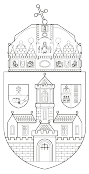


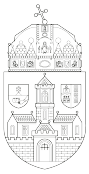
ÓBUDAI EGYETEM
ÓBUDA UNIVERSITY

NEUMANN JÁNOS
INFORMATIKAI KAR

SZAKDOLGOZAT

OE-NIK
2022.

Hallgató neve: **Érsok Máté**
Hallgató törzskönyvi száma: **T/006469/FI12904/N**



ÓBUDAI EGYETEM
ÓBUDA UNIVERSITY

NEUMANN JÁNOS
INFORMATIKAI KAR

SZAKDOLGOZAT

Honeypotok szerepe támadások felderítésében, API környezetben

Hallgató: **Érsok Máté**
Intézményi konzulent: **Vörösné Dr. Bánáti-Baumann Anna**
Külső konzulens: **Szarvák Anikó**

HALLGATÓI NYILATKOZAT

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

Budapest, 2021. 05. 13.

.....
hallgató aláírása

Óbudai Egyetem
Neumann János Informatikai Kar
Biomatika és Alkalmazott Mesterséges Intelligencia Intézet

SZAKDOLGOZAT FELADATLAP

Hallgató neve: **Érsok Máté**
Törzskönyvi száma: T/006469/FI12904/N
Neptun kódja: [redacted]

A dolgozat címe:

Honeypotok szerepe támadások felderítésében, API környezetben
The role of honeypots in attack detection, API environment

Intézményi konzulensek: Vörösné Dr. Bánáti-Baumann Anna, Szarvák Anikó
Külső konzulens:

Beadási határidő: 2021. május 15.

A záróvizsga tárgyai: Szoftvertechnológia és grafikus felhasználói interfész tervezése
Kiberbiztonság specializáció

A feladat

Rohamosan fejlődő világunkban már szinte minden vállalat alapja egy stabil, megbízható informatikai infrastruktúra. Ez magába foglalja a gyors és biztonságos működéshez, illetve üzemeltetéshez szükséges hardver és szoftver komponenseket, hálózati elemeket, üzleti folyamatokat támogató alkalmazásokat és azokat az egyéb szolgáltatásokat, amelyek szükségesek a rendszer fenntartásához. Egy ilyen infrastruktúra biztonsága a vállalat működése szempontjából meglehetősen kritikus és ez a fontos rendszer könnyen kiberbűnözők célpontjává válhat. E fenyegetés ellen többféle módszerrel igyekeznek védekezni, ezek közül az egyik honeypot rendszer alkalmazása. A honeypot igazából egy csapda, amely egy valós vállalati rendszerelemnek álcázza magát, de nem tartalmaz valódi adatokat, nem kommunikál az éles környezettel, viszont sokkal könnyebben feltörhető. Ezzel a megoldással egyrészt egy elszigetelt környezetbe csalogatható a támadó, ahol nem jelent veszélyt a vállalati szolgáltatásokra, másrészt monitorozhatóvá és elemezhetővé válik a támadás, aminek segítségével egy esetleges valódi támadás megakadályozható lesz.
A szakdolgozat célja, hogy bemutassa a biztonsági műveleti központ (BMK) oldaláról a honeypot-tal szemben támasztott szempont- és követelményrendszert, illetve a ráépülő riasztásokat és szolgáltatásokat.
A hallgató feladata, hogy mutassa be egy honeypot rendszer és egy BMK kapcsolata által létrejövő kihasználható lehetőségeket, valamint a BMK oldaláról megtervezze és implementálja a szükséges megoldásokat.

A dolgozatnak tartalmaznia kell:

- a honeypot rendszer használati eseteit API környezetben,
- a BMK oldaláról, a honeypot-al szemben támasztott szempont- és követelményrendszrert,
- a rendszerben rejlő kihasználható lehetőségek ismertetését,
- a BMK-ra épülő szolgáltatások és riasztások előnyeit,
- a szolgáltatások és riasztások implementációját, tesztjét és annak kiértékelését.

ÓBUDAI EGYETEM * Neumann János Informatikai Kar * 1. Budapest

Ph.

.....
Dr. Eigner György
mb. intézetigazgató

A szakdolgozat elévülésének határideje: **2023. május 15.**
(OE TVSz 55.§ szerint)

A dolgozatot beadásra alkalmasnak tartom:

.....
külső konzulens

.....
intézményi konzulens

ÓBUDAI EGYETEM
ÓBUDA UNIVERSITY

Neumann János Informatikai Kar
9. sz. melléklet

KONZULTÁCIÓS NAPLÓ

Hallgató neve: ERSOK MÁTÉ

Neptun kód: [redacted]

Tagozat: ÜZEMMÉRNÖK-INFORMATIKUS

Telefon: [redacted]

Levelezési cím (pl: lakcím): [redacted]

Szakdolgozat / Diplomamunka[1] címe magyarul:
HONEYPOTOK SZEREPE TÁMADÁSOK FELDERÍTÉSÉBEN, API KÖRNYEZETBEN

Szakdolgozat / Diplomamunka[2] címe angolul:
THE ROLE OF HONEYPOTS IN ATTACK DETECTION, API ENVIRONMENT

Intézményi konzulens: VÖRÖSNÉ Dr. BÁNÁTI-BAUMANN ANNA

Külső konzulens: SZARVÁK ANIKÓ

Kérjük, hogy az adatokat nyomtatott nagybetűkkel írja!

Alk.	Dátum	Tartalom	Aláírás
1.	10.21.	FELADAT KIÍRÁS	Bánáti
2.	10. 28.	SZERVEZETI FELÉPÍTÉS	Bánáti
3.	11.10.	HIÁNYOSSÁGOK MEGBESZÉLÉSE	Bánáti
4.	12. 03.	PREZENTÁCIÓ	Bánáti

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 / Diplo-mamunka 2 / Diplomamunka 3 / Diplomamunka 4[3] tantárgy követelményét teljesítette, beszámolóra / védésre[4] bocsátható.

Budapest, 20.20 december 10.

Bánáti Anna
intézményi konzulens

[1] Megfelelő aláhúzandó!
[2] Megfelelő aláhúzandó!
[3] Megfelelő aláhúzandó!
[4] Megfelelő aláhúzandó!

KONZULTÁCIÓS NAPLÓ

Hallgató neve: ÉRSOK MATÉ

Neptun kód: [redacted]

Tagozat: ÜZEMMÉRNÖK-INFORMATIKUS

Telefon: [redacted]

Levelezési cím (pl: lakcím): [redacted]

Szakdolgozat / Diplomamunka[1] címe magyarul:

HONEYPOTOK SZEREPE TÁMADÁSOK FELDERÍTÉSÉBEN, API KÖRNYEZETBEN

Szakdolgozat / Diplomamunka[2] címe angolul:

THE ROLE OF HONEYPOTS IN ATTACK DETECTION, API ENVIRONMENT

Intézményi konzulens: Dr. BANÁTI-BAUMANN ANNA

Külső konzulens: SZARVÁK ANIKÓ

Kérjük, hogy az adatokat nyomtatott nagybetűkkel írja!

Alk.	Dátum	Tartalom	Aláírás
1.	02. 10.	TESZTELÉSI KÖRNYEZET KIALAKÍTÁSA	
2.	03. 17.	TESZTELÉS ESEMÉNYEINEK MEGGYŰJTÉSE	
3.	04. 07.	EREDMÉNYEK ELEMZÉSE	
4.	04. 21	TOVÁBBFEJLESZTÉSI LEHETŐSÉGEK 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 / Diplo-mamunka 2 / Diplomamunka 3 / Diplomamunka 4[3] tantárgy követelményét teljesítette, beszámolóra / védésre[4] bocsátható.

Budapest, 2021.05.13.

.....
intézményi konzulens

[1] Megfelelő aláhúzandó!
[2] Megfelelő aláhúzandó!
[3] Megfelelő aláhúzandó!
[4] Megfelelő aláhúzandó!

Tartalomjegyzék

1. Bevezetés

Napjainkra az emberiség jelentős hányada hallott már a kiberbűnözésről és találkozott vele közvetve, vagy közvetlenül. Ehhez képest viszont egész kevesen vannak tisztában azzal, hogy pontosan milyen támadási formák léteznek, hogyan lehetne ezek ellen teljes biztonsággal védekezni. Mindez azért van, mert a támadók és velük együtt technikáik folyamatosan fejlődnek. Az információs rendszerekben kezelt adatok mennyisége és az adatkezelés sebessége egyre növekszik, ami egyre nagyobb erőforrást, egyre több informatika eszközt igényel. Azonban ez a megnövekedett eszközpark, nagyobb támadási felületet biztosít kiberbűnözőknek. Emellett az eszközök, és a rajtuk futó alkalmazások diverzitása további támadási formák megjelenését eredményezheti.

1.1. Fenyegetések

Rengeteg támadási formáról hallhatunk, találhatunk információt, melyek legtöbb esetben károsíthatnak egy vállalatot vagy államigazgatási szervet, hátráltathatják a hatékony működésben. A támadási lehetőségek száma szinte végtelen, de vannak közöttük láthatóan népszerűbbek. A DOS vagy DDOS - (distributed) denial of service, amely (elosztott) szolgáltatás megtagadást jelent. Ezen támadási forma a vállalat valamely szolgáltatását igyekszik oly mértékben leterhelni, hogy az szolgáltatás kieséshez vezessen. A malware-ek rengeteg formája okozhat további problémákat egy informatikai rendszeren. Ebbe a csoportba tartoznak a már régóta jelenlévő vírusok és férgek, de a napjainkban meglehetősen népszerű ransomware (zsarolóvírus) és spyware (kémprogram) megoldások is. Ha ez még nem lenne elég, az SQL injection (SQL injektálás) vagy épp a Cross-Site Scripting (weboldalak közötti szkriptelés) is kellőképpen nagy fenyegetést jelent a vállalatok számára. A kiberbiztonsági szakemberek számára tehát adott a feladat, meg kell teremteni az ilyen és hasonló támadások elleni biztonsági eszközöket. [1]

1.2. Védekezési lehetőségek

A biztonság fogalma A magyar nyelv értelemező szótára szerint: „A dolgoknak, életviszonyoknak olyan rendje, olyan állapot, amelyben kellemetlen meglepetésnek, zavarnak, veszélynek nincs v. alig van lehetősége, amelyben ilyentől nem kell félni. Kollektív biztonság; biztonság okáért ¬ v. kedvéért. A világ békéje és biztonsága. Rend, nyugalom és biztonság a termelés alapja. ‖ a. Vkinek, vminek (veszélytől, kártól, jogtalan beavatkozástól, bántódástól való) védett állapota, helyzete”[2]. A biztonság tehát csak egy állapot, amelyet különböző védekezi mechanizmusokkal képesek lehetünk fenntartani. Legalapvetőben három alappillérrel jellemezhető, ez a három a bizalmasság (confidentality), sértetlenség (integrity) és a hozzáférhetőség (availability), melyeket röviden csak CIA hármas modellként emlegetünk. A modell iránymutatást ad, hogy pontosan milyen információbiztonsági szempontok alapján kell megtervezni és üzemeltetni egy ilyen infrastruktúrát.

Ahogyan a támadásnál, úgy védekezésnél is jó néhány technika közül válogathatunk. Amellett, hogy követjük a megfelelő ajánlásokat – alkalmazások frissítése, megfelelő jelszókezelés, alhálózatokra bontás stb. – konkrét hardveres és szoftveres eszközt is bevethetünk. Tűzfalak használatával figyelhetjük és szabályozhatjuk a kimenő és a bejövő forgalmat, mind hálózati mind host oldalon. Alkalmazhatunk továbbá behatolás detektáló (IDS) és behatolás megelőző rendszereket is a biztonság növelése érdekében. Az IDS érzékelését csak utólagos elemzésre és szabályok felállítására, az IPS-t viszont valós idejű megakadályozásra is használhatjuk.

Azért, hogy teljes képet kaphassunk rendszerünk hiányosságairól, gyenge pontjairól fontos lehet különféle sérülékenységi vizsgálatokat végeznünk. Ezzel feltárhatunk minden olyan szoftveres vagy hardveres ismert sebezhetőséget, melyeket frissítéssel, cserével célszerű javítanunk. Egy másik biztonsági próbája lehet a rendszernek a penetration testing (behatolási vizsgálat), mely során szerződésben rögzített paraméterek alapján egy etikus hacker megkísérel betörni a rendszerbe. Amennyiben ez sikerül, a teszt alapján kijavíthatóak azok a biztonsági rések, amelyeken át tudott jutni.

Igazán feltérképezni naplóbejegyzések alapján tudjuk, melyek minden eszközön keletkeznek kellő gyakorisággal. Ezen log adatokat érdemes központosítva kezelni a könnyebb elérhetőség és kereshetőség érdekében, például egy SIEM rendszer alkalmazásával.

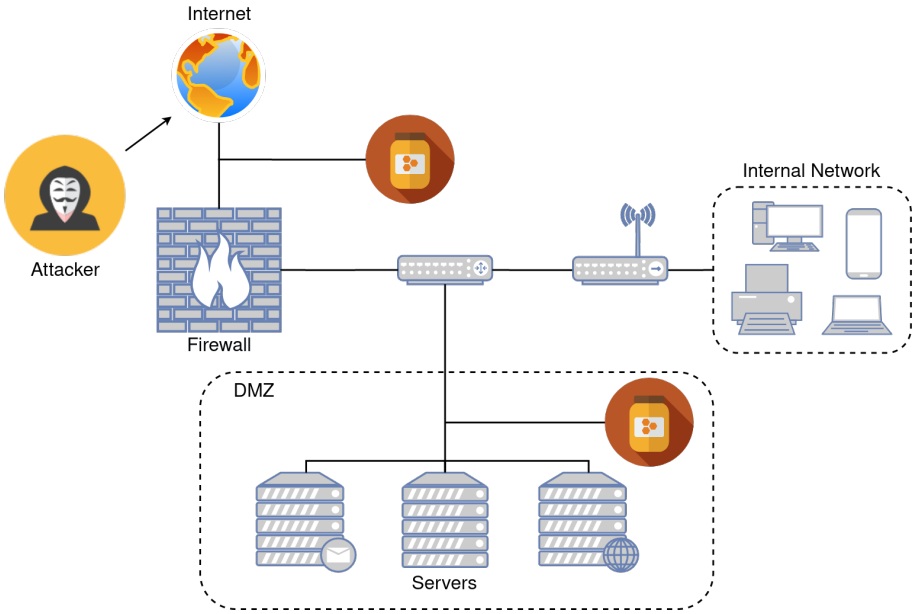
Ennél offenzívabban is végezhetünk kutatást a lehetséges támadásokról, amennyiben nem csak védekezünk, hanem csapdát is állítunk támadóinknak. Ezen a területen a csapda egy honeypot-ot jelent, amely segítségével képesek lehetünk megfigyelni a támadók szokásait és eszközeit, időt nyerve felkészíteni az infrastruktúrát a valós támadások elleni védekezésre.

1.3. Projekt célja

A projekt célja megvizsgálni milyen különbségek lehetnek egy vállalati és egy kutatási környezetben alkalmazott honeypot rendszer között. A két különböző követelmény rendszerhez igazodva, hogyan leheteséges megfelelő honeypot megoldást alkalmazni úgy, hogy az általa gyűjtött adatok kellően hasznos információval szolgáljanak az elemzők számára. Fontos ismertetni, hogy milyen gyakorlati haszna lehet, amennyiben a keletkezett adatok központosítva kerülnek feldolgozásra, hogyan illetve milyen szempontrendszer alapján kerülhetnek implementálásra riasztások és szolgáltatások, amelyek segíthetik az elemzők munkáját a támadások felderítésében. Fontosnak tartom megemlíteni, hogy ezen projekt hallgatótársammal, Balogh Ádámmal közreműködésben készült a téma és a feladatok felosztásával. Az általam implemtált rendszer az ő dolgozatában kiválasztott és implementált rendszerre épül.

2. Honeypot

A honeypot egy olyan informatikai biztonsági eszköz, amely egy valós, de könnyebb feltörhető rendszerelemnek álcázza magát, ezzel felkeltve a támadó figyelmét. Amennyiben megfelelő honeypot-ot használunk és jól választjuk a monitorozás alatt gyűjtött adatok mennyiségét és minőségét egészen pontos képet kaphatunk arról, hogy mivel szemben is kellene védekeznünk.



1. ábra. Honeypot elhelyezése egy hálózatban

2.1. Mi az a honeypot?

Ezen honeypot megoldásokat különböző szempontok szerint is csoportosíthatjuk. Megkülönböztethetünk fizikai és virtuális honeypotot. A fizikai honeypot egy valós készüléken, míg a virtuális egy ténylegesen nem létezőn hostján keresztül kapcsolódik a hálózathoz. A honeypottal való kapcsolat gyakorisága alapján kategorizálhatjuk az egyes megvalósításokat, ez alapján beszélünk alacsony és magas interakciójú, illetve pure honeypotokról. Míg az alacsony interakciójú csak egy viszonylag szűk részét szimulálja a hálózatnak, addig a magas sokkal komplexebb folyamatokat és szolgáltatásokat is magában foglal. A pure honeypot mindkettőt túlszárnyalva egy teljes értékű – az éleshez hasonló – környezetet futtat a kiberbűnözők csalogatására.

2.2. Használati esetek

Honeypotot több okból is létrehozhatunk. Amennyiben céljuk a tanulmányozás, hogy milyen támadások gyakoriak, pontosan milyen területről érkeznek, akkor egy research honeypotról beszélünk. Viszont, ha egy vállalat védelmének szolgálatában működik,

abban az esetben production honeypotról van szó. Egy research honeypot általában nagyobb mennyiségű információ gyűjtésére konfigurált, egy production honeypot azonban sokkal inkább ki van hegyezve azokra veszélyforrásokra, amely a szervezet informatikai infrastruktúrájára jelent fenyegetést.

Egy honeypot sokoldalúan képes működni, attól függően, hogy mire használjuk az általa gyűjtött információkat. Mivel ez az informatikai csapda könnyen kapcsolatba hozható más vállalati szolgáltatásokkal is, ezért ez az adathalmaz meglehetősen sokrétűen feldolgozható, rengeteg kihasználható lehetőséget biztosít a cég részére.

A legegyszerűbb felhasználási opció, ha a gyűjtés alapján fekete listát hozunk létre, melyben tiltjuk azon forrásokról a hozzáférést, amelyek korábban kapcsolatba léptek a honeypottal. Ez azért is életszerű, mivel a honeypottal egyetlen alkalmazottnak sem kell érintkeznie, nem képezi a szervezetnél működő éles környezet részét. Ezáltal, ha valaki mégis kapcsolatba kerül vele, az enyhén szólva is gyanús.

Előfordulhat azonban, hogy egy vállalati gépről érkezik forgalom a honeypot felé, ilyenkor jobb esetben csak egy kevésbé tájékozott kíváncsi kollégával, rosszabb esetben viszont egy kompromittálódott eszközzel van dolgunk. A honeypot adta lehetőség által az ilyen és ehhez hasonló problémákra még az előtt fény derülhet, hogy rosszabbra fordulna a helyzet.

Alternatíva lehet még az is, hogy egy IDS/IPS rendszer kiegészítésére használjuk. Ez a rendszer igyekszik minél hatékonyabban figyelni és elemezni a hálózati forgalmat, az IDS passzívan, míg az IPS aktívan teszi ezt. Mindkét technológia mintákat használ a rosszindulatú forgalom detektálására, ezért előfordulhat, hogy rosszul ítélnek meg egy valótlan fenyegetést vagy épp szó nélkül átengednek egy valósat. Ezen pontatlanság javítására használhatjuk fel a honeypot által gyűjtött információkat és új minták létrehozásával fejleszthetjük az IDS/IPS rendszert, csökkenthetjük az álpozitív riasztások számát. Természetesen az együttműködés lehet kölcsönönös is abban az esetben, ha mondjuk az IDS képes a honeypotra juttatni a támadót.[3]

3. Security Operation Center

3.1. SOC feladata és működése

Egy Security Operation Center (továbbiakban SOC, Biztonsági Műveleti Központ) leginkább a Nasa Küldetés Irányító Központjához hasonítható, egy fal tele monitorokkal, amelyet egy egész teremnyi specialista figyel keresve a legapróbb jelet bármiféle támadásra. A valóság ennél egy kicsit evilágibb. Legegyszerűbben megfogalmazva a SOC egy gyakorlott szakértőkből álló csapat, akiknek rendelkezésére áll egy megfelelő eszközkészlet.[4] Feladatuk a szervezet teljes hálózati infrastruktúrájának monitorozása támadások és fenyegetések jeleit keresve. Ezzel megteremve annak a lehetőséget, hogy szinte azonnal képesek legyenek reagálni a bekövetkezett eseményekre. Ahelyett, hogy passzívan riasztásokat várnának, vadászhatnak is a támadásokra, mellyel még inkább növelhetik a rendszer biztonságát.

3.2. SIEM

A SOC legfontosabb része a SIEM (Security Information and Event Management, Biztonsági Információ- és Eseménykezelés), amely egy központosított formában gyűjti és dolgozza fel az információs rendszer valamennyi eszközéről és szolgáltatásából érkező naplóbejegyzéseket. Az összegyűjtött adatokból értelmezhető információt hoz létre az által, hogy a hasonlókat, közös vonásokkal rendelkezőket egy kötegbe fogja össze. Ezen információt összeveti a rendelkezésére álló fenyegetési mintázatokkal és amennyiben talál egyezést, előre definiált szabályok alapján riasztást küld. A riasztás különböző csatornákon érkezhet az elemzőkhöz, akár egyszerű szöveges üzenet formájában, akár egy biztonsági irányítópult felületére. Kizárólag a riasztás célja fontos, ami pedig az, hogy felkeltse a szakemberek figyelmét a megszokottól eltérő működésre.[5]

3.3. Szolgáltatások és riasztások

Egy ilyen jelentős adatmennyiséget kezelő, megközelítőleg az infrastruktúrában minden eszköztől naplóadatot gyűjtő rendszerben a kialakítható további szolgáltatások listája végtelenül hosszú. Az eszközök, alkalmazások fontossága és a bekövetkezett események súlyossága alapján a szolgáltatás- és riasztásrendszert különféle kritikussági szinteken definiálhatjuk. Ezeket a biztonsági kategóriákat kulcsfontosságú precízen megszabni, annak érdekében, hogy kellően releváns tájékoztatást nyújtson az elemzők számara. Ez megtehető az értesítés gyakoriságának, a tájékoztató csatorna, valamint informálni kívánt szakértői csoport jól megfontolt megválasztásával. Amennyiben egy jelentéktelenebb esetről van szó meghatározhatunk például egy olyan szabályt, mely alapján az értesítés csak megszabott gyakorisággal (óránként egyszer, naponta egyszer) hívja fel a szakértők figyelmét. Ezzel szemben, hogyha egy rendkívül kritikus ügyet észlel, előnyös ezt azonnal a rendszert megfigyelői között, minél nagyobb körben elterjesztenie azért, hogy a kivizsgálás a lehető leghamarabb megkezdődhessen. Ezen riasztásoknál többféle kommunikációs csatorna is szóba jöhet. Történhet emailben, a

szakemberek vállalati mobileszközeire egy pillanat alatt megérkező leküldéses értesítés (push notification) formájában, vagy a szervezeti üzenetküldő platform elkülönített csatornáján, vagy akár egy erre célra létrehozott szolgáltatás segítségével is. Ezek mellett a SIEM rendszer irányítópultokba (dashboard) rendezve is felvilágosítja az elemzőket a történt eseményekről. Ebben a formában lehetőség nyílik többféle kimutatás konfigurálására, melyekre rápillantva azonnal észrevehető a szokásostól eltérő működés.[4] [6]

4. Tervezés

4.1. SOC oldali követelmények

A honeypot által gyűjtött információk felhasználása egy logkezelő rendszer segítségével kell történjen, amely centralizáltan gyűjti, tárolja, valamint rendszerezi a beérkezett naplóadatokat. A vállalati környezetben ezek az Elastic Stack alkalmazáskészlete által kerülnek feldolgozásra, ezáltal a honeypotnak valamilyen formában összekapcsolhatónak kell lennie a már meglévő rendszerrel.

A honeypot szolgáltatásáról/szolgáltatásairól az infrastruktúrára fenyegetést jelentő eseteket naplózni kell, és ezen naplózás szerint kritikussági szintekre bontva, definiálni riasztásokat, amelyek gyorsítják az elemzők munkáját. Ezen riasztások célalkalmazása a vállalati csevegő platform egy erre elkülönített csatornája. Az riasztási üzenetnek tartalmaznia kell, hogy a honeypot miatt generálódott, valamint, néhány fontosabb információt a kiváltó okról. Ez lehet például a megtámadott szolgáltatás neve, a támadó forrás címe, vagy éppen a támadó által bevitt adatok, parancsok.

A csapdaként működő alkalmazásoknak és szolgáltatásoknak illeszkednie kell a valós vállalati környezetbe. Szükséges elkülöníteni a honeypotról érkező naplóbejegyzéseket egy külön csatorna vagy index létrehozásával, hogy ezen logok izolálva legyenek a vállalati alkalmazások gyűjteményeitől.

Fontos kidolgozni egy megoldást, mely segítségével a honeypot működése meghatározott gyakorisággal automatikusan tesztelhető anélkül, hogy fals pozitív riasztás keletkezne. Ezt érdemes egy honeypottal azonos környezetben üzemelő szerveren megvalósítani, ezáltal minimalizálni a teszt közben felmerülő hibalehetőségeket, amely forrása lehet egy téves tűzfalkonfiguráció, vagy akár egy hálózati kiesés.

Amennyiben lehetséges, célszerű a keletkezett adatokat vizualizálni, így gyorsítva az elemzők munkáját:a szemléltetés főbb pontjait, a behatolások gyakoriságát szolgáltatásokra lebontva, a támadók IP címét helyadatok felhasználásával térképen szemléltetve, a felhasznált bejelentkezési adatokat, bevitt parancsokat. [6] [5]

4.2. Választott honeypot ismertetése

A honeypot megoldások vizsgálatával Balogh Ádám foglalkozott, összegyűjtve az elérhető programokat és szempontrendszer felállítása mellett vetette össze a lehetséges opciókat. Az eredmény az előzetes elképzelésekhez képest jelentősen változott, egy egyszerű honeypot szolgáltatás helyett két egészen komplex honeypot rendszer került kiválasztásra. Ez a két szolgáltatás egymástól függetlenül működik, viszont az általuk gyűjtött adatok akár egyszerre is használhatóak, legegyszerűbb esetben például validálhatják egymást.

4.3. Honeytrap

A *HoneyTrap* [7] egy nyílt forráskódú honeypot keretrendszer, amely meglehetősen sokrétűen konfigurálható. Szimulálhat egyetlen egyszerű szolgáltatást, vagy akár egy

komplex honeypot architektúrát is. Konfigurálástól függően lehetőségünk van akár egy alacsony akár egészen magas elérési szintű honeypot rendszer készítésére. Akár több eszközön is telepíthetjük, így növelve a gyűjtött adatok mennyiségét. Ezen megvalósítás előnye, hogy meglehetősen széles a támogatott operációs rendszerek és architektúrák száma azáltal, hogy a *HoneyTrap* teljes szolgáltatáskészlete egyetlen docker konténerben is képes futni.

4.4. T-Pot

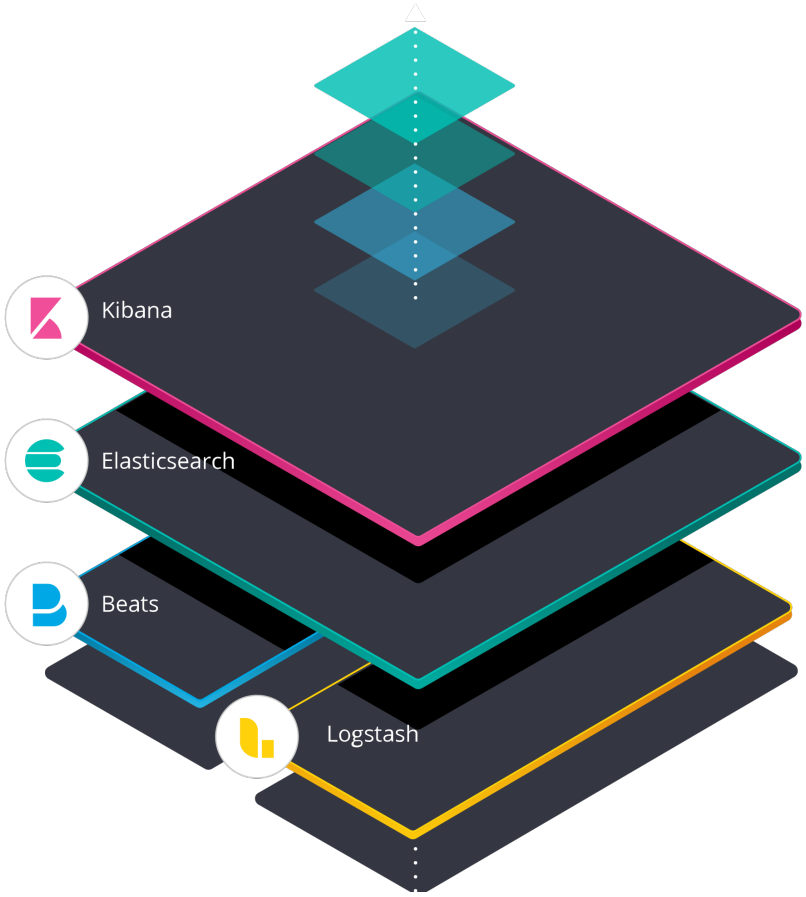
A *T-Pot* [8] egy nyílt forráskódú honeypot gyűjtemény, amelyet a Telekom fejlesztett, egymással összhangban működő, konténerizált honeypot szolgáltatásokból áll. A szolgáltatások egyesével konfigurálhatóak, illetve lehetőségünk van további konténerek hozzáadására is. A csomag alapértelmezetten tartalmaz egyéb funkcionalitást ellátó komponenseket is, amelyek tovább bővítik a gyűjtött adatok mennyiségét, vagy épp segítik az adatok elemzését. Ilyen például a Suricata vagy a P0f, amelyek a T-Pot gazdagépére érkező teljes hálózati forgalmat figyelik, illetve a Geoip-Attack-Map és a Spiderfoot, melyek pedig az elemzési fázist gyorsíthatják fel.

4.5. *Elastic Stack*

A Biztonsági Műveleti Központban a naplóadatok központosított kezelésére előzetesen az *Elastic Stack* [9] (ELK Stack) került megvalósításra. Egy nyílt forráskódú elosztott rendszer, amely flexibilitása és skálázhatósága miatt alkalmazható szinte bármely informatikai rendszer adatainak centralizált gyűjtésére. Ez a rendszer felel a naplóállományok gyűjtése mellett a rendszerezéséért, illetve tárolásáért. Három alapvető komponens alkotja: az Elasticsearch, a Logstash valamint a Kibana.

Az Elasticsearch egy kereső és elemző motor, amely az ELK leglényegesebb, kiadását tekintve legidősebb komponense. Legyen szó struktúrált vagy struktúrálatlan szövegről, numerikus vagy térinformatikai adatokról az Elasticsearch képes keresést és elemzést végezni közel valós időben. A begyűjtött bejegyzéseket indexekben tárolja, amely további szűrési opciókat tesz lehetővé. Ezen indexek a felhasználási cél alapján definiálhatóak, egyszerű kezelésüket minták létrehozásával segíthetjük. Ezen mintákban definiálható, hogy milyen mezőkre bontsa fel a gyűjtött naplóadatokat, hogyan tárolja és archiválja. A további komponensek stabil és hatékony működését az Elasticsearch elengedhetetlen működése teszi lehetővé.

Az Elasticsearch munkáját segíti egy Logstash nevezetű adatgyűjtő motor. Három alapvető műveletet valósít meg, begyűjt, módosít és továbbít. Begyűjtés során az input lehet szöveges file, syslog vagy *beat*-ek amelyek folyamatok eseményeit tartalmazzák és a Beat modul küldi őket. A módosításnál szöveges feldolgozás, mező létrehozás, átnevezés vagy kihagyás valósul meg, mely segítségével kívánt formára alakíthatóak a különböző rendszerekben keletkezettt logadatok. Az utolsó folyamat, a továbbítás, ezen rész valósítja meg az adatok megfelelő cél felé való küldését. A cél lehet a már említett Elasticsearch, egy fájl amelyben a módosított adatok írásra kerülnek, vagy akár más naplóadatkezelő megoldás is.



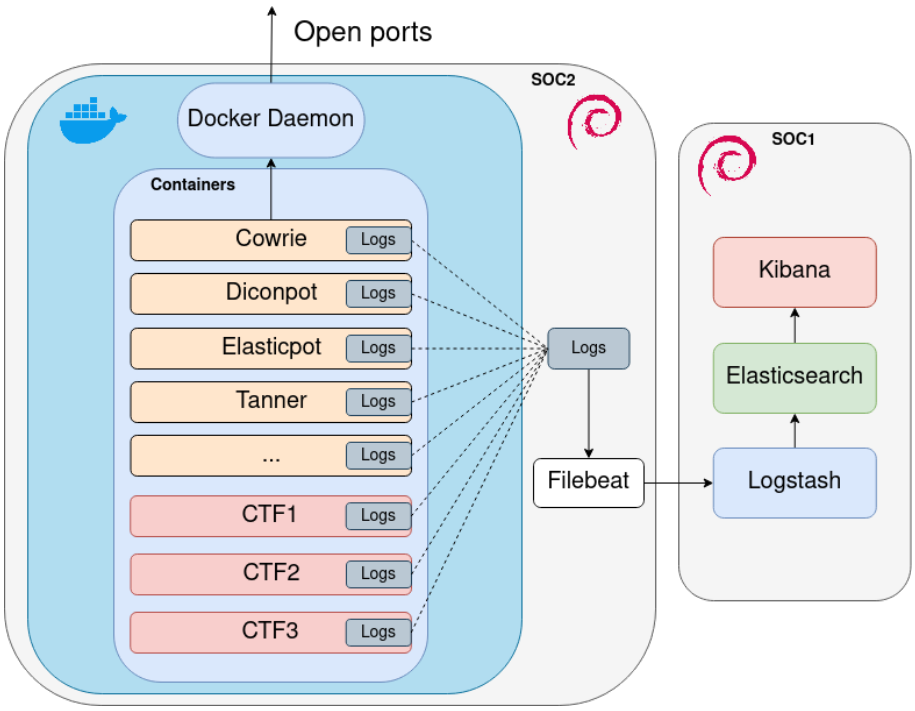
2. ábra. Elastic Stack komponenseinek kapcsolata[9]

Végfelhasználói szempontból az ELK leglátványosabb alkotóeleme a Kibana. Feladata rendkívűl sokszínű, felhasználói interfészt biztosít a teljes ELK rendszerhez. Segítsével a gyűjtött adatok alakot ölthetnek, átlátható és könnyen szűrhető formában jeleníthetőek meg. A bejegyzések szabályok alapján való összekapcsolásával további grafikonokat, kimutatásokat készíthetünk, amelyek segítenek a rendszer történéseinek megértésében. Irányítópultok létrehozásával felgyorsíthatjuk az elemzést azáltal, hogy így egy szempillantás alatt észrevehető a normál működéstől való eltérés, legyen szó akár valamilyen rendszerhibáról, akár egy biztonsági incidensről. Ezen felül itt elvégezhetjük szinte a teljes Elastic konfigurálását és menedzselését.

Az egyes végpontokon az Elasticsearch és a Logstash folyamatait megelőzve a Filebeat segíthet az adatok előkészítésében. Ez egy meglehetősen alacsony erőforrásigényű agent, amely a keletkezett események és adatok aggregálását végzi, majd az aggregálást követően küldi tovább a Logstashnek, vagy az Elasticsearch egy meghatározott indexe felé. A forrás a célhoz hasonlóan igény szerint testreszabható, megadható hol keresse a naplófájlokat vagy éppen milyen szabály mentén gyűjtse be azokat. A Filebeat lehetőséget biztosít arra, hogy az informatika rendszerünk bármely eszközéről gyűjthessünk adatokat, különösebb erőforrásigény nélkül.

4.6. Architectúra bemutatása

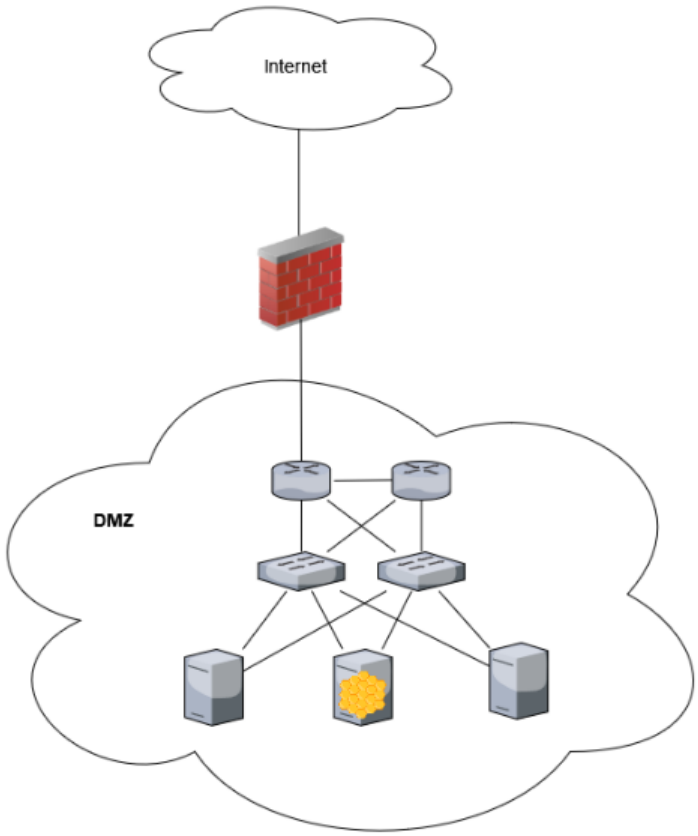
A kiválasztott honeypot rendszerek implementálására az egyetemi SOC laborban kerül sor. A Honeytrap és a T-Pot egy-egy virtuális Debian szerverre kerül feltelepítésre. Mindkét rendszer Docker alapon működik melynek előnye, hogy egy esetleges hiba vagy leállás esetén automatikusan újraindul. Biztonsági szempontból is megfelelőbb ez a konfiguráció, hiszen így az egyes honeypotok elszeparált környezetben működhetnek, a támadó a konténerekből nem tud a host gép felé kommunikálni. Az adatok elemzése szempontjából szükséges komponens az ELK Stack, amely korábban telepítésre került a laborban. A honeypot rendszerek ennek a központi naplóemlemzőnek juttatják el a gyűjtött adatokat, a kapcsolatot az alábbi 3. ábra szemlélteti.



3. ábra. Egy honeypot rendszer és az ELK kapcsolata

4.7. Honeypot által gyűjtött naplóállományok felhasználása

Egy informatikai rendszerben meglehetősen sok adat keletkezik, már az eszközök alapállapotú futása közben is, ezért egy honeypot esetén különösen fontos megválasztani a felderítés szempontjából fontos naplóállományok csoportját. Egy ilyen csapda esetében érdemes azokkal a jelzésekkel foglalkozni, amelyek egy interakció során jönnek létre, mert ezek nyújthatnak információt a támadóról. Természetesen egyéb naplóadatokatnak is tulajdoníthatunk jelentőséget, például egy egyszerű „Igen, életben vagyok” jellegű bejegyzésen keresztül láthatjuk, hogy a rendszerünk továbbra is működik és várja



4. ábra. Honeypot elhelyezése egy hálózatban

a beérkező kéréseket. Jelen rendszernél hasonló megfontolás alapján kezeltük ezen állományokat.

Az adatok a konténerizált rendszerbe egy a gazdagépről felcsatolt könyvtárban kerülnek tárolásra, így az adatok lokálisan is folytatólagosak lehet abban az esetben is ha a konténert valamilyen hiba folytán újra kell telepíteni. Innen a naplóadatok elszállításáért egy *Filebeat* [9] nevű logszállító alkalmazás felel, amely a definiált elérési útvonalakról gyűjti a bejegyzéseket, és az Elasticsearch megfelelő indexeibe juttatja el azokat. Eközben megtörténik a fájlokban lévő adatok meghatározott szabály alapján történő mezőkre bontása. Ez az adathalmaz már könnyen kereshető és szűrhető, mely lehetőséget ad arra, hogy a Kibana felületén irányítópultokba rendezve megjelenítsük, további folyamatokban automatizálva használjuk, kialakíthassunk egy szolgáltatási- és riasztási rendszert.

5. Implementáció

5.1. Honeypot telepítési tapasztalatok

A honeypotok a fejlesztőik által ajánlott módszerekkel, a hivatalos dokumentációk alapján kerültek telepítésre, ennek részletes leírását hallgatótársam, Balogh Ádám dolgozata tartalmazza. Mindkét honeypot rendszer Debian operációs rendszeren, Docker konténer alapon került telepítésre. A Honeytrap egyetlen, minden szolgáltatást tartalmazó szeparált környezetben működik. Ezzel ellentétben a T-Pot minden szolgáltatását külön-külön konténerbe helyezi. A T-Pot telepítési folyamatát felgyorsította, hogy a fejlesztők biztosítanak egy Debian alapú telepítési képfájlt, amely már tartalmaz minden szükséges csomagot, és függőséget a honeypot rendszer telepítéséhez.

5.2. Naplózás

A honeypot szolgáltatások által írt naplófájlok minden esetben a konténereken kívül, azaz a host gépen is tárolásra kerülnek az elérhetőség és biztonsági mentés szempontjából. Ezen adatok egységesen a `/data` mappába kerülnek, minden szolgáltatás külön almappát használ.

5.3. Honeypotok bekötése *Elasticsearch*-be

A naplóadatok honeypotról történő elszállításáért a Filebeat felel, melynek konfigurálását három részre oszthatjuk: input, szűrés és output. Az input során definiáltuk, a honeypotok által gyűjtött logfájlok elérési utját. A Filebeat ezeket a forrásokat figyeli és változás esetén elküldi az új bejegyzéseket. A szűrésben a megvalósítható az adatok csoportokba bontása, amelyet egy, a honeypot nevének megfelelő típus megadásával tettünk meg. Végezetül az outputban kerül beállításra, hogy pontosan hova kerüljön továbbításra az összegyűjtött és felcímkézett adat. Lehetőségünk van közvetlenül az Elasticsearchbe juttatni, vagy további feldolgozásra a Logstashbe küldeni. Utóbbi mellett döntöttünk, mivel voltak olyan bejegyzések, amelyeket a Filebeat nem tudott különálló mezőkre bontani.

A Logstash a Filebeathez képest sokkal intenzívebb, precízebb szűrésre és feldolgozásra képes, szerkezetünk viszont egészen hasonló. A Logstash is meglehetősen sok forrás típust különböztet meg, amelyből - az előbbiekben leírtak alapján kikövetkeztethető módon - a Filebeat által biztosított *beat* típust alkalmaztuk. A forrás variáns megadása után, a korábban hozzárendelt típus alapján további szűrést végezhetünk az adathalmazon. Meghatározhatjuk, hogy a már független mezőkben szereplő adat milyen formátumot kövessen (pl. időbélyeg) vagy esetleg módosíthatjuk a mezők nevét, hozzáfűzhetünk tartalmukhoz. Ahogyan esetünkben is, többsoros bejegyzések keletkezése során, úgynevezett *grok* mintát kell létrehoznunk, amely alapján a Logstash elvégezheti a mezőkre bontást. A minta leköveti a bejegyzés szerkezetét és feldarabolja a karakterláncot, illetve ezen részekből új mezőket hoz létre. A mintának tartalmaznia kell az egyes részekhez tartozó új mezők nevét, a benne tárolt adat

típusát[10]. A feldolgozott, teljes egészében cellákra bontott adat az Elasticsearchbe kerül.

```
input {
    file {
        path => "/tmp/access_log"
        start_position => "beginning"
    }
    }

    filter {
    if [path] =~ "access" {
        mutate { replace => { "type" => "apache_access" } }
        grok {
        match => { "message" => "%{COMBINEDAPACHELOG}" }
        }
    }
    date {
        match => [ "timestamp" , "dd/MMM/yyyy:HH:mm:ss␣Z" ]
    }
    }

    output {
    elasticsearch {
        hosts => ["localhost:9200"]
    }
```

5. ábra. Logstash konfiguráció minta [11]

Az átláthatóság és hatékony kereshetőség érdekében az Elasticsearchben meg kell határozni, hogy a beérkező rekordok melyik indexhez kerüljenek hozzárendelésre. Az indexeket sablonok létrehozásával könnyen kezelhetjük, ezek definiálják mi szerepelhet mezőikben. Honeypot adatait - hasonló formátumuk és egyező célfelhasználásuk miatt - egyetlen indexhez rendeltük, melyek így már egységesen szűrhetőek rekordjaik alapján, valamint konzisztens alapot bizosítanak a rájuk épülő szolgáltatásoknak.

6. ábra. Egy index bejegyzései a Kibana felületén

5.4. Adatok vizualizálása és szűrése

Az adatok vizualizálását a Kibana végzi, mely lehetőségeket nyújt az Elasticsearchben rendszerezett adatok irányítópultokba való rendezésére. A T-Pot rendszere alapértelmezetten biztosít ilyen irányítópultok készítéséhez való objektumokat, amelyeket importáltunk a Kibanába. Ezt természetesen igény szerint testreszabhatjuk, valamint az indexelt adatokra építve teljesen új is készíthető.

A irányítópultok panelekből épülnek fel, amelyek egy, kettő vagy több mező adattartalmának összerendelésével hozhatóak létre. Ez a kimutatási felület sokféle formát ölthet, támogatott a különféle oszlop-, kör- és vonaldiagramok készítése, továbbá adatainkat automatikusan táblázatba rendezheti vagy akár - geolokációs adatok esetén - elhelyezheti egy térképen. Amennyiben nem vagyunk elégedettek a Kibana alapértelmezéseivel, manuálisan elvégezhetjük a panelek finomhangolását, szűrést végezhetünk, állíthatjuk a diagrammok osztását és természetesen a színezetét.

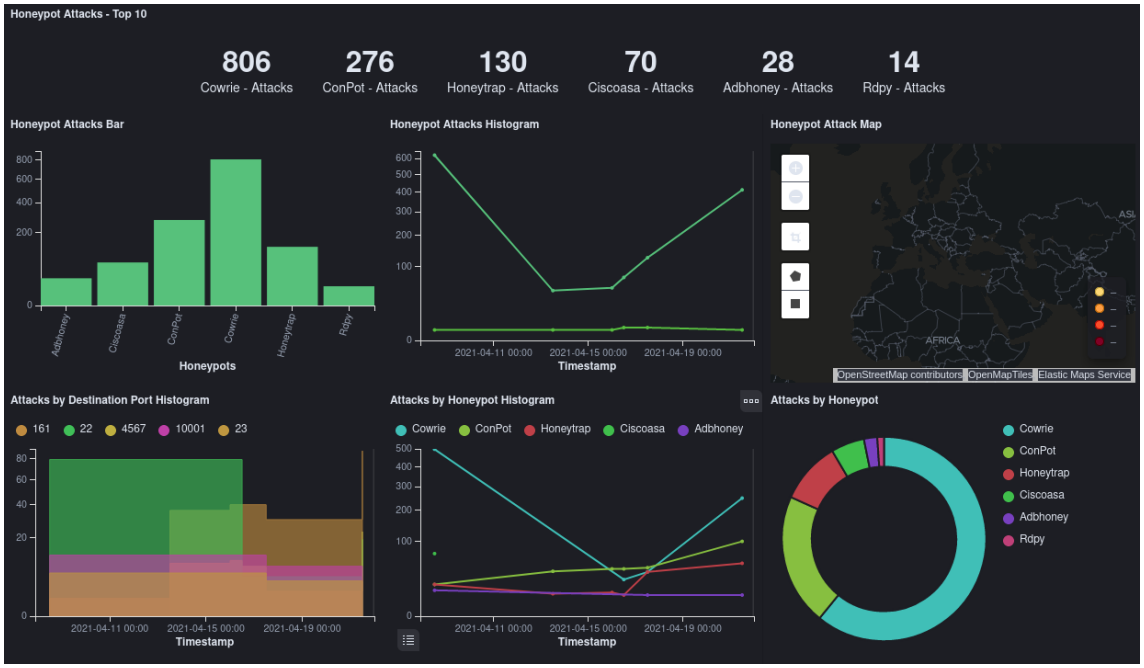
A panelek elrendezése igények alapján tetszőlegesen állítható, az Elasticsearch bármely indexének rekordjai felhasználhatók egy irányítópultban. Honeypot rendszerünk esetében több ilyen nézetet is létrehoztunk, amelyek eltérő szempontból mutatják az összegyűjtött adatot. A nézetek között szerepel olyan, amely szemlélteti a teljes honeypot rendszer hozzávetőlegesen minden adatát (*lásd 8. ábra*), ennek célja, hogy egy pillantás alatt észrevehető legyen bármely alrendszert ért támadás, illetve egy összesítet képet ad az átlag támadások formájáról, esetleg helyéről.

A támadók technikáinak részletes elemzésehez egy sokkal inkább szűrt adathalmazból érdemes kimutatásokat készíteni. A szűrés alapja, lehet egyetlen honypotra vagy akár egy konkrét szolgáltatásra vonatkozó, így sokkal pontosabb képet kaphatunk arról mivel is próbálkozott a betokodó. Ez az egyes honeypotoknál a szolgáltatások különbsége miatt meglehetősen eltérő eredményt hozhat,de egy erre megfelelő példa a 14. ábrán tekinthető meg a következő fejezetben.

7. ábra. Az irányitópultokat alkotó panel készítése

5.5. Riasztások konfigurálása

Mivel a honeypot alkalmazásának legnagyobb előnye, hogy időt nyerhetünk a támadókkal szemben, ezért célszerű, hogy bizonyos eseményekről azonnal értesüljünk egy esetleges támadás vagy támadási kísérlet alkalmával. Ehhez nem elég, ha van egy irányópult amelyet folyamatosan figyelünk, hiszen ebben az esetben előfordulhat, hogy a vizualizációs panelek közül épp egy másikat figyelünk. A figyelem felkeltésére riasztásokat alkalmazunk. Az általam létrehozott riasztási rendszer az ELK stackben már jelenlévő naplóeseményekre épül, és bizonyos időintervallumonként, leellenőrzi a bejegyzéseket a meghatározott riasztási szabályok alapján. Amennyiben létrejön olyan bejegyzés, amely a szabálynak megfelelő, akkor küldd egy riasztást egy meghatározott csatornára. Erre célra én az *Elastalert*[12] alkalmazást használtam, amely egy webhook segítségével juttatja el az ilyen riasztás-üzeneteket az egyetemi Microsoft Teams egy csatornájára. Ezáltal gyors és figyelemfelkeltő, akár hangjelzéssel járó értesítés keletkezik, amely, tartalmazza a riasztási szabálynak megfelelő naplóbejegyzés meghatározott mezőit. Az Elastalert komponens - a honeypotokhoz hasonlóan - konténerben fut az ELK hoston, és lokálisan hozzáfér az Elasticsearch szolgáltatási portjához. Ezen keresztül tud lekérdezéseket beküldeni és a kapott válasz, valamint az érvényben lévő szabályok alapján eldönti, hogy kell e riasztást indítani. A konfigurációja meglehetősen egyszerű, az érvényesítendő szabályok elérése mellett meg kell adnunk, hogyan és milyen gyakorisággal küldjön kérést az Elasticsearch felé, a 9. ábrán látható egy példakonfiguráció. A szabályok alapját ugyanolyan lekérdezések adják. mint amelyek segítségével a Kibana felületén tudjuk lekérdezni az adatokat. Meghatározandó a riasztás típusa, gyakorisága, a naplóbejegyzéseket tartalmazó index az Elasticsearchben, illetve a riasztás csatornája. A 10. ábrán egy riasztási szabály szerepel, amely egy adott



8. ábra. Összesített irányítópult a Kibana felületén

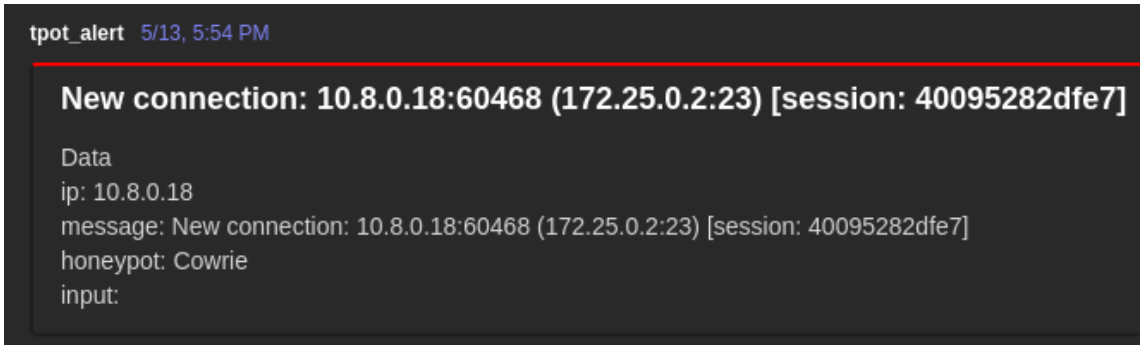
honeypotra, egy adott IP címtartomány felől érkező kéréseket figyeli, és riasztást küld egy Teams csatornára (11. ábra).

```
rules_folder: /usr/share/elastalert/rules
run_every:
    minutes: 1
buffer_time:
    minutes: 15
es_host: localhost
es_port: 9200
verify_certs: false
es_send_get_body_as: GET
verify_certs: false
writeback_index: elastalert_status
alert_time_limit:
    days: 2
```

9. ábra. Elastalert példa konfiguráció [12]

```
type: "frequency"
index: "logstash-*"
is_enabled: true
num_events: 1
timeframe:
  minutes: 5
timestamp_field: "@timestamp"
timestamp_type: "iso"
alert_text_args:
  - "message"
  - "src_ip"
  - "type"
  - "input"
filter:
  - query:
      query_string:
        query: "type: Cowrie AND src_ip: 10.8.*"
alert:
  - ms_teams
ms_teams_alert_summary: Alert
ms_teams_webhook_url:
'https://obudaiegyetem.webhook.office.com/webhookb2/xyz'
```

10. ábra. Részlet egy Elastalert riasztásból



11. ábra. Egy riasztás az egyetemi Microsoft Teams csatornán

6. Tesztelés

6.1. Módszer kiválasztása

Az elkészült rendszerünk tesztelésére többféle módszer közül is választhatunk. Talán a legidőigényesebb és kevésbé hatékony metódus, ha mi magunk állunk neki kidolgozni a rengeteg támadási formát és ezeket manuálisan végrehajtjuk. Egy optimálisabb esetben bevethetünk egy automatikus eljárást a támadások végrehajtására. Amennyiben az internet felé is elérhetővé tesszük, biztosan megtalálják automata szkennerek, ez is mutathat egyfajta képet a fenyegető veszélyekről, ezzel kapcsolatos tapasztalatainkat a 6.4 fejezetben részletezem.

Ennél azonban egy átfogóbb és életszerűbb eredményeket hozó tesztelést szerettem volna végrehajtani. Arra voltam kíváncsi, hogy mennyire hatékony a honeypot rendszer különböző valós támadókkal szemben, akik különféle metodikával támadnak, más sérülékeny szolgáltatást helyeznek előtérbe és egymástól eltérő idő után próbálkoznak inkább egy másik szolgáltatással vagy módszerrel. Ezen adatok segítségével szerettem volna megbecsülni, hogy egy SOC-ban „védekező” csapatnak mennyi idő nyerhető az első honeypotra érkező kéréstől számítva. Az eredmények alapján további optimalizációs lehetőséget adhat, hogy látjuk mely honeypot-ok a legnépszerűbbek, melyek kötötték le leginkább a támadók figyelmét.

A módszerem, az volt, hogy a honeypot rendszerbe egy „zászlófoglaló” (*capture the flag*) játékot hoztam létre és ezt versenyként került meghírdetésre egyetemi hallgatók számára. Így kellően eltérő felderítési és támadási technikákat figyelhettem meg és ez módszer még tanulási, gyakorlási lehetőséget is biztosított hallgatótársaim számára.

6.2. Tesztelés megvalósítása

A verseny, illetve ezzel együtt az eredményeket szolgáltató tesztkörnyezet a *T-Pot* honeypot rendszerből, valamint három kihívás szolgáltatásból áll. A *Honeytrap* tesztből való kihagyása mellett azért döntöttem, mivel a *T-Pot* is rendelkezik hasonló szolgáltatásokkal, illetve kutatási honeypot lévén nagyobb mennyiségű információval tud szolgálni.

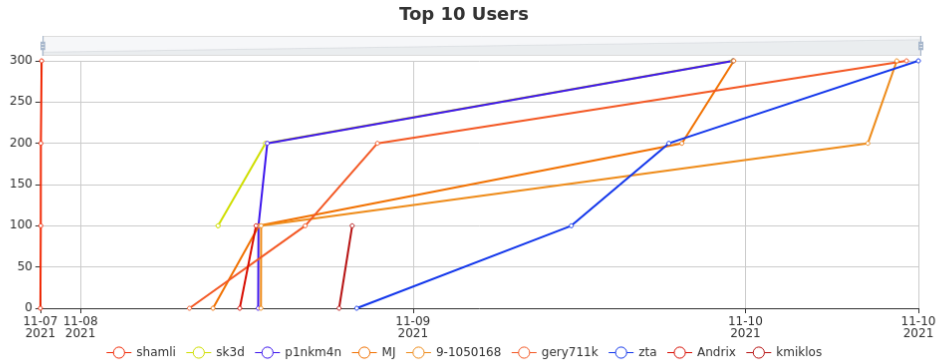
A kihívásokat - amelyeket a honeypotokhoz hasonlóan - szintén egy-egy docker konténerben futtatottak, három széles körben használt szolgáltatás, az ssh, ftp és http sérülékeny konfigurálásával hoztam létre. Mindhárom esetében egy „zászló” (*flag*) került elrejtésre, amely a hallgatók számára meghirdetett játék alapját adja. Ezen kihívásokat a honeypot konténerek közé rejtettem, így a rájuk váró feladat nem csupán annyi volt, hogy valamilyen módon kinyerjék a zászlót egy meghatározott szolgáltatástól. Az igazi megmérettetés az volt, hogy hogyan találják meg ezeket az alkalmazásokat a sok honeypot között.

A tesztrendszer elsődleges tervek szerint kizárólag az egyetemen elérhető formában került telepítésre és így is kezdődött meg a tesztelés. A hibrid oktatás miatti távollét és több lelkes érdeklődő kérésének eleget téve egy későbbi szakaszban felhő alapú szerverre is másolásra került az egyetemen működő környezet. Ebben, a már párhuzamos

működésben egyrészt több hallgatót be lehetett vonni, mivel otthonról is elérték. Másrészt automata szkennerekről is sikerült információt gyűjtenünk, amely jó alapul szolgálhat egy jövőbeli kutatáshoz.

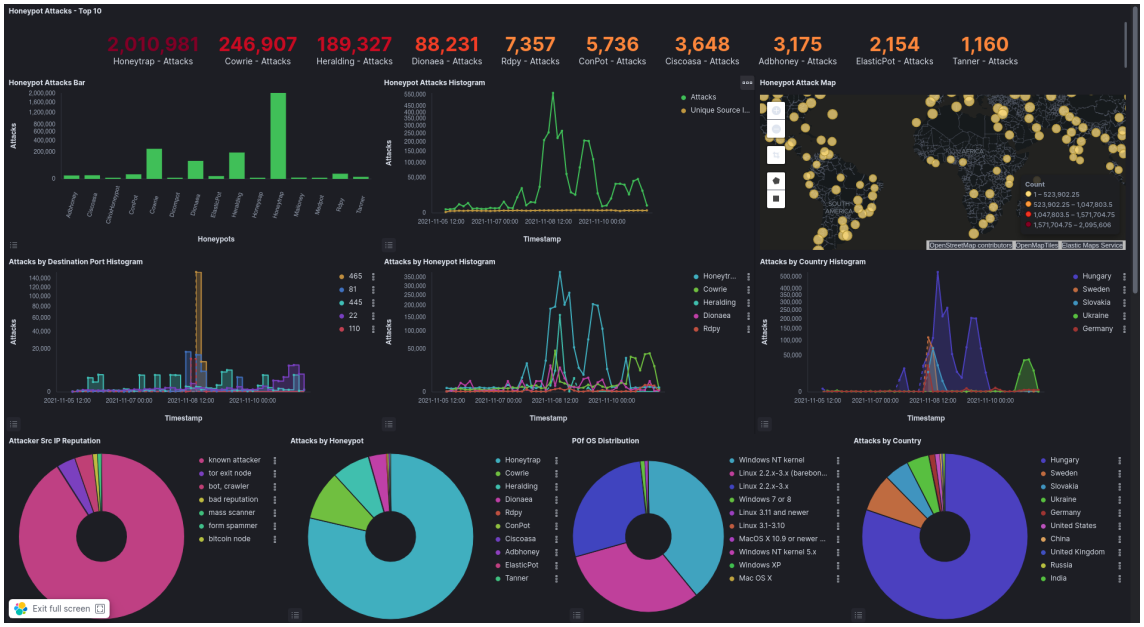
A támadások megfigyelésére a honeypotok naplói mellett, felhasználtam a T-Pot-ban elérhető *Suricata* IDS rendszert is. Ezt azért éreztem szükségesnek, mivel így nem csak az egyes honeypotok konténereibe érkezett kéréseket tudtam tanulmányozni, hanem az összeset, ami a rendszerre érkezett, továbbá így egyszerűen elemezhettem a kihívásokat, amelyeknél problémát okozott a naplóadatok továbbítása.[13]

A verseny menedzselését a CTFd nevezetű nyílt forráskódú capture the flag keretrendszer segítségével végeztem, amely egy webes felületet biztosít egy ilyen esemény szervezésére. Ezen a felületen regisztráltak a hallgatók és nyomonkövethető volt előrehaladásuk a verseny során az egyes zászlók validálásával, a résztvevők feladatmegoldási ütemét szemlélteti a 12. ábra. A teszt két hétig tartott 20 résztvevővel. Kizárólag a honeypotokra több millió kérés érkezett, amely megfelelő nagyságú adathalmazt biztosított az elemzéshez.



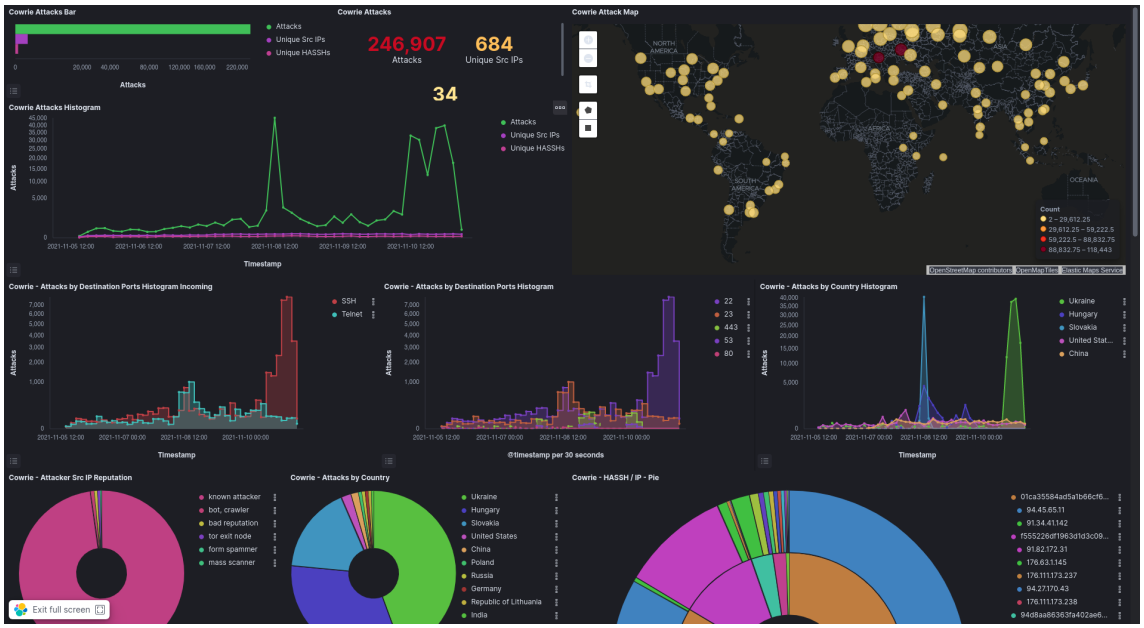
12. ábra. A hallgatók előrehaladása a CTFd felületén szemléltetve

A beérkezett adatok vizsgálatát a Kibana felületének irányítópultjai, az egyes résztvevőkre vonatkozó lekérdezések, valamint az előbbiekben említett CTFd platform összevetésével végeztem el. A T-Pot összesítő dashboardja segítségével átfogó képet kaphattam arról, hogyan alakul a kérések megoszlása az összes vagy akár egyetlen forrás címre szűrve időegységekre lebontva. Összegyűjti, hogy melyek a leggyakrabban megcímzett honeypot szolgáltatások, IP címek geolokációja alapján a világ mely pontjáról lettek detektálva támadások, valamint milyen autentikációs paraméterekkel történt próbálkozás.



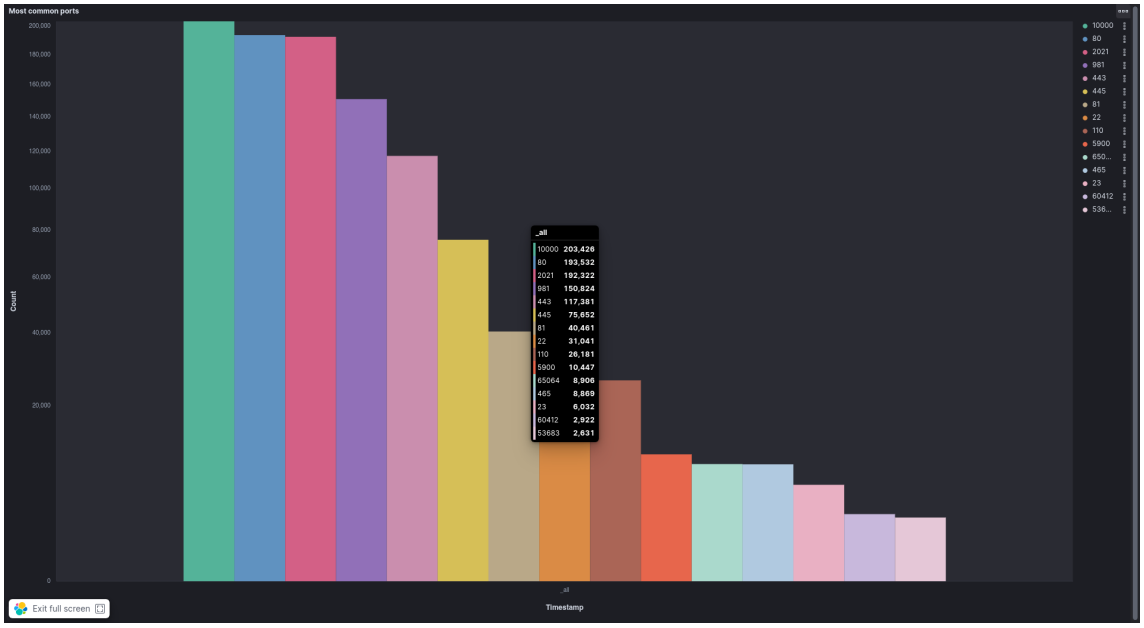
13. ábra. Összesítő irányítópult

Az egyes honeypotokra vonatkozó felület alapján analizálhattam, hogy a szolgáltatásoktól függően milyen támadási módszerek kerültek alkalmazásra. Az egyes kérésekre vonatkozó időbeli eloszlás megmutatta, hogy mennnyi időt töltöttek a hallgatók a szolgáltatások felderítésével, mennyi időre volt szükségük az egyes kihívások megtalálásához. Ennek segítségével tudtam megbecsülni mennnyi idő nyerhető a védekező oldal számára.



14. ábra. Egyetlen honeypot, a Cowrie irányítópultja

Az alapértelmezetten elérhető vizualizációs panelek mellett egyes esetekben egyedieket is létrehoztam. Többnyire akkor volt ilyenre szükség, ha nem kizárólag honeypotra vonatkozó, hanem a verseny egészét érintő adatot szerettem volna szemléltetni. Ilyen például az alábbi 15. ábra, amely tartalmazza a kihívások, valamint azon portok listáját, amelyen nem működött szolgáltatás, azonban érkezett rá kérés.



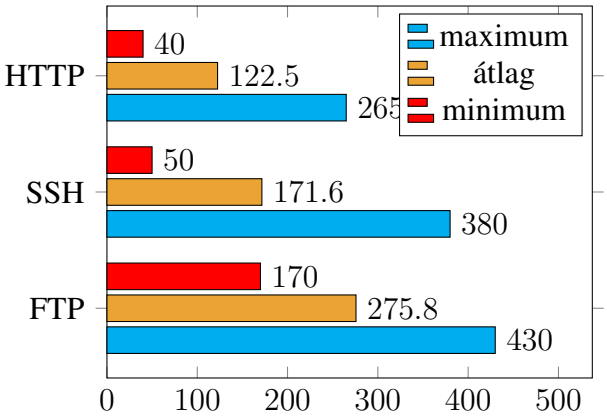
15. ábra. A 15 leggyakrabban megcímzett port

6.3. Eredmények értékelése

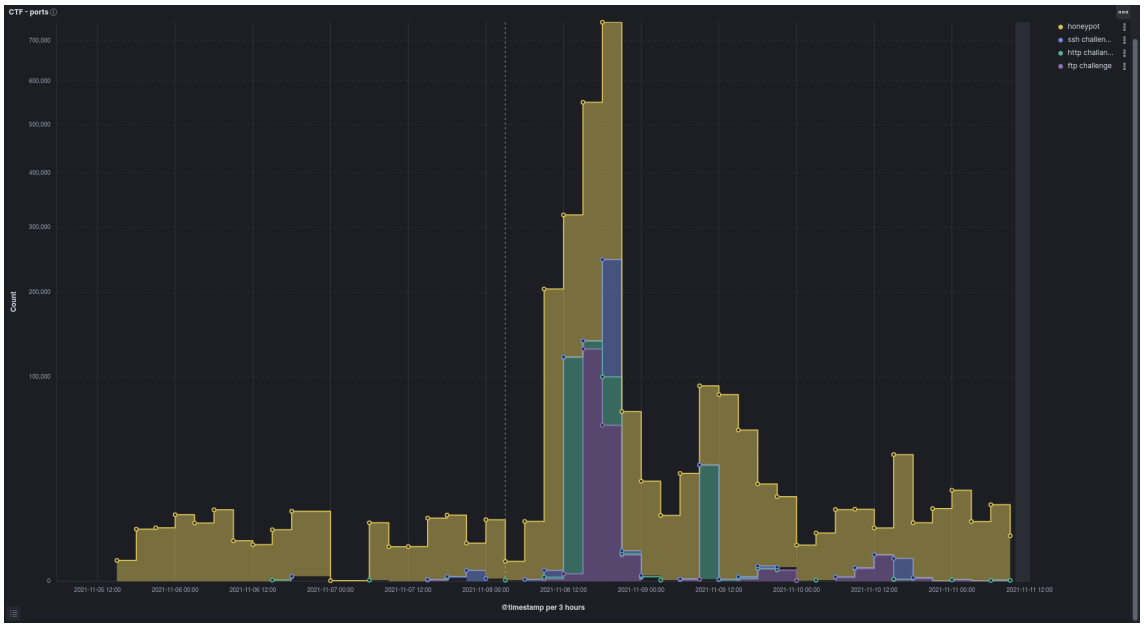
Az eredmények értékelése során az egyik legfontosabb szempont, hogy mennyire hatékonyan sikerült elterelni a támadók figyelmét a lényeges szolgáltatásokról, ez esetben a zászlókról. Ezt a különböző szolgáltatásokkal eltöltött idő mérésével tudjuk legpontosabban lemérni. Ez alapján a honeypotok kifejezetten hatékonynak mondhatók, átlagosan a támadó támadással eltöltött idejének csupán tíz százalékában foglalkozott a kihívásokat futtató szolgáltatásokkal, ez egyben azt is jelenti, hogy a támadás teljes ideje közel tízszeresére növekedett. Ez már önmagában elkönyvelhető sikerként, hiszen egy éles helyzetben a honeypot a támadás észlelésén túl értékes perceket, órákat nyerne a SOC üzemeltetőinek.

Az erdemények értékelése elsősorban az ELK stack Kibana komponense segítségével történt, azonban hasznos információval szolgált például az előbb említett CTFd felület is. Honeypotokra érkező kérések és az egyes hallgatók CTF megoldásainak összevetésével az volt megállapítható, hogy mennyi ideje lehet a biztonsági szakértőknek egy SOC-ban az első támadást jelző riasztástól számítva. A 16. ábra szemlélteti, hogy még a leggyorsabb résztvevőnek is 40-50 percre volt szüksége a megfeleő szolgáltatás megtalálásához és feltöréséhez a honeypotok között. Ez az idő már önmagában sok mindenre elég, de átlagosan elmondható, hogy ezáltal a védekező oldal számára akár 2-3 óra is nyerhető

a felkészülésre.



16. ábra. Zászló megtalálásáig eltelt idő (percben)



17. ábra. Kérések időbeli eloszlása honeypot szolgáltatások és kihívások között

A honeypotok további célja, a támadás módjának megismerése. Erről szintén rengeteg adatot tudtam kinyerni. A legnépszerűbb támadás/felderítési technika a portok letapogatása volt, de ez nem meglepő, hiszen ez a legelterjedtebb módja a felderítésnek. További gyakori támadás volt a bruteforce, amellyel rengetegen kiderítették a szándékosan gyenge felhasználónév - jelszó párosokat. Már kevésbé gyakran, de többször eszleltem hallgatók által használt automatizált scripteket, amely pár másodperc alatt végigpróbált több száz, az adott szolgáltatásra jellemző sérülékenységet.

További érdekes megfigyelés volt az UDP protokoll mellőzése, a forgalom csupán 0.2 százalékát tette ki. Ez egyben azt is jelenti, hogy az UDP protokollt használó

szolgáltatásokat a támadók többsége fel sem fedezte. Ez egyrészt érthető, hiszen a legismertebb szolgáltatások TCP-n keresztül üzemelnek, másrészt furcsa, hogy a legtöbb támadó még csak egy udp portletapogatást sem indított, emiatt több szolgáltatásról nem is tudott.

6.4. Automata szkennerek

Az internetet fürkésző automatikus szkennereknek hála a világ számos pontjáról detektáltunk felderítési és/vagy támadási mechanizmust. Ezekről tapasztalataim alapján általánosságban elmondható, hogy egy támadás során meglehetősen hasznosak lehetnek a egyes rendszerek feltérképezésére, illetve a kezdeti információgyűjtésre. Többnyire egy jól megírt szkript formájában működnek, amelyek specifikus szolgáltatásokat keresnek, illetve meghatározott támadási formákat vetnek be ezek feltörésére, azonban nem fedik le egy tényleges támadó képességeit. Egy megfelelően konfigurált hálózatra nem jelentenek veszélyt, azonban hiányosságok esetén könnyen megtalálhatják annak gyengepontjait. Ezen adatok részletesebb értékelése a jövőben megvalósítandó tervek között szerepel.

18. ábra. Világtérkép a honeypotra érkező támadásokról

7. Összegzés

A hagyományos védelmi megoldások önmagukban már nem elegendőek az egyre bővülő IT infrastruktúrák elleni támadások megelőzősére. Melletük célszerű komplexebb megoldásokat is alkalmazni, amely segíti a védekező csapat munkáját, ilyen lehet például egy honeypot alkalmazása. A tesztelés során gyűjtött adatok elemzése megmutatta, hogy milyen előnyökkel jár egy honeypot rendszer üzemeltetése egy IT infrastruktúrában. Segítségével már a valós támadás kezdete előtt értesülhetünk a betolakodókról és amíg a honeypot leköti a figyelmüket, felkészülhetünk a valós rendszerek elleni kísérletekre.

Dolgozatomban egy ilyen eszközt felhasználva szerettem volna megmutatni, hogy az általa gyűjtött adatok hogyan használhatóak fel, milyen szolgáltatások építhetőek rájuk és mennyire működnek hatékonyan egy valóshoz egészen hasonló környezetben. A Security Operation Center szerinti követlményeknek eleget téve, egy meglévő rendszer komponenseinek felhasználásával és általam implementáltak segítségével célom volt létrehozni egy átlátható és könnyen kereshető adathalmazt, amely felgyorsítja a támadási technikák elemzését. Továbbá azonnali értesítés lehetőségére törekedtem, mely felhívja a figyelmet egy támadási kísérletre. Az implemtált megoldások mellett szerettem volna megmutatni, hogy hogyan használható egy honeypot a támadók technikáinak elemzésére és egy szimulált vállalati infrastruktúrában milyen szempontok szerint próbálják feltérképezni a hálózatot, milyen szolgáltatásokat részesítenek előnyben.

Az implementált megoldásokat akár egy valós infrastruktúrában is felhasználhatónak, az elemzés során elért eredményeket pedig sikeresnek gondolom. A honeypotok rendkívül hasznos eszköznek bizonyoltak, és a hagyományos védelmi megoldások mellett tovább növelhetik egy rendszer biztonságát.

Természetesen a létrejött rendszer rengeteg féleképpen továbbfejleszthető. A honeypotok a valós környezethez igazodva konfigurálhatóak, hogy így teljesen beleolvadjanak az infrastruktúrába. Tovább javítható a honeypot rendszer hatékonysága gépi tanulás alkalmazásával, amennyiben ennek segítségével képesek lehetünk meghatározott bemeneti paraméterek (például beérkezett kérések száma) segítségével dinamikusan változtatni a honeypotok szolgáltatását, tartalmát, elérhetőségét, és a futtatok példányok számát. A csapdarendszer hitelesebbé tehető, az egyes honeypot szolgáltatások között kommunikáció szimulálásával. A vizualizáció és a riasztási rendszer igény szerint bővíthető további irányítópultokkal valamint szűrési- és riasztási szabályokkal az infrastruktúrában működő szolgáltatásoknak megfelelően. Az adatgyűjtés szemponjából érdemes lenne rendszert nagyobb számú részvevővel is tesztelni, ez közeljövőben megvalósítandó célok között szerepel.

A dolgozatom témája további kutatási projekt keretében került továbbfejlesztésre, szaktársam és oktatóim közreműködésével. A közös munka eredményeképpen már létrejött két TDK dolgozat, egy cikk a SAMI konferenciára és további dokumentumok elkészítése várható a közeljövőben.

A honeypotok szolgáltatási rendszerének megtervezése és implementálása mérföldkőhöz érkezett, remélem a jövőben még sok érdekes kutatásnak szolgálhat alapjául.

8. Summary

Traditional defence solutions alone are no longer enough to prevent attacks on ever-expanding IT infrastructures. In addition, it is advisable to use more complex solutions that help the work of the defensive team, such as the use of a honeypots. Analysis of the data collected during testing showed the benefits of operating a honeypot system in an IT infrastructure. It allows us to be aware of intruders before the real attack begins, and as long as the honeypot catches their attention, we can prepare for attempts against real systems.

In my dissertation, I wanted to use such a tool to show how the data it collects can be used, what services can be built on it, and how efficiently they work in a real-world environment. Meeting the requirements of the Security Operation Center, my goal was to create a transparent and easy-to-search data set that accelerates the analysis of attack techniques using the components of an existing system and the ones I implemented. I also sought the possibility of an immediate notification to draw attention to an attempted attack. In addition to the implemented solutions, I wanted to show how a honeypot can be used to analyze attacker techniques and what aspects of a simulated enterprise infrastructure they are trying to map the network to, what services they prefer.

I consider the implemented solutions to be usable even in a real infrastructure, and the results obtained during the analysis to be successful. Honeypots have proven to be an extremely useful tool and can further enhance the security of a system in addition to traditional defence solutions.

Of course, the resulting system can be improved in many ways. Honeypots can be configured to adapt to the real environment to fully integrate into the infrastructure. The efficiency of the honeypot system can be further improved by using machine learning, as it allows us to dynamically change the service, content, availability, and number of instances of honeypots using specific input parameters (such as the number of requests received). The trap system can be made more authentic by simulating communication between each honeypot service. The visualization and alarm system can be expanded as required with additional dashboards, filtering and alarm rules according to the services operating in the infrastructure. From the point of view of data collection, it would be worthwhile to test the system with a larger number of participants, which is one of the goals to be achieved in the near future.

The topic of my dissertation was further developed within a further research project with the participation of my classmate and lecturers. As a result of the joint work, two TDK dissertations, an article for the SAMI conference and further documents are expected in the near future.

The design and implementation of the honeypot service system arrived at a milestone, I hope it can serve as a basis for much more interesting research in the future.

9. Köszönetnyílvánítás

Köszönettel tartozom, Vörösné Dr. Bánáti Annának, a dolgozattal kapcsolatos szakmai és formai kérdésekben való segítségnyújtásért, valamint azért a hihetetlenül nagy tudásért amivel rendelkezik, és igyekezett ebből még többet átadni az évek során.

Továbbá szeretnék köszenetet mondani külső konzulensemnek és volt munkatársamnak, Szarvák Anikónak. Ő tette lehetővé, hogy a dolgozatban részletezett rendszer kipróbálásra kerülhessen vállalati környezetben. Szakmai tapasztalatival, tanácsaival és segítő szándékú kérdéseivel egy teljesen új látásmódot sikerült átadnia.

Végezetül, a dolgozat nem jöhetett volna létre szaktársam, munkatársam és barátom, Balogh Ádám nélkül, akivel rengeteg álmatlan éjszaka árán sikerült az egyes közös projektek megvalósítása. Közös kutatásunk eredményeként emelném ki TDK 2. helyezésünket, amely dolgozattal OTDK-ra is továbbjutottunk.

Ábrák jegyzéke

Hivatkozások

[1] Nitin Naik és Paul Jenkins. „Discovering hackers by stealth: Predicting fingerprinting attacks on honeypot systems”. *2018 IEEE International Systems Engineering Symposium (ISSE)*. IEEE. 2018, 1–8. old.

[2] Munk Sándor. „Információbiztonság vs. informatikai biztonság”. *Hadmérnök különszám. A Robothadviselés* 7 (2007).

[3] Rasmi-Vlad Mahmoud és Jens Myrup Pedersen. „Deploying a University Honeypot: A case study”. *CEUR Workshop Proceedings*. 2443. köt. CEUR Workshop Proceedings. 2019, 27–38. old.

[4] Júlia Murınová. „Application log analysis”. *Relatório técnico, Masarykova Univerzita Fakulta Informatiky* (2015).

[5] Moukafih Nabil és tsai. „SIEM selection criteria for an efficient contextual security”. *2017 International Symposium on Networks, Computers and Communications (ISNCC)*. IEEE. 2017, 1–6. old.

[6] Afsaneh Madani, Saed Rezayi és Hossein Gharaee. „Log management comprehensive architecture in Security Operation Center (SOC)”. *2011 International Conference on Computational Aspects of Social Networks (CASoN)*. IEEE. 2011, 284–289. old.

[7] *HoneyTrap Documentation*. 2021. URL: https://docs.honeytrap.io/ (elérés dátuma 2021. 04. 18.).

[8] telekom-security. *T-Pot honeypot*. 2021. URL: https://github.com/telekom-security/tpotce (elérés dátuma 2021. 04. 22.).

[9] Elasticsearch B.V. *Elastic Stack*. 2021. URL: https://www.elastic.co/ (elérés dátuma 2021. 04. 18.).

[10] StreamSets. *Grok Patterns*. 2021. URL: https://streamsets.com/documentation/datacollector/latest/help/datacollector/UserGuide/Apx-GrokPatterns/GrokPatterns_title.html (elérés dátuma 2021. 04. 18.).

[11] Elasticsearch B.V. *Logstash configuration examples*. 2021. URL: https://www.elastic.co/guide/en/logstash/current/config-examples.html (elérés dátuma 2021. 04. 18.).

[12] Yelp. *Elastalert documentation*. 2021. URL: https://elastalert.readthedocs.io/en/latest/ (elérés dátuma 2021. 04. 18.).

[13] OISF. *Suricata*. 2021. URL: https://suricata.io/ (elérés dátuma 2021. 11. 02.).