ÓBUDAI EGYETEM
ÓBUDA UNIVERSITY

NEUMANN JÁNOS
INFORMATIKAI KAR

SZAKDOLGOZAT

OE-NIK	Hallgató neve:	**Palik Bence Dávid**
2022	Hallgató törzskönyvi száma:	**T/006342/FI12904/N**

Óbudai Egyetem
Neumann János Informatikai Kar
Kiberfizikai Rendszerek Intézet

SZAKDOLGOZAT FELADATLAP

Hallgató neve: **Palik Bence Dávid**
Törzskönyvi száma: T/006342/FI12904/N
Neptun kódja: [redacted]

A dolgozat címe:

Honeypot adatok tárolása és vizualizációja
Storage and visualisation of honeypot data

Intézményi konzulens: Dr. Fleiner Rita
Külső konzulens: Vránics Dávid Ferenc

Beadási határidő: 2022. május 15.

A záróvizsga tárgyai: Számítógép architektúrák
Big Data és üzleti intelligencia specializáció

A feladat

A számítógéphálózatok irányába végzett automatizált behatolási kísérletek jelentős adatmennyiséget generálhatnak, amelyek manuális elemzése időigényes lehet. A szakdolgozat kapcsán a hallgató feladata egy olyan rendszer megtervezése és megvalósítása, amely lehetővé teszi egy tesztkörnyezetben üzemelő honeypot rendszerben keletkezett alacsony szintű adatok strukturált tárolását és megjelenítését. A feladat végrehajtása során kutassa fel és vizsgálja meg az elérhető honeypot eszközöket. Ismertesse és hasonlítsa össze a lehetséges adattárolási technológiákat. Valósítson meg egy olyan webalkalmazást, amely képes a gyűjtött adatok vizuális megjelenítésére.

A dolgozatnak tartalmaznia kell:

- a feladat pontos ismertetését,
- a követelmények specifikációját,
- az elérhető honeypot eszközök bemutatását,
- a lehetséges adattárolási technológiák összehasonlítását,
- a választott szoftverkomponensek indoklását,
- a tervezés és megvalósítás részletes leírását,
- a továbbfejlesztési lehetőségek számba vételét.

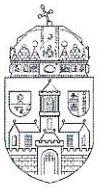
Fleiner R.
.....
Dr. Fleiner Rita
intézetigazgató

A szakdolgozat elévülésének határideje: **2024. május 15.**
(OE TVSz 55.§ szerint)

A dolgozatot beadásra alkalmasnak tartom:

.....
külső konzulens

Fleiner R.
.....
intézményi konzulens



ÓBUDAI EGYETEM
ÓBUDA UNIVERSITY

Neumann János Informatikai Kar

HALLGATÓI NYILATKOZAT

Alulírott hallgató kijelentem, hogy a szakdolgozat saját munkám eredménye, a felhasznált szakirodalmat és eszközöket azonosíthatóan közöltem. Az elkészült szakdolgozatomban található eredményeket az egyetem és a feladatot kiíró intézmény saját céljára térítés nélkül felhasználhatja.

Budapest, 2022. május 1.

Dalik Jenő

.....

hallgató aláírása

ÓBUDAI EGYETEM
ÓBUDA UNIVERSITY

Neumann János Informatikai Kar

KONZULTÁCIÓS NAPLÓ

Hallgató neve: Palik Bence Dávid

Neptun Kód: [redacted]

Tagozat: Mérnökinformatikus BSc 2018 nappali

Telefon: [redacted]

Levelezési cím (pl.: lakcím): [redacted]

Szakdolgozat / Diplomamunka[6] címe magyarul:

Honeypot adatok tárolása és vizualizációja

Szakdolgozat / Diplomamunka[7] címe angolul:

Storage and visualisation of honeypot data

Intézményi konzulens: Fleiner Rita

Külső konzulens: Vránics Dávid Ferenc

Kérjük, hogy az adatokat nyomtatott nagy betűkkel írja!

Alk.	Dátum	Tartalom	Aláírás
1.	2021.09.26.	Szakdolgozat témájának megbeszélése	Fleiner R
2.	2021.10.05.	Feladatlap megbeszélése	Fleiner R
3.	2021.11.29.	Irodalomkutatás megbeszélése	Fleiner R
4.	2021.12.06.	Dokumentumok beadásának megbeszélése	Fleiner R

A Konzultációs naplót összesen 4 alkalommal, az egyes konzultációk alkalmával kell láttamoztatni bármelyik konzulenssel.

A hallgató a Szakdolgozat I. / Szakdolgozat II. (BSc) vagy Diplomamunka 1 / Diplomamunka 2 / Diplomamunka 3 / Diplomamunka 4 [8] tantárgy követelményét teljesítette, beszámolóra / védésre [9]bocsátható.

Fleiner R

.....

Intézményi konzulens

Budapest, 2021.12.10

---

[6] Megfelelő aláhúzandó!

[7] Megfelelő aláhúzandó!

[8] Megfelelő aláhúzandó!

[9] Megfelelő aláhúzandó!

ÓBUDAI EGYETEM
ÓBUDA UNIVERSITY

Neumann János Informatikai Kar

KONZULTÁCIÓS NAPLÓ

Hallgató neve: Palik Bence Dávid

Neptun Kód: [redacted]

Tagozat: Mérnökinformatikus BSc 2018 nappali

Telefon: [redacted]

Levelezési cím (pl.: lakcím):

Szakdolgozat / Diplomamunka[6] címe magyarul:

Honeypot adatok tárolása és vizualizációja

Szakdolgozat / Diplomamunka[7] címe angolul:

Storage and visualisation of honeypot data

Intézményi konzulens: Fleiner Rita

Külső konzulens: Vránics Dávid Ferenc

Kérjük, hogy az adatokat nyomtatott nagy betűkkel írja!

Alk.	Dátum	Tartalom	Aláírás
1.	2022.02.04.	Architektúra áttekintése	
2.	2022.03.11.	Biztonsági kérdések megbeszélése	
3.	2022.04.08.	Továbbfejlesztési lehetőségek áttekintése	
4.	2022.05.05.	Dolgozat beadásának megbeszélése	

A Konzultációs naplót összesen 4 alkalommal, az egyes konzultációk alkalmával kell láttamoztatni bármelyik konzulenssel.

A hallgató a Szakdolgozat I. / Szakdolgozat II. (BSc) vagy Diplomamunka 1 / Diplomamunka 2 / Diplomamunka 3 / Diplomamunka 4 [8] tantárgy követelményét teljesítette, beszámolóra / védésre [9]bocsátható.

.....

Intézményi konzulens

Budapest, 2021.12.10

---

[6] Megfelelő aláhúzandó!

[7] Megfelelő aláhúzandó!

[8] Megfelelő aláhúzandó!

[9] Megfelelő aláhúzandó!

Tartalomjegyzék

1. BEVEZETÉS

Az interneten elérhető kiszolgálók szinte folyamatosan ki vannak téve támadási kísérleteknek. A támadások leggyakoribb célpontja az SSH protokoll alapértelmezett portja (TCP 22). [1] [2] Az SSH elleni egyik legegyszerűbb támadás a jelszó alapú hitelesítés próbálkozáson alapuló támadása, amely során a támadó automatizált eszközökkel, jellemzően valamilyen már meglévő, bejelentkezési adatokat tartalmazó szótárakat (wordlists) felhasználva próbál csatlakozni a szolgáltatáshoz, kipróbálva az összes rendelkezésre álló felhasználónév és jelszó párost, annak érdekében, hogy hozzáférést szerezzen a rendszerhez. Természetesen az egyéb különböző szolgáltatások alapértelmezett portjaira is érkeznek rosszindulatú hálózati csomagok. Az üzemeltetők létrehozhatnak honeypot (csali) rendszereket, amelyeken éles (pl. üzleti folyamatokat támogató) szolgáltatások nem érhetőek el, azokat csupán szimulálják. Ezeket a rendszereket szándékosan hagyják kitéve a lehetséges támadásoknak, hogy a támadásokról, illetve magukról a támadókról adatokat szolgáltassanak. Az automatizált támadások során azonban olyan nagy mennyiségű adat, pl. naplóbejegyzések jöhetnek létre, amelyeket már manuálisan, egyenként értelmezni megterhelő, idő- és tárhelypazarló[1]. Ezekből az adatokból célszerű időszakosan valamilyen tömör, megfelelő szempontok mentén felépített vizuális összesítést létrehozni, amely az üzemeltető részére könnyen értelmezhető képet ad az eddigi támadásokról, azok forrásáról, ami alapján következtethetnek a jövőbeni veszélyekre.

A szakdolgozat során megvalósítandó feladat egy megfelelően kialakított tesztkörnyezetben üzemeltetett, az SSH autentikációs kísérletek, valamint a beérkezett hálózati csomagok megfelelő adatait rögzítő honeypot rendszerből származó adatok tárolását lehetővé tevő megoldás kiválasztása, végül az adatok összegzett vizuális megjelenítését megvalósító alkalmazás megtervezése és fejlesztése. A feladat motivációját egyéni érdeklődés adta: véleményem szerint a begyűjtött adatokat, és a feladat megoldása során szerzett tapasztalatokat a későbbiekben hasznosíthatom hasonló környezetben üzemeltetni tervezett, személyes, vagy üzleti célokat szolgáló, éles szolgáltatásokat megvalósító rendszerek esetén. Dolgozatom struktúrája így a következő:

1. Megvizsgálom a honeypot rendszerek általános elméleti hátterét.
2. Meghatározom a megjelenítést végző alkalmazásra vonatkozó követelményeket.
3. Létrehozom a rendszer üzemeltetéséhez szükséges tesztkörnyezetet.
4. Megvizsgálom a lehetséges honeypot eszközöket és az adattárolás lehetőségeit.
5. Kiválasztom a feladat megoldásához szükséges eszközöket.
6. Megtervezem az adattárolás struktúráját.
7. Megtervezem és implementálom a megjelenítést megvalósító alkalmazást.
8. Ismertetem a jövőbeni továbbfejlesztési lehetőségeket.

[1] A túlzott naplózás miatt nagymennyiségű naplók keletkezése DoS támadásokra is lehetőséget adhat.

2. HONEYPOT RENDSZEREK ELMÉLETI HÁTTERE

2.1. Kapcsolódó fogalmak

2.1.1. Honeypot

A honeypot tágabb értelemben „biztonsági erőforrás, amelynek értéke annak vizsgálatában[2], megtámadásában, illetve kompromittációjában rejlik” [3], illetve „információs rendszer erőforrás, amelynek értéke annak illetéktelen, vagy meg nem engedett használatában rejlik” [4], szűkebb értelemben kb. „szorosan figyelemmel kísért számítástechnikai erőforrás, amelyet vizsgálatnak[2], támadásnak, vagy kompromittációnak szándékozunk kitenni” [5], továbbá „számítógép, illetve számítógép-rendszer, melynek célja kibertámadások lehetséges célpontjainak imitálása.” [6]. Dolgozatom további részében a számítógéphálózatokban elhelyezett olyan csapda jellegű rendszert értek honeypot alatt, amely a támadó számára éles rendszereket, szolgáltatásokat szimulál, azonban a valódi éles rendszerektől, szolgáltatásoktól függetlenül, amelyek működését nem befolyásolja. [7]

2.1.2. Honeytoken

A honeytoken fogalom is egy, a fenti átfogóbb definícióknak megfelelő erőforrásra utal, pl. egy fájl, valamilyen rendszerparaméter, vagy felhasználói adat, azonban az „nem az információ feldolgozó eszközt, hanem a tárolt információt használja fel csalinak.” [8] Például egy, a támadó szempontjából értékesnek tűnő (kitüntetett felhasználói név), azonban az adott rendszeren korlátozott jogosultságokkal bíró (már bejelentkezési jogosultsággal sem rendelkező) felhasználói fiók lehet honeytoken, hiszen amennyiben a belépési próbálkozás naplózásra kerül, az információt szolgáltathat potenciális támadásra vonatkozóan. [4]

A továbbiakban honeytoken technikák és koncepciók részletesebb tárgyalása, illetve felhasználása nem célja dolgozatomnak.

2.1.3. Honeynet

Egy honeynet jellemzően olyan, általában az éles rendszerekkel megegyező (elsősorban szoftver-) konfigurációval rendelkező számítógépek, illetve honeypot rendszerek hálózata, amelyek közös irányítás, felügyelet alá tartoznak. [3] [4]

[2] A kifejezés itt az ellenérdekelt fél által végzett vizsgálatra (pl. felderítési, szkennelési tevékenység) utal.

2.2. Működés jellege

Működésük jellege, tehát a támadás módja szerint a honeypot rendszerek következő két csoportját különböztethetjük meg:

2.2.1. Aktív honeypot

Az aktív, azaz kliensoldali honeypot a kártékony tartalmak, támadási módszerek (pl. weboldalba ágyazott, a webböngésző alkalmazás sérülékenységét kihasználó káros kódok) begyűjtését végzi valamilyen forrásból (pl. web, peer-to-peer fájlcserélő hálózatok, azonnali üzenetküldő rendszerek), jellemzően az adott alkalmazás automatizált futtatásával. Az észlelés az automatizált futtatás előtti és utáni állapot, bizonyos rendszerjellemzők összehasonlítása alapján valósulhat meg. [8]

2.2.2. Passzív honeypot

A passzív, azaz szerveroldali honeypot a hálózaton elérhető szerveroldali szolgáltatás(ok) (pl. SSH, HTTP) imitálását végzi. [8]

Dolgozatom a továbbiakban kizárólag ezen szerveroldali honeypot megvalósításokkal foglalkozik.

2.3. Feladat

2.3.1. Védelmi célú honeypot[3]

Egy szervezet informatikai hálózati, hálózatbiztonsági infrastruktúrájába illeszkedő védelmi célú honeypot rendszerek elsősorban a védendő éles rendszerek minél hatásosabb imitálását célozzák. [4] Ezen szerepüket a biztonság három fő alapelvének (prevenció, detekció, reakció) szempontjából vizsgálhatjuk. [3] [7]

Prevenció alatt olyan védelmi intézkedéseket értünk, amelyek célja megelőzni a biztonsági incidenseket. Mivel a honeypot rendszerek egyik alapvető tulajdonsága a lehetséges támadásoknak való szándékos kitettség, így a preventív intézkedésekhez ebből adódóan közvetlenül nem járulnak hozzá, azonban bizonyos esetekben számba vehető pszichológiai hatásokkal rendelkeznek: a honeypot megtéveztheti a támadót, számára éles rendszer látszatát keltve, ezzel lekötve erőforrásait az éles rendszerek támadása elől[4], továbbá amennyiben a támadó előtt a honeypot lelepleződik, elriaszthatja a további támadási kísérletektől[5]. Fontos megjegyezni azonban, hogy automatizált támadások,

[3] Production honeypot.
[4] Megtévesztés.
[5] Elrettentés.

illetve kellően motivált emberi támadó esetén ezek a hatások nem feltétlenül érvényesülnek. [3] [7]

Detekció alatt a lehetséges jogosulatlan tevékenységek észlelésére, érzékelésére irányuló intézkedéseket értjük, amelyek megvalósíthatók pl. szabályalapú hálózati IDS[6] technológiák alkalmazásával. Ezen rendszerek esetén jelentős mértékű emberi beavatkozást igényelhet az esetlegesen nagyszámú fals pozitív[7] észlelés kezelése, amely rendszerint a szabályrendszer módosítását vonja maga után. Túl laza szabályrendszer azonban fals negatív[8] eseteket eredményezhet. Mivel egy honeypot jellemzően nem folytat hálózati forgalmat legitim rendszerekkel (kivétel pl. az üzemeltető tevékenység kapcsán keletkező szándékos forgalom), ezért ezeknek a hibáknak az előfordulása minimálisnak tekinthető, így a detekcióhoz jelentős mértékben hozzájárulhat. [3] [4] [5] [7] [8]

Reakció alatt a támadás bekövetkezését követő tevékenységek összességét értjük, beleértve az incidens megszüntetését, a károk csillapítását, továbbá a bizonyítékok összegzését. Utóbbi során felmerül a probléma, hogy egy éles rendszer esetében magából a rendeltetésszerű használatból származó adatok, pl. a naplófájlokban keletkező bejegyzések, fájl időbélyegek megváltozása stb. gyakorlatilag zajként foghatók fel, a támadásról kinyerhető adatok használhatóságát, azok bizonyító értékét csökkentik. További problémát jelenthet, hogy egyes üzleti rendszereket az üzemeltető szervezetek esetleg az incidens elhárításával összefüggésben sem választhatnak le a hálózatról, mivel azok (üzleti szempontból) kritikus fontosságú szolgáltatásokat nyújtanak, ez azonban hátráltathatja a hiteles és pontos bizonyítékgyűjtést. Honeypot rendszerek esetén ezek a problémák nem állnak fenn, mivel éles szolgáltatások nem futnak rajtuk, ebből adódóan ilyen módon a keletkezett bizonyítékok nem szennyeződnek, valamint ezek a rendszerek bármikor leválaszthatóak a hálózatról elemzés céljából, ezért felhasználhatóak digitális nyomforrásként. [3] [4] [7]

2.3.2. Kutatási célú honeypot[9]

A kutatási célú honeypot rendszereket nem kimondottan egy adott szervezet hálózatára irányuló támadások megértésének, megelőzésének vagy észlelésének céljából üzemeltetik. Ezek pl. eddig ismeretlen támadóeszközök és technikák felfedezését teszik lehetővé, illetve hozzájárulnak egy általános képet alkotni a támadók hátteréről, motivációiról, céljairól, továbbá későbbi lehetséges támadások előrejelzését támogathatják. [3] [4] [7]

---

[6] „A behatolásjelző rendszerek (IDS, Intrusion Detection System) célja, hogy a hálózati eszközök monitorozásával detektálják azok rendellenes vagy hibás viselkedését, működését.” [9]
[7] A rendszer a normál hálózati forgalmat ártalmasként detektálja.
[8] A rendszer nem detektálja az ártalmas hálózati forgalmat.
[9] Research honeypot.

2.4. Interakció szintje

A honeypot rendszerek csoportosíthatóak a támadó részére biztosított interakció lehetőségének, tehát az imitáció valószerűségének szintje alapján.

2.4.1. Alacsony interakciójú[10] honeypot

Egy alacsony interakciójú honeypot általában egy szolgáltatás vagy protokoll minimális részét valósítja meg, amely a támadó részére kevés mozgásteret nyújt. Előnye, hogy viszonylag egyszerűen üzembe helyezhető. Bár az üzemeltető fél részére alacsonyabb szintű adatokat, illetve kevesebb információt szolgáltat, mint egy magasabb interakciójú honeypot, azonban a jelentősen korlátozott számú szimulált funkció miatt, az alacsony komplexitásából adódóan, megfelelő implementáció esetén valószínűleg kevésbé sérülékeny, tehát kisebb valószínűséggel kompromittálható, ezért ez a típus nyújtja a legkisebb kockázatot. A limitált funkcionalitásból adódóan egy támadó azonban nagy valószínűséggel felfedezheti a rendszer csapda mivoltát. [3] [4] [5] [8]

2.4.2. Közepes interakciójú[11] honeypot

Egy közepes interakciójú honeypot összetetteb funkcionalitást valósít meg egy alacsony interakciójúnál. Gyakorlatilag bizonyos előre definiált beérkező forgalomra, tevékenységre előre definiált válaszokat képes adni, azonban ebben az esetben is csak egy szimulált szolgáltatás fut, amely az operációs rendszerhez szándékosan nem ad hozzáférést a támadónak. Általában a bonyolultabb üzembe helyezése, konfigurációja több időt vesz igénybe, azonban sokkal több adatot képes biztosítani egy támadásról. Mivel magasabb komplexitású, ezért nagyobb a valószínűsége az implementációban lévő hibáknak, ezáltal a kompromittációnak is, így az alacsony interakciójú honeypot rendszernél nagyobb kockázatot nyújt, viszont kisebb az esélye, hogy a támadó előtt lelepleződik. [3]

2.4.3. Magas interakciójú[12] honeypot

Egy magas interakciójú honeypot magasabb szintű funkciókat valósít meg, akár a rendszer valódi, vagy valódinak látszó (pl. hamis adatokat tartalmazó adatbázis) erőforrásaihoz is hozzáférést adva, egy éles rendszert teljesen valószerűen imitálva. Ebben az esetben gyakorlatilag valós szerveroldali alkalmazást vagy alkalmazásokat futtatunk az operációs rendszer felett, pl. a szervezet éles rendszereivel teljesen megegyező rendszerek üzembe helyezésével, viszont az ezek által nyújtott szolgáltatásokat más (üzleti) céllal nem vesszük igénybe. Jellemzően ez a típus igényli a legtöbb konfigurációt és manuális testre szabást, azonban így nyerhető ki a legtöbb

[10] Low-interaction.
[11] Medium-interaction.
[12] High-interaction.

információ egy támadásról, illetve támadóról. A legmagasabb kockázatot nyújtja, kompromittáció esetén a támadó számára egy teljesértékű rendszer áll rendelkezésre, amelyről akár további támadásokat indíthat, ebből kifolyólag csak erősen szabályozott környezetben ajánlott üzemeltetésük. [3] [4] [5] [8]

2.5. Virtualizáció

Egy következő lehetséges csoportosítás a fizikai és a virtuális honeypot megvalósítások között tesz különbséget.

2.5.1. Fizikai honeypot

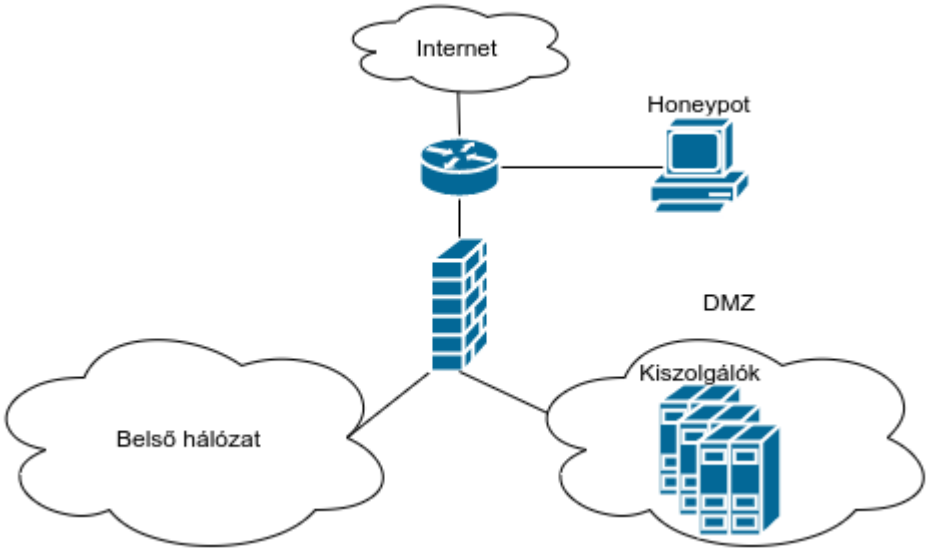
Fizikai honeypot esetén a rendszer valós számítógépen kerül futtatásra. Ez sok esetben magas interakciójú megvalósítást jelent, így bár a csapdának a támadó előtti lelepleződése kevésbé valószínű, a teljes rendszer kompromittációjának megnő az esélye. Jellemzően költségesebb és időigényesebb (fizikai hardver beszerzése, karbantartása) egy virtuális honeypot üzemeltetésénél. Nagyobb IP cím tartomány esetén a teljes hálózatot lefedni csapda rendszerekkel sokkal nagyobb befektetéssel járna. Ezen hátrányai miatt kevésbé elterjedt. [5] [8]

2.5.2. Virtuális honeypot

Virtuális honeypot esetén a fizikai hardver felett valamilyen virtualizációs környezetben fut a rendszer. Előnye, hogy egy fizikai gépen több honeypot is működhet (akár különböző operációs rendszereken), a különböző szolgáltatások szeparált, szabályozottabb környezetben futhatnak, valamint a felügyeletük, telepítésük, kompromittáció esetén a helyreállításuk jelentősen egyszerűbbé válik. Mivel az üzemeltetést a virtualizáció jelentősen egyszerűsiti, ideális magas interakciójú honeypot megvalósításához, tényleges hálózati alkalmazások szabályozottabb futtatásához, szemben a szimulált szolgáltatásokkal. [4] [5] [8]

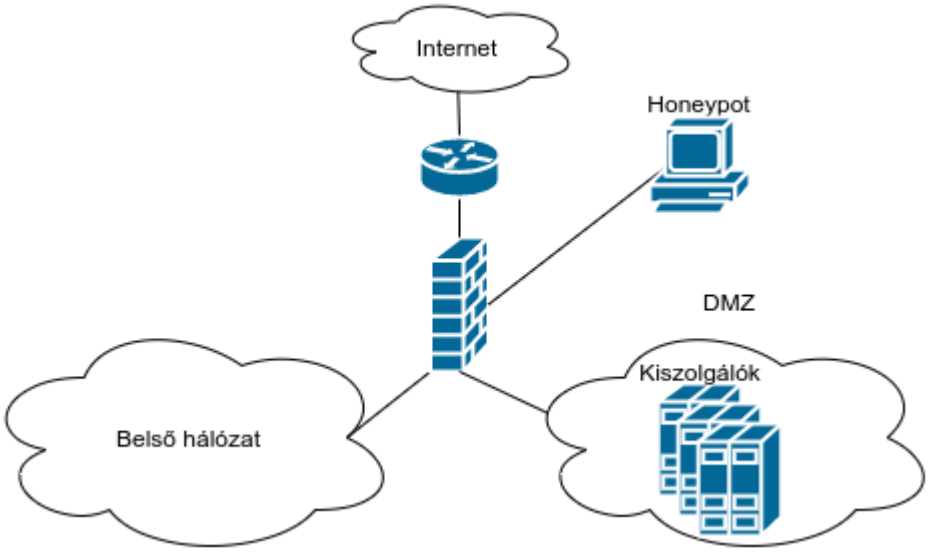
2.6. Elhelyezés

2.6.1. Szervezeti hálózatban



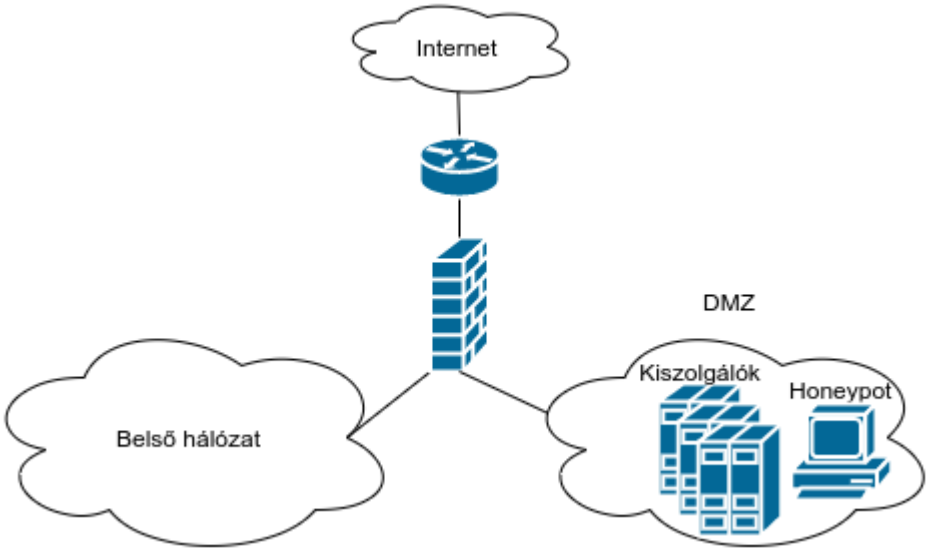
1. ábra: Honeypot külső elhelyezése. [3]

Határvédelmi tűzfalon kívüli elhelyezés (1. ábra) esetén, ha egy támadó detektálta a csapdát, esetleg felhagyhat a tevékenységével, mivel nem pazarolná erőforrásait további lehetséges honeypot rendszerekre; azonban ellenkező hatást kiváltva épp felhívhatja a figyelmet a hálózatra, ösztönözheti annak további felderítését, megtámadását, ezzel gyakorlatilag kockázatnak kitéve azt, ami prevenció szempontjából nem bizonyul előnyösnek. Az ide elhelyezett rendszer érzékeli ugyan a támadásokat, viszont a kívülről érkező automatikus felderítési kísérletek nagy száma miatt a szervezeti hálózatban a detekcióhoz kevéssé járulhat hozzá, mivel túl sok támadást jelezne, ezzel feleslegesen lekötve az üzemeltetőt. Bizonyos esetekben viszont pont ezen kísérletek megfigyelése lehet a cél, például egy viszonyítási alap felállításához a tűzfal felé érkező külső forgalom tekintetében. Szintén kevés haszna van ennek az elhelyezésnek reakció szempontjából, ugyanis az ebből a célból telepített honeypot rendszereket sokkal inkább azokkal az éles rendszerekkel azonos hálózatban célszerű elhelyezni, amelyeket imitálni hivatottak. A külső hálózaton elhelyezett honeypot rendszer ezen hátrányok miatt védelmi célra kevésbé alkalmas, viszont mivel nagy valószínűséggel támadják meg, ezért kutatási célú rendszer esetén előnyös, mivel kevesebb konfigurációt igényel, és a nagyszámú támadásról sok adatot szolgáltathat. [3] [4]



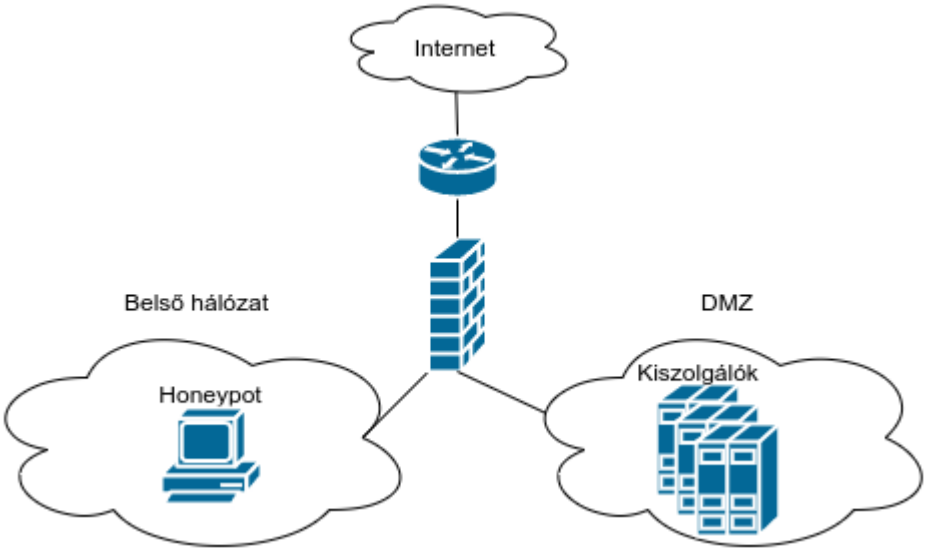
2. ábra: Honeypot elhelyezése tűzfalon belül. [3]

Minden esetben biztosítani kell, hogy az esetlegesen kompromittálódott rendszerünket a támadó ne használhassa ugródeszkaként további rendszerek megtámadására, beleértve a szervezetünk és más szervezetek rendszereit is, ezért a külső elhelyezéshez képest célszerűbb lehet a tűzfal mögötti, szeparált hálózaton történő elhelyezés (2. ábra), ami lehetővé teszi csali rendszerről kimenő forgalom korlátozását is. [3] [4]



3. ábra: Honeypot elhelyezése DMZ hálózaton. [3]

DMZ hálózaton lévő kiszolgálók védelmére sokkal célravezetőbb a velük azonos hálózaton elhelyezett (3. ábra) csapda, mind prevenció, detekció, és reakció szempontjából. Ezen a hálózaton viszont éles rendszerek találhatók, így ide kutatási célú honeypot telepítése nem javasolt, mivel kompromittációja esetén azt a támadó továbbra is könnyedén használhatja az itt elhelyezkedő éles rendszerek támadására. [3] [4]



4. ábra: Honeypot elhelyezése belső hálózaton. [3]

Honeypot közvetlenül belső hálózaton történő elhelyezése (4. ábra) a belső fenyegetések (pl. alkalmazottak által folytatott szándékos károkozás) felderítése szempontjából előnyös lehet, viszont hátránya, hogy kompromittáció esetén a belső hálózaton belüli forgalom korlátozására valószínűleg kevésbé van lehetőség a megfelelő eszközök (pl. tűzfal) hiányában. [3] [4]

2.6.2. Felhőszolgáltatásként

Az NIST[13] meghatározása szerint a felhőalapú számítástechnika alapvető jellemzői az igény szerinti önkiszolgálás (a felhasználó[14] szüksége szerint kiválaszthatja a szolgáltatásokat és hozzáférhet azokhoz, mivel a megrendelés és a teljesítés automatizált, nem igényel manuális beavatkozást), a széles hálózati hozzáférés (a felhasználó speciális hálózati kapcsolat nélkül, többféle kliens segítségével is igénybe veheti a szolgáltatást), az erőforrás-összevonás (a szolgáltató kihasználja, hogy a felhasználó általában nem folyamatosan veszi igénybe az erőforrásait, ezért a kihasználatlan erőforrásokat más felhasználó részére ajánlja ki), a bővítés szempontjából való rugalmasság (a szolgáltató rendelkezik annyi tartalékkal, hogy a felhasználó részére bármikor többlet erőforrást tudjon biztosítani), valamint a mért szolgáltatás (az elszámolás a használt erőforrások, szolgáltatások típusa alapján meghatározott, a mért felhasználás szerinti mennyiségén alapszik). [10]

Felhőüzemeltetési modell szempontjából megkülönbözteti a nyilvános (a szolgáltatást nyújtó rendszereket és erőforrásokat külső szolgáltató birtokolja, irányítja, üzemelteti), magán (a szolgáltató maga a felhasználó), közösségi (fél-nyilvános felhő, a felhasználói

[13] National Institute of Standards and Technology, USA Nemzeti Szabványügyi és Technológiai Intézete.
[14] A felhasználó itt maga a felhőszolgáltatás igénybe vevője, amely lehet egy szervezet vagy személy is.

általában néhány, valamilyen közös céllal működő szervezet, akik között megoszlik az üzemeltetés felelőssége is), és hibrid (különböző, az előzőekben felsorolt módszerek valamilyen összekapcsolása) megoldásokat. [10]

Ezen túlmenően három alapvető felhőszolgáltatási modellt definiál, ahol a szolgáltató különböző szintű erőforrásokat üzemeltet és biztosít a felhasználó részére: IaaS[15] esetén alapszintű infrastruktúrát (fizikai, vagy virtuális gépek, hálózat, tárolás), PaaS[16] esetén operációs rendszert, valamilyen fejlesztői platformot, illetve adatbázist, SaaS[17] esetén alkalmazás szintű szolgáltatásokat. [10]

Brown és tsai. [11] egy 2012-ben publikált projekt keretein belül különböző publikus felhőszolgáltatók IaaS szolgáltatásait vették igénybe honeypot eszközök telepítésére, amely során képesek voltak alapvető támadói profilokat felállítani, beleértve a forrás lehetséges földrajzi elhelyezkedését, a támadások során használt operációs rendszereket, a leggyakrabban megcélzott hálózati szolgáltatásokat, illetve az autentikációs kísérletek során használt felhasználóneveket és jelszavakat.

Ezek alapján az elsősorban IaaS szolgáltatásként igénybe vett virtuális gépeken futó csapda rendszerek főként kutatási [11] célokat szolgálhatnak.

2.7. Adatmennyiség és -minőség

Egy szervezeti hálózatban alapesetben is nagymennyiségű, biztonsági szempontból releváns adat keletkezhet, pl. hálózati infrastruktúra eszközök (switch, router, wireless vezérlő, wireless access point stb.), biztonsági eszközök (tűzfal, IDS, IPS stb.), szerver és kliens operációs rendszerek, és alkalmazások naplói. [12] Egy honeypot rendszer előnye, hogy mivel akaratlagosan van kitéve esetleges támadásoknak, így minden rajta folyó tevékenység kezdettől fogva gyanús, ezért az innen származó naplók kevés zajjal rendelkeznek, így adatminőség szempontjából értékesek, mégis aránylag kis adatmennyiséget képviselnek. [3] [5]

2.8. Jogi és etikai megfontolások

Dolgozatom célja elsősorban nem jogi szempontok elemzése, azonban az üzemeltetés során célszerű lehet a fizikai lokáció (az adott ország) jogrendszerét alapul venni. A globális internet felől nyilvánosan elérhető rendszerek esetén felmerülhetnek nemzetközi jogi kérdések is. A továbbiakban a legrosszabb eshetőség elvét követve feltételezem, hogy az üzemeltetőt minden esetben személyes felelősség terheli az esetlegesen kompromittálódott honeypot rendszerből indított rosszindulatú tevékenységért.

[15] Infrastructure as a Service: Infrastruktúra, mint szolgáltatás.
[16] Platform as a Service: Platform, mint szolgáltatás.
[17] Software as a Service: Szoftver, mint szolgáltatás.

3. KÖVETELMÉNY SPECIFIKÁCIÓ

A megjelenítést végző alkalmazás célja egy honeypot rendszeren keletkezett adatokból tömör vizuális összesítések (vizualizációk) létrehozása, amely az arra jogosult felhasználók (a csapda üzemeltetői) részére könnyen értelmezhető képet adhat az eddigi támadási kísérletekről, azok forrásáról. Az alkalmazásnak lehetővé kell tennie, hogy egy időben több felhasználó, több eszközről hozzáférhessen a vizualizációkhoz, ezért egy webes felhasználói felület kialakítása szükséges.

A webalkalmazás használatához a felhasználó bejelentkezése szükséges, amihez érvényes felhasználói névvel és jelszóval kell rendelkeznie. Az alkalmazásnak támogatnia kell a kétfaktoros autentikációt, az RFC 6238 szabványnak megfelelő TOTP[18] implementáció biztosításával, ha azt a felhasználó engedélyezi.

A vizualizációk oszlopdiagram formájában lesznek elkészítve. A felhasználó beállíthatja a részére megjeleníteni kívánt vizualizációkat és azok megjelenésének sorrendjét. A megjeleníteni kívánt időszakot, valamint az „N leggyakoribb" típusúak esetén a leggyakoribb értékek számát (N értékét) szintén a felhasználó állíthatja be. Lehetőség lesz az adatokat táblázatos formában is megjeleníteni. Ezek a beállítások a felhasználói munkamenetek között megmaradnak.

A jelszó alapú SSH autentikációs kísérletekre vonatkozóan kiválasztható vizualizációk:

- Kísérletek száma, N leggyakoribb forrás IPv4 cím szerint.
- Kísérletek száma, N leggyakoribb felhasználónév-jelszó páros szerint.
- Kísérletek száma, N leggyakoribb felhasználónév szerint.
- Kísérletek száma, N leggyakoribb jelszó szerint.
- Kísérletek száma, hónap napjai szerint.
- Kísérletek száma, év hónapjai szerint.

A beérkezett hálózati csomagokra vonatkozóan kiválasztható vizualizációk:

- Csomagok száma, N leggyakoribb forrás IPv4 cím szerint.
- Csomagok száma, a felsőbb rétegbeli protokoll szerint.
- Csomagok száma, hónap napjai szerint.
- Csomagok száma, év hónapjai szerint.

A beérkezett TCP szegmensekre vonatkozóan kiválasztható vizualizációk:

- TCP szegmensek száma, az N leggyakoribb cél port szerint.

[18] Time-based One Time Password, időalapú egyszeri jelszó.

A beérkezett UDP datagramokra vonatkozóan kiválasztható vizualizációk:

- UDP datagramok száma, az N leggyakoribb cél port szerint.

A beérkezett ICMP csomagokra vonatkozóan kiválasztható vizualizációk:

- ICMP csomagok száma, az N leggyakoribb típus-kód mező szerint.

Az alkalmazásnak a következő felhasználói szerepköröket és hozzájuk tartozó funkcionalitást kell megvalósítania:

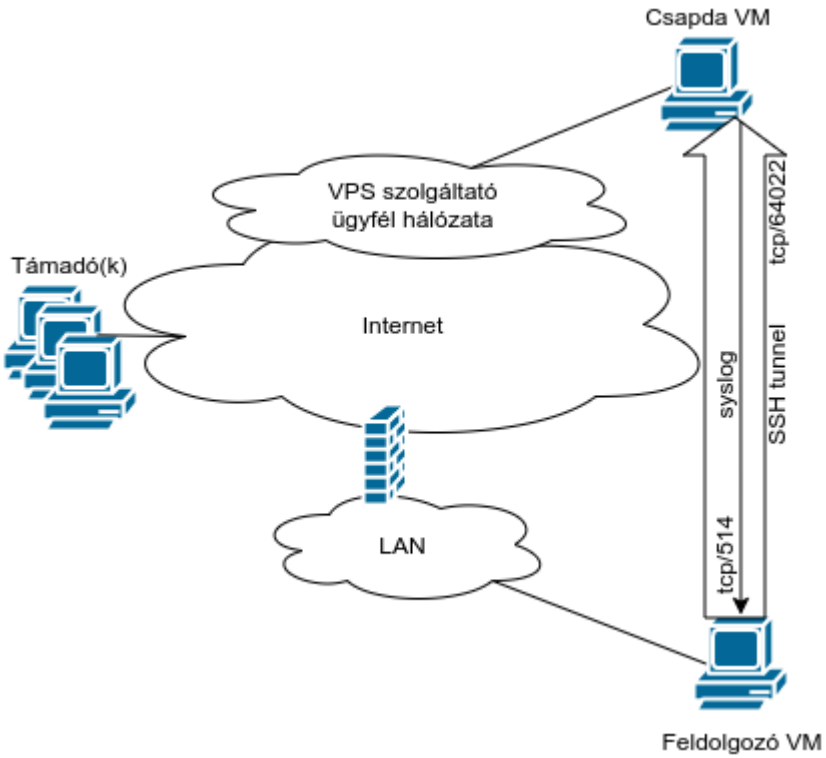
- Olvasó szerepkörű felhasználó:
  - Bejelentkezés a web alapú felhasználói felületre.
  - Vizualizációk megtekintése.
  - Vizualizációk megjelenítési beállításainak módosítása:
    - Megjelenítés engedélyezése.
    - Megjelenítés letiltása.
    - Táblázatos megjelenítési formátum engedélyezése.
    - Táblázatos megjelenítési formátum tiltása.
    - „N leggyakoribb” típusú vizualizáció N értékének beállítása.
    - Megjelenő időszak beállítása.
  - Bejelentkezési beállítások módosítása:
    - Bejelentkezéshez használt jelszó módosítása.
    - TOTP alapú kétfaktoros hitelesítés engedélyezése.
    - TOTP alapú kétfaktoros hitelesítés letiltása.
  - Kijelentkezés, a felhasználói munkamenet megszakítása.
- Adminisztrátor szerepkörű felhasználó:
  - Minden, az olvasó szerepkörben elérhető funkcionalitás.
  - Az összes felhasználó menedzselése:
    - Új felhasználó felvétele:
      - Felhasználónév megadása.
      - Jelszó megadása.
    - Felhasználó bejelentkezési adatainak módosítása:
      - Felhasználónév módosítása.
      - Jelszó módosítása.
    - Felhasználó törlése.

4. TESZTKÖRNYEZET KIALAKÍTÁSA

A csapda rendszernek az SSH szolgáltatásra vonatkozóan biztosítania kell a jelszó alapú autentikáció során használt felhasználói nevet, jelszót, és forrás IPv4 címet. Honeypot eszközök kiválasztása során törekedni kell a minél biztonságosabb szoftverek alkalmazására, hogy a rendszer kompromittációjának esélye minél kisebb legyen, ezért elsősorban az alacsony interakciójú megvalósításokat vettem számításba.

A beérkező hálózati csomagok tekintetében szükséges az IPv4 csomag fejlécének forrás IP cím mezője, beágyazott felsőbb rétegbeli protokoll esetén annak azonosítója, beágyazott ICMP fejléc esetén szükséges a típus és a kód mező, beágyazott TCP szegmens vagy UDP datagram esetén fejlécének cél port mezője.

Az eszközöknek támogatnia kell a syslog alapú naplózást, mivel az üzemeltetésben elterjedt formátumról van szó, így az esetleg később integrálásra kerülő egyéb alkalmazások naplói is közös rendszerben lesznek tárolhatóak a meglévőkkel, ami a feldolgozást megkönnyíti.

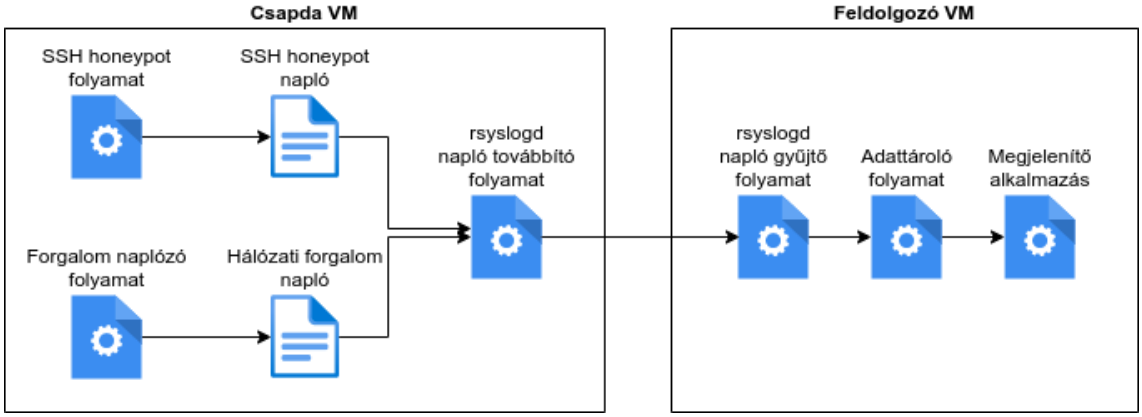


5. ábra: Tesztkörnyezet hálózati architektúrája.

A rendszert valós támadások adataival is szeretném tesztelni, azonban a szakdolgozat időpontjában nem áll rendelkezésemre olyan szervezeti hálózat, amelyben elhelyezett honeypot felé nagy számú támadási kísérlet lenne várható, ezért a csapda rendszert egy hazai szerverbérlési szolgáltatásokat nyújtó cég infrastruktúrájában már korábban bérelt,

az internet felől egy IPv4 címen elérhető virtuális gépen (Csapda VM) valósítom meg (5. ábra). A bejövő hálózati forgalomra alapértelmezetten nincs csomagszűrés beállítva, ezáltal lehetővé téve annak teljes naplózását. A virtuális gépen Debian 11 disztribúció fut, mivel egy elterjedt, jól dokumentált, ingyenesen használható operációs rendszer, amelyre számos szoftver előre csomagoltan elérhető.

Fontos az adatok tárolásának és feldolgozásának a csapdától szeparált megvalósítása, hogy a keletkezett adatok a csapda esetleges kompromittációja esetén se vesszenek el (pl. támadó szándékos károkozása miatt). Mivel nem rendelkezek több bérelt virtuális géppel, a feldolgozásért felelős komponenseket egy tűzfal és NAT mögötti otthoni helyi hálózatban lévő virtuális gépen (Feldolgozó VM) állítom üzembe (5. ábra). A virtuális gépek közötti biztonságos kapcsolatot a Feldolgozó VM felől a Csapda VM felé létrehozott SSH tunnel segítségével teszem lehetővé, amin keresztül a Csapda VM napló továbbító folyamata csatlakozik a Feldolgozó VM naplógyűjtő folyamatához, továbbítva a honeypot adatokat (6. ábra).



6. ábra: Az adatok továbbítása.

A naplók továbbításához (6. ábra) az rsyslog alkalmazást használom, mivel a csapda rendszeren futó operációs rendszer alapértelmezett naplókezelője, széles körben elterjedt, aktívan karbantartott, nyílt forráskódú szoftver, amely számos kimeneti modulja segítségével különféle adatbázisba is képes továbbítani az adatokat. [13]

5. ELÉRHETŐ HONEYPOT ESZKÖZÖK

Olyan kereskedelmi forgalomban elérhető, klasszikus értelemben vett honeypot eszközök, amelyek a dolgozat elkészítésének időpontjában is gyártói, illetve fejlesztői támogatásban részesülnek, eddigi kutatásom alapján nem érhetőek el, ezért a továbbiakban a fellelhető, aktívan fejlesztett open-source megoldásokra koncentráltam.

Az egyes eszközöket az nmap 7.80 verziójú hálózati feltérképező eszköz, annak kiválasztott szkriptjei, és az OpenSSH 8.4 kliens segítségével (7. ábra) vizsgáltam meg.

```
$ uname -srvmo
Linux 5.10.0-9-amd64 #1 SMP Debian 5.10.70-1 (2021-09-30) x86_64 GNU/Linux
$ cat /etc/debian_version
11.1
$ nmap --version
Nmap version 7.80 ( https://nmap.org )
Platform: x86_64-pc-linux-gnu
Compiled with: liblua-5.3.3 openssl-1.1.1j libssh2-1.9.0 libz-1.2.11 libpcre-8.3
9 libpcap-1.10.0 nmap-libdnet-1.12 ipv6
Compiled without:
Available nsock engines: epoll poll select
$ nmap --script-help ssh-auth-methods.nse
Starting Nmap 7.80 ( https://nmap.org ) at 2021-12-03 09:26 CET

ssh-auth-methods
Categories: auth intrusive
https://nmap.org/nsedoc/scripts/ssh-auth-methods.html
  Returns authentication methods that a SSH server supports.

  This is in the "intrusive" category because it starts an authentication with a
  username which may be invalid. The abandoned connection will likely be logged.
$ nmap --script-help ssh-brute.nse
Starting Nmap 7.80 ( https://nmap.org ) at 2021-12-03 09:27 CET

ssh-brute
Categories: brute intrusive
https://nmap.org/nsedoc/scripts/ssh-brute.html
  Performs brute-force password guessing against ssh servers.
$ ssh -V
OpenSSH_8.4p1 Debian-5, OpenSSL 1.1.1k  25 Mar 2021
```

7. ábra: Vizsgálathoz használt eszközök.

5.1. SSH szolgáltatás

5.1.1. ssh-honeypotd

Az ssh-honeypotd egy jelenleg is aktívan fejlesztett és karbantartott, C nyelven írt alacsony interakciójú, szerveroldali SSH honeypot. [14] Az eszköz a bejelentkezési adatokat naplózza. Az ssh-honeypotd 2.1.2 verzióját használtam, amit a tesztrendszeren a megfelelő csomagfüggőség telepítése után, forráskódból fordítva futtattam (8. ábra).

```
$ sudo apt-get install -qq -o=Dpkg::Use-Pty=0 libssh-dev
Selecting previously unselected package libssh-dev:amd64.
(Reading database ... 414498 files and directories currently installed.)
Preparing to unpack .../libssh-dev_0.9.5-1+deb11u1_amd64.deb ...
Unpacking libssh-dev:amd64 (0.9.5-1+deb11u1) ...
Setting up libssh-dev:amd64 (0.9.5-1+deb11u1) ...
$ git clone -q https://github.com/sjinks/ssh-honeypotd.git
$ cd ssh-honeypotd/
$ make
cc  -fvisibility=hidden -Wall -Werror -Wno-error=attributes -Wno-unknown-pragmas
  -c "main.c" -MMD -MP -MF"main.dep" -MT"main.dep" -o "main.o"
cc  -fvisibility=hidden -Wall -Werror -Wno-error=attributes -Wno-unknown-pragmas
  -c "globals.c" -MMD -MP -MF"globals.dep" -MT"globals.dep" -o "globals.o"
cc  -fvisibility=hidden -Wall -Werror -Wno-error=attributes -Wno-unknown-pragmas
  -c "cmdline.c" -MMD -MP -MF"cmdline.dep" -MT"cmdline.dep" -o "cmdline.o"
cc  -fvisibility=hidden -Wall -Werror -Wno-error=attributes -Wno-unknown-pragmas
  -c "pidfile.c" -MMD -MP -MF"pidfile.dep" -MT"pidfile.dep" -o "pidfile.o"
cc  -fvisibility=hidden -Wall -Werror -Wno-error=attributes -Wno-unknown-pragmas
  -c "daemon.c" -MMD -MP -MF"daemon.dep" -MT"daemon.dep" -o "daemon.o"
cc  -fvisibility=hidden -Wall -Werror -Wno-error=attributes -Wno-unknown-pragmas
  -c "worker.c" -MMD -MP -MF"worker.dep" -MT"worker.dep" -o "worker.o"
cc  -fvisibility=hidden -Wall -Werror -Wno-error=attributes -Wno-unknown-pragmas
  -c "log.c" -MMD -MP -MF"log.dep" -MT"log.dep" -o "log.o"
cc main.o globals.o cmdline.o pidfile.o daemon.o worker.o log.o -lssh  -pthread
 -o ssh-honeypotd
$ ./ssh-honeypotd -v
ssh-honeypotd 2.1.2
Copyright (c) 2014-2021, Volodymyr Kolesnykov <volodymyr@wildwolf.name>
License: MIT <http://opensource.org/licenses/MIT>
```

8. ábra: Az ssh-honeypotd futtatása.

Hogyha az SSH alapértelmezett portján kell elérhetővé tenni, és nincs lehetőség port átirányításra, lehetőség van az alkalmazást root felhasználóként elindítani, ami ezután nemprivilegizált felhasználó és csoport (alapértelmezetten daemon/daemon, vagy nobody/nogroup) jogosultságra fog átváltani, így a szoftverben lévő esetleges sérülékenység kihasználása esetén sem szerezhet egy támadó egyből root jogosultságot a rendszeren. A honeypot a jelszó alapú SSH autentikációt támogatja (9. ábra).

```
# ./ssh-honeypotd -P /run/ssh-honeypotd.pid
# ps -o pid,user,group,cmd --pid $(cat /run/ssh-honeypotd.pid)
    PID USER     GROUP    CMD
   4268 nobody   nogroup  ./ssh-honeypotd -P /run/ssh-honeypotd.pid
# nmap --script ssh-auth-methods.nse -p22 localhost
Starting Nmap 7.80 ( https://nmap.org ) at 2021-12-01 14:38 CET
Nmap scan report for localhost (127.0.0.1)
Host is up (0.000066s latency).
Other addresses for localhost (not scanned): ::1

PORT   STATE SERVICE
22/tcp open  ssh
| ssh-auth-methods:
|   Supported authentication methods:
|_    password

Nmap done: 1 IP address (1 host up) scanned in 0.38 seconds
```

9. ábra: Az ssh-honeypotd által támogatott autentikációs metódus.

Az eszköz az autentikációs kísérleteknek syslog vagy stderr kimenetre történő naplózását teszi lehetővé. Egy kísérlet alapján egy egysoros bejegyzés keletkezik, amely tartalmazza a használt felhasználónevet, forrás IP címet, forrás portot, protokoll verziót, cél IP címet, cél portot, és a jelszót (10. ábra).

```
Dec  1 14:45:19 machine ssh-honeypotd[4268]: Failed password for administrator f
rom 127.0.0.1 port 56892 ssh2 (target: 127.0.0.1:22, password: barbie)
Dec  1 14:45:19 machine ssh-honeypotd[4268]: Failed password for webadmin from 1
27.0.0.1 port 56894 ssh2 (target: 127.0.0.1:22, password: barbie)
Dec  1 14:45:19 machine ssh-honeypotd[4268]: Failed password for sysadmin from 1
27.0.0.1 port 56896 ssh2 (target: 127.0.0.1:22, password: barbie)
Dec  1 14:45:19 machine ssh-honeypotd[4268]: Failed password for netadmin from 1
27.0.0.1 port 56898 ssh2 (target: 127.0.0.1:22, password: barbie)
Dec  1 14:45:19 machine ssh-honeypotd[4268]: Failed password for guest from 127.
0.0.1 port 56900 ssh2 (target: 127.0.0.1:22, password: barbie)
Dec  1 14:45:19 machine ssh-honeypotd[4268]: Failed password for user from 127.0
.0.1 port 56902 ssh2 (target: 127.0.0.1:22, password: barbie)
Dec  1 14:45:19 machine ssh-honeypotd[4268]: Failed password for web from 127.0.
0.1 port 56904 ssh2 (target: 127.0.0.1:22, password: barbie)
Dec  1 14:45:19 machine ssh-honeypotd[4268]: Failed password for test from 127.0
.0.1 port 56906 ssh2 (target: 127.0.0.1:22, password: barbie)
Dec  1 14:45:19 machine ssh-honeypotd[4268]: Failed password for root from 127.0
.0.1 port 56908 ssh2 (target: 127.0.0.1:22, password: 666666)
Dec  1 14:45:19 machine ssh-honeypotd[4268]: Failed password for admin from 127.
0.0.1 port 56910 ssh2 (target: 127.0.0.1:22, password: 666666)
Dec  1 14:45:19 machine ssh-honeypotd[4268]: Failed password for administrator f
rom 127.0.0.1 port 56912 ssh2 (target: 127.0.0.1:22, password: 666666)
Dec  1 14:45:19 machine ssh-honeypotd[4268]: Failed password for webadmin from 1
27.0.0.1 port 56914 ssh2 (target: 127.0.0.1:22, password: 666666)
Dec  1 14:45:19 machine ssh-honeypotd[4268]: Failed password for sysadmin from 1
27.0.0.1 port 56916 ssh2 (target: 127.0.0.1:22, password: 666666)
Dec  1 14:45:19 machine ssh-honeypotd[4268]: Failed password for netadmin from 1
27.0.0.1 port 56918 ssh2 (target: 127.0.0.1:22, password: 666666)
```

10. ábra: Példa az ssh-honeypotd syslog kimenetére.

5.1.2. cowrie

A cowrie egy jelenleg is aktívan fejlesztett, Python nyelven írt szerveroldali SSH és telnet honeypot, amely közepes vagy magas interakciójú működést támogat. A jelszó alapú autentikáción túl közepes interakciójú módban egy Unix shell emulációját valósítja meg, míg magas interakciójú módban a támadó és egy valódi SSH vagy telnet szolgáltatás közé ékelhető, ezáltal lehetővé téve a támadó bejelentkezést követő további tevékenységeinek nyomon követését is. A szolgáltatás alapértelmezett konfigurációval közepes interakciójú módban fut és a TCP 2222 porton elérhető. [15]

A cowrie 2.3.0 verzióját használtam. A rendszerszintű csomagfüggőségek telepítésén túl Python virtuális környezet létrehozása (11. ábra), valamint a Python csomagfüggőségek telepítése is szükséges. Nem valósítja meg a folyamat felhasználói jogosultságainak váltását, ezért éles használat esetén célszerű dedikált nemprivilegizált felhasználó és csoport nevében futtatni, és az alapértelmezett port használatához port átirányítást konfigurálni [15] [16], viszont a vizsgálathoz ezek a lépések kihagyhatóak.

```
$ sudo apt-get install -y git python3-virtualenv libssl-dev libffi-dev build-ess
ential libpython3-dev python3-minimal authbind virtualenv &> /dev/null
$ virtualenv -q --python=python3 cowrie-env
$ wget -q https://github.com/cowrie/cowrie/archive/refs/tags/v2.3.0.tar.gz
$ tar zxf v2.3.0.tar.gz
$ source cowrie-env/bin/activate
(cowrie-env) $ pip install -qU pip
(cowrie-env) $ pip install -qUr cowrie-2.3.0/requirements.txt
(cowrie-env) $ cowrie-2.3.0/bin/cowrie start &> /dev/null
(cowrie-env) $ cowrie-2.3.0/bin/cowrie status
cowrie is running (PID: 16934).
(cowrie-env) $ nmap --script +ssh-auth-methods.nse -p2222 localhost
Starting Nmap 7.80 ( https://nmap.org ) at 2021-12-02 13:17 CET
Nmap scan report for localhost (127.0.0.1)
Host is up (0.00013s latency).
Other addresses for localhost (not scanned): ::1

PORT     STATE SERVICE
2222/tcp open  EtherNetIP-1
| ssh-auth-methods:
|   Supported authentication methods:
|     publickey
|_    password

Nmap done: 1 IP address (1 host up) scanned in 0.36 seconds
```

11. ábra: A cowrie által támogatott autentikációs metódusok.

Az alkalmazás számos más kimeneti lehetőség mellett (pl. SQLite, MySQL) támogatja a syslog naplózást [15], azonban ezt külön kell engedélyezni (12. ábra), az alapértelmezett konfigurációval JSON formátumú naplófájlt hoz létre.

```
(cowrie-env) $ cat >> cowrie-2.3.0/etc/cowrie.cfg
[output_localsyslog]
enabled = true
facility = AUTH
format = text
(cowrie-env) $ cowrie-2.3.0/bin/cowrie restart
```

12. ábra: A cowrie syslog alapú naplózásának engedélyezése.

Az eszköz a következő syslog bejegyzéseket hozta létre (13. ábra):

- Bejegyzés új SSH kapcsolat tényéről, ami tartalmazza a forrás IP címet, a forrás portot, a cél IP címet, a cél portot, és egy egyedi SSH munkamenet azonosítót.
- Bejegyzés, ami tartalmazza a kliens által küldött azonosító sztringet.
- Bejegyzés, ami tartalmaz egy a kliens implementációra jellemző ujjlenyomatot.
- Bejegyzés a sikertelen bejelentkezés tényéről, ami tartalmazza a használt felhasználónevet és jelszót.
- Bejegyzés a kapcsolat megszakadásáról.

```
Dec  9 17:37:10 machine cowrie: [cowrie.ssh.factory.CowrieSSHFactory] New connec
tion: 127.0.0.1:41422 (127.0.0.1:2222) [session: 2ebc0bce521d]
Dec  9 17:37:10 machine cowrie: [HoneyPotSSHTransport,678,127.0.0.1] Remote SSH
version: SSH-2.0-OpenSSH_8.4p1 Debian-5
Dec  9 17:37:10 machine cowrie: [HoneyPotSSHTransport,678,127.0.0.1] SSH client
hassh fingerprint: ae8bd7dd09970555aa4c6ed22adbbf56
Dec  9 17:37:12 machine cowrie: [HoneyPotSSHTransport,678,127.0.0.1] login attem
pt [teszt/Teszt] failed
Dec  9 17:37:13 machine cowrie: [HoneyPotSSHTransport,678,127.0.0.1] Connection
lost after 3 seconds
```

13. ábra: Példa a cowrie syslog kimenetére.

5.2. Egyéb forgalom

5.2.1. tcpdump

A tcpdump a libpcap hordozható C/C++ könyvtárra épülő parancssori csomagelemző eszköz, amely számos népszerű operációs rendszerre elérhető. [17] Az alkalmazás lehetővé teszi a hálózati interfészekről érkező és azokon küldött csomagok (bizonyos mezőinek) szöveges formátumban történő megjelenítését (14. ábra).

```
$ sudo tcpdump -q -i lo -n 'ip and (tcp or udp or icmp)'
tcpdump: verbose output suppressed, use -v[v]... for full protocol decode
listening on lo, link-type EN10MB (Ethernet), snapshot length 262144 bytes
13:31:57.638280 IP 127.0.0.1.44545 > 127.0.0.1.23: tcp 0
13:31:57.638299 IP 127.0.0.1.23 > 127.0.0.1.44545: tcp 0
13:31:57.638311 IP 127.0.0.1.44545 > 127.0.0.1.80: tcp 0
13:31:57.638326 IP 127.0.0.1.80 > 127.0.0.1.44545: tcp 0
13:31:57.638331 IP 127.0.0.1.44545 > 127.0.0.1.80: tcp 0
13:31:57.638342 IP 127.0.0.1.44545 > 127.0.0.1.443: tcp 0
13:31:57.638347 IP 127.0.0.1.443 > 127.0.0.1.44545: tcp 0
13:32:04.998197 IP 127.0.0.1.49131 > 127.0.0.1.161: UDP, length 60
13:32:04.998218 IP 127.0.0.1 > 127.0.0.1: ICMP 127.0.0.1 udp port 161 unreachabl
e, length 96
13:32:04.998232 IP 127.0.0.1.49131 > 127.0.0.1.631: UDP, length 0
13:32:04.998239 IP 127.0.0.1 > 127.0.0.1: ICMP 127.0.0.1 udp port 631 unreachabl
e, length 36
13:32:04.998248 IP 127.0.0.1.49131 > 127.0.0.1.137: UDP, length 50
13:32:06.099678 IP 127.0.0.1.49132 > 127.0.0.1.137: UDP, length 50
13:32:07.974441 IP 127.0.0.1 > 127.0.0.1: ICMP echo request, id 9726, seq 1, len
gth 64
13:32:07.974461 IP 127.0.0.1 > 127.0.0.1: ICMP echo reply, id 9726, seq 1, lengt
h 64
```

14. ábra: Példa a tcpdump használatára hálózati forgalom megfigyeléséhez.

Az alapértelmezett kimenetben található bejegyzések a következő formátumot követik: időbélyeg, hálózati protokoll, forrás cím és port, cél cím és port, felsőbb rétegbeli protokoll típusa, és protokolltól függően egyéb kiegészítő információk.

Az eszköz beépítetten nem támogatja a syslog alapú naplózást, azonban további konfigurációval lehetőség van ennek megvalósítására.

5.2.2. nftables naplózás

Az nftables a netfilter projekt részeként a Linux kernel hálózati csomag klasszifikációs alrendszere. Menedzseléséhez az nft parancssori segédprogram használható [18], melynek segítségével konfigurálhatjuk (15. ábra) a hálózati csomagok naplózását is.

```
$ sudo cat /etc/nftables.conf
#!/usr/sbin/nft -f
flush ruleset
table inet filter {
        chain input {
                type filter hook input priority 0; policy drop;
                ip6 version 6 drop
                log prefix "CSOMAG " drop
        }
        chain forward {
                type filter hook forward priority 0; policy drop;
        }
        chain output {
                type filter hook output priority 0; policy accept;
        }
}
$ sudo nft -c -f /etc/nftables.conf
$ sudo nft -f /etc/nftables.conf
```

15. ábra: Példa nftables naplózás konfigurációra.

A megadott szabályoknak megfelelő csomagokról így a kernel naplóban szöveges bejegyzések (16. ábra) jönnek létre.

```
$ sudo dmesg -W
[31178.639793] CSOMAG IN=lo OUT= MAC=00:00:00:00:00:00:00:00:00:00:00:00:08:00 S
RC=127.0.0.1 DST=127.0.0.1 LEN=28 TOS=0x00 PREC=0x00 TTL=57 ID=62325 PROTO=UDP S
PT=49540 DPT=631 LEN=8
[31179.640939] CSOMAG IN=lo OUT= MAC=00:00:00:00:00:00:00:00:00:00:00:00:08:00 S
RC=127.0.0.1 DST=127.0.0.1 LEN=28 TOS=0x00 PREC=0x00 TTL=51 ID=11113 PROTO=UDP S
PT=49541 DPT=631 LEN=8
[31180.675296] CSOMAG IN=lo OUT= MAC=00:00:00:00:00:00:00:00:00:00:00:00:08:00 S
RC=127.0.0.1 DST=127.0.0.1 LEN=60 TOS=0x00 PREC=0x00 TTL=64 ID=25603 DF PROTO=TC
P SPT=47342 DPT=80 WINDOW=65495 RES=0x00 SYN URGP=0
[31181.676527] CSOMAG IN=lo OUT= MAC=00:00:00:00:00:00:00:00:00:00:00:00:08:00 S
RC=127.0.0.1 DST=127.0.0.1 LEN=60 TOS=0x00 PREC=0x00 TTL=64 ID=54027 DF PROTO=TC
P SPT=47344 DPT=80 WINDOW=65495 RES=0x00 SYN URGP=0
[31182.695858] CSOMAG IN=lo OUT= MAC=00:00:00:00:00:00:00:00:00:00:00:00:08:00 S
RC=127.0.0.1 DST=127.0.0.1 LEN=84 TOS=0x00 PREC=0x00 TTL=64 ID=14896 DF PROTO=IC
MP TYPE=8 CODE=0 ID=5105 SEQ=1
```

16. ábra: Példa nftables naplóbejegyzésekre.

A naplóbejegyzésekben különböző szóközökkel elválasztott mezők találhatók, amelyek közül a legfontosabbak: forrás IP cím (SRC), protokoll (PROTO), cél port (DPT); ICMP esetén típus (TYPE) és kód (CODE).

A kernel naplóbejegyzések fogadása alapértelmezetten támogatott az rsyslog szoftverben.

6. LEHETSÉGES ADATTÁROLÁSI MEGOLDÁSOK

6.1. Elválasztott szövegfájl

Felmerülhet a naplófájlok valamilyen egyedi szempont szerint kialakított, mezőnként elválasztott szövegfájlokká, pl. CSV formátumra történő átalakítása. Ezek a formátumok általában még ember által is viszonylag könnyen értelmezhetőek, és már használhatóak bizonyos mértékű gépi feldolgozáshoz, viszont nagyobb mennyiségű, vagy összetett adatoknál nem praktikusak.

6.2. Relációs adatbázis

Egy jól megtervezett, relációs adatmodellre épülő adatbázis egyszerűen átlátható struktúrát eredményez. A modell alapja, hogy az azonos típusú egyedek táblákban (relációkban), tehát az egyedeket soroknak, a tulajdonságokat oszlopoknak megfeleltető adattípusban tárolódnak. A redundancia beszúrási-, módosítási-, és törlési anomáliákat eredményez, azonban a normálformákra épülő adatbázis tervezés lehetővé teszi ezek kiküszöbölését. [19]

A DB-Engines 2021. decemberi, 152 rendszert tartalmazó összehasonlítása [20] alapján a két legnépszerűbb ingyenesen használható relációs adatbáziskezelő rendszert vettem számba.

6.2.1. MySQL

A MySQL egy C/C++ nyelven fejlesztett, aktívan karbantartott, nyílt forráskódú relációs adatbáziskezelő rendszer. [21] Előnye az aránylag alacsony komplexitás és egyszerű üzembe helyezés. Hátránya, hogy az alapvető adattípusokon kívül nem támogat továbbiakat (pl. hálózati címek), tábla törlés (DROP TABLE) esetén nem támogatja a CASCADE (függőségeket is törölje) funkciót, illetve, hogy alapvetően alkalmasabb a nagy számú olvasást igénylő feladatokhoz. [22] [23]

6.2.2. PostgreSQL

A PostgreSQL egy C nyelven fejlesztett, aktívan karbantartott, nyílt forráskódú relációs adatbáziskezelő rendszer. Előnye bizonyos objektum-orientált koncepciók (objektum, osztály, öröklődés) támogatása, a beépített hálózati cím jellegű adattípusok (cidr, inet, macaddr, macaddr8) [24], DROP TABLE CASCADE funkció támogatása, valamint alkalmas nagyszámú írást és olvasást igénylő feladatokhoz is. [22] [23]

7. SZOFTVERKOMPONENSEK KIVÁLASZTÁSA

A vizsgált SSH honeypot megoldások közül az ssh-honeypotd alacsony interakciójú eszközt választottam, mivel a cowrie szoftvernél kevesebb funkciót támogat, ezáltal alacsonyabb is a komplexitása, ebből adódóan egyszerűbb a konfigurációja és kisebb a kompromittáció valószínűsége is. Bár mindkét megoldás támogatja a syslog alapú naplózást, a cowrie kimenete még a csak jelszó alapú autentikációt támogató kapcsolat esetén is több olyan adatot szolgáltat, amelyek a megoldás szempontjából nem relevánsak, ezek kezelése viszont a feldolgozás komplexitását is feleslegesen növelné.

A bejövő hálózati csomagok adatainak naplózásához az nftables naplózást választottam. A konfigurációs segédprogramot leszámítva nem igényel a felhasználói térben futó külön folyamatot. Az adott rendszeren legmagasabb jogosultsággal folyamatosan futó folyamatként is felfogható Linux kernel beépített megoldása, ebből adódóan elvárható a stabil működése. Fontos szempont továbbá, hogy míg az nftables naplózás a megfelelő (a rendszeren alapértelmezett rsyslog) naplókezelő szoftverrel integrált működést eleve lehetővé teszi, a tcpdump használata azonban várhatóan többlet fejlesztést és konfigurációt igényelne. A vizsgált adattárolási megoldások közül a PostgreSQL adatbáziskezelő rendszerrel megvalósított relációs adatbázist ítélem megfelelőnek a honeypot adatok tárolására. A csapda rendszerből származó adatok alkalmasak a megfelelően kialakított relációs adatbázis struktúrában való tárolásra. Az adatbáziskezelő rendszer melletti legfontosabb érv a hálózati cím jellegű adattípusok használatának beépített lehetősége, valamint az, hogy alkalmas a gyakori írást és olvasást igénylő feladatok megvalósítására.

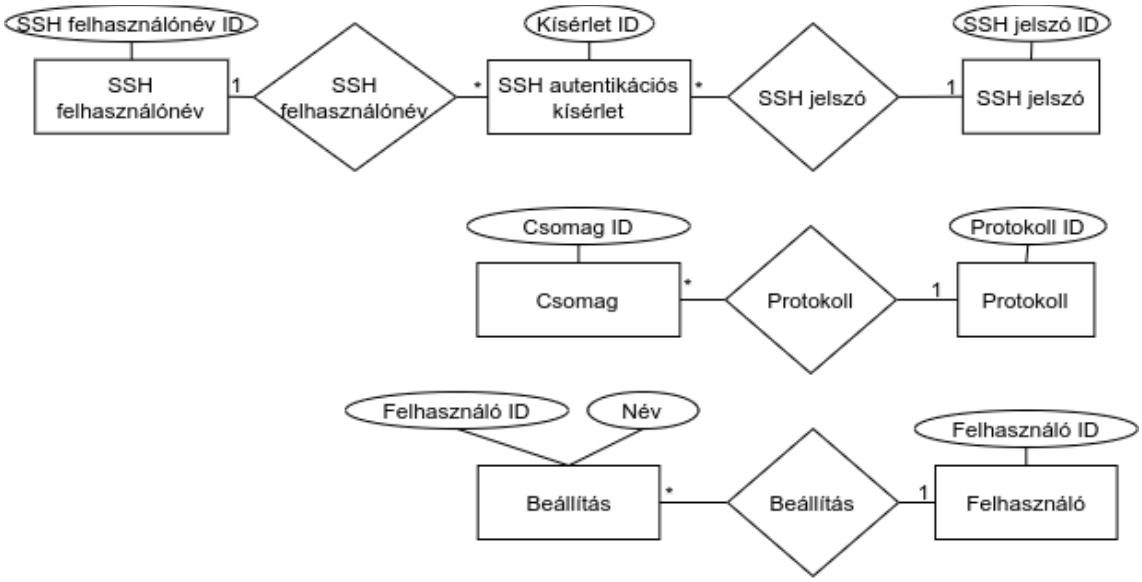
A megjelenítésért felelős webalkalmazást a Microsoft által fejlesztett ASP.NET Core környezetben fogom implementálni, mivel egy nyílt forráskódú, cross-platform keretrendszer, amely támogatja a tanulmányaim során megismert C# programnyelvet és a Visual Studio integrált fejlesztői környezetet. [25] A szoftver felépítése az MVC[19] architekturális tervezési mintát fogja követni, mivel az lehetővé teszi az alkalmazásban a szétválasztás elvének megvalósítását, és a használt keretrendszerhez is kifejezetten támogatott, webalkalmazások fejlesztéséhez javasolt módszer. [26] Az adatbázis elérését az Npgsql [27] ADO.NET Data Provider komponenssel fogom megvalósítani. Az autentikációt és autorizációt az ASP.NET Core Identity [28] API segítségével implementálom. A vizualizációk kliensoldali megjelenítéséhez a Chart.js [29] JavaScript könyvtárat használom, mivel egyszerűen konfigurálható, és többféle diagram formátumot is támogat. A webes felület kialakításakor lehetőség szerint reszponzív megjelenést szeretnék elérni, amihez felhasználom a Bootstrap [30] könyvtárat.

[19] Model-View-Controller.

8. ADATBÁZIS

8.1. Tervezés

A tervezés során meghatároztam a fontosabb egyedtípusokat. Az egyedtípusok kapcsolatait az alábbi egyed-kapcsolat diagram szemlélteti (17. ábra), amely alapján a későbbiekben létrehoztam a táblák szerkezetét.



17. ábra: Egyed-kapcsolat diagram.

8.2. Megvalósítás

A honeypot adatok részére a következő táblákat alakítottam ki (1-6. táblázatok).

Mezőnév	Típus	Megszorítás	Magyarázat
Id	BIGINT	GENERATED ALWAYS AS IDENTITY	SSH autentikációs kísérlet egyedi azonosítója.
		PRIMARY KEY	
Timestamp	TIMESTAMP WITH TIME ZONE	NOT NULL	Időbélyeg.
SrcIp	INET	NOT NULL	Forrás IPv4 cím.
SshUserId	BIGINT	NOT NULL	SSH felhasználónév egyedi azonosítója.
		FOREIGN KEY	
SshPassId	BIGINT	NOT NULL	SSH jelszó egyedi azonosítója.
		FOREIGN KEY	

1. táblázat: SshAuth tábla szerkezete.

Mezőnév	Típus	Megszorítás	Magyarázat
Id	BIGINT	GENERATED ALWAYS AS IDENTITY PRIMARY KEY	SSH felhasználónév egyedi azonosítója.
Username	VARCHAR	UNIQUE	SSH felhasználónév.

2. táblázat: SshUser tábla szerkezete.

Mezőnév	Típus	Megszorítás	Magyarázat
Id	BIGINT	GENERATED ALWAYS AS IDENTITY PRIMARY KEY	SSH jelszó egyedi azonosítója.
Password	VARCHAR	UNIQUE	SSH jelszó.

3. táblázat: SshPass tábla szerkezete.

Mezőnév	Típus	Megszorítás	Magyarázat
Id	BIGINT	GENERATED ALWAYS AS IDENTITY PRIMARY KEY	Csomag egyedi azonosítója.
Timestamp	TIMESTAMP WITH TIME ZONE	NOT NULL	Időbélyeg.
ProtocolId	SMALLINT	NOT NULL FOREIGN KEY	Felsőbb rétegbeli protokoll egyedi azonosítója.
SrcIp	INET	NOT NULL	Forrás IPv4 cím.
DstPort	INT	NOT NULL	Célport (TCP, UDP), vagy Típus és kód (ICMP).

4. táblázat: Packet tábla szerkezete.

Mezőnév	Típus	Megszorítás	Magyarázat
Id	SMALLINT	PRIMARY KEY	Felsőbb rétegbeli protokoll egyedi azonosítója.
Name	VARCHAR	UNIQUE NOT NULL	Rövid név.

5. táblázat: Protocol tábla szerkezete.

Az Identity API teljeskörű támogatásához további, a feladat megoldása szempontjából kevésbé lényeges adatszerkezetek létrehozása is szükségessé vált. Ezek részletesen nem kerülnek tárgyalásra, közülük csupán az AspNetUsers táblát említeném meg, amely az alkalmazásszintű felhasználók adatait tárolja, elsődleges kulcsa az Id mező, amely a felhasználók egyedi azonosítója.

A vizualizációk beállításait tárolja a VizSettings tábla (6. táblázat), amelyben a UserId mező idegen kulcsként az AspNetUsers tábla Id mezőjére hivatkozik, ezzel összekapcsolva a beállításokat a felhasználókkal.

Mezőnév	Típus	Megszorítás	Magyarázat
UserId	VARCHAR	FOREIGN KEY	Felhasználó egyedi azonosítója.
VizName	VARCHAR	PRIMARY KEY	Vizualizáció egyedi azonosítója.
Enabled	BOOLEAN	NOT NULL	Láthatóság.
IsTbl	BOOLEAN	NOT NULL	Táblázat forma.
N	SMALLINT		N érték, vagy NULL.
DateFrom	TIMESTAMP WITH TIME ZONE	NOT NULL	Időszak kezdete, vagy időszak.
DateTo	TIMESTAMP WITH TIME ZONE		Időszak vége, vagy NULL.
Priority	SMALLINT		Megjelenítés sorszáma.

6. táblázat: VizSettings tábla szerkezete.

Az adatok eléréséhez két adatbázis szintű szerepkört hoztam létre, egyet az adatbetöltő komponens adatbáziskapcsolatához (syslog), és egy másikat a webalkalmazás adatbáziskapcsolatához (hvizweb). A legfontosabb elvárások, hogy a betöltő komponens nem törölheti, vagy írhatja felül a honeypot adatokat, valamint a felhasználói adatokhoz semmilyen formában sem férhet hozzá (7. táblázat); továbbá a webalkalmazás is csak olvashatja a honeypot adatokat. A hvizweb szerepkör részére ezen kívül az Identity API használata miatt létrehozott táblákhoz, valamint a vizualizáció beállításokat tároló táblához további jogosultságok szükségesek (8. táblázat). Fontos megjegyezni, hogy a sémahasználat jellegű hozzáférés (9. táblázat) nem jelent alapértelmezetten a sémán belüli objektumokra vonatkozóan semmilyen további jogosultságot, csupán azok láthatóságát biztosítja.

	syslog				hvizweb			
	I	S	U	D	I	S	U	D
SshUser	✓	✓	✗	✗	✗	✓	✗	✗
SshPass	✓	✓	✗	✗	✗	✓	✗	✗
SshAuth	✓	✓	✗	✗	✗	✓	✗	✗
Packet	✓	✓	✗	✗	✗	✓	✗	✗
Protocol	✓	✓	✗	✗	✗	✓	✗	✗

7. táblázat: Szerepkörök jogosultságai a honeypot táblákon.[20]

	syslog				hvizweb			
	I	S	U	D	I	S	U	D
AspNetRoles	✗	✗	✗	✗	✓	✓	✓	✓
AspNetUsers	✗	✗	✗	✗	✓	✓	✓	✓
AspNetRoleClaims	✗	✗	✗	✗	✓	✓	✓	✓
AspNetUserClaims	✗	✗	✗	✗	✓	✓	✓	✓
AspNetUserLogins	✗	✗	✗	✗	✓	✓	✓	✓
AspNetUserRoles	✗	✗	✗	✗	✓	✓	✓	✓
AspNetUserTokens	✗	✗	✗	✗	✓	✓	✓	✓
VizSettings	✗	✗	✗	✗	✓	✓	✓	✓

8. táblázat: Szerepkörök jogosultságai a webalkalmazás táblákon.[20]

	syslog	hvizweb
hviz	USAGE	USAGE
SCHEMA	✓	✓

9. táblázat: Szerepkörök jogosultságai a séma objektumon.

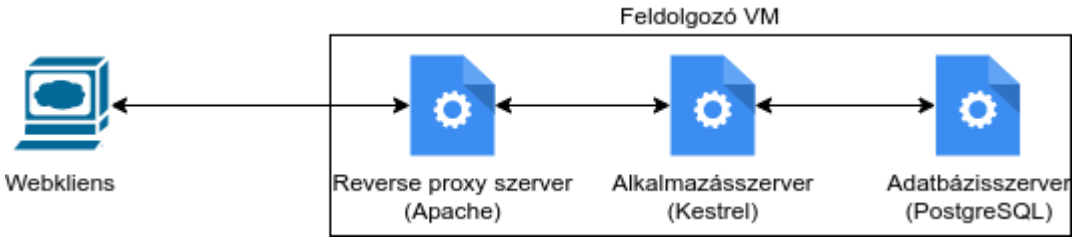
Az adattáblákon túl létrehoztam a beérkező syslog bejegyzések ellenőrzését, és feldarabolását végző PL/pgSQL tárolt eljárást[21] is, amely ugyan látható a teljes sémán belül, viszont alapértelmezés szerint SECURITY INVOKER paraméterrel jön létre, tehát mindig a meghívó szerepkör jogosultságaival kerül végrehajtásra, így az csak a saját adatbázisobjektumaihoz férhet hozzá.

A PostgreSQL adatbáziskezelő rendszer legfontosabb beállításait tartalmazó alapvető konfiguráció, az adatbázis struktúra létrehozásáért felelős kód, valamint a tárolt eljárás megvalósítása megtalálhatóak a 15.1-15.3 mellékletekben.

[20] INSERT INTO, SELECT FROM, UPDATE, DELETE FROM.
[21] hviz.parse_log(varchar, varchar)

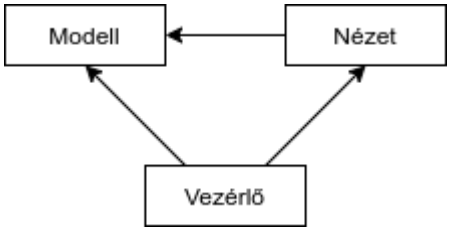
9. MEGJELENÍTŐ ALKALMAZÁS

9.1. Rendszerterv



18. ábra: A webalkalmazás architektúrája.

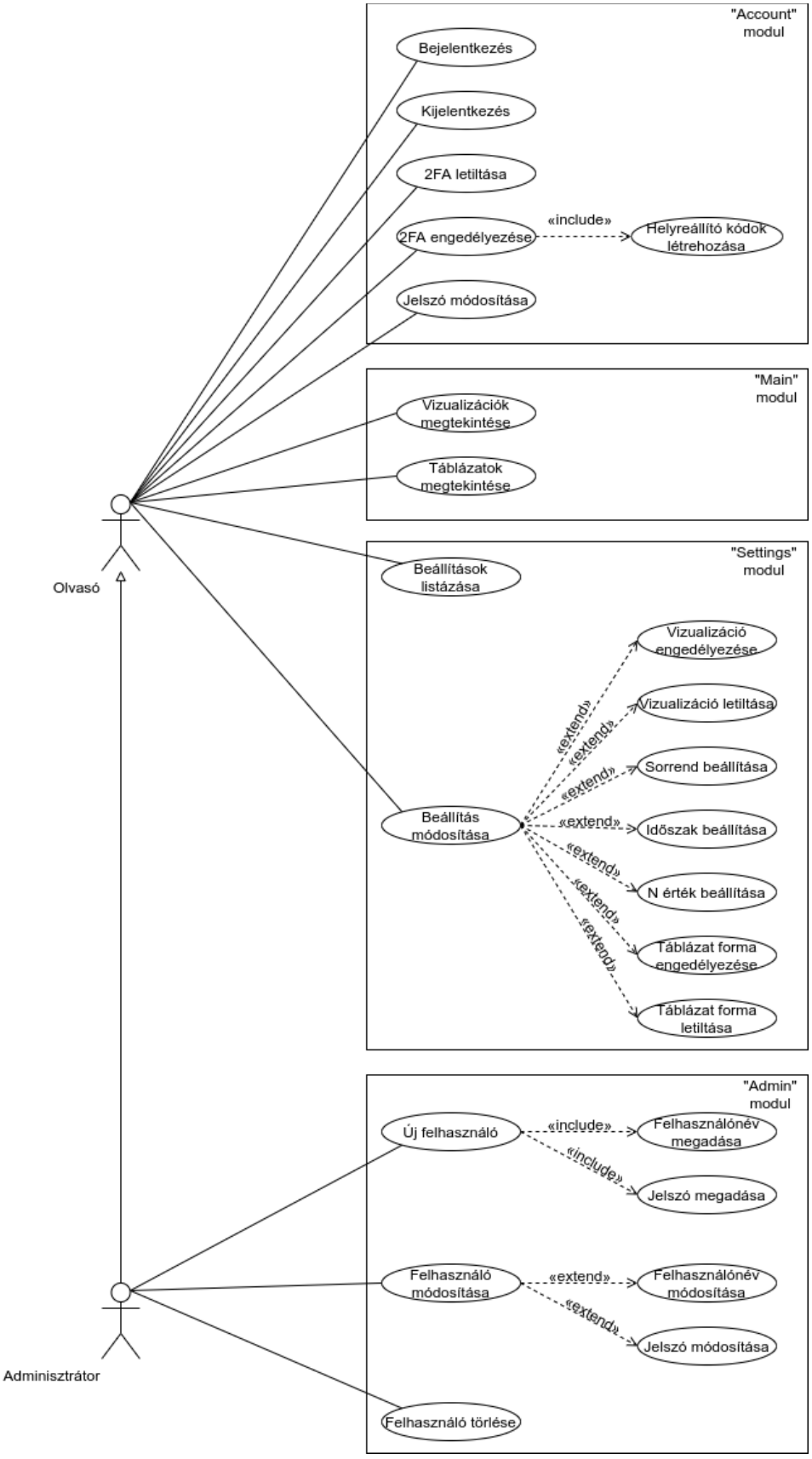
Az adatok vizuális megjelenítéséért felelős webalkalmazás a 18. ábrán látható architektúrában helyezkedik el. A webkliens szerepét a felhasználó webböngészője valósítja meg. Az alkalmazásszerver futtatja a webalkalmazást. A kliens és az alkalmazás közötti kapcsolatot egy reverse proxy szerver biztosítja, amely a kliens felől érkező HTTP kéréseket fogadja és továbbítja az alkalmazásszervernek, valamint az alkalmazás felől érkező válaszokat továbbítja a kliens részére [31]. Az alkalmazás a megjelenített adatokat az adatbázisszervertől kérdezi le.



19. ábra: MVC architektúra. [26]

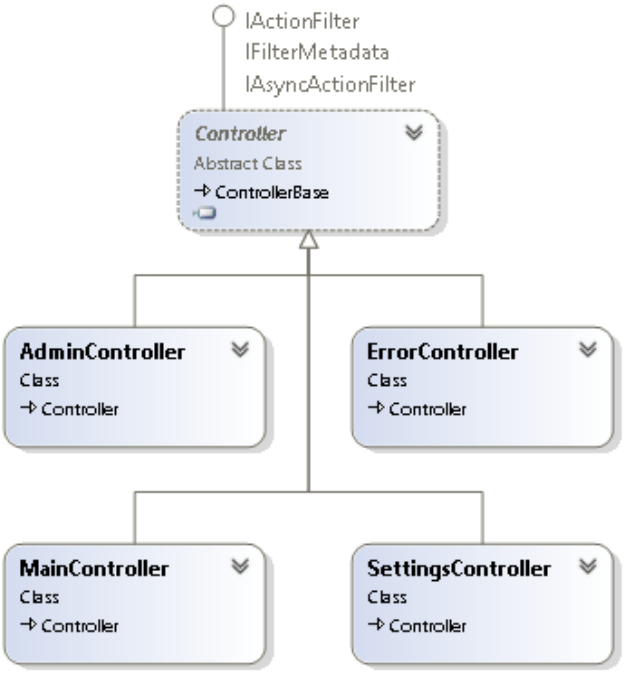
A webalkalmazás egy rétegből áll, és az MVC (19. ábra) architekturális tervezési mintát implementálja, amely a Modell, Nézet és Vezérlő szerepeket különíti el. A felhasználói kérések a Vezérlőhöz kerülnek, amely megvalósítja a felhasználói műveleteket, illetve a lekérdezéseket, majd továbbítja a felhasználó felé a megfelelő Nézetet, amelynek átadja a megfelelő Modell formájában a megjeleníteni kívánt adatokat. [26]

Az alkalmazásban található tartozó két felhasználói szerepkör, az általuk használt négy fő modul, illetve a modulok által elérhető funkcionalitás a szoftver használati eset diagramján (20. ábra) látható. Az Account modul a bejelentkezéshez és a felhasználók bejelentkezési beállításainak módosításához szükséges, a Main modul a honeypot adatok megjelenítését teszi lehetővé, a Settings modul a vizualizációk beállításainak kezelését valósítja meg, illetve az Admin modul a felhasználók és bejelentkezési adataik kezelését biztosítja.

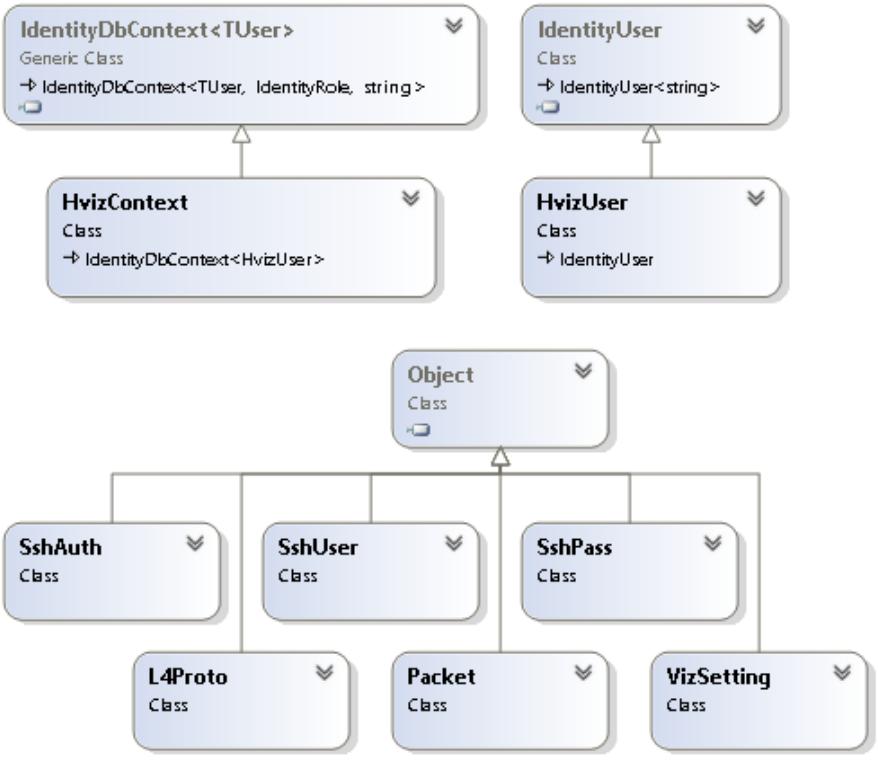


20. ábra: A webalkalmazás használati eset diagramja.

9.2. Implementáció



21. ábra: Vezérlő osztályok.



22. ábra: Entitás osztályok.

23. ábra: Modell osztályok.

A hitelesítést és engedélyezést az Identity API segítségével implementáltam. A bejelentkezési felület, valamint a bejelentkezési beállításokat kezelő felület megvalósításához a keretrendszer beépített Razor Pages alapú sablonjait használtam fel. A jól elkülöníthető szerepköröknek és funkcióknak köszönhetően az autorizációt a vezérlő osztályokban (21. ábra) a felhasználóhoz rendelt szerepkör (Administrator, vagy Reader) vizsgálatával[22] valósítottam meg.

Az adatbázis entitásoknak megfelelő entitás osztályokat, valamint a megfelelő DbContext osztályt (22. ábra), ami az előbbiek adatbázistáblákhoz rendelését valósítja meg, Entity Framework Core segítségével hoztam létre, ezek a Hviz.Data névtéren belül érhetők el. A Main modulhoz tartozó üzleti logikát (tehát a lekérdezések megvalósítását) egyszerűsége miatt a Hviz.Controllers.MainController vezérlő osztály Index metódusán belül implementáltam LINQ segítségével.

Az egyes vizualizációkhoz tartozó nézeteket Partial View-ként hoztam létre, amelyek a Main Index nézetét egészítik ki. A kialakított modell osztályok (23. ábra) a VizViewModel osztály leszármazottai.

A Settings modulhoz tartozó CRUD műveleteket a Hviz.Controllers.SettingsController vezérlő osztályban valósítottam meg, amely a Hviz.Data.VizSetting entitást használja. Új felhasználó esetén nem találhatóak vizualizáció beállítások az adatbázisban, ezért az alapértelmezett beállítások létrehozása is itt történik, az Index metódusban.

A felhasználók alapszintű adminisztrációját a Hviz.Controllers.AdminController vezérlő osztály megfelelő metódusai (Index, NewUser, ChangeUsername, ChangePassword, DeleteUser) teszik lehetővé.

A webalkalmazás forráskódja megtalálható a fájlmellékletben (Hviz/).

[22] [Authorize(Roles="…")] annotáció a vezérlő osztályon.

10. TESZTELÉS

A rendszer működéséhez a következő alapértelmezett adatokat szükséges az adatbázisban előre tárolni: támogatott protokollok, alkalmazás szintű szerepkörök, alapértelmezett belépési adatok[23], és a felhasználó szerepkörhöz rendelése. A szükséges rekordokat létrehozó kód megtalálható a 15.4. mellékletben.

VM	Szoftver	Verzió
Csapda VM	ssh-honeypotd	2.1.2
Csapda VM, Feldolgozó VM	Linux kernel	#1 SMP Debian 5.10.113-1 (2022-04-29)
	OpenSSH	8.4p1 Debian-5
	OpenSSL	1.1.1n 15 Mar 2022
	rsyslogd	8.2102.0 (aka 2021.02)
Feldolgozó VM	PostgreSQL	13.5 (Debian 13.5-0+deb11u1)
	Microsoft .NET SDK	6.0.300
Teszt VM	Microsoft Windows	8.1
	Mozilla Firefox	100.0
	Google Chrome	101.0.4951.67
	Microsoft Edge	101.0.1210.39

10. táblázat: Teszteléshez használt szoftverek.

A 4. fejezetben tárgyalt, korábban kialakított tesztkörnyezetet kiegészítettem egy, a megjelenítés teszteléséhez szükséges további virtuális géppel, majd telepítettem és konfiguráltam a webalkalmazást, illetve a rendszer működéséhez szükséges további szoftvereket (10. táblázat). A feladat szempontjából legfontosabb beállításokat tartalmazó alapvető konfigurációk megtalálhatóak a 15.5. mellékletben.

A tesztadatok begyűjtését két körben, először 2022. május 11-én 15:40 és 20:40 között, majd másodjára 2022. május 12. 06:00 és 10:00 között végeztem.[24] A gyűjtött adatok megtalálhatóak a fájlmellékletben (tesztadatok.sql).

A következő lépésben az alkalmazás manuális tesztelését végeztem, a követelményspecifikációban meghatározott szempontok szerint, amelynek eredménye a 11-12. táblázatokban található.

[23] Felhasználónév: „Administrator”, jelszó: „Administrator”.
[24] Közép-európai nyári idő szerint.

Teszteset	V[25]	K[25]	T[2525]
Vizualizációk megjelenítése (bejelentkezés nélkül)	✗	✗	✓
Bejelentkezés (jelszó)	✓	✓	✓
2FA konfigurálása	✓	✓	✓
Bejelentkezés (jelszó és 2FA)	✓	✓	✓
Bejelentkezés (jelszó és helyreállító kód)	✓	✓	✓
Vizualizációk megjelenítése konfiguráció nélkül	✗	✗	✓
Megjelenítés konfigurálása	✓	✓	✓
Engedélyezett vizualizációk megjelenítése	✓	✓	✓
Letiltott vizualizációk megjelenítése	✗	✗	✓
Prioritás hozzárendelése	✓	✓	✓
Megjelenítés sorrendben	✓	✓	✓
Megjelenített időszak beállítása	✓	✓	✓
N érték beállítása („N leggyakoribb” típus)	✓	✓	✓
Maximum N megjelenő adat („N leggyakoribb” típus)	✓	✓	✓
N érték beállítása (egyéb típus)	✓	✓	✓
Maximum N megjelenő adat (egyéb típus)	✗	✗	✓
Megjelenítés táblázatos formában	✓	✓	✓
SSH kísérletek (N leggyakoribb forrás IPv4 cím)	✓	✓	✓
SSH kísérletek (N leggyakoribb felhasználó-jelszó)	✓	✓	✓
SSH kísérletek (N leggyakoribb felhasználónév)	✓	✓	✓
SSH kísérletek (N leggyakoribb jelszó)	✓	✓	✓
SSH kísérletek (hónap napjai)	✓	✓	✓
SSH kísérletek (év hónapjai)	✓	✓	✓
Csomagok (N leggyakoribb forrás IPv4 cím)	✓	✓	✓
Csomagok (felsőbb rétegbeli protokoll)	✓	✓	✓
Csomagok (hónap napjai)	✓	✓	✓
Csomagok (év hónapjai)	✓	✓	✓
TCP szegmensek (N leggyakoribb cél port)	✓	✓	✓
UDP datagramok (N leggyakoribb cél port)	✓	✓	✓
ICMP csomagok (N leggyakoribb típus-kód)	✓	✓	✓

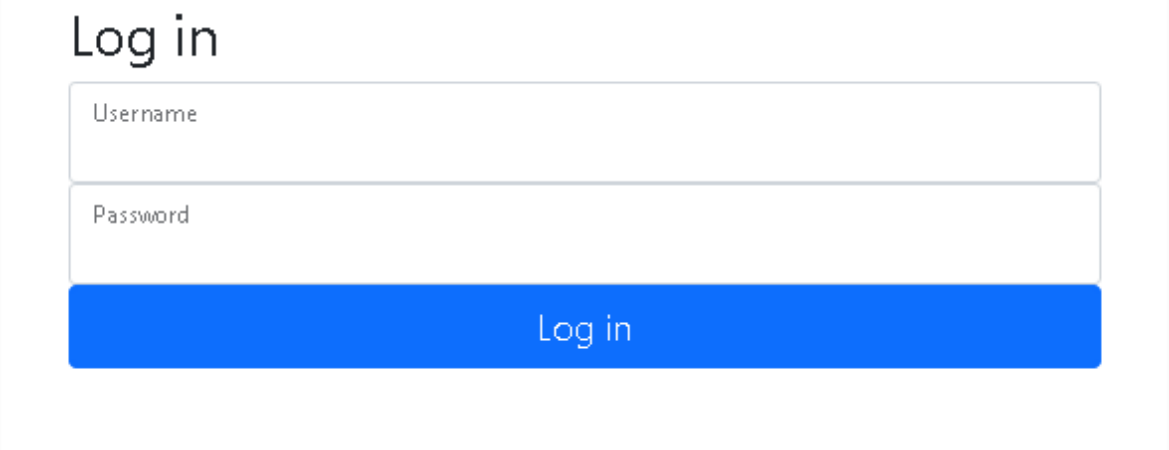
11. táblázat: Manuális tesztelés eredményei (olvasói funkciók).

[25] Várt eredmény, kapott eredmény, teszt eredménye.

Teszteset	V[26]	K[26]	T[26]
Felhasználók megjelenítése (bejelentkezés nélkül)	✗	✗	✓
Felhasználók megjelenítése (adminisztrátor)	✓	✓	✓
Felhasználók megjelenítése (olvasó)	✗	✗	✓
Felhasználó létrehozása (bejelentkezés nélkül)	✗	✗	✓
Felhasználó létrehozása (adminisztrátor)	✓	✓	✓
Felhasználó létrehozása (olvasó)	✗	✗	✓
Felhasználó nevének módosítása (bejelentkezés nélkül)	✗	✗	✓
Felhasználó nevének módosítása (adminisztrátor)	✓	✓	✓
Más felhasználó nevének módosítása (olvasó)	✗	✗	✓
Felhasználó jelszavának módosítása (bejelentkezés nélkül)	✗	✗	✓
Felhasználó jelszavának módosítása (adminisztrátor)	✓	✓	✓
Más felhasználó jelszavának módosítása (olvasó)	✗	✗	✓
Felhasználó törlése (bejelentkezés nélkül)	✗	✗	✓
Felhasználó törlése (adminisztrátor)	✓	✓	✓
Felhasználó törlése (olvasó)	✗	✗	✓

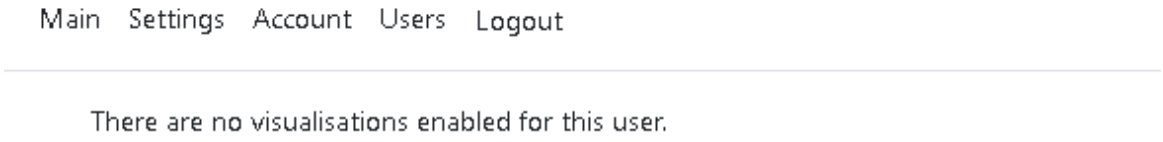
12. táblázat: Manuális tesztelés eredményei (adminisztrátori funkciók)

A megjelenés minden vizsgált böngészőben közel azonosnak bizonyult, funkcionális eltérést nem találtam, az oldal minimum 640*480 felbontás mellett volt praktikusan használható. A manuális tesztelés végeztével megállapítottam, hogy a webalkalmazás a követelményeknek megfelelően működik (24-32. ábrák).

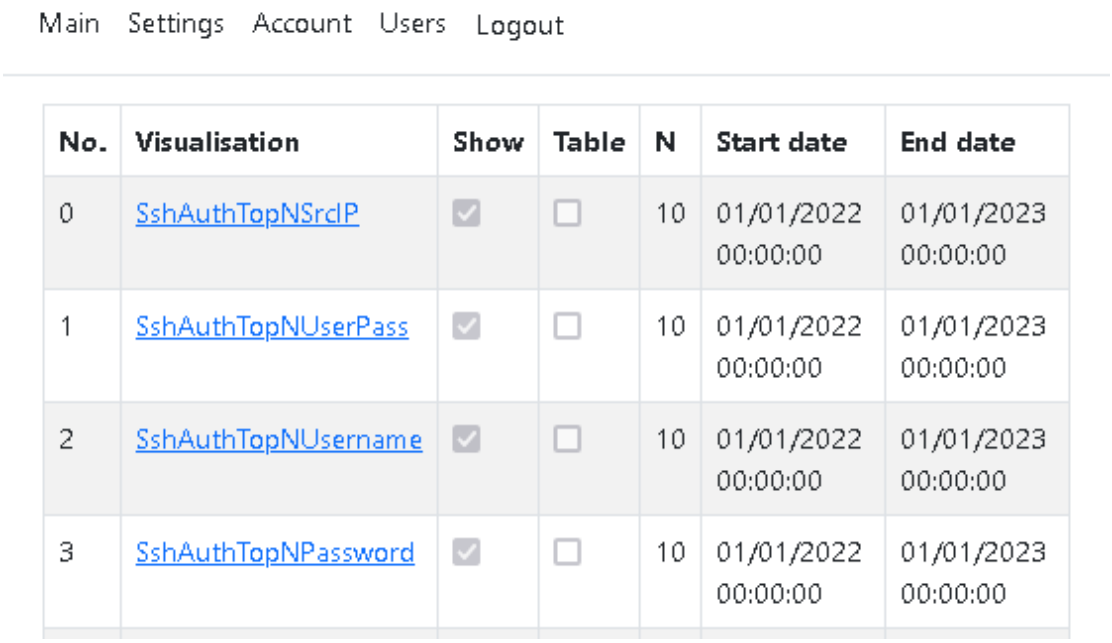


24. ábra: Bejelentkezési felület.

[26] Várt eredmény, kapott eredmény, teszt eredménye.

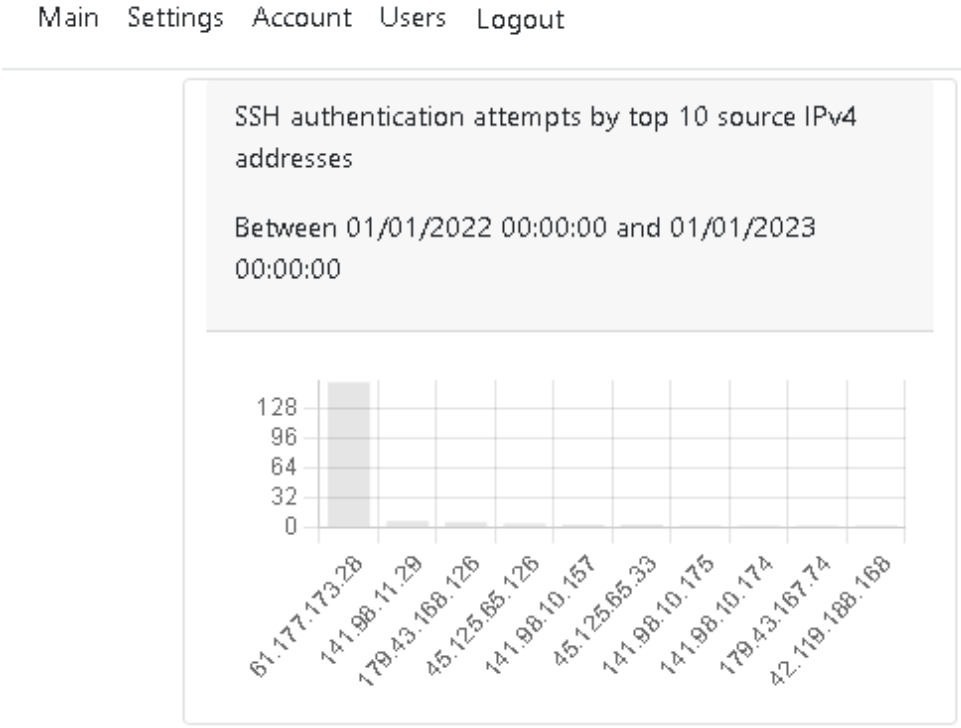


25. ábra: Alapértelmezett kezdőoldal.

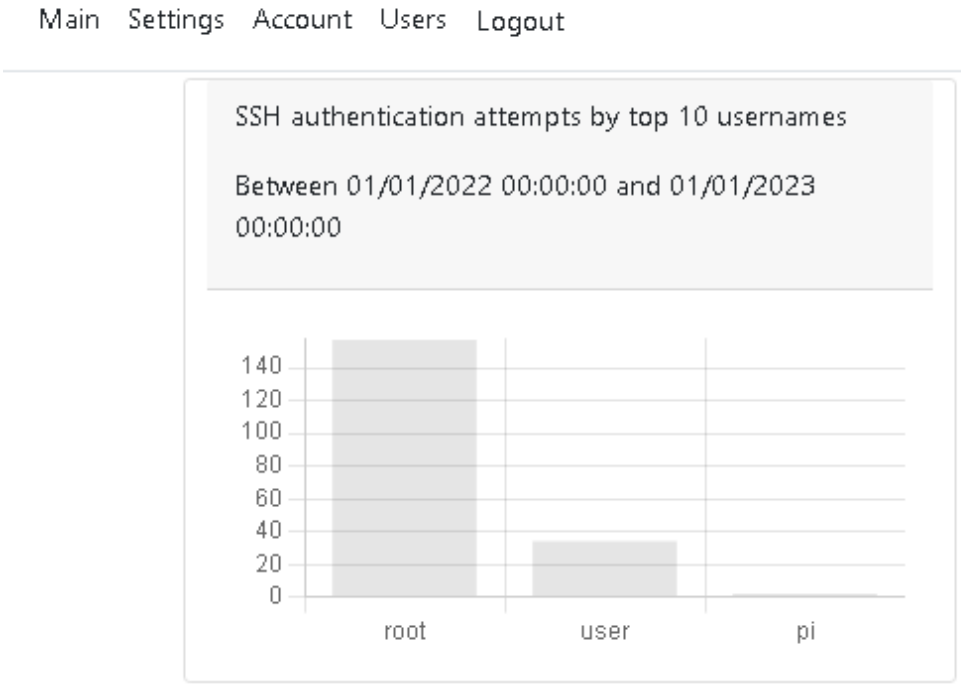


No.	Visualisation	Show	Table	N	Start date	End date
0	SshAuthTopNSrcIP	☑	☐	10	01/01/2022 00:00:00	01/01/2023 00:00:00
1	SshAuthTopNUserPass	☑	☐	10	01/01/2022 00:00:00	01/01/2023 00:00:00
2	SshAuthTopNUsername	☑	☐	10	01/01/2022 00:00:00	01/01/2023 00:00:00
3	SshAuthTopNPassword	☑	☐	10	01/01/2022 00:00:00	01/01/2023 00:00:00

26. ábra: Beállítások oldal.



27. ábra: Kísérletek száma a leggyakoribb forráscímek szerint.

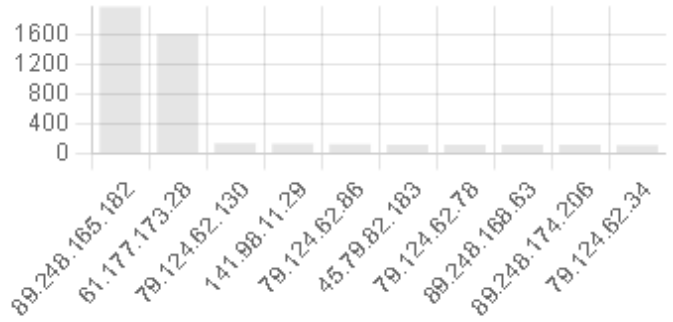


28. ábra: Kísérletek száma a leggyakoribb felhasználónevek szerint.

Main Settings Account Users Logout

Packets by top 10 source IPv4 addresses

Between 01/01/2022 00:00:00 and 01/01/2023 00:00:00



29. ábra: Csomagok száma a leggyakoribb forráscímek szerint.

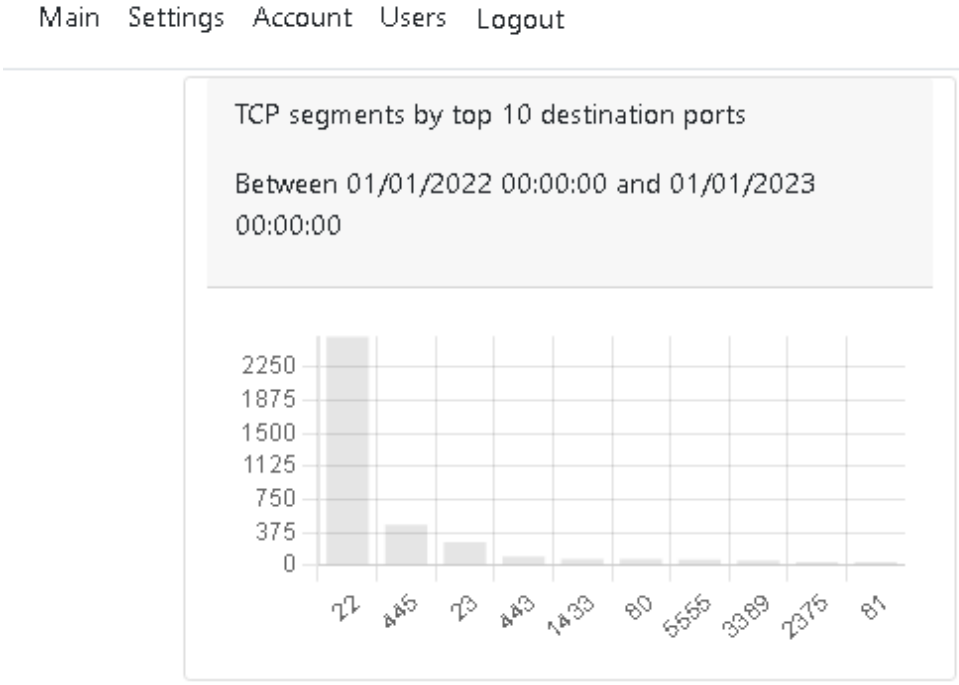
Main Settings Account Users Logout

Packets by protocols

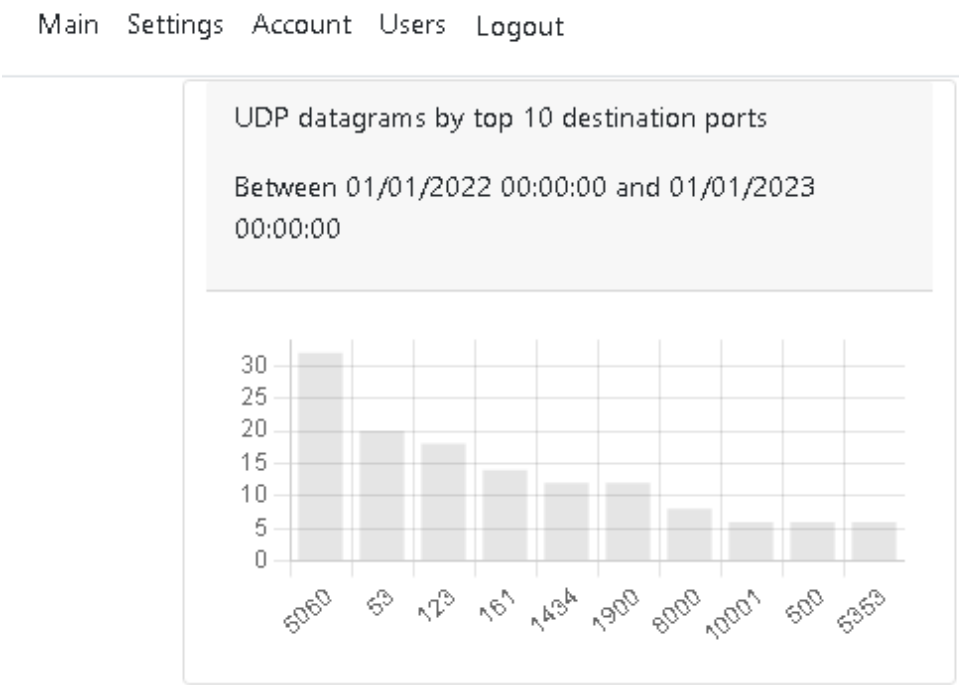
Between 01/01/2022 00:00:00 and 01/01/2023 00:00:00

Protocol	Packets
ICMP	169
UDP	247
TCP	9087

30. ábra: Csomagok száma protokollok szerint.



31. ábra: TCP szegmensek száma a leggyakoribb célportok szerint.



32. ábra: UDP datagramok száma a leggyakoribb célportok szerint.

11. TOVÁBBFEJLESZTÉSI LEHETŐSÉGEK

Az elkészült szoftverben a jövőben szeretném megvalósítani az IPv6 címek használatának lehetőségét, valamint az eddig támogatott szállítási rétegbeli protokollokon kívül újak támogatását is.

További lehetőség az elkészült webalkalmazás behatóbb vizsgálata biztonsági szempontból, tehát az esetleges sérülékenységek (pl. SQL Injection, Cross-Site Scripting, Mass Assignment/Overposting) feltárása, továbbá ezek javítása, mindenképpen a már meglévő funkcionalitás megtartásával.

A fentieken kívül fontosnak tartom további alkalmazás szintű (pl. DNS) alacsony interakciójú honeypot eszközök adatforrásként történő integrálásának lehetőségét is, így szükség lehet további adatszerkezetek kialakítására. Különböző IP címeken elérhető, de azonos szolgáltatást emuláló rendszerek hozzákapcsolásának támogatásához az adatszerkezetet ki kell egészíteni a forrásrendszerek azonosításához szükséges mezőkkel.

A dolgozat során felhasznált nyílt forrású honeypot eszköz licensze megengedi a forráskód módosítását, így lehetőség van annak új funkciókkal történő kibővítésére, továbbfejlesztésére is, például publikus kulcs alapú autentikáció emulálásával, vagy ismerten sebezhető algoritmusok (pl. SHA-1-et használó Diffie-Hellman group kulcscsere algoritmus) támogatásának emulálásával, amelyek további, kifinomultabb támadási kísérletekre ösztönözhetik a támadókat, ezzel további adatokat szolgáltatva.

12. ÖSSZEFOGLALÁS

Szakdolgozatom fő motivációját a begyűjtött honeypot adatoknak, és a feladat megoldása során szerzett tapasztalatoknak a későbbiekben hasonló környezetben (VPS szolgáltatónál) üzemeltetni tervezett, személyes, vagy üzleti célokat szolgáló, éles szolgáltatásokat nyújtó rendszerek esetén való felhasználásának a lehetősége adta. A kapcsolódó feladatokat a 2021/2022/1. és 2021/2022/2. szemeszter folyamán végeztem:

Megvizsgáltam a honeypot rendszerek általános elméleti hátterét, a kapcsolódó fogalmakat, az üzemeltetésük során felmerülő legfontosabb kérdéseket, beleértve a különböző honeypot típusokat (a működésük jellege, feladatuk, és interakciójuk szintje szerint), valamint a virtualizáció, és a hálózatban való elhelyezés lehetőségeit. Meghatároztam egy honeypot adatok vizuális megjelenítését végző alkalmazásra vonatkozó követelményeket. Megterveztem és kialakítottam a rendszer üzemeltetéséhez szükséges tesztkörnyezet, tehát az adatok begyűjtéséhez és továbbításához szükséges környezet architektúráját. Megvizsgáltam és összehasonlítottam a felhasználható, adatforrásként funkcionáló honeypot eszközöket, amelyek közül egy nyílt forrású alacsony interakciójú SSH honeypot megvalósítást, valamint a hálózati csomagok adatainak gyűjtéséhez a Linux kernel beépített naplózását használtam fel. Megvizsgáltam és összehasonlítottam a lehetséges adattárolási lehetőségeket, amelyek közül egy PostgreSQL alapú relációs adatbázist választottam a feladat megoldásához. Megterveztem és létrehoztam a honeypot adatok, valamint a megjelenítő alkalmazás konfigurációjának tárolásához szükséges adatbázis struktúrát. Megterveztem a vizuális megjelenítést megvalósító alkalmazás felépítését, amely az MVC architekturális tervezési mintát követi. A fejlesztési folyamat során C# nyelven implementáltam a megtervezett szoftvert ASP.NET Core környezetben. A megvalósított tesztkörnyezetbe telepítettem, majd a tesztkörnyezetből származó valós adatokkal sikeresen teszteltem a fejlesztett alkalmazást. Meghatároztam a rendszer jövőbeni továbbfejlesztésének lehetséges irányait.

13. SUMMARY

The main motivation behind my thesis is that the collected honeypot data and the gained experience may be applicable during the operation and administration of systems that are planned to provide services for personal or business usage deployed in a similar network environment (i.e., VPS hosting). The following tasks have been completed during the 2021/2022/1 and 2021/2022/2 semesters:

The general theoretical background of honeypot systems, the related terminology, the most important topics regarding their operation, including: the types of honeypots (differentiated by the characteristics of their operation, their role, and the level of interaction), and the different options for virtualisation and deployment were examined. The requirements of an application which is capable of visualising specific honeypot data were defined. The architecture of the test environment which can collect and forward the specified honeypot data was designed and implemented. Different honeypot tools were examined and compared. An open-source low-interaction SSH honeypot application was chosen as a data source of password based SSH authentication attempts. The capability of Linux kernel to log basic packet headers was chosen as a source of network traffic data. Various options of data storage were examined and compared. A PostgreSQL-based relational database was chosen for storage. The database structure for storing honeypot data and the configuration of the visualisation application was designed and implemented. The architecture of the visualisation application was designed utilising the MVC pattern. The application was implemented in C# and the ASP.NET Core environment. The application was installed into the test environment. The manual tests of the application were successful. Further possible development tasks of the system were defined.

14. IRODALOMJEGYZÉK

[1] Alert Logic Critical Watch Report. SMB Threatscape 2019, (https://www.alertlogic.com/assets/critcal-watch-report/CWR19_ExecReport.pdf), utoljára megtekintve: 2022.05.13.

[2] Eseményészlelés | Nemzeti Kibervédelmi Intézet, (https://nki.gov.hu/szolgaltatasok/tartalom/esemenyeszleles/), utoljára megtekintve: 2022.05.13.

[3] Spitzner, L.: Honeypots: tracking hackers. *Addison-Wesley*, 2003.

[4] Grimes, R. A.: Honeypots for Windows. *Apress*, 2006.

[5] Provos, N., Holz, T.: Virtual honeypots: from botnet tracking to intrusion detection. *Addison-Wesley*, 2007.

[6] Symanovich, S.: What is a honeypot? How it can lure cyberattackers, (https://us.norton.com/internetsecurity-iot-what-is-a-honeypot.html), utoljára megtekintve: 2022.05.13.

[7] Szabó, A.: Preventív hálózatvédelmi rendszerek alkalmazási lehetőségei a támadások detektálására, valamint a módszerek elemzésére I. rész. *Hadmérnök*, VI. évfolyam 4. szám, 2011, 239-249. old.

[8] Szabó, A.: Preventív hálózatvédelmi rendszerek alkalmazási lehetőségei a támadások detektálására, valamint a módszerek elemzésére II. rész. *Hadmérnök*, VII. évfolyam 2. szám, 2012, 312-334. old.

[9] Gyurák, G.: Informatikabiztonság II. *Pécsi Tudományegyetem*, 2015.

[10] Rountree, D., Castrillo, I.: The basics of cloud computing: Understanding the fundamentals of cloud computing in theory and practice. *Syngress*, 2013.

[11] Brown, S., Lam, R., Prasad, S., Ramasubramanian, S., Slauson, J.: Honeypots in the Cloud. *University of Wisconsin-Madison*, 2012.

[12] Sparenberg, K.: Top 10 Log Sources You Should Monitor, (https://www.dnsstuff.com/top-10-log-sources-you-should-monitor), utoljára megtekintve: 2022.05.13.

[13] Rsyslog Doc Documentation, (https://readthedocs.org/projects/rsyslog/downloads/pdf/latest/), utoljára megtekintve: 2022.05.13.

[14] sjinks/ssh-honeypotd: A low-interaction SSH honeypot written in C,
(https://github.com/sjinks/ssh-honeypotd),
utoljára megtekintve: 2022.05.13.

[15] cowrie/cowrie: Cowrie SSH/Telnet Honeypot,
(https://github.com/cowrie/cowrie),
utoljára megtekintve: 2022.05.13.

[16] cowrie dokumentáció,
(https://cowrie.readthedocs.io/_/downloads/en/latest/pdf/),
utoljára megtekinve: 2022.05.13.

[17] Home | TCPDUMP & LIBPCAP,
(https://www.tcpdump.org),
utoljára megtekintve: 2022.05.13.

[18] netfilter/iptables project homepage - The netfilter.org "nftables" project,
(https://netfilter.org/projects/nftables/),
utoljára megtekintve: 2022.05.13.

[19] Szabó, B.: Adatbázisfejlesztés és üzemeltetés I. *EKF Líceum Kiadó*, 2014.

[20] DB-Engines Ranking – popularity ranking of relational DBMS,
(https://db-engines.com/en/ranking/relational+dbms),
utoljára megtekintve: 2021.12.10.

[21] mysql/mysql-server: MySQL Server,
(https://github.com/mysql/mysql-server),
utoljára megtekintve: 2022.05.13.

[22] Wolfe, M.: MySQL vs PostgreSQL. Comparing the Relational Database Management System to the Object-Relational Database Management System.
(https://towardsdatascience.com/mysql-vs-postgresql-3d48891452a),
utoljára megtekintve: 2022.05.13.

[23] Chen, B.: PostgreSQL vs. MySQL: What You Need to Know,
(https://fivetran.com/blog/postgresql-vs-mysql),
utoljára megtekintve: 2022.05.13.

[24] PostgreSQL: Documentation: 14: 8.9. Network Address Types
(https://www.postgresql.org/docs/current/datatype-net-types.html),
utoljára megtekintve: 2021.12.10.

[25] Overview of ASP.NET Core | Microsoft Docs,
(https://docs.microsoft.com/en-us/aspnet/core/introduction-to-aspnet-core?view=aspnetcore-6.0),
utoljára megtekintve: 2022.05.13.

[26] Overview of ASP.NET Core MVC | Microsoft Docs,
(https://docs.microsoft.com/en-us/aspnet/core/mvc/overview?view=aspnetcore-6.0),
utoljára megtekintve: 2022.05.13.

[27] Npgsql - .NET Access to PostgreSQL | Npgsql Documentation,
(https://www.npgsql.org/index.html),
utoljára megtekintve: 2022.05.13.

[28] Introduction to Identity on ASP.NET Core | Microsoft Docs,
(https://docs.microsoft.com/en-us/aspnet/core/security/authentication/identity?view=aspnetcore-6.0),
utoljára megtekintve: 2022.05.13.

[29] Chart.js | Open source HTML5 Charts for your website,
(https://www.chartjs.org),
utoljára megtekintve: 2022.05.13.

[30] Bootstrap · The most popular HTML, CSS, and JS library in the world.,
(https://getbootstrap.com),
utoljára megtekintve: 2022.05.13.

[31] Host ASP.NET Core on Linux with Apache | Microsoft Docs
(https://docs.microsoft.com/en-us/aspnet/core/host-and-deploy/linux-apache?view=aspnetcore-6.0),
utoljára megtekintve: 2022.05.09.

15. MELLÉKLETEK

15.1. melléklet: PostgreSQL konfiguráció

```
# /etc/postgresql/13/main/postgresql.conf

data_directory = '/var/lib/postgresql/13/main'
hba_file = '/etc/postgresql/13/main/pg_hba.conf'
external_pid_file = '/var/run/postgresql/13-main.pid'
listen_addresses = '127.0.0.1'
port = 5432
max_connections = 100
unix_socket_directories = '/var/run/postgresql'
password_encryption = scram-sha-256
ssl = on
ssl_cert_file = '/etc/ssl/certs/ssl-cert-snakeoil.pem'
ssl_key_file = '/etc/ssl/private/ssl-cert-snakeoil.key'
shared_buffers = 128MB
dynamic_shared_memory_type = posix
max_wal_size = 1GB
min_wal_size = 80MB
log_line_prefix = '%m [%p] %q%u@%d '
log_timezone = 'UTC'
cluster_name = '13/main'
stats_temp_directory = '/var/run/postgresql/13-main.pg_stat_tmp'
datestyle = 'iso, mdy'
timezone = 'UTC'
lc_messages = 'C'
lc_monetary = 'C'
lc_numeric = 'C'
lc_time = 'C'
default_text_search_config = 'pg_catalog.english'
include_dir = 'conf.d'
```

```
# /etc/postgresql/13/main/pg_hba.conf

# TYPE  DATABASE  USER      ADDRESS       METHOD
local   all       postgres                trust
host    hviz      syslog    127.0.0.1/32  scram-sha-256
host    hviz      hvizweb   127.0.0.1/32  scram-sha-256
```

15.2. melléklet: Adatszerkezetet létrehozó kód

```
-- \c postgres
CREATE DATABASE hviz WITH
 TEMPLATE = "template0"
 ENCODING = "UTF8"
 LOCALE = "C";
CREATE ROLE "syslog" WITH LOGIN PASSWORD 'jelszó' ;
CREATE ROLE "hvizweb" WITH LOGIN PASSWORD 'jelszó' ;

-- \c hviz
CREATE SCHEMA hviz;

CREATE TABLE hviz."SshUser" (
 "Id" BIGINT GENERATED ALWAYS AS IDENTITY,
 "Username" VARCHAR UNIQUE,
 CONSTRAINT "PK_SshUser" PRIMARY KEY ("Id"));
```

```
CREATE TABLE hviz."SshPass" (
 "Id" BIGINT GENERATED ALWAYS AS IDENTITY,
 "Password" VARCHAR UNIQUE,
 CONSTRAINT "PK_SshPass" PRIMARY KEY ("Id"));

CREATE TABLE hviz."SshAuth" (
 "Id" BIGINT GENERATED ALWAYS AS IDENTITY,
 "Timestamp" TIMESTAMP WITH TIME ZONE NOT NULL,
 "SrcIp" INET NOT NULL,
 "SshUserId" BIGINT NOT NULL,
 "SshPassId" BIGINT NOT NULL,
 CONSTRAINT "PK_SshAuth" PRIMARY KEY ("Id"),
 CONSTRAINT "FK_SshAuth_SshUser_SshUserId"
  FOREIGN KEY ("SshUserId") REFERENCES hviz."SshUser" ("Id"),
 CONSTRAINT "FK_SshAuth_SshPass_SshPassId"
  FOREIGN KEY ("SshPassId") REFERENCES hviz."SshPass" ("Id"));

CREATE TABLE hviz."Protocol" (
 "Id" SMALLINT,
 "Name" VARCHAR UNIQUE NOT NULL,
 CONSTRAINT "PK_Protocol" PRIMARY KEY ("Id"));

CREATE TABLE hviz."Packet" (
 "Id" BIGINT GENERATED ALWAYS AS IDENTITY,
 "Timestamp" TIMESTAMP WITH TIME ZONE NOT NULL,
 "ProtocolId" SMALLINT NOT NULL,
 "SrcIp" INET NOT NULL,
 "DstPort" INT NOT NULL,
 CONSTRAINT "PK_Packet" PRIMARY KEY ("Id"),
 CONSTRAINT "FK_Packet_Protocol_ProtocolId"
  FOREIGN KEY ("ProtocolId") REFERENCES hviz."Protocol" ("Id"));

CREATE TABLE hviz."AspNetRoles" (
 "Id" VARCHAR,
 "Name" VARCHAR(256),
 "NormalizedName" VARCHAR(256),
 "ConcurrencyStamp" VARCHAR,
 CONSTRAINT "PK_AspNetRoles" PRIMARY KEY ("Id"));

CREATE TABLE hviz."AspNetUsers" (
 "Id" VARCHAR,
 "UserName" VARCHAR(256),
 "NormalizedUserName" VARCHAR(256) UNIQUE,
 "Email" VARCHAR(256),
 "NormalizedEmail" VARCHAR(256),
 "EmailConfirmed" BOOLEAN NOT NULL,
 "PasswordHash" VARCHAR,
 "SecurityStamp" VARCHAR,
 "ConcurrencyStamp" VARCHAR,
 "PhoneNumber" VARCHAR,
 "PhoneNumberConfirmed" BOOLEAN NOT NULL,
 "TwoFactorEnabled" BOOLEAN NOT NULL,
 "LockoutEnd" TIMESTAMP WITH TIME ZONE,
 "LockoutEnabled" BOOLEAN NOT NULL,
 "AccessFailedCount" INT NOT NULL,
 CONSTRAINT "PK_AspNetUsers" PRIMARY KEY ("Id"));
```

```
CREATE TABLE hviz."AspNetRoleClaims" (
 "Id" INT GENERATED BY DEFAULT AS IDENTITY,
 "RoleId" VARCHAR NOT NULL,
 "ClaimType" VARCHAR,
 "ClaimValue" VARCHAR,
 CONSTRAINT "PK_AspNetRoleClaims" PRIMARY KEY ("Id"),
 CONSTRAINT "FK_AspNetRoleClaims_AspNetRoles_RoleId"
  FOREIGN KEY ("RoleId") REFERENCES hviz."AspNetRoles" ("Id")
  ON DELETE CASCADE);

CREATE TABLE hviz."AspNetUserClaims" (
 "Id" INT GENERATED BY DEFAULT AS IDENTITY,
 "UserId" VARCHAR NOT NULL,
 "ClaimType" VARCHAR,
 "ClaimValue" VARCHAR,
 CONSTRAINT "PK_AspNetUserClaims" PRIMARY KEY ("Id"));

CREATE TABLE hviz."AspNetUserLogins" (
 "LoginProvider" VARCHAR(128),
 "ProviderKey" VARCHAR(128),
 "ProviderDisplayName" VARCHAR,
 "UserId" VARCHAR NOT NULL,
 CONSTRAINT "PK_AspNetUserLogins"
  PRIMARY KEY ("LoginProvider", "ProviderKey"),
 CONSTRAINT "FK_AspNetUserLogins_AspNetUsers_UserId"
  FOREIGN KEY ("UserId") REFERENCES hviz."AspNetUsers" ("Id")
  ON DELETE CASCADE);

CREATE TABLE hviz."AspNetUserRoles" (
 "UserId" VARCHAR,
 "RoleId" VARCHAR,
 CONSTRAINT "PK_AspNetUserRoles" PRIMARY KEY ("UserId", "RoleId"),
 CONSTRAINT "FK_AspNetUserRoles_AspNetRoles_RoleId"
  FOREIGN KEY ("RoleId") REFERENCES hviz."AspNetRoles" ("Id")
  ON DELETE CASCADE,
CONSTRAINT "FK_AspNetUserRoles_AspNetUsers_UserId"
 FOREIGN KEY ("UserId") REFERENCES hviz."AspNetUsers" ("Id")
 ON DELETE CASCADE);

CREATE TABLE hviz."AspNetUserTokens" (
 "UserId" VARCHAR,
 "LoginProvider" VARCHAR(128),
 "Name" VARCHAR(128),
 "Value" VARCHAR,
 CONSTRAINT "PK_AspNetUserTokens"
  PRIMARY KEY ("UserId", "LoginProvider", "Name"),
 CONSTRAINT "FK_AspNetUserTokens_AspNetUsers_UserId"
  FOREIGN KEY ("UserId") REFERENCES hviz."AspNetUsers" ("Id")
  ON DELETE CASCADE);

CREATE INDEX "IX_AspNetRoleClaims_RoleId"
 ON hviz."AspNetRoleClaims" ("RoleId");
CREATE UNIQUE INDEX "RoleNameIndex"
 ON hviz."AspNetRoles" ("NormalizedName")
 WHERE "NormalizedName" IS NOT NULL;
CREATE INDEX "IX_AspNetUserClaims_UserId"
 ON hviz."AspNetUserClaims" ("UserId");
CREATE INDEX "IX_AspNetUserLogins_UserId"
 ON hviz."AspNetUserLogins" ("UserId");
```

```
CREATE INDEX "IX_AspNetUserRoles_RoleId"
 ON hviz."AspNetUserRoles" ("RoleId");
CREATE INDEX "EmailIndex"
 ON hviz."AspNetUsers" ("NormalizedEmail");
CREATE UNIQUE INDEX "UserNameIndex"
 ON hviz."AspNetUsers" ("NormalizedUserName")
 WHERE "NormalizedUserName" IS NOT NULL;

CREATE TABLE hviz."VizSettings" (
 "UserId" VARCHAR,
 "VizName" VARCHAR,
 "Enabled" BOOLEAN NOT NULL,
 "IsTbl" BOOLEAN NOT NULL,
 "N" SMALLINT,
 "DateFrom" TIMESTAMP WITH TIME ZONE NOT NULL,
 "DateTo" TIMESTAMP WITH TIME ZONE,
 "Priority" SMALLINT,
 CONSTRAINT "PK_VizSettings" PRIMARY KEY ("UserId", "VizName"),
 CONSTRAINT "FK_VizSettings_AspNetUsers_UserId"
  FOREIGN KEY ("UserId") REFERENCES hviz."AspNetUsers" ("Id")
  ON DELETE CASCADE);

GRANT USAGE ON SCHEMA hviz TO hvizweb, syslog;

GRANT INSERT, SELECT ON
 hviz."SshUser",
 hviz."SshPass",
 hviz."SshAuth",
 hviz."Packet",
 hviz."Protocol"
TO syslog;

GRANT SELECT ON
 hviz."SshUser",
 hviz."SshPass",
 hviz."SshAuth",
 hviz."Packet",
 hviz."Protocol"
TO hvizweb;

GRANT INSERT, SELECT, UPDATE, DELETE ON
 hviz."AspNetRoles",
 hviz."AspNetUsers",
 hviz."AspNetRoleClaims",
 hviz."AspNetUserClaims",
 hviz."AspNetUserLogins",
 hviz."AspNetUserRoles",
 hviz."AspNetUserTokens",
 hviz."VizSettings"
TO hvizweb;
```

15.3. melléklet: Feldolgozásért felelős tárolt eljárás

```
CREATE PROCEDURE hviz.parse_log(message VARCHAR,timereported VARCHAR)
LANGUAGE PLPGSQL AS $$ DECLARE
 msg VARCHAR[];
 sshuser_id BIGINT;
 sshpass_id BIGINT;
 sshauth_srcip INET;
 icmp_msg VARCHAR[];
 protocol_id SMALLINT;
 packet_srcip INET;
 packet_dstport INT;
 timestamp TIMESTAMP WITH TIME ZONE;
BEGIN
msg := regexp_match(message, '^\s*Failed password for (.*) from (.*)
port .* ssh2 \(target: .*:.*, password: (.*)\)\s*$');
IF msg IS NOT NULL THEN
 timestamp := timestamptz(timereported);
 sshauth_srcip := CAST(msg[2] AS INET);
 IF msg[1] IS NOT NULL AND msg[3] IS NOT NULL
 AND sshauth_srcip IS NOT NULL THEN
  SELECT "Id" FROM hviz."SshUser"
   WHERE "Username" = msg[1] INTO sshuser_id;
  IF sshuser_id IS NULL THEN
   INSERT INTO hviz."SshUser"("Username") VALUES (msg[1]);
   SELECT "Id" FROM hviz."SshUser"
    WHERE "Username" = msg[1] INTO sshuser_id; END IF;
  SELECT "Id" FROM hviz."SshPass"
   WHERE "Password" = msg[3] INTO sshpass_id;
  IF sshpass_id IS NULL THEN
   INSERT INTO hviz."SshPass"("Password") VALUES (msg[3]);
   SELECT "Id" FROM hviz."SshPass"
    WHERE "Password" = msg[3] INTO sshpass_id; END IF;
  IF sshuser_id IS NOT NULL AND sshpass_id IS NOT NULL THEN
   INSERT INTO hviz."SshAuth" (
    "Timestamp", "SrcIp", "SshUserId", "SshPassId")
    VALUES (timestamp, sshauth_srcip, sshuser_id, sshpass_id);
 END IF; END IF; END IF;
msg := regexp_match(message,
'^\s*\[.*\]
CSOMAG.*SRC=([\d.]+).*PROTO=(TCP|UDP|ICMP).*(?:SPT|TYPE)=([\d]+).*(?:D
PT|CODE)=([\d]+).*\s*$');
IF msg IS NOT NULL THEN
 timestamp := timestamptz(timereported);
 packet_srcip := CAST(msg[1] AS INET);
 SELECT "Id" FROM hviz."Protocol"
  WHERE "Name" = msg[2] INTO protocol_id;
 IF protocol_id IS NOT NULL THEN
  IF protocol_id = 1 THEN -- ICMP. TYPE és CODE 8 bitesek, l. RFC 792.
   packet_dstport := (CAST(msg[3] AS INT) << 8) + CAST(msg[4] AS INT);
  ELSE -- TCP vagy UDP.
   packet_dstport := CAST(msg[4] AS INT);
  END IF;
  INSERT INTO hviz."Packet" (
   "Timestamp","ProtocolId","SrcIp","DstPort")
   VALUES (timestamp, protocol_id, packet_srcip, packet_dstport);
END IF; END IF; END; $$;
```

15.4. melléklet: Alapértelmezett adatokat létrehozó kód.

```
INSERT INTO hviz."Protocol" ("Id", "Name")
 VALUES (1, 'ICMP'), (6, 'TCP'), (17, 'UDP');

INSERT INTO hviz."AspNetRoles" (
 "Id",
 "Name",
 "NormalizedName",
 "ConcurrencyStamp")
VALUES (
 gen_random_uuid(),
 'Administrators',
 'ADMINISTRATORS',
 gen_random_uuid());

INSERT INTO hviz."AspNetUsers" (
 "Id",
 "UserName",
 "NormalizedUserName",
 "EmailConfirmed",
 "PasswordHash",
 "SecurityStamp",
 "ConcurrencyStamp",
 "PhoneNumberConfirmed",
 "TwoFactorEnabled",
 "LockoutEnabled",
 "AccessFailedCount")
VALUES (
 gen_random_uuid(),
 'Administrator',
 'ADMINISTRATOR',
 false,
'AQAAAAEAACcQAAAAELw6Q+73lrlNqeSxFOwpPSekbtVi3O8XoJ8hxXgXZ02jslCzT6ssx
b4x6ifgTH4kYw==',
 gen_random_uuid(),
 gen_random_uuid(),
 false,
 false,
 true,
 0);

INSERT INTO hviz."AspNetUserRoles" ("UserId", "RoleId") VALUES
 ((SELECT "Id" FROM hviz."AspNetUsers" LIMIT 1),
 (SELECT "Id" FROM hviz."AspNetRoles" LIMIT 1));

INSERT INTO hviz."AspNetRoles" VALUES
 (gen_random_uuid(), 'Readers', 'READERS', gen_random_uuid());
```

15.5. melléklet: Tesztkörnyezet konfigurációja

```
# Csapda VM
# /etc/systemd/system/ssh-honeypotd.service

[Unit]
Description=Low-interaction SSH honeypot
Requires=syslog.service
Requires=network.target
After=syslog.service
After=network.target

[Service]
Type=forking
ExecStart=/opt/ssh-honeypotd/ssh-honeypotd \
        --address 0.0.0.0 \
        --port 22 \
        --pid /run/ssh-honeypotd.pid \
        --name ssh-honeypotd \
        --user nobody \
        --group nogroup
Restart=always
RestartSec=1s

[Install]
WantedBy=default.target
```

```
# Csapda VM
# /etc/nftables.conf

#!/usr/sbin/nft -f
flush ruleset
table inet filter {
 chain input {
  type filter hook input priority 0; policy drop;
  iifname ne "lo" ip daddr 127.0.0.0/8 drop
  ip6 version 6 drop
  ip daddr 255.255.255.255 drop
  ip daddr 79.139.59.255 drop
  iifname "lo" ip saddr 127.0.0.0/8 ip daddr 127.0.0.0/8 accept
  tcp dport 64022 accept                    # SSH tunnel
  log prefix "CSOMAG " tcp dport 22 accept # SSH honeypot
  meta l4proto tcp ct state { established, related } accept
  meta l4proto udp ct state { established, related } accept
  log prefix "CSOMAG " drop
 }
 chain forward {
  type filter hook forward priority 0; policy drop;
 }
 chain output {
  type filter hook output priority 0; policy accept;
 }
}
```

```
# Csapda VM
# /etc/rsyslog.conf

module(load="imuxsock")
module(load="imklog")
module(load="omfwd")
$ActionFileDefaultTemplate RSYSLOG_TraditionalFileFormat
$FileOwner root
$FileGroup adm
$FileCreateMode 0640
$DirCreateMode 0755
$Umask 0022
$WorkDirectory /var/spool/rsyslog
$ActionQueueType LinkedList
$ActionQueueFileName queue
$ActionResumeRetryCount -1
$ActionQueueSaveOnShutdown on

:programname,isequal,"ssh-honeypotd" @@127.0.0.1:64514
:msg,contains,"CSOMAG"               @@127.0.0.1:64514
```

```
# Csapda VM
# /etc/ssh/sshd_config

Port 64022
AddressFamily inet
ListenAddress 0.0.0.0
HostKey /etc/ssh/ssh_host_ed25519_key
SyslogFacility AUTH
LogLevel INFO
PermitRootLogin no
PubkeyAuthentication yes
PasswordAuthentication no
PermitEmptyPasswords no
ChallengeResponseAuthentication no
UsePAM no
AllowAgentForwarding no
AllowTcpForwarding no
X11Forwarding no
PrintMotd no
AcceptEnv LANG LC_*
Subsystem sftp /usr/lib/openssh/sftp-server
AllowUsers srvadmin tunnel
Match User tunnel
      AllowTcpForwarding yes
      PermitTTY no
```

```
# Feldolgozó VM
# /etc/systemd/system/tunnel.service

[Unit]
Description=SSH tunnel for syslog forwarding
Requires=syslog.service
Requires=network.target
After=syslog.service
After=network.target

[Service]
Type=simple
User=tunnel
Group=tunnel
ExecStart=/usr/bin/ssh \
          -oServerAliveInterval=10 \
          -oExitOnForwardFailure=yes \
          -p64022 \
          -g \
          -T \
          -N \
          -R 127.0.0.1:64514:127.0.0.1:514 tunnel@csapda-vm
Restart=always
RestartSec=1s

[Install]
WantedBy=default.target
```

```
# Feldolgozó VM
# /etc/rsyslog.conf

module(load="imuxsock")
module(load="imklog")
module(load="imtcp")

$ActionFileDefaultTemplate RSYSLOG_TraditionalFileFormat
$FileOwner root
$FileGroup adm
$FileCreateMode 0640
$DirCreateMode 0755
$Umask 0022
$WorkDirectory /var/spool/rsyslog
$ActionQueueType LinkedList
$ActionQueueFileName queue
$ActionResumeRetryCount -1
$ActionQueueSaveOnShutdown on

template(name="postgres-syslog" type="list" option.stdsql="on") {
 constant(value="CALL hviz.parse_log ('")
 property(name="msg")
 constant(value="','")
 property(name="timereported" dateformat="pgsql" date.inUTC="on")
 constant(value="')")
}

input(type="imtcp" port="514")
module(load="ompgsql")

if ($programname == 'ssh-honeypotd') or ($msg contains 'CSOMAG') then{
 action(type="ompgsql" server="127.0.0.1" db="hviz" uid="syslog"
 pwd="jelszó" template="postgres-syslog")}
```

```
# Feldolgozó VM
# /etc/systemd/system/hvizweb.service

[Unit]
Description=Hvizweb
After=network.target

[Service]
WorkingDirectory=/var/www/Hviz
ExecStart=/usr/bin/dotnet /var/www/Hviz/Hviz.dll
Restart=always
RestartSec=10
User=www-data
Environment=ASPNETCORE_ENVIRONMENT=Production

[Install]
WantedBy=multi-user.target
```

```
# Feldolgozó VM
# /etc/apache2/sites-available/hviz.conf

<IfModule mod_headers.c>
 <VirtualHost *:*>
  RequestHeader set "X-Forwarded-Proto" expr=%{REQUEST_SCHEME}
 </VirtualHost>
</IfModule>

<VirtualHost *:80>
 Redirect permanent / https://feldolgozo-vm/
</VirtualHost>

<IfModule mod_ssl.c>
 <VirtualHost *:443>
  Protocols h2 http/1.1
  ServerName feldolgozo-vm
  ServerAlias *.feldolgozo-vm
  ProxyPreserveHost on
  ProxyPass / http://localhost:5000/
  ProxyPassReverse / http://localhost:5000/
  SSLEngine on
  SSLCertificateFile    /etc/ssl/certs/ssl-cert-snakeoil.pem
  SSLCertificateKeyFile /etc/ssl/private/ssl-cert-snakeoil.key
 </VirtualHost>
</IfModule>
```