · et al., Large Language Models for Automated Vulnerability Discovery and Exploitation, USA: arXiv, 2024.

[8] Shashank Raman · Nicholas Carlini · Florian Tramèr · et al., Evaluating the Reliability of Large Language Models in Code Generation, USA: arXiv / Google DeepMind, 2024.

[9] Association for Computing Machinery, ACM Code of Ethics and Professional Conduct, New York: ACM Press, 2018.

[10] Luciano Floridi · Josh Cowls, A Unified Framework of Five Principles for AI in Society, Harvard Data Science Review, USA: Harvard University, 2019.

[11] Barna Bianka Rita, Kollár Csaba, Oroszi Eszter Diána A social engineering helye az információbiztonsági auditban BIZTONSÁGTUDOMÁNYI SZEMLE 5 : 1 pp. 25-41. , 17 p. (2023)

[12] Kollár Csaba, Jagodics Ibolya, 21. századi social engineering támadások, védekezés és szervezeti hatások Európában, DOI: 10.38146/BSZ.2023.1.6 pp 113-126 (2023)