VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
FACULTY OF INFORMATION TECHNOLOGY

ÚSTAV INFORMAČNÍCH SYSTÉMŮ
DEPARTMENT OF INFORMATION SYSTEMS

DETEKCE BEZPEČNOSTNÍCH INCIDENTŮ V POČÍTAČOVÉ SÍTI NEMOCNICE

DETECTION OF SECURITY INCIDENTS IN HOSPITAL COMPUTER NETWORK

SEMESTRÁLNÍ PROJEKT
TERM PROJECT

AUTOR PRÁCE **JIŘÍ PISK**
AUTHOR

VEDOUCÍ PRÁCE **Ing. PETR MATOUŠEK, Ph.D., M.A.**
SUPERVISOR

BRNO 2022

Vysoké učení technické v Brně
Fakulta informačních technologií

Ústav informačních systémů (UIFS) Akademický rok 2021/2022

Zadání bakalářské práce

Student: **Pisk Jiří**
Program: Informační technologie
Název: **Detekce bezpečnostních incidentů v počítačové síti nemocnice**
Detection of Security Incidents in Hospital Computer Network
Kategorie: Počítačové sítě

Zadání:

1. Seznamte se s použitím honeypotů pro detekci bezpečnostních incidentů. Prozkoumejte honeypot určené k detekci průniku do sítě (SSH, Telnet), šíření malware, spamů, webové služby, NTP, VoIP či detekci průniků do sítě Wifi (např. T-pot, mnh).
2. Popište počítačovou síť v nemocnici, její topologii, používané služby a aktuální způsob implementace zabezpečení.
3. Navrhněte nasazení vybraných honeypotů do infrastruktury nemocnice. Uveďte jaké služby a v jaké části sítě chcete sledovat.
4. Analyzujte data získaná z honeypotů. Vytvořte automatizovaný způsob detekce zájmových dat z logů honeypotu.
5. Diskutujte nalezené hrozby a navrhněte způsob rozšíření zabezpečení počítačové sítě nemocnice proti nalezeným hrozbám.
6. Zhodnoťte přínos své práce i možnost uplatnění v praxi.

Literatura:

- Juan David Guarnizo, et al: SIPHON: Towards Scalable High-Interaction Physical Honeypots. In *Proceedings of the 3rd ACM Workshop on Cyber-Physical System Security* (*CPSS '17*). New York.
- Jouni Ihanus, Tero Kokkonen: Modelling Medical Devices with Honeypots, Internet of Things, Smart Spaces, and Next Generation Networks and Systems, Springer, 2020.
- Guidance on Cybersecurity for medical devices, Medical Device Coordination Group, MDCG 2019-16 rev. 1, 2019.
- Principles and Practices for Medical Device Cybersecurity, International Medical Device Regulators Forum, 2020.
- Proactive Detection of Security Incidents. Honeypots, ENISA, 2012.

Podrobné závazné pokyny pro vypracování práce viz https://www.fit.vut.cz/study/theses/

Vedoucí práce: **Matoušek Petr, Ing., Ph.D., M.A.**
Vedoucí ústavu: Kolář Dušan, doc. Dr. Ing.
Datum zadání: 1. listopadu 2021
Datum odevzdání: 11. května 2022
Datum schválení: 26. října 2021

Abstrakt

Tato práce se zabývá honeypoty jako systému brzké detekce útočníka v produkční síti Nemocnice Jihlava, analýzou získaných dat a porovnání dostupných řešení pro nasazení honeypotového systému. V praktické části je nasazeno několik instancí open source honeypot platformy TPot včetně jedné instance určené pro monitoring celého systému. Nazazený systém je dále testován automatizovanými penetračními testy a na základě výsledků testování je navržen a implementován systém ohlašování detekovaných incidentů.

Abstract

This work explores honeypots as an early threat detection system in the production network of Jihlava hospital, analysis of the data collected from those honeypots as well as a comparisson of currently available solutions for honeypot deployment. As part of the practical section multiple instances of TPot (an open source honeypot platform) are deployed as well as one instance serving as a monitor for the whole system. This implementation is then tested using automated penetration testing tools and the results used to design and implement automatic alerts to detected incidents.

Klíčová slova

Honeypot, Honeynet, Informační bezpečnost ve zdravotnictví, detekce malware, analýza bezpečnostních incidentů

Keywords

Honeypot, Honeynet, Information security in healthcare, malware detection, security incident analysis

Citace

PISK, Jiří. *Detekce bezpečnostních incidentů v počítačové síti nemocnice*. Brno, 2022. Semestrální projekt. Vysoké učení technické v Brně, Fakulta informačních technologií. Vedoucí práce Ing. Petr Matoušek, Ph.D., M.A.

Detekce bezpečnostních incidentů v počítačové síti nemocnice

Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně pod vedením pana doktora Matouška. Další informace mi poskytl Bc. Petr Málek, DiS vedoucí oddělení správy sítí a hardware Nemocnice Jihlava. Uvedl jsem všechny literární prameny, publikace a další zdroje, ze kterých jsem čerpal.

.....
Jiří Pisk
9. května 2022

Poděkování

Tímto bych rád poděkoval doktoru Matouškovi, vedoucímu této práce, za tpělivost a pochopení při komplikacích a za podněty a rady při konzultacích. Dále děkuji Nemocnici Jihlava za umožnění testování praktické části práce. Obrovský dík patří zejména vedoucímu oddělení správy sítí a hardware Petru Málkovi za podporu a motivaci.

Obsah

Kapitola 1

Úvod

Síťová bezpečnost je téma, které je v dnešní době neustále aktuální a podstatné pro jakoukoliv entitu připojenou do internetu. Je to oblast, na kterou se klade stále větší důraz ať už v multimiliardových společnostech tak u malých podnikatelů či ve státním sektoru. A vyjímkou nejsou ani nemocnice. Naopak po vlně kyberútoků cílících právě na nemocnice v posledních letech je síťová bezpečnost mezi hlavními tématy takřka každé nemocnice v České republice. Jedním z moderních trendů bezpečnosti je technologie klamu (Deception technology), která má svůj počátek v takzvaném honeypotu. Honeypot je lidově řečeno jakási past na útočníka v síti. Jedná se o zdánlivě zranitelné systémy které se útočníkovi prezentují jako snadný cíl ovšem ve skutečnosti zaznamenávají jeho akce a varují správce sítí o útočníkově přítomnosti.

Cílem této práce je prozkoumat dostupné honeypoty a navrhnout jejich nasazení v počítačové síti nemocnice Jihlava za účelem detekce útočníků, kterým se podařilo do této sítě proniknout. Snahou je navrhnout řešení, které bude pokrývat nejběžněji využívané služby nemocnice, bude snadno udržovatelné a rozšiřitelné. Výstupy z těchto honeypotů musí být v rámci řešení lehce shromažďovány, analyzovány, zpracovávány a předávány dalším nástrojům.

V první kapitole této práce je vysvětlen princip funkce honeypotů a jejich dělení dle míry interakce, kterou útočníkovi nabízí. Dále je prozkoumána řada honeypotů volně dostupných jako otevřený software, které je možné pro účely práce využít. Blíže jsou rozepsány platformy pro nasazení a správu honeypotů. Zmíněna jsou i komerční řešení společností FortiNet a Attivo Networks, se kterými se v rámci práce podařilo vyjednat dočasnou bezplatnou ukázku pro porovnání. Druhá kapitola práce nastiňuje současný stav a prostředky zabezpečení v nemocnici a dále se věnuje návrhu a popisu implementace nasazení honeypotové platformy Tpot do několika vybraných segmentů nemocniční sítě. Ve třetí kapitole probíhá testování a analýza výsledků tohoto řešení pomocí několika nástrojů pro automatické penetrační testování. Výsledky této kapitoly jsou nadále využity pro návrh systému zpracování logů honeypotů a automatizovaného hlášení incidentů, kterému se věnuje poslední kapitola.

Výsledkem práce by tedy měl být snadno udržovatelný a rozšiřitelný systém detekce útočníků v síti pomocí honeypotů, který se stane stálou součástí zabezpečení nemocnice a bude dále využíván správcem bezpečnosti.

Kapitola 2

Detekce bezpečnostních incidentů pomocí honeypotů

Honeypoty jsou aplikace či zdroje využívané k detekci, analýze a případně zábraně nepovoleného vniknutí útočníka do sítě[10]. Základním principem je vytvoření jakési "pasti" na případné útočníky, tedy systému, který se tváří jako potenciální cíl (často úmyslně zranitelný), ovšem na první pohled vypadající jen jako další počítač, server, aktivní prvek nebo IoT zařízení v síti. Zdroje a služby, které takový systém nabízí, nejsou ve skutečnosti využívány v produkčním provozu a vzniká tak předpoklad, že jakákoliv entita pokoušející se o spojení a interakci s honeypotem je entitou podezřelou[7]. Právě tento předpoklad, je jednou z výhod využití honeypotů, neboť je velice nízká pravděpodobnost falešného poplachu narozdíl od IDS systémů, které musí analyzovat běžný provoz v síti[6]. Pokud tedy honeypot ohlásí aktivitu, správce si může být jistý, že se jedná o problém. V lepším případě jde o chybnou konfiguraci některého z jiných systémů v síti, v tom horším se jedná o útočníka.

Veškeré interakce jsou honeypotem monitorovány a zaznamenávány pro pozdější analýzu. V závislosti na typu a konfiguraci honeypotu mohou vybrané interakce vyvolat varování pro správce systému, nebo spustit automatizované procesy k odklonění či zablokování opakované aktivity útočníka v síti. Monitorování a záznam veškeré aktivity však může být jednou z nevýhod využití honeypotů, protože během útoků může vznikat obrovské množství těchto záznamů, jak lze pozorovat v kapitole 4. Pro praktické využití je tedy nutné tyto data automatizovaně filtrovat a zpracovávat. Nicméně takto zpracovávat je nutné pouze interakce cílící na honeypot samotný. Honeypot je tedy zcela nezávislý celkové zátěži sítě do které je umístěn[9]. Tento fakt ovšem rovněž přichází i s jednou z velkých nevýhod honeypotů, kterou je fakt, že pokud se útočník interakci s honeypotem zcela vyhne, pak jej honeypot nikdy neodhalí. V rámci této práce má ovšem systém honeypotů, sloužit jako doplněk stávajících bezpečnostních systému, nikoliv jejich náhradou.

Jednou z dalších výhod je pak velký výběr dostupných řešení. V dnešní době již existuje celá řada honeypotů a nástrojů pro jejich správu. Rychle vznikají nové volně dostupné a otevřené projekty, které cílí na aktuální zranitelnosti. Příkladem může být například honeypot s názvem Log4pot[1] cílící na zranitelnosti log4j, jiné projekty mohou být roky staré nicméně stále udržované a aktualizované. Rovněž existuje celá řada komplexních honeypotových systémů nabízených jako komerční produkty. Jedná se zejména o produkty dodavatelů software a hardware, jako je například společnost FortiNet, kde je často kladen důraz na provázání s ostatními produkty dané společnosti.

[1] Dostupný z: https://github.com/thomaspatzke/Log4Pot [2.5.2022].

2.1 Dělení honeypotů

Dle míry interakce, kterou honeypoty útočníkovy nabízí je lze dělit do tří základních typů[7]:

- Honeypoty s nízkou úrovní interakce
- Honeypoty se střední úrovní interakce (Hybridní honeypoty)
- Honeypoty s vysokou úrovní interakce

2.1.1 Honeypoty s nízkou úrovní interakce (Low-interaction honeypoty)

Low-Interaction honeypoty, tedy honeypoty s nízkou úrovní interakce, nabízejí útočníkovi pouze omezenou interakci. Nejčastěji se jedná o programy pouze simulující danou službu či zranitelnost, nikoliv o reálné systémy. Výhodou takovýchto honeypotů je především menší náročnost na systémové zdroje a snadná údržba. Honeypot a tím i jeho správce má plnou kontrolu nad útokem, protože útočník nemůže "vybočit" z množiny akcí, na kterou je honeypot připraven. Tato skutečnost je ovšem i největší nevýhodou low-interaction honeypotů, neboť neočekávané chování útočníka obvykle vede k snadnému prozrazení honeypotu. Protože se nejedná o reálný systém, nelze takovýmto honeypotem odhalit případné nové zranitelnosti[7]. Low-interaction honeypoty je tedy vhodné využít především jako sensory pro detekci výskytu útočníka v síti než k analýze útoků či detekce zero-day (nedávno objevené zranitelnosti, které může útočník využít k útoku na systém[2]) zranitelností.

2.1.2 Honeypoty se střední úrovní interakce (Hybridní honeypoty)

Honeypoty se střední úrovní interakce si lze představit jako střední cestu mezi jejich low a high interaction protějšky. Tyto honeypoty poskytují útočníkovi vyšší míru interakce než low-interaction honeypoty většinou ve snaze přimět útočníka, aby pokračoval v dalších krocích útoku. Stále se však jedná pouze o simulaci, nikoliv o reálný systém[9].

2.1.3 Honeypoty s vysokou úrovní interakce (High-interaction honeypot)

Honeypoty s vysokou úrovní interakce jsou zpravidla reálné systémy obohacené nástroji pro detekci a sběr dat o jejich chování a interakcích s útočníkem. Protože se jedná o reálné systémy, je pro útočníka mnohem těžší rozpoznat, že se jedná o honeypot. Tento typ honeypotů nabízí plnou interakci a může tedy odhalit i doposud neznámé zranitelnosti, pokud je útočník při svém útoku využije. Jejich hlavní nevýhodou je mnohem vyšší náročnost na výpočetní zdroje v porovnání s low-interaction honeypoty a to nejen pro udržení chodu celého systému, ale i pro značně větší objem dat, který honeypot sbírá a který je třeba analyzovat. Další nevýhodou pak je nižší míra kontroly nad akcemi útočníka, což může vést ke kompromitaci systému a zneužití honeypotu útočníkem k dalšímu šíření po síti. High-Interaction honeypoty jsou proto nejčastěji využívány k výzkumu a analýze chování útočníků nebo v místech kde je velká pravděpodobnost objevu nových zranitelností[10].

2.2 Honeypoty dostupné jako otevřený software

Honeypotů, které jsou dostupné v rámci otevřeného softwaru, existuje v dnešní době spousta, mnoho z nich je však již není podporováno a cílí na zastaralé verze služeb či dávno opravené zranitelnosti. Při výběru honeypotů pro nasazení je tedy třeba dbát na to, jak jsou jejich

repozitáře aktivní a udržované. Tyto honeypoty také často bývají modulární a je nabízeno více služeb najednou. Pokud má být honeypotů nasazeno více, je tedy nutné vybrat je tak, aby se vzájemně nepřekrývaly.

Mezi poměrně známé patří například low-interaction ssh honeypot Kippo[2] či spíše jeho medium-interaction následovník Cowrie[3]. Známý je také například honeypot Dionaea[4], cílící na zranitelnosti celé řady protokolů a zaměřující se především na zachycení a uchování použitého malware k další analýze, následník honeypotu Nepenthes. Rovněž existuje celá řada honeypotů SMTP zaměřených na boj se spamem jako jsou SpamHole[5] nebo Shiva[6]. Pro simulaci webových služeb lze využít některý z HTTP honeypotů, kterými jsou například Nodepot[7] (honeypot simulující NodeJS aplikaci) nebo třeba Snare[8] s jeho evaluačním nástrojem Tanner[9] fungující v režimu sensor (SNARE) a controller (TANNER) či jejich předchůdce honeypot Glasopft[10].

Najde se i řada dalších honeypotů cílících různé další služby. Honeypot ADBHoney[11] simulující android zařízení, které využívá Android Debug Bridge server, Elasticpot[12] pro předstírání existující služby Elasticsearch [13] (volně dostupný otevřený nástroj pro prohledávání a analýzu textových, numerických ču pozičních dat ve strukturované i nestrukturované formě[3]), RDPy[14] jako falešný klient pro Microsoft Remote Desktop protokol.

Do zvláštní kategorie patří honeypoty SCADA jakým je například honeypot Conpot[15]. Vzhledem k tématu práce nelze jistě opomenout honeypoty cílící na zdravotnická zazřízení, kterými jsou honeypoty Medpot[16] a DICOMPOT[17]. Medpot simuluje protokoly FHIR (Fast Healthcare Interoperability Resources) a HL7 (Health Level 7), což jsou mezinárodně využívané protokoly pro přenos klinických a administrativních dat ve zdravotnicví. DICOMPOT se zaměřuje, jak jeho název napovídá, na mezinárodní standard pro správu a přenos obrazových medicínských dat DICOM (Digital Imaging and Communications in Medicine[1]).

Mezi honeypoty dostupnými jako otevřený software rozhodně najdeme i řadu honeypotů s vysokou úrovní interakce. Předem zmiňovaný honeypot Cowrie může sloužit také jako SSH proxy, čímž se v podstatě mění na vysoko-interaktivní honeypot. Dále je vhodné zmínit nástroj Shadow Daemon [18]. Ačkoliv se jedná spíše o firewall či filter potenciálně škodlivých parametrů příchozích dotazů HTTP pro webové aplikace, lze Shadow Deamon v kombinaci s nástražnou web aplikací považovat za honeypot s vysokou úrovní interakce. Další možností pro vysoko-interaktivní webový honeypot, tentokrát založený na PHP, je nástroj HoneypotCreator, který transformuje jednoduché webové aplikace na vysoko-interaktivní honeypoty.

[2]Dostupný z: https://github.com/desaster/kippo [2.5.2022].
[3]Dostupný z: https://github.com/cowrie/cowrie [2.5.2022].
[4]Dostupný z: https://github.com/DinoTools/dionaea [2.5.2022].
[5]Dostupný z: https://github.com/n1co/SpamHole [2.5.2022].
[6]Dostupný z: https://github.com/shiva-spampot/shiva [2.5.2022].
[7]Dostupný z: https://github.com/schmalle/Nodepot [2.5.2022].
[8]Dostupný z: https://github.com/mushorg/snare [2.5.2022].
[9]Dostupný z: https://github.com/mushorg/tanner [2.5.2022].
[10]Dostupný z: https://github.com/mushorg/glastopf [2.5.2022].
[11]Dostupný z: https://github.com/huuck/ADBHoney [2.5.2022].
[12]Dostupný z: https://github.com/bontchev/elasticpot [2.5.2022].
[13]Dostupný z: https://www.elastic.co/elasticsearch [2.5.2022].
[14]Dostupný z: https://github.com/citronneur/rdpy [2.5.2022].
[15]Dostupný z: https://github.com/mushorg/conpot [2.5.2022].
[16]Dostupný z: https://github.com/schmalle/medpot [2.5.2022].
[17]Dostupný z: https://github.com/nsmfoo/dicompot [2.5.2022].
[18]Dostupný z: https://github.com/zecure/shadowd [2.5.2022].

Data z nich je pak možné analytovat nástrojem HIHAT[19] (High Interaction Honeypot Analysis Tool) vydaný stejnými autory[8].

Výběr je mezi dostupnými honeypoty opravdu veliký a bezpochyby by bylo možné poskládat vhodnou sadu honeypotů pro nemocnici a spravovat každý samostatně. Nicméně jedním z cílů práce je, aby nasazené řešení bylo pokud možno co nejsnáze udržitelné a rovněž lehce rozšiřitelné. Jednou z možností, jak takového řešení dosáhnout, je namísto samostatných honeypotů využít jeden z dostupných nástroju pro nasazení a správu honeypotů. Takovýchto možností se rovněž ve světě otevřeného software najde hned několik. Jedná se o nástroje jako je SIREN[20], Nova[21], HoneyTrap[22], MHN[23] (či novější komunitní verze Community Honey Network[24]) a Tpot[25]. Tyto nástroje jsou jakousi nástavbou nad existujícími honeypoty a ačkoliv nenabízí svoje vlastní honeypoty, poskytují nástroje a rozhraní pro snadnější nasazení a správu těch existujících.

2.2.1 Nova

Nova[26] je nástroj pro konfiguraci, monitoring, nasazení a správu honeypotů vyvinutý společností DataSoft, který má ovšem dostupnou také otevřenou verzi. Nástroj nabízí možnost nasazení řady honeypotů s nízkou úrovní interakce v rámci své komponenty Haystack. Nova poskytuje webové rozhraní pro správu honeypotů, analýzu a vizualizaci nasbíraných dat. Kromě webového rozhraní Nova nabízí také NovaCLI příkazovou řádku. Nova využíva strojové učení ke klasifikaci opakujících se vzorů v síťové komunikaci a může fungovat jako systém IDS (Intrusion Detection System). Rovněž je k dispozici několik způsobů podání varovných hlášení, která může Nova zaslat správci emailem, reportovat pomocí protokolu syslog do nástrojů pro sběr logů, nebo využít svoje webové rozhraní. V době vzniku této práce je však nástroj již značně zastaralý s poslední aktualizací vydanou v roce 2015. Pro využití v praktické části práce se tedy jeví jako nevhodný.

2.2.2 HoneyTrap

Honeytrap[27] je framework pro správu honeypotů nabízející možnost nasazení honeypotů na jeden či více serverů. Využitím agentů pak lze přesměrovat provoz ze vzdálených serverů do centrálního serveru HoneyTrapu. Narozdíl od ostatních zde zmiňovaných HoneyTrap nabízí podporu více operačních systémů (Linux, MacOS a Windows). Nástroj rovněž umožnuje nasazení honeypotů s vysokou úrovní interakce pomocí linuxových kontejnerů (LXC). Jeho základní sadu honeypotů lze rožšířit o existující honeypoty jako například Cowrie či další výše zmiňované. Unikátní vlastností je také možnost provozovat více protokolů na jednom portu pomocí detekce obsahu příchozí komunikace. Honeytrap umožňuje sbírat, filtrovat a dále předávat logy z honeypotů například nástroji Elasticsearch, Kafka, Splunk nebo pouze do souborů či výstupem na konzoli.

[19] Dostupný z: https://github.com/honeynet/HIHAT [2.5.2022].
[20] Dostupný z: https://github.com/blaverick62/SIREN [2.5.2022].
[21] Dostupný z: https://github.com/DataSoft/Nova [2.5.2022].
[22] Dostupný z: https://github.com/honeytrap/honeytrap [2.5.2022].
[23] Dostupný z: https://github.com/pwnlandia/mhn [2.5.2022].
[24] Dostupný z: https://github.com/CommunityHoneyNetwork/CHN-Server [2.5.2022].
[25] Dostupný z: https://github.com/telekom-security/Tpotce [2.5.2022].
[26] Dostupný z: https://github.com/DataSoft/Nova [2.5.2022].
[27] Dostupný z: https://github.com/honeytrap/honeytrap [2.5.2022].

2.2.3 Modern Honey Network

Modern Honey Network (MHN)[28] je platforma pro správu a sběr dat z honeypotů. MHN nabízí databázi pro sběr logů z honeypotů a webové rozhraní pro jejich správu. Prvotní instalace je bez honeypotů, které je třeba doinstalovat pomocí instalačních skriptů. Na výběr je opět několik známějších honeypotů zmíněných výše jako třeba Cowrie či Dionaea. Existuje také oddělená verze MHN s názvem CommunityHoneyNetwork[29] s cílem nástroj dále rozvíjet a udržovat. Bohužel v době vzniku této práce se zdají být obě verze již neudržované.

2.2.4 Tpot

T-Pot[30] je platforma vyvinutá společností T-Mobile využívající řadu známých honeypotů. Zajišťuje snadné nasazení a správu honeypotů s využitím virtualizační platformy Docker a nástroje docker-compose. Hned v základu nabízí několik typů nasazení (sad honeypotů a dalších nástrojů), kterými jsou následující:

- **Standard** - Obsahuje základní sadu nástrojů a známých honeypotů.
- **NextGen** - Obsahuje sadu nástrojů a honeypotů připravovaných pro nové verze Tpotu, které ještě nejsou součástí Standard balíčku.
- **Industrial** - Obsahuje SCADA honeypoty. Určeno pro nasazení do průmyslových sítí.
- **Medical** - Obsahuje zdravotnické honeypoty DICOMPOT a Medpot.
- **Collector** - Balíček sloužící jako centrální server pro sběr dat ze sensorů.
- **Sensor** - Snížené nároky na HW zdroje, protože neobsahuje nástroje pro analýzu dat, ale pouze honeypoty samotné. Vhodné pro nasazení v kombinaci s Collector balíčkem.

Největší výhodou Tpotu je ovšem možnost vytvářet vlastní balíčky (docker-compose soubory), tím jej plně přizpůsobit potřebám nemocnice a přitom stále docílit plné integrace. Honeypoty jsou spouštěny ve virtualizovaných kontejnerech pomocí nástroje docker a pravidelně mazány a opětovně sestaveny z jejich docker obrazů. Doba uchování vlastních dat a logů z honeypotů je delší, v základu 30 dní ovšem lze ji volně přenastavit v konfiguračním souboru nástroje logrotate.

Kromě honeypotů samotných nabízí Tpot několik dalších nástrojů pro snadnější správu a monitoring. Těmi jsou zejména ELK stack[31], tedy kombinace nástrojů Elasticsearch, Logstash a Kibana sloužících ke sběru (Logstash), zpracování (Elasticsearch) a vizualizaci (Kibana) logů nasazených honeypotů. Dále nástroj Cockpit, který poskytuje webové rozhraní pro správu serveru, umožňuje sledovat využití systémových zdrojů, ovládat aktuálně běžící docker kontejnery či ručně spouštět nové, provádět aktualizace a restart systému. Cockpit také nabízí přístup k terminálu přes HTTPS.

[28] Dostupný z: https://github.com/pwnlandia/mhn [2.5.2022].
[29] Dostupný z: https://github.com/CommunityHoneyNetwork/CHN-Server [2.5.2022].
[30] Dostupný z: https://github.com/telekom-security/tpotce [2.5.2022].
[31] Dostupný z: https://www.elastic.co/elastic-stack/ [2.5.2022].

Název	**Popis**	**Nabízené porty**	**Poslední aktualizace**
adbhoney	Android honeypot	5555	6.5.2021
ciscoasa	Honeypot pro cisco adaptive security appliance	5000, 8443	16.8.2018
citrixhoneypot	Honeypot pro detekci zranitelnosti CVE-2019-19781 zařízení Citrix ADC	443	15.1.2020
conpot	ICS/SCADA honeypot s nízkou úrovní interakce vyvýjený jako součást honeynet projektu	80, 102, 161, 502, 623, 1025, 2404, 10001, 44818, 47808, 50100	20.2.2022
cowrie	medium-interaction SSH/TELNET honeypot, nástupce honeypotu Kippo	22, 23	19.3.2022
ddospot	honeypotová platforma pro detekci DDoS útoků která podporuje protokoly DNS, NTP, SSDP, CHARGEN nebo odposlouchání vlastního UDP portu	19, 53, 123, 1900, vlastní	27.12.2020
dicompot	DICOM server honeypot pro zdravotnická zařízení	11112	22.10.2020
dionaea	Honeypot zaměřený na zachycení a uchování použitého malware pro analýzu. Podporuje například protokoly SMB, MSSQL nebo HTTP	21, 42, 69, 135, 443, 445, 1433, 1723, 1883, 3306, 8081	8.2.2021
elasticpot	Elastic search server honeypot	9200	19.8.2020
endlessh	SSH past která útočníkovy posílá nekonečnou odpověď při pokusu o SSH připojení	22	30.4.2021
glutton	Generický SSH a TCP honeypot s nízkou úrovní interakce	22	18.10.2021
heralding	Jednoduchý honeypot sbírající pokusy o přihlašení a přihlašovací údaje. Podporuje SSH, HTTP, HTTPS, SMTP a další.	21, 22, 23, 25, 80, 110, 143, 443, 993, 995, 1080, 5432, 5900	13.11.2021
hellpot	Vtipný HTTP honeypot, který po navázání spojení útočníkovi posílá úryvky z knihy "The Birth of Tragedy (Hellenism and Pessimism)" od Friedricha Nietzche v nekonečné smyčce.	80, 445	1.4.2022
ipphoney	Internet Printing Protocol honeypot simulující internetovou tiskárnu.	631	16.10.2020
log4pot	Honeypot zaměřený na detekci útoků na zranitelnosti Log4j.	80, 443, 8080, 9200, 25565	23.1.2022
mailoney	Jednoduchý SMTP honeypot.	25	13.3.2021

medpot	Honeypot pro protokoly HL7 a FHIR simulující zařízení ve zdravotnictví	2575	1.4.2021
redishoneypot	High-interaction honeypot pro Remote Dictionary Server	6379	23.4.2021
rdpy	Implementace RDP v pythonu obsahující RDP honeypot.	3389	10.4.2020
snare	Honeypot sensor pro webové aplikace umožnující konvertovat existující webové stránky na honeypoty.	80	13.1.2021
tanner	Řídící nástroj pro honeypot snare. Zpracovává události ze snare sensorů a rozhoduje o navazujících akcích.	-	16.1.2022

Tabulka 2.1: Honeypoty dostupné v nástroji Tpot

2.3 Dostupná komerční řešení

Také v komerční sféře existuje mnoho produktů založených na honeypotech. Většina těchto produktů klade důraz na jednoduchost nasazení, komplexní analýzu získaných výsledků a blízkou integraci s ostatními produkty daného výrobce. Při výběru komerčních řešení je nutno brát na uváženou další faktor a tím je samozřejmě cena. Protože se nejedná zdaleka o levné produkty (ceny se pohybují v řádech stovek tisíc korun i více), jsou v rámci této práce zmíněny pouze dva, u kterých se podařilo vyjednat ukázku či zkušební zapůjčení a nasazení. Jsou jimi FortiDeceptor od společnosti FortiNet a BotSink vyvíjený společností Attivo Networks.

2.3.1 FortiDeceptor

FortiDeceptor je komerční honeypotové řešení od společnosti Fortinet. Nabízí celou řadu "návnad" se střední a vysokou úrovní interakce, pomocí kterých detekuje bezpečnostní incidenty. Kromě toho nabízí i analyzační nástroje a automatizované reakce na bezpečnostní incidenty a pokusy o vniknutí. Systém je plně integrovaný s dalšími Fortinet zařízeními jako například FortiGate firewally ci FortiSIEM. Kromě toho umožňuje FortiDeceptor vytváření vlastních instalací honeypotů, konfigurací vlastního virtuálního stroje a opatření jej FortiDefender nástrahami. V rámci práce se podařilo vyjednat krátkou zkušební licenci pro nasazení nástroje v nemocnici Jihlava jako virtuálního serveru.

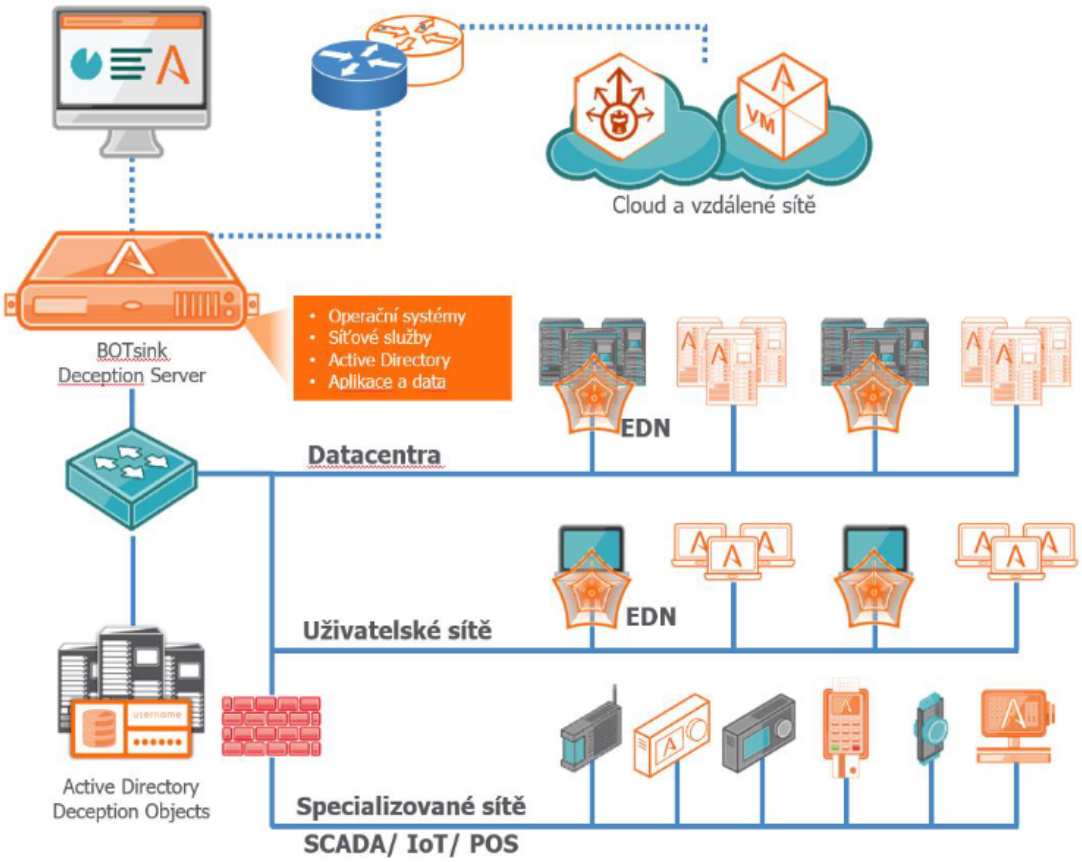
Pokusné nasazení nástroje v nemocnici se potýkalo s řadou problémů. Nástroj FortiDeceptor vyžaduje zapnutý promiskuitní mód na virtuálním switchi, do kterého je připojen, což je ovšem proti bezpečnostní politice nemocnice. Samotné nasazení honeypotů z nástroje se neobešlo bez potíží, honeypoty se totiž v rámci FortiDeceptoru tvářily jako zapnuté, ovšem nebylo možno k nim nikterak přistoupit. Pro vytvoření vlastních honeypotů je zapotřebí další licence a i standardní licence je omezena počtem nasazených honeypotů. V rámci zápůjčky dostala nemocnice licenci na pět honeypotů, ovšem objevil se problém, kdy při smazání jednoho z nasazených FortiDeceptor neuvolnil využitou licenci.

Ačkoliv některé funkce nástroje, jako například možnost vkládat vlastní soubory či uživatele pro snadnější maskování honeypotu, byly velmi vítané, nástroj zaměstnance oddělení

správy sítí příliš neoslovil. Díky problémům objevených během zkušebního nasazení a nevyhovující licenční politice, byl nástroj vyhodnocen jako nevhodný pro využití v nemocniční síti.

2.3.2 BotSink

Nástroj BotSink je přispůsobitelná honeypotová platforma vyvinutá společností Attivo Networks pro snadné nasazení a správu honeypotů v podnikové síti. Nástroj umožnuje výběr z předem připravených honeypotů nebo vytvoření vlastních honeypot instalací podobně jako u FortiDeceptoru. BotSink navíc využívá strojové učení a po rychlém oskenování sítě sám navrhne nejvhodnější rozmístění honeypotů v síti. V kombinaci se systémem Attivo Endpoint Detection Net umožňuje rychle reagovat na pokusy o proniknutí do sítě a odklonění útočníka na některý z honeypotů vytvářením falešných odpovědí a přesměrování. Rovněž i zde je kladen důraz na integraci s partnerskými komponenty v síti umožňující rychle automatizovaně reagovat na případné hrozby.



Obrázek 2.1: Schéma BotSink modelu[32]

[32] Převzato z materiálů společnosti Attivo Networks.

2.4 Shrnutí

V této kapitole byl představen princip funkce honeypotů, jejich dělení a některé z výhod či nevýhod jejich využití. Dále byla zkoumána dostupná řešení pro nazasení a provoz honeypotů z oblasti otevřeného software, kde se nabízí celá řada potenciálně vhodných honeypotů pro nasazení v nemocnici. Z diskutovaných platforem pro nasazení a správu honeypotů byl pro nasazení v nemocnici zvolen nástroj Tpot, zejména díky jeho snadné rozšiřitelnosti a také díky tomu, že je v době vzniku práce nejaktuálnějším. Výběr platformy Tpot pak zúžil možnosti pro výběr samotných honeypotů na ty, které jsou dále popsány v tabulce 2.1.

V poslední části kapitoly byly popsány dostupné honeypot systémy z komerční sféry, u kterých se podařilo vyjednat ukázku či pokusné nasazení. Ani jedna z těchto možností však nebyla zaměstnanci oddělení správy sítě vyhodnocena jako vhodná pro nasazení v nemocnici. Zbytek práce se tedy zabývá návrhem a nasazením systému honeypotů na platformě Tpot do několika míst v nemocniční síti.

Kapitola 3

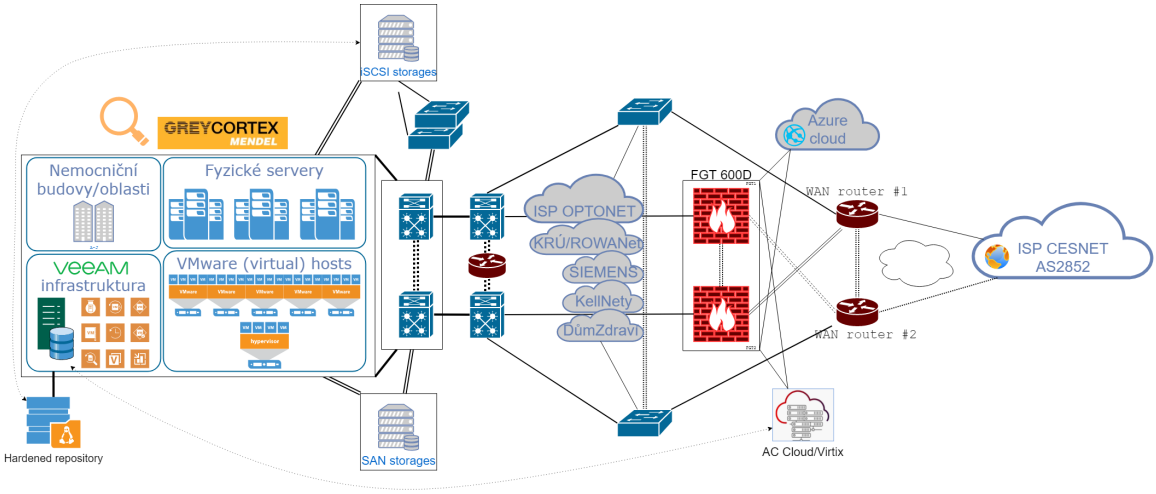
Nasazení honeypotů v nemocnici Jihlava

Hlavním cílem této práce bylo rozšířit zabezpečení počítačové sítě nemocnice Jihlava o sadu honeypotů. K dosažení tohoto cíle, bylo nejprve nutné seznámit se s existujícími prostředky pro zabezpečení nemocniční sítě a následně navrhnout systém honeypotů tak, aby mohl být využit v běžném provozu oddělení správy sítí. Právě tomuto cíli se věnuje následující kapitola. První část této kapitoly přibližuje počítačovou síť nemocnice Jihlava a jejího zabezpečení. Na základě této analýzy je dále navržena sada honeypotů a systém jejich nasazení do podsítí nemocnice. V poslední části se kapitola věnuje implementaci tohoto systému a možnosti jeho budoucího rozšíření.

Počítačová síť nemocnice Jihlava (popsaná na obrázku 3.1) v posledních několika letech vyrostla do poměrně značných rozměrů. V současné době nemocnice Jihlava provozuje dvě plně vybavená datacentra a v nich přibližně 170 serverů, 130 z nich virtualizovaných na ESXi klastru platformy VMWare s plánovaným rozšířením o nový klastr složený ze tří serverů v nadcházejícím roce. Síť propojuje celkem devět budov nemocnice a její návrh odpovídá topologii dvojité hvězdy. Z páteřních prvků umístěných v datacentrech jsou vedeny optické spoje do datových rozvoden v jednotlivých budovách, kam jsou následně připojeny všechny další prvky v dané části budovy. Některé další služby jako Office365, zaměstnanecké VDI (Virtual Desktop Infrastructure)[1] či ověřování a SSO (Single Sign On)[2] jsou provozovány v cloudu (Microsoft Azure). Nemocnice je do internetu připojena jednak optickým spojem poskytovatele OptoNet a také dvěma spoji do sítě vysokých škol a Akademie věd České republiky CESNET (optický spoj 10 GB do Českých Budějovic a optický spoj 1 GB přímo v Jihlavě).

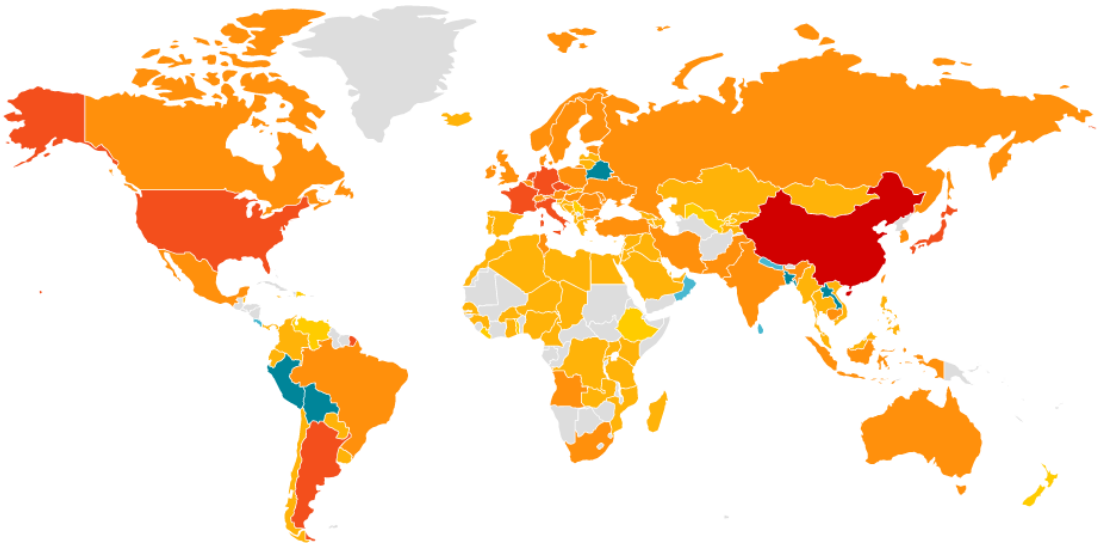
[1] Umožňuje zaměstnancům na vyžádání vytvořit virtuálního klienta, pomocí kterého mohou přistoupit do interní sítě nemocnice.

[2] Autentikační schéma umožňující jednotné přihlášení uživatele do nezávislých systémů. Uživatel se autentifikuje pouze v jedné službě a je automaticky autentifikován i v ostatních.



Obrázek 3.1: Schéma sítě nemocnice Jihlava

Jako asi každá větší organizace čelí i nemocnice Jihlava běžně bezpočtu skenů a pokusů o průnik, nicméně valná většina těchto pokusů je samozřejmě automatizovaná činnost botnetů. Na obrázku 3.2 jsou vidět nejčastější země původu těchto automatizovaných útoků. Jakožto nemocnice je, zejména díky současné geopolitické situaci, ale také potenciálním cílem pro politicky motivované útoky či pokusy o rozvrácení infrastruktury na území České republiky[5].



Obrázek 3.2: Geografické zobrazení detekovaných bezpečnostních incidentů

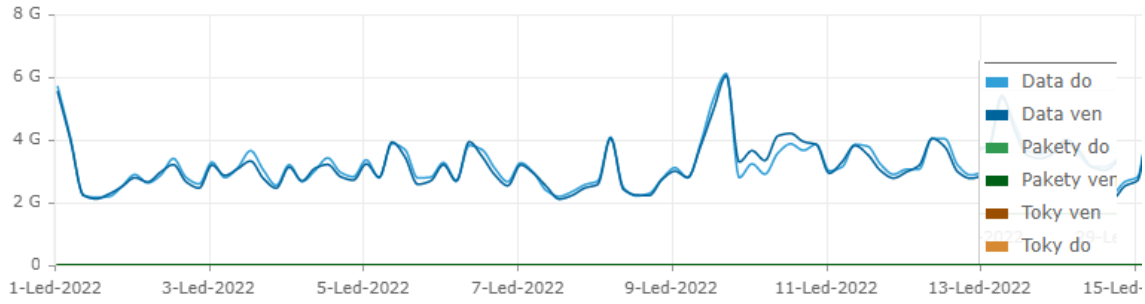
Informační systém nemocnice Jihlava je v evidenci Národního úřadu pro kybernetickou bezpečnost (NÚKIB) veden jako významný informační systém, z čehož vyplývá řada opatření a bezpečnostních politik, které musí nemocniční IT infrastruktura dodržovat.

3.1 Současné prostředky zabezpečení

Základním prvkem zabezpečí nemocnice je segmentace. V současné době je nemocnice rozdělena do více než 150ti sítí v pěti zónách zabezpečenými firewally FortiGate. Veškerá komunikace napříč segmenty je monitorována systémem IPS (Intrusion Prevention System) a auditována pomocí systému SEM (Security Event Management) Logmanager[3]. Přístup dodavatelů technologií je zajištěn pomocí šifrované VPN, ovšem v současné době probíhá postupné nahrazení VPN přístupů systémem CyberArk[4], který jednak umožňuje bližší monitorování, například možnost nahrávat každou session a ukládat tento záznam pro pozdější kontrolu, ale především také separací serverů na síťové úrovni. Vzdálený uživatel tedy "vidí" a je schopen přistoupit pouze na server CyberArk a vzdálený přístup do vnitřních síťových segmentů, kde se nachází samotné aplikační servery je možná pouze z tohoto serveru v kontrolované session. Síťový provoz mezi páteřními sítěmi a virtuálním prostředím VMWare klastru je dále sledován monitorovacím nástrojem Greycortex Mendel[5]. Některé události jsou dále přeposílávány do cloudové služby Azure Sentinel pro podrobnější analýzu. Detekované bezpečnostní incidenty dále sleduje a zkoumá správce bezpečnosti (zaměstnanec nemocnice a manažer kyberbezpečnosti dle směrnic NÚKIB) a případně vyvolá patřičné kroky k nápravě nalzených zranitelností.

3.1.1 Nástroj GreyCortex Mendel

GreyCortex Mendel je nástroj pro detekci a automatizovou odpověď na incidenty v síti nabízející hlubokou viditelnost v podnikových sítích a pokročilou detekci a reakce na hrozby [4]. V nemocnici Jihlava plní převážně funkci monitorování síťových toků a detekce potenciálních incidentů a je ve správě správce kybernetické bezpečnosti nemocnice. Během řešení bakalářské práce se povedlo vyjednat k tomuto nástroji přístup a bylo tak možné blíže nahlédnout do nemocniční sítě. To bylo důležité zejména pro správné naplánování umístění honeypotů, které je blíže popsáno v následující kapitole.



Obrázek 3.3: Graf toku dat do a z nemocniční sítě

Jak je vidět na obrázku 3.3, sítí nemocnice podle nástroje Mendel proteče každou vteřinu v průměru zhruba 3GB dat, přičemž nástroj Mendel monitoruje 83 z celkových 150 existujících podsítí. Na obrázku 3.4 pak můžeme vidět data pro 32 nejvíce využívaných podsítí, ze kterých byly v rámci této práce po konzultaci s vedoucím oddělení správy sítí a harware vybrány čtyři podsítě, do kterých byl nasazen senzor.

[3]https://www.logmanager.cz/ [2.5.2022].
[4]https://www.cyberark.com/ [2.5.2022].
[5]https://www.greycortex.com/mendel [2.5.2022].

Název	Toky	Toky do	Toky ven
All Internal	71.14 M	58.03 M	70.8 M
IPCAM, novy seg	997.15 k	1.11 M	992.2 k
Private B	31.0 M	34.56 M	30.95 M
Private C	65.36 k	18.9 k	64.49 k
ROWANET_FREE_KRU_550	1.8 M	117.78 k	1.76 M
WiFi VIP, segment CGN RFC 6598	2.91 M	1.15 M	2.86 M
segment 1 pocitacu bud E	2.25 M	363.87 k	2.25 M
C_USER_PC1_248	3.07 M	332.62 k	3.05 M
Private A	539.85 k	508.22 k	533.15 k
segment 3 pocitacu bud E	2.11 M	270.7 k	2.11 M
MGMT_HW_servers_110	102.59 k	9.27 k	102.59 k
segment 1 pocitacu bud G	2.11 M	266.68 k	2.11 M
segment 1 pocitacu bud B	2.06 M	346.45 k	2.05 M
segment VOIP TU	239.4 k	100.62 k	239.4 k
segment 2 pocitacu bud G	1.29 M	146.61 k	1.28 M
VRF IPTV	15.23 M	12.89 M	15.23 M
segment 1 pocitacu bud D	2.72 M	403.92 k	2.72 M
segment 1 pocitacu bud F	187.35 k	54.56 k	187.08 k
Link Local IPv6 Addresses	857.03 k	3.15 k	857.03 k
C_USER_PC2_249	95.42 k	60.62 k	95.3 k
segment 2 pocitacu bud B	129.12 k	269.52 k	128.69 k
SS1_sub1_192	93.86 k	115.38 k	93.86 k
ICT LAN	276.33 k	172.39 k	272.83 k
segment 1 pocitacu bud A	101.26 k	46.66 k	101.21 k
DMZ_AZURE_212.0/23	32.86 k	7.66 k	32.84 k
Multicast: Administratively scoped	3.76 k	3.04 M	
DMZ_AZURE_208.0/22	363.59 k	85.94 k	363.59 k
HW_GO_90	52.95 k	35.15 k	52.8 k
segment 1 pocitacu bud J	25.12 k	10.27 k	25.12 k
TOPENI_ENESA	79.72 k	33.91 k	79.72 k
Automatically created subnet	32.77 k	11.62 k	32.77 k
WIFI MGMT sit pro iAP iAP3x a vyssi	33.6 k	31.61 k	33.6 k
SW COREMGMT_B_WIFI verze iAP3x	14.84 k	13.24 k	14.84 k

Obrázek 3.4: 32 nejaktivnějších subnetů v nemocnici Jihlava

Nástroj Mendel rovněž nabízí několik funkcí pro správu incidentů a jejich automatizované ohlašování, nicméně správce bezpečnosti nemocnice krom něj využívá pro tyto účely i

další nástroje, jako je například LogManager[6]. Nutno podotknout, že valná většina současných incidentů se týká převážně vnitřních bezpečnostních politik a provozních problémů. Průniky útočníků do vnitřních podsítí nemocnice jsou velice vzácné.

3.1.2 Zálohování

Vzhledem k stále se zvyšujícímu počtu útoků ransomware mimo jiné cílících i na české nemocnice, je vhodné do kategorie bezpečnosti zařadit i systémy zálohování, jakožto poslední linii v obraně proti tomuto typu malware. V nemocnici Jihlava se na většině serverů provádí zálohy nástrojem Veeam Backup and Replication[7]. Nástroj běží na dedikovaném fyzickém serveru a v pravidelných intervalech provádí zálohu takřka celé serverové infrastruktury. Dodržuje se při tom takzvané pravidlo 3-2-1 pro zálohování. Udržují se tedy vždy tři kopie produkčních dat, vždy na dvou různých médiích, z nichž jedna se nachází v jiné lokalitě pro případ úplného kolapsu sítě, například v důsledku přírodní katastrofy. Pro uchování záloh se v nemocnici využívá deduplikační jednotky ExaGrid, kde jsou aplikovány časové zámky pro manipulace s daty a data jsou kompletně separována od zbytku sítě. Tyto zálohy jsou dále replikovány do druhé jednotky ExaGrid, umístěné na krajském úřadě v Jihlavě. Tato replikace probíhá po vyhrazeném privátním optickém vlákně napřímo, tedy neprochází nemocniční sítí ani internetem. Pro jakoukoliv manipulaci se zálohami uloženými v ExaGridu je zapotřebí nejen účet administrátora, ale také schválení operace správcem bezpečnosti (vynuceno ExaGridem samotným). V neposlední řadě jsou zálohy pravidelně kontrolovány technologií SureBackup, kdy jsou servery ze záloh pravidelně obnoveny do odděleného virtuálního prostředí a podrobeny sérii skriptů pro test konzistence.

3.2 Implementace

V rámci praktické části bakalářské práce byla v nemocnici Jihlava nasazena sada několika honeypotů. Po bližším zvážení a průzkumu dostupných honeypotů (popsaných v podkapitole 2.2) byl pro nasazení zvolen nástroj Tpot, jelikož nabízí poměrně snadnou instalaci a integraci velkého množství honeypotů, není náročný na údržbu a nabízí širokou sadu nástrojů pro analýzu a zpracování nasbíraných dat. Tpot lze instalovat jako samostatnou platformu, kdy jsou nástroje pro zpracování logů z honeypotů nainstalovány na stejný systém, na kterém běží samotné honeypoty. Využitím nástroje Docker[8] pro virtualizaci komponent, je zaručeno oddělení jednotlivých nástrojů od sebe, kdy každý pracuje ve svém vlastním virtuálním prostředí. V takovémto návrhu by pak každá nasazená instance Tpotu pracovala nezávisle na ostatních a uchovávala všechna data na svých lokálních úložištích, zpracovávala a odesílala do nástroje LogManager[9]. Toto řešení by vyhovovalo pro krátký výzkumný projekt o malém počtu instancí. Aby však mělo produkční nasazení do sítě nemocnice smysl i do budoucna, musí být snadněji rozšiřitelné a spravovatelné. K těmto účelům se mnohem lépe hodí návrh nasazené honeypotů v režimu monitor-sensor, kdy jedna instance hraje roli monitoru a všechny další jsou jí podřízeny a slouží jako sensory.

Monitor sám o sobě neobsahuje žádné honeypoty, ale pouze nástroje pro jejich zpracování a vizualizaci. Sensory potom naopak obsahují pouze honeypoty, ovšem již není nutné instalovat a spouštět další nástroje. Sensory pouze odesílají logy z honeypotů do instance

[6] https://www.logmanager.cz/ [2.5.2022]
[7] https://www.veeam.com/vm-backup-recovery-replication-software.html [2.5.2022]
[8] https://www.docker.com/ [2.5.2022]
[9] https://www.logmanager.cz/ [2.5.2022]

monitoru. Díky tomu lze mimo jiné u sensorů slevit hardwarové nároky zhruba na polovinu minimálních požadavků monitoru. Samozřejmě celkové vytížení zdrojů závisí na počtu a typu provozovaných honeypotů. Vyšší úroveň interakce obvykle přináší vyšší nároky na systémové zdroje než "prostá" simulace služeb honeypoty s nízkou úrovní interakce.

3.2.1 Návrh nasazení honeypotů do sítě

V době vzniku této práce bylo vestavěné řešení monitor-sensor u nástroje Tpot stále ve vývinu a nebylo funkční do takové míry, aby bylo vhodné pro nasazení v nemocnici. Znamená to tedy, že bylo třeba vlastní instance Tpotu rozšířit o způsob přenosu dat na monitor.

Jedním z možných řešení tohoto problému je využití separátního logovacího nástroje, jako například plugin pro Logstash Lumberjack[10]. V takovémto řešení by tedy existovala instance Logstash na sensoru, která by shromažďovala logy honeypotů a předávala je nástroji Lumberjack, který by zajistil přenos přes SSL do monitoru, kde by tato data převzala lokální instance Logstash. Ta potom data zpracuje dle konfigurace a uloží jako dokumenty pro ElasticSearch.

Vzniká tedy jakési schéma Honeypot -> Logstash -> Lumberjack -> Logstash -> ElasticSearch. Toto řešení bylo v rámci práce testováno a fungovalo dle očekávání.

Vhodnějším řešením je však použití nástroje FileBeat[11]. FileBeat je nástroj z rodiny Elastic určený k monitorování, sběru a přesměrování logů. Díky tomu, že se jedná o rozšíření pro Elastic stack, umožňuje rovněž mnohem jednoduší integraci s Logstash. Pro reálné nasazení byl tedy na sensorech použit nástroj FileBeat, který monitoruje výskyt nových logů některého z honeypotů v adresáři `/data/HONEYPOTNAME` a v případě nových dat je označí a opět šifrovaným SSL spojením na portu 64299 předá nástroji Logstash na monitoru. Tam se data dále zpracují a nakonec uloží ve formátu pro ElasticSearch. Pomocí nástroje Kibana pak lze tyto data procházet a vizualizovat do grafů ve webovém grafickém rozhraní. Tím je tedy vytvořena kompletní Elastic pipeline.

Dalším krokem v rámci návrhu bylo určit, kolik sensorů je třeba nasadit, do kterých částí nemocniční sítě a jaké honeypoty na nich provozovat. Po konzultaci s oddělením sítí a hardware byly vybrány čtyři sítě VLAN pro nasazení sensorů i sada honeypotů na každou z těchto sítí. Jedná se o následující:

- **Vlan1** - Pozůstatek z dřívějších let. Ačkoliv se nové systémy do této sítě VLAN již nepřidávají, některé zastaralejší servery v ní stále zůstávají.
- **ICT síť VLAN** - Interní síť VLAN pro ICT oddělení. Spadají do ní všechny pracovní stanice administrátorů a další interní ICT systémy, které představují lákavý cíl pro útočníky.
- **Zaměstnanecká WIFI síť VLAN** - Síť VLAN pro připojení zaměstnanců přes WiFi. Pro připojení je třeba se autorizovat, nicméně zaměstnanci připojují k této síti osobní zařízení, což představuje potenciální cestu pro škodlivý software.
- **Uživatelská síť VLAN budovy E** - Síť pro uživatelská PC na budově E. Tato síť je převážně využívána zdravotnickým personálem.

[10] https://github.com/logstash-plugins/logstash-input-lumberjack a https://github.com/logstash-plugins/logstash-output-lumberjack [2.5.2022]

[11] https://www.elastic.co/beats/filebeat [2.5.2022]

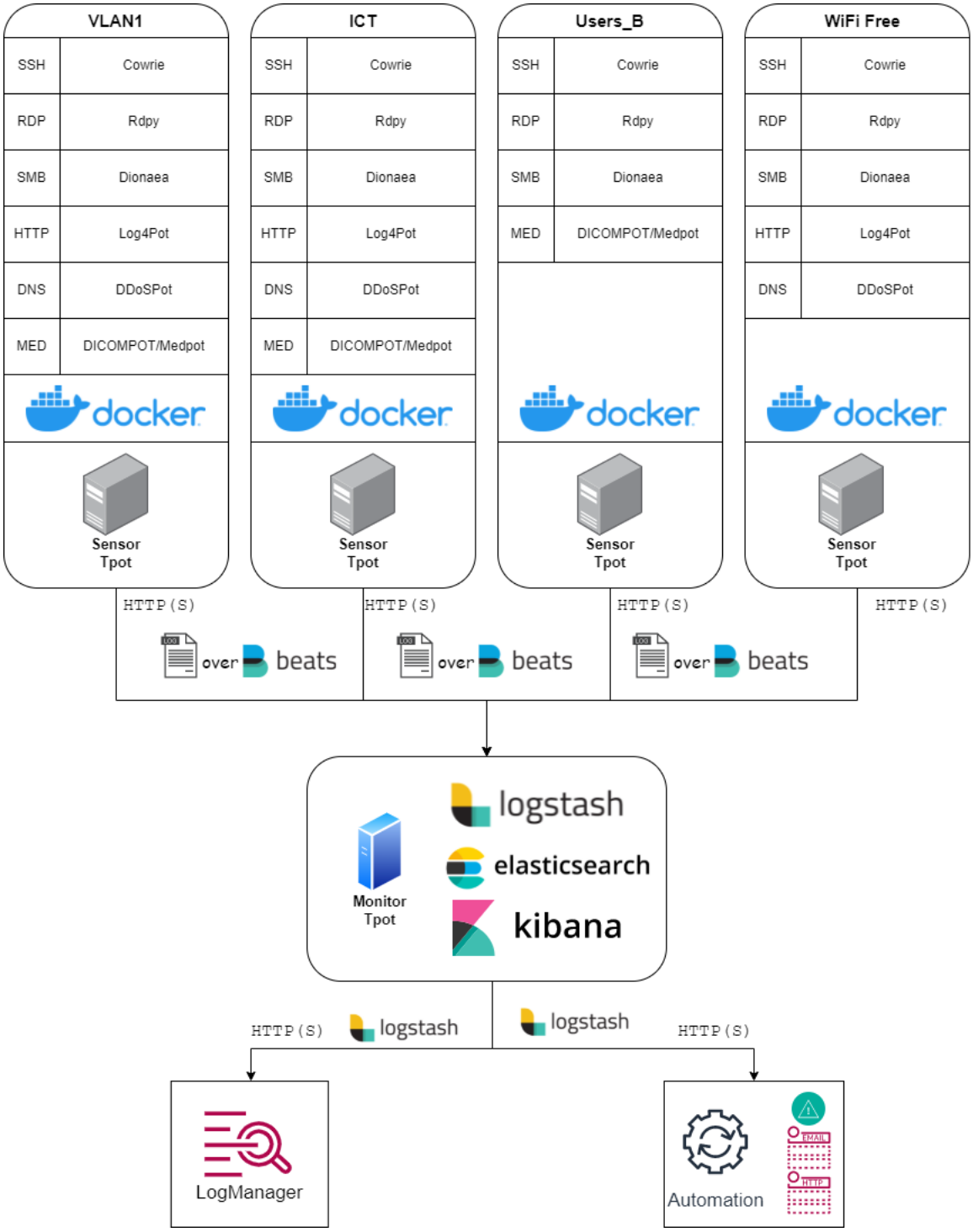
Výběr samotných honeypotů je založen na statistice nejzranitelnějších služeb v nemocniční síti získané nástrojem GreyCortex Mendel, jejíž výsledek lze vidět na obrázku 3.5. Tyto služby nástroj Mendel v nemocnici vyhodnotil jako nejrizikovější. Některé porty je na obrázku možné vidět dvakrát. To je způsobeno tím, že nástroj Mendel detekoval otevřený port, ale nebyl jednoznačně schopen určit službu, která na něm běží. Ve sloupci na pravé straně obrázku je pak možno vidět počet záznamů událostí detekovaných nástrojem Mendel.

Riziko	Služba	Hosts	Události
Vysoký	SMB (445)	1.08 k	6.0 M
Vysoký	135	1.98 k	566.8 k
Vysoký	RDP (3389)	1.13 k	348.2 k
Vysoký	HTTP (80)	2.91 k	186.8 k
Vysoký	445	1.47 k	7.1 M
Vysoký	DNS (53)	309	3.6 M
Vysoký	HTTPS (443)	5.25 k	1.2 M
Vysoký	443	4.9 k	30.1 k
Vysoký	80	753	17.8 k
Vysoký	NTP (123)	138	12.4 k

Obrázek 3.5: Statistika zranitelných služeb v nemocniční síti dle nástroje Mendel

K "pokrytí" těchto služeb je tedy zvolena sada několika honeypotů. Pro služby HTTP a HTTPS byl zvolen honeypot Log4Pot, zejména díky vlně Log4j zranitelností a rozsahu jejich rozšíření, což z tohoto honeypotu tvoří zajímavý cíl pro potenciálního útočníka. Dále honeypot RDPy pro RDP a honeypot Dionaea, který nabízí celou řadu protokolů mimo jiné i SMB. Přidán byl rovněž honeypot ssh Cowrie. Ani pokusy o zneužití ostatních portů a služeb však nezůstanou bez povšimnutí, protože na všech sensorech běží také nástroj P0f[12] sloužící pro získávání otisků operačního systému a po provedení experimentů v kapitole 5 byl rovněž přidán monitorovací nástroj Suricata[13]. Celkově jsou tedy sensory schopné zaznamenat a ohlásit takřka jakoukoliv interakci. Samotný monitor je pak umístěn rovněž v interní ICT síti. Tím tedy vznikl finální návrh nasazení který popisuje obrázek 3.6.

[12]https://github.com/p0f/p0f [2.5.2022]
[13]https://github.com/OISF/suricata [2.5.2022]



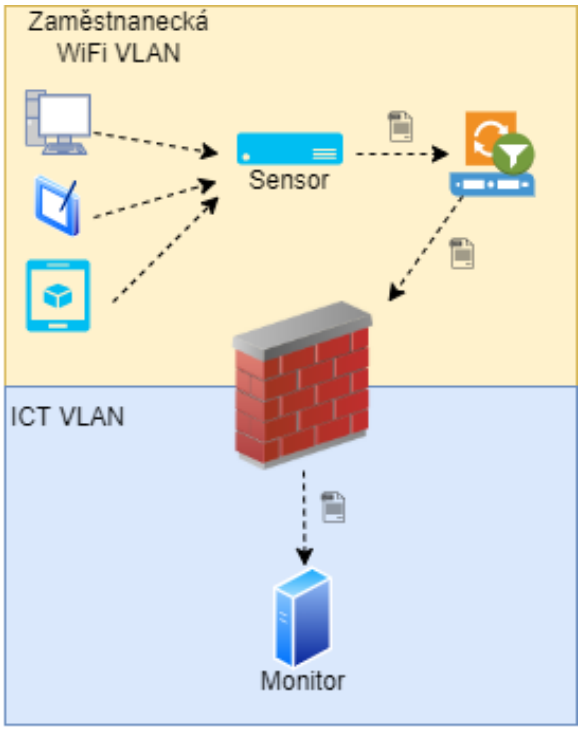
Obrázek 3.6: Schéma nasazení honeypotů do sítě

Problém s nasazením do zaměstnanecké WiFi VLAN

V průběhu implementace předchozího návrhu došlo k problému s nasazením sensoru do zaměstnanecké WiFi sítě. Tato síť slouží pro zaměstnance nemocnice jakožto přístupový bod k internetu na jejich osobních zařízeních a ačkoliv je zapotřebí autentizace RADIUS

pro přístup, je tato síť stejně oddělena ve vlastní zóně firewall a komunikace do vnitřních sítí nemocnice je zakázána. Při nasazení sensoru do této sítě by tedy nebylo možné zajistit komunikaci s monitorem umístěným v interní ICT VLAN. Jako možné řešení tohoto problému byla testována funkce Tpotu na dvou síťových rozhraních, nicméně tuto funkčnost Tpot nenabízí. Jediným způsobem, jak takové funkce docílit, je konfigurace interních routovacích pravidel přímo v linuxovém prostředí sensoru a pevná definice IP adres, na kterých mají honeypoty naslouchat a FileBeats odesílat. Toto řešení bylo konzultováno s vedoucím oddělení správy sítí a hardware a správcem bezpečnosti a bylo vyhodnoceno jako příliš rizikové. Chyba v konfiguraci by totiž mohla znamenat nechtěné vytvoření cesty do interních sítí nemocnice a pokud by byl sensor kompromitován, útočníkovi by poskytl možnost obejít firewall.

Jinou možností řešení tohoto problému by mohla být instalace proxy serveru (viz obrázek 3.7), který by přijímal logy ze sensoru a komunikaci filtroval. Toho lze docílit například využitím nástroje Logstash či vlastní implementací filtrovacího programu. Tento proxy server by pak měl umožněn průchod firewallem do monitoru.



Obrázek 3.7: Schéma nasazení s proxy serverem

3.2.2 Příprava Tpotu

Před instalací bylo nejprve potřeba nadimenzovat a připravit virtuální severy pro instalaci. Po konzultaci a testování požadavků jednotlivých konfigurací byly zvoleny následující parametry.

	CPU	RAM	Storage
Monitor	2 core	8 GB	128GB [14]
Sensory	2 core	4 GB	16 GB

Tabulka 3.1: Hardwarové specifikace

Samotný postup instalace je poměrně jednoduchý. Tpot nabízí v podstatě dvě varianty instalace a to buď přímo z obrazu ISO nebo jako nadstavba nad operačním systémem Debian 10 (Buster). Metody byly v rámci práce testovány obě, ovšem instalace na již existující Debian 10 instalaci se nepodařila z důvodů konfliktů verzí nástrojů. Bylo však možné vygenerovat vlastní ISO obraz pro přímou instalaci.

Instalace pomocí předem připraveného obrazu ISO proběhla bez větších potíží. Během instalace není možné vybrat vlastní sadu honeypotů a je nutné nainstalovat jeden z nabízených balíků. Po instalaci je však možné vytvořit vlastní konfigurační soubor a pomocí skriptu konfiguraci změnit. Jak již bylo nastíněno v předchozí kapitole, nástroj Tpot umožnuje vytvořit vlastní konfiguraci docker-compose a posléze na ni přejít spuštěním skriptu `/opt/tpot/bin/tped.sh`, který je součástí nástroje Tpot a vždy jej lze najít v adresáři `/opt/tpot/bin`. Tento skript umožnuje vybrat a automaticky nasadit novou sadu honeypotů a podpůrných nástrojů definovaných v konfiguračních souborech docker-compose umístěných v adresáři `/opt/tpot/ect/compose/`.

Pro samotnou instalaci byla využita nejnovější verze Tpot obrazu ISO[15]. Při instalaci je nutné zvolit heslo pro defaultního administrátora s uživatelským jménem `tsec` a posléze uživatelské jméno a heslo pro přihlášení do webového rozhraní. Pro přehlednost bylo u všech instancí zvoleno jméno `observer`. Tpot umožňuje později vytvořit více uživatelů ve webovém rozhraní. Následně je nutné vybrat jeden z předem připravených balíků (pro účely práce nezáleží který, protože bude stejně v zápětí překonfigurován). U všech instancí byl zvolen balík sensor z důvodů rychlejší instalace (neinstaluje se elastic stack). Ihned po ukončení instalace byla služba Tpot vypnuta příkazem `systemctl stop Tpot` v rámci přípravy na překonfigurování. Posledním společným krokem pro monitor i sensor byla změna hostname. V základu Tpot vygeneruje hostname náhodně a je tedy nutné jej pro přehlednost změnit ručně.

3.2.3 Konfigurace sensorů

Konfigurace sensorů pro tento návrh je velice jednoduchá. Nejprve bylo nutné upravit konfigurační soubor /opt/Tpot/etc/compose/sensor.yml. Jedná se o soubor docker-compose[16], který určuje jakou sadu honeypotů a podpůrných nástrojů má Tpot spouštět. Ze souboru byly odebrány všechny nástroje pro analýzu a sběr logů a všechny honeypoty, které nejsou součástí návrhu. Kromě webového serveru Nginx[17] a vybraných honeypotů byly všechny ostatní služby smazány a nakonec přidána konfigurace Filebeat. Protože Tpot repozitáře neobsahují docker image pro Filebeat, byl při konfiguraci využit oficiální obraz z docker.elastic.co pro nástroj Filebeat verze 7.9.2. Při konfiguraci bylo třeba připojit adresář data a další dva konfigurační soubory. To je možné vidět na obrázku 3.8, který ukazuje výtah souboru sensor.yml. Pro účely této práce je důkežitá zejména možnost `volumes` která

[14] Možno rozšířit v závislosti na množství dat.
[15] Dostupný z: https://github.com/telekom-security/tpotce/releases/tag/20.06.2 [21.12.2021].
[16] Formát dle https://docs.docker.com/compose/ [2.5.2022].
[17] https://www.nginx.com/ [2.5.2022]

popisuje mapování adresářů nacházejícím se v souborovém systému sensoru na adresáře ve virtuálním prostředí kontejneru docker pro danou službu (v tomto případe Filebeats). Pro správnou funkci sensoru je nutno mapovat nejen adresář `/data`, kde se nachází samotné logy z honeypotů, ale také konfigurační soubory pro službu Filebeats. Možnost `:ro` na konci řádků s konfiguračními soubory, říká, že ve virtuálním prostředí jsou tyto soubory v režimu pouze pro čtení. Možnost `image` definuje odkud má nástroj Docker stáhnout obraz, který se má spustit.

```
    ...
## Filebeat service
  filebeat:
    container_name: filebeat
    restart: always
    image: "docker.elastic.co/beats/filebeat:7.9.2"
    user: root
    env_file:
     - /opt/Tpot/etc/compose/elk_environment
    volumes:
     - /data:/data
     - /opt/elk/filebeat.yml:/usr/share/filebeat/filebeat.yml:ro
     - /opt/elk/fields.yml:/usr/share/filebeat/fields.yml:ro
        ...
```

Obrázek 3.8: Ukázka sensor.yml

Následně bylo nutné provést konfiguraci nástroje Filebeat tak, aby sbíral data z patřičných souborů s logy a odeslal je do logstash na monitoru. Toho lze docílit právě v souboru filebeat.yml (lze vidět na obrázku 3.9), kde je nutné v sekci `filebeat.inputs` definovat cesty ke všem vstupním souborům (logům honeypotů v adresáři /data), dále je možné je označit typem (který honeypot logy vytvořil) přidáním vlastního pole v možnosti `fields`. Filebeat umožňuje přidávat a zahazovat pole z logů, což lze využít k přidání informací, o který sensor se vlastně jedná. To lze nakonfigurovat v sekci `filebeat.processors` ve stejném konfiguračním souboru. Ke každému logu je tedy možností `add_fields` připojena položka `host\_info` obsahující pole pro IP adresu, název zařízení a řetězec s názvem sensoru, ze kterého pochází. Nakonec stačí ve výstupní konfiguraci přidat navázání na logstash pomocí IP adresy a portu, případně definovat cestu k SSL certifikátu.

Vzniká ovšem konflikt v názvu klíčového slova `input.type`. Protože většina honeypotů ve svém logu využivá klíčového slova `input` jako název jednoho z polí logu, což je ale zároveň klíčovým slovem ve Filebeat a dochází tak ke konfliktu. Naštěstí se tento problém dá lehce vyřešit změnou klíčového slova v nástroji Filebeat. K tomu stačí předefinovat toto klíčové slovo v soubor fields.yml. Pro účely této práce byla tedy zvolena verze `input_type` namísto `input.type`.

```
filebeat.inputs:
  - enabled: true
    paths:
      - /data/cowrie/log/cowrie.json
    fields:
      type: Cowrie
    fields_under_root: true
    json.keys_under_root: true
    json.overwrite_keys: true

    ... ostatni honeypoty stejnym zpusobem ...

processors:
    ...
    # Oznaceni sensoru
    - add_fields:
      target: host_info
      fields:
        exit_ip: "${MY_EXTIP}"
        in_ip: "${MY_INTIP}"
        hostname: "${MY_HOSTNAME}"
        sensor: "SensorICT"

    ...
output.logstash:
  hosts: ["MONITOR_IP:64299"]
  ...
```

Obrázek 3.9: Konfigurace nástroje Filebeat

Posledním krokem bylo spuštění předem zmíněného skriptu /opt/Tpot/bin/tped.sh a zvolení sensor balíčku. Tpot automaticky stáhne chybějící docker obrazy, spustí nakonfigurované služby a honeypoty. Tento postup byl opakován u všech sensorů.

3.2.4 Konfigurace monitoru

Při konfiguraci monitoru bylo prvním krokem vytvoření souboru docker-compose /opt/Tpot/etc/compose/monitor.yml. Jedná se o standardní formát pro docker-compose[18]. Pro účely monitoru byly konfigurovány pouze služby Elasticsearch, Elasticsearch-head, Kibana, Logstash a Nginx. Ačkoliv instance monitoru může sama sloužit jako platforma pro honeypoty, pro účely této práce se jedná pouze o platformu pro sběr, zpracování a monitorování. Je tedy vhodné tyto funkce nechat oddělené od samotných honeypotů. Důležitou změnou oproti ostatním konfiguracím v této části je konfigurace služby Logstash, kde bylo třeba specifikovat cestu k vlastní konfiguraci nástroje podobně jako pro nástroj Filebeat v předchozí podkapitole na obrázku 3.8.

Následně bylo nutné vytvořit konfigurační soubor pro logstash. Nejjednodušším způsobem je zkopírovat a upravit již existující základní verzi tohoto konfiguračního souboru, který lze najít v adresáři `/opt/Tpot/docker/elk/logstash/dist`. V tomto souboru lze definovat další manipulaci s logy přicházejícími ze sensorů. Pro tuto část práce však stačilo nakonfigurovat vstup z nástroje Filebeat na sensorech. Logstash toto umožňuje prostým využitím objektu "beats" ve vstupní části konfiguračnímu souboru. Stačilo tedy definovat vstupní port 64299. Podobně jako u sensorů byla provedena rekonfigurace skriptem /opt/Tpot/bin/tped.sh a zvolen nově vzniklý balík monitor.yml. Po skončení skriptu je monitor připraven k používání.

[18]Formát dle https://docs.docker.com/compose/ [2.5.2022].

Nyní už tedy stačilo jen přihlásit se do webového rozhraní nástroje Kibana a začít tvořit přehledy pro analýzu dat. Práce s nástrojem Kibana je velice jednoduchá a lze poměrně rychle vytvářet grafické přehledy z logů, které jsou zpracovány pomocí ElasticSearch. Pro sledování přímo z monitoru byl připraven jednoduchý dashboard.

Samotné procházení logů ze sensorů je opět v nástroji Kibana Discover snadné, logy lze volně prohledávat, filtrovat a zobrazovat detaily. V rámci návrhu je připravena konfigurace pro odesílání logů do systému LogManager[19], který nemocnice využívá pro centralizovanou správu a analýzu logů z většiny serverů a aktivních prvků v síti nemocnice. Implementace této části je omezena nutností další konfigurace ze strany správce LogManageru a z časových důvodů tedy nebyla v době vzniku této práce dokončena. Nicméně v návrhu setrvává a je schválena, takže je možné bezpečně předpokládat že k ní v blízké době dojde. Tato funkcionalita bude opět zajištěna nástojem Logstash s využitím výstupního pluginu pro výstup Syslog do LogManageru. Automatické odmazávání zastaralých logů (nastaveno na 30 dní) je zajištěno nástrojem Logrotate[20]. Sensory mažou svoje lokální úložiště každé dva dny naprosto stejným mechanizmem. Důvod k tomuto krátkodobému uchovávání logů sensory je případný výpadek spojení s monitorem či monitoru samotného, například při aktualizaci. Pokud k podobnému výpadku dojde, nástroj Filebeats zajistí, že po obnovení spojení budou všechny lokální logy opět odeslány do monitoru ke zpracování.

Kromě analýzy na monitoru nebo v LogManageru je jakákoliv interakce se sensory rovněž reportována a vyvolá vytvoření nového incidentu ve vlastní instanci nástroje GitLab provozovaného lokálně v nemocnici, kterou oddělení správy sítí a hardware využívá jako správce incidentů a k rozdělování úkolů mezi členy oddělení. Implementace této funkce je blíže popsána v kapitole 5.

Rozšiřitelnost řešení

Jedním z hlavních cílů tohoto projektu bylo, aby navržené a implementované řešení bylo snadno rozšiřitelné, neboť se jedná o systém, který zůstane součástí produkční infrastruktury nemocnice. Implementace řešení s tímto cílem počítá a jak sensory tak monitor jsou virtualizovné na platformě VMWare a díky tomu jednoduše hardwarově dimenzovatelné a rozšiřitelné. Nicméně nic nebrání ani budoucí instalaci fyzických sensorů či migraci monitoru na fyzickou platformu. Sensory jsou díky dockerizované struktuře Tpotu snadno rozšiřitelné o další honeypoty či výměnu těch stávajících. Přidání zcela nového sensoru do některé z dalších sítí nemocnice je opět poměrně nenáročné a sestává z několika kroků:

1. **Žádost o server** - Každé zařízení připojované do interní sítě nemocnice musí být povoleno a projít procesem vzniku serveru dle bezpečnostních politik nemocnice. Tento proces zahrnuje několik kroků, mezi nimiž jsou například schválení správcem bezpečnosti nemocnice, schválení náměstkem ICT, přidělení IP adresy, zařazení do zálohování, zařazení do monitorovacích systémů atd.

2. **Instalace Tpot** - Po vzniku nového serveru proběhne instalace Tpot v implicitním nastavení. Součástí tohoto procesu je nastavení IP, hostname atd.

3. **(Volitelné) Úprava konfigurace** - Pokud je třeba změnit složení honeypotů, které sensor nabízí, je třeba doupravit konfigurační docker-compose soubor sensoru dle potřeby. V opačném případě nejsou nutné úpravy žádné.

[19] https://www.logmanager.cz/ [2.5.2022]
[20] https://github.com/logrotate/logrotate [2.5.2022]

4. **Nahrání konfigurace** - Po instalaci v implicitním nastavení stačí nahrát připravené konfigurační soubory (sensor.yml,filebeats.yml a fields.yml), spustit skript tped.sh, který je popsán v podkapitole 3.2.2 a vybrat nový konfigurační soubor z nabídky. Případně ještě lze dodat certifikáty pro využití SSL šifrování při odesílání logů ze sensoru do monitoru.

Monitor již není dále třeba upravovat a celý proces od bodu dva zabere v průměru zhruba 15-20 minut. Jedinou komplikovanější operací při rožšiřování tohoto systému je napojení na systém hlášení incidentů do GitLabu při změně sady honeypotů na sensorech, kdy je potřeba definovat strukturu zájmových dat v konfiguraci logstash na monitoru a upravit program zajištující vytváření incidentů. Tento proces je blíže popsán v kapitole 5

3.3 Shrnutí

V této kapitole byla blíže prozkoumána a popsána počítačová síť nemocnice Jihlava a současné prostředky jejího zabezpečení. Byla provedena analýza sítě pomocí nástroje GreyCortex, na základě které byl stanoven seznam služeb vhodných pro imitaci honeypoty (obrázek 3.5). Dále byla zvolena vhodná metoda implementace systému monitor-sensor a stanoveny podsítě, do kterých budou sensory nasazeny. Jedná se celkem o tři sensory nesoucí následující sadu honeypotů:

SensorICT	**SensorLAN**	**SensorUsrB**
Cowrie	Cowrie	Cowrie
Rdpy	Rdpy	Rdpy
Dionaea	Dionaea	Dionaea
Dicompot	Dicompot	Dicompot
Medpot	Medpot	Medpot
Log4pot	Log4pot	
DDOSpot	DDOSpot	

Tabulka 3.2: Vybrané honeypoty na sensorech

Tyto sensory přenáší vygenerované logy přes HTTP pomocí nástroje Filebeats do monitoru umístěného v ICT VLAN, který je uchovává k další analýze. Celý návrh lze vidět na obrázku 3.6. V poslední části kapitoly je popsán postup implementace tohoto návrhu a možnost jeho rozšíření o další sensory. Tím je tedy splněn jeden z hlavních cílů práce.

Kapitola 4

Testování nasazených honeypotů

V rámci bakalářské práce byl nasazený systém podroben několika automatizovaným penetračním testům, které popisuje tato kapitola. Pro tento účel bylo připraveno testovací prostředí, které je popsáno v následující sekci. Kapitola se pak dále věnuje popisu provedených testů a hodnocení jejich výsledků. Cílem je zjistit, jak nasazené honeypoty reagují na interakce a pokusy o zneužití, jaké produkují logy a co se z nich dá vyčítat. Na základě výsledků těchto testů byl následně navržen systém automatického ohlašování incidentů, který je popsán v kapitole 5.

4.1 Příprava prostředí

Pro účely testování byl vytvořen nový testovací server s operačním systémem Kali Linux verze 2022.1[1]. Kali už v základu přichází s řadou nástrojů vhodných pro penetrační testování, mezi nimiž je mimo jiné i nástroj nmap[2] sloužící pro průzkum sítě a skenování portů.

Dále byl na tento server nainstalován nástroj OpenVAS[3], scanner zranitelností od společnosti Greenbone. Tento nástroj byl zvolen namísto původně plánovaného nástroje Nesus[4], zejména protože narozdíl od Nessus, není třeba pro komunitní verzi OpenVAS žádná licence. Instalace nástroje byla velice jednoduchá, protože je dostupný v balíčkovacím nástroji apt-get. Po instalaci je třeba provést automatizované nastavení spuštěním příkazu `gvm-setup`, který stáhne databáze zranitelností. Tento proces může být poměrně zdlouhavý (trval zhruba hodinu). Třetím nástrojem využitým pro testování honeypotů je nástroj Sn1per[5], což je rovněž otevřený software pro automatizované útoky za účelem penetračního testování. Instalace nástroje byla opět velice jednoduchá, kdy stačilo pouze naklonovat repozitář z GitHubu. a spustit instalační skript.

Samotný testovací server byl umístěn v interní ICT síti z důvodu snadného testování. ICT VLAN je totiž využívaná oddělením ICT pro pracovní stanice správcu a je z ní dostupná většina ostatních podsítí v nemocnici. Během testů byl spuštěn packet sniffer Wireshark a byl tedy pořízen záznam komunikace s cílem.

[1] https://www.kali.org/get-kali/#kali-bare-metal [10.4.2022]
[2] https://nmap.org/ [10.4.2022]
[3] https://openvas.org/ [10.4.2022]
[4] https://www.tenable.com/products/nessus [10.4.202]
[5] https://github.com/1N3/Sn1per [13.4.2022]

4.2 Nmap skenování

4.2.1 Test č.1: Základní Nmap sken

Prvním testem v pořadí byl základní SYN sken. Cílem bylo otestovat, jak honeypot reaguje na skenování, jakým způsobem reaguje na interakci s uzavřenými porty a jaké výstupy je možno při takové aktivitě očekávat. Očekávaným výsledkem bylo několik otevřených portů nejběžnějších služeb jako HTTP, SSH, ms-sql, atd. Výsledek by se měl co nejvíce blížit obrázku 3.5. Cíl útoku (honeypot sensor) by měl tento sken zalogovat a zjistit alespoň základní informaci o zdrojové IP adrese, případně otisk operačního systému pomocí nástroje P0f. Jelikož se jedná pouze o SYN sken, samotné honeypoty zřejmě na tuto interakci reagovat nebudou.

Test byl spuštěn z připraveného testovacího serveru popsaného výše a sestával ze tří skenů nástrojem nmap v implicitním nastavení (tedy příkazem `nmap SENSOR_IP`) cílícím na všechny tři nasazené sensory.

Výsledek

Z výsledku na obrázku 4.1 je možné vidět, že nástroj Nmap detekuje několik otevřených portů nabízejících služby, které jsou v sítích nemocnice Jihlava dostupné. Drobným nedostatkem je velké množství těchto služeb na jediném zařízení, což může potenciálně působit příliš nerealisticky a prozradit tak honeypot. Řešením této obtíže je do budoucna rozdělit honeypoty do více sensorů v jedné síti. Díky modulárnosti navrženého řešení to nepředstavuje větší problém.

```
Starting Nmap 7.92 ( https://nmap.org ) at 2022-04-11 11:51 CEST
Nmap scan report for 172.19.191.47
Host is up (0.00027s latency).
Not shown: 983 closed tcp ports (reset)
PORT STATE SERVICE
21/tcp open ftp
22/tcp open ssh
23/tcp open telnet
42/tcp open nameserver
80/tcp open http
81/tcp open hosts2-ns
135/tcp open msrpc
443/tcp open https
445/tcp open microsoft-ds
1433/tcp open ms-sql-s
1723/tcp open pptp
3306/tcp open mysql
3389/tcp open ms-wbt-server
5060/tcp open sip
5061/tcp open sip-tls
8080/tcp open http-proxy
9200/tcp open wap-wsp
MAC Address: 00:50:56:A0:89:AE (VMware)

Nmap done: 1 IP address (1 host up) scanned in 0.23 seconds
```

Obrázek 4.1: Výstup NMap skenu sensoru ICT

Dle očekávání na SYN sken jednotlivé honeypoty nereagují. Nicméně sensor je stále schopen tuto aktivitu zachytit díky nástroji P0f, který na packety SYN reaguje a pokouší se získat otisk operačního systému. Ačkoliv to není mnoho informací, důležité je, že je aktivita zaznamenána. Vzhledem k principu fungování honeypotů a vzhledem k umístění sensorů v síti lze jakoukoliv cílenou komunikaci se sensorem považovat za podezřelou. Nástroj P0f navíc zaznamená zdrojovou IP adresu. Jak je vidět na obrázku 4.2, je schopen rozeznat, že

se jedná o sken nástroje Nmap a tuto informaci předat v poli `app`. Krom toho lze také v logu z nástroje P0f najít IP adresu zařízení, ze kterého útok přišel.

Tímto způsobem byly oskenovány všechny sensory, nicméně výsledek je dle očekávání stejný. Jediný podstatný rozdíl je ve složení služeb u sensoru umístěného do uživatelské podsítě budovy B, které je dáno rozdílem v nasazených honeypotech.

@timestamp	Apr 11, 2022 @ 11:51:42.000
@version	1
app	NMap SYN scan
dest_ip	172.19.191.47
DestPort	443
dist	<= 8
host_info.exit_ip	195.113.227.242
host_info.hostname	ict-chroustalj
host_info.in_ip	172.19.191.47
host_info.sensor	SensorICT
mod	syn
params	random_ttl
raw_sig	4:56+8:0:1460:1024,0:mss::0
src_ip	172.19.191.46
src_port	38791
subject	cli
T-Pot Hostname	vdi-chroustalj
t-pot_ip_ext	195.113.227.242
t-pot_ip_int	172.19.191.48
Type	P0f

Obrázek 4.2: Záznam z nástroje P0f na sensoru detekující Nmap sken

4.2.2 Test č.2: Agresivní Nmap sken

Daším testem, kterému byly sensory podrobeny, byl agresivní sken všech portů TCP za účelem otestovat, zda honeypoty opravdu naslouchají na všech portech definovaných v konfiguraci a zda nabízí dané služby. Test se nazývá agresivním, protože kromě obyčejného skenování portů navíc využívá Nmap scripting engine, což je součást nástroje Nmap, která provádí detekci operačního systému cíle, testuje verze služeb a provádí základní penetrační testy, jako například pokusy o prolomení hesel. Využití NMap scripting engine tedy mělo ukázat reakci samotných honeypotů a ověřit, jaké údaje jsou honeypoty schopny zaznamenat a do jakého detailu.

Z testovacího Kali prostředí byl tedy opět spuštěn Nmap scan tentokrát s parametry `nmap -p0- -v -A -T4`. Je to tedy skenn všech TCP portů (-p0-), detekce OS, verzí služeb a scipty (-A) a nejrychlejší časování (-T4).

Očekávaným výsledkem byly otevřené porty nabízené honeypoty dle konfigurace, které popisuje tabulka 4.1

Honeypot	Očekávaný port
Dionaea	20
	21
	42
	81
	135
	445
	1433
	1723
	1883
	3306
	5060
	5061
	27017
Medpot	2575
Dicompot	11112
RDpy	3389
Log4pot	80
	443
	8080
	9200
	25565
Cowrie	22
	23

Tabulka 4.1: Očekávané otevřené porty dle honeypotů

Dále byly očekávané otevřené obslužné porty 64295, 64297, 64294 a také port 64299, který využívá služba Filebeat pro odesílání logů do monitoru.

Výstup

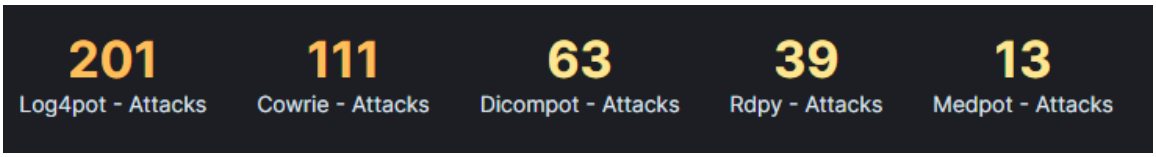
Test dopadl více méně dle očekávání. Je možno vidět téměř všechny očekávané porty a na nich běžící služby, nicméně se objevilo i několik anomálií. V prvé řadě je možno vidět, že testování některých služeb komplikuje tcpwrapper, který ukončuje spojení před testem služby. Příčinou může být pravděpodobně volba agresivního skenu. Test s parametrem -sV pravděpodobně dopadne úspěšněji v detekci služeb sensoru. Port 2575, na kterém je provozován honeypot medpot, Nmap identifikuje jako Apache Spark. Služby na portech 23 a 3389 Nmap označuje jako nerozpoznaný. Jedná se o honeypot RDpy operující na standardním RDP portu 3389 a telnet z honeypotu Cowrie opět na standartním portu 23.

Nmap dle očekávání odhalil administrativní porty 64294 a 64295, které správně identifikoval jako Cockpit web service a SSH respektive. Dalším potenciálním problémem, který je možné vidět na obrázku 4.3, může být identifikace operačního systému jako Debian linux, když služba SMB se tváří jako Windows 7 Professional, což je samozdřejmě podezřelé a může prozradit přítomnost honeypotu.

```
Device type: general purpose
Running: Linux 5.X
OS CPE: cpe:/o:linux:linux_kernel:5
OS details: Linux 5.0 - 5.3
Uptime guess: 11.996 days (since Thu Mar 31 10:09:41 2022)
Network Distance: 2 hops
TCP Sequence Prediction: Difficulty=250 (Good luck!)
IP ID Sequence Generation: All zeros
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel
...
Host script results:
| smb-os-discovery:
| OS: Windows 7 Professional 7600 (Windows 7 Professional 6.1)
| OS CPE: cpe:/o:microsoft:windows_7::-:professional
| NetBIOS computer name: ADFS\x00
| Workgroup: DACH\x00
|_ System time: 2022-04-12T09:03:43+01:00
| smb-security-mode:
| account_used: guest
| authentication_level: user
| challenge_response: supported
|_ message_signing: disabled (dangerous, but default)
|_clock-skew: mean: -31m15s, deviation: 40m33s, median: -59m56s
|_smb2-time: Protocol negotiation failed (SMB2)
```

Obrázek 4.3: Výtah NMap skenu sensoru ICT

Toto skenování ovšem samozdřejmě nezůstalo bez povšimnutí ze strany sensoru a honeypoty vygenerovaly poměrně velké množství logů popisujících útok, jak je možné vidět na obrázku 4.4.



Obrázek 4.4: Generované logy podle honeypotu po scanu sensoru UsrB

Jelikož s vyjímkou Cowrie se jedná o honeypoty s nízkou úrovní interakce, lze z nich získat relativně omezená data. Většina honeypotů více méně pouze detekuje navázání spojení a je schopna získat základní informace z příchozích packetů jako je zdrojová IP adresa a port, případně další drobné informace jako třeba dotaz HTTP u honeypotu Log4pot viditelný na obrázku 4.5.

V tomto ohledu je vyjímkou honeypot Cowrie jako jediný hybridní honeypot z nasazených. Cowrie podává podrobné logy o komunikaci SSH a Telnet, včetně verzí a použitých šifrovacích algoritmů. Nástroj P0f je znovu schopen identifikovat sken Nmap a určit operační systém útočníka.

Celkově tedy ze skenu nástrojem Nmap je honeypot schopen podat informace o IP adrese útočníka, využitých a napadených portech, operačním systému útočníka, identifikaci nmap útoku a verzi použitých nástrojů, případně konkrétní dotazy na honeypoty.

Field	Value
t _id	7zYoHYABgUtu59VeCY2d
t _index	logstash-2022.04.12
# _score	-
t _type	_doc
@timestamp	Apr 12, 2022 @ 11:43:24.075
t @version	1
t host_info.exit_ip	195.113.227.242
t host_info.hostname	ict-chroustalj
t host_info.in_ip	172.19.191.47
t host_info.sensor	SensorICT
t message	{"reason": "request", "timestamp": "2022-04-12T09:43:21.837409", "correlation_id": "aeeeb845-5460-4b4b-802c-80e55d765676", "server_port": 8080, "client": "172.19.191.46", "port": 35795, "request": "GET /service/index_pri.php HTTP/1.0", "headers": {}}
t T-Pot Hostname	vdi-chroustalj
t t-pot_ip_ext	195.113.227.242
t t-pot_ip_int	172.19.191.48
t Type	Log4pot

Obrázek 4.5: Ukázka logu z Log4Pot

4.3 Automatizované penetrační testy

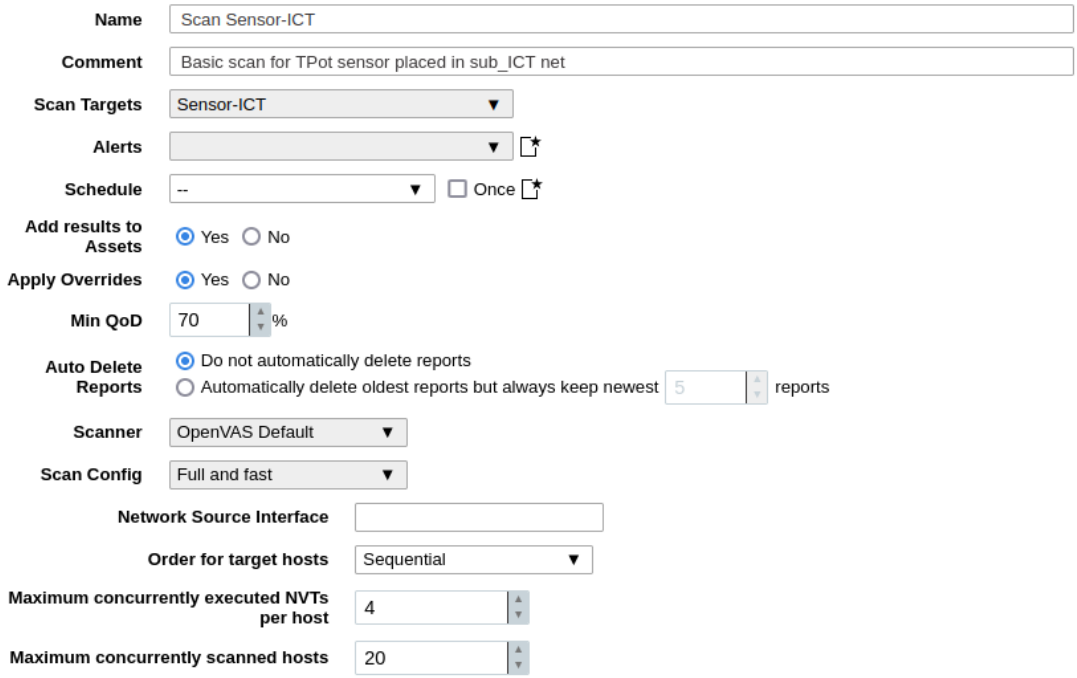
Po skenování nástrojem Nmap byly sensory dále podrobeny automatizovanému penetračnímu testování a skenu zranitelností.

4.3.1 Test č.3: Penetrační test pomocí nástroje OpenVAS

Dalším z testů byl automatizovaný sken zranitelností pomocí nástroje OpenVAS. Hlavním cílem tohoto experimentu bylo zjistit, do jaké míry se honeypot prezentuje jako zranitelný cíl, a také ověřit, zda neobsahuje nazamýšlené zranitelnosti, díky kterým by mohlo dojít ke kompromitaci sensoru samotného.

Očekávaným výsledkem je systém prezentující několik vážných zranitelností. Vzhledem k přítomnosti honeypotu Log4pot se očekávala minimálně zranitelnost Log4J. Nebylo by však překvapením, kdyby nástroj byl schopen odhalit že se jedná o honeypoty.

Scan byl proveden z připraveného Kali prostředí nejnovější verzí nástroje OpenVAS. Test byl spuštěn v nastavení (jak ukazuje obrázek 4.6 Full and Fast režimu s minimální kvalitou detekce 70%. Jeho provedení trvalo přibližně 20 minut pro každý ze sensorů.



Obrázek 4.6: Nastavení OpenVAS testu

Výstupy

Dle očekávání odhalil tento test velké množství zranitelností. Nižší počet zranitelností u sensoru umístěného v uživatelské síti budovy B byl rovněž očekávaným výsledkem, neboť tento sensor obsahuje menší sadu honeypotů, aby lépe zapadl mezi ostatní zařízení provozované v dané podsíti. Obrázek 4.7 nabízí podrobnější pohled na konkrétní zranitelné porty dle skenu OpenVAS a jejich závažnost. Je ovšem nutno podotknout, že dle zachycené komunikace nebyly testovány všechny porty, jako například obslužné porty sensoru.

SensorICT

Service (Port)	Threat Level
1723/tcp	High
3306/tcp	High
80/tcp	High
443/tcp	High
general/tcp	High
9200/tcp	High
8080/tcp	High
445/tcp	High
22/tcp	Medium
23/tcp	Medium
3306/tcp	Medium
22/tcp	Low
3306/tcp	Low
general/tcp	Low

SensorLAN

Service (Port)	Threat Level
80/tcp	High
9200/tcp	High
443/tcp	High
general/tcp	High
1723/tcp	High
8080/tcp	High
445/tcp	High
3306/tcp	High
23/tcp	Medium
22/tcp	Medium
3306/tcp	Medium
general/tcp	Low
22/tcp	Low
3306/tcp	Low

SensorUsrB

Service (Port)	Threat Level
general/tcp	High
3306/tcp	High
22/tcp	Medium
3306/tcp	Medium
23/tcp	Medium
general/tcp	Low
22/tcp	Low
3306/tcp	Low

Obrázek 4.7: Zranitelnosti dle OpenVAS

Nalezené zranitelnosti jsou různorodé, ovšem většina z nich je poměrně stará. V některých případech sahají více než deset let zpět, jako například u SMB služby honeypotu Dionaea, kde byly OpenVAS objeveny zranitelnosti sahající až do roku 2010. Operační systém byl dle OpenVAS identifikován jako Windows 7 Professional. Hned několik zranitelností bylo objeveno v nabízeném MySql a MSSql, dále byly hlášeny zranitelnosti v SSH serveru nabízeného honeypotem Cowrie, který akceptuje DSS klíče a navíc podporuje algoritmy založené na MD5.

Kromě výše zmíněných je rovněž přítomno několik hlášení zranitelnosti Log4J. Toto byl očekávaný výsledek vzhledem k nasazení honeypotu Log4Pot.

Z výsledků je tedy možné pozorovat, že sensor se prezentuje jako velice zranitelný systém. Možná až příliš zranitelný, vzhledem k jeho umístění ovšem je nutno podotknout, že všecha interakce s honeypoty byla zalogována a sensor by tedy splnil svůj účel detekce potenciálního útočníka. OpenVAS test rovněž neodhalil, že se jedná o honeypot, nicméně i tak by se v případě reálného útočníka mohl sensor zdát jako příliš podezřelý cíl.

4.3.2 Test č.4: Penetrační test pomocí nástroje Sn1per

Následujícím testem byla opět automatizovaná sada útoků za účelem odhalení zranitelností, tentokrát využitím nástroje Sn1per[6]. Jedná se o otevřenou softwarovou platformu pro automatizované penetrační testování. Nástroj nabízí i komerční professional verzi, ovšem pro účely této práce byla využita verze zdarma. Test byl proveden celkem dvakrát, jednou ve "stealth" režimu, kdy byly využity pouze pasivní metody, a posléze ve standardním plném režimu. Cílem tohoto testu bylo zejména porovnat odhalené zranitelnosti vůči testu předchozímu a zjistit, co dokáží o honeypotech odhalit.

Test byl opět spuštěn z připraveného prostředí Kali a cílen na sensor umístěný v síti ICT. Očekávaným výsledkem bylo odhalení zranitelností, které ukázal test pomocí OpenVAS případně několik nových. Pro standardní sadu testů byl rovněž očekáván negativní výsledek pro obslužné porty sensoru.

Výstupy

V pasivním režimu nebyl Sn1per schopen odhalit příliš mnoho informací. Nalezl pouze tři zranitelnosti v HTTP serveru na portu 80. I s využitím pouze pasivních metod skenování byl sensor opět schopen rychle odhalit interakci a pomocí nástroje P0f identifikovat informaci, že se jedná o Nmap sken. Samotné honeypoty sice na skenování nereagovali, nástroj ale P0f sám o sobě poskytuje dost informací pro identifikaci potenciálně nepřátelské recon aktivity.

Standardní režim už odhalil podstatně více. Ačkoliv celkový počet 59 (úrovně low a vyšší) zranitelností je o něco nižší než kolik odhalil OpenVAS, Sn1per našel několik odlišných.

```
================================================================================
Sc0pe Vulnerability Report by @xer0dayz
================================================================================
Critical: 4
High: 1
Medium: 48
Low: 6
Info: 43
Score: 223
================================================================================
```

Obrázek 4.8: Souhrn nalezených zranitelností standardním scanem nástroje Sn1per

Nejzajímavějším výstupem z tohoto testu je fakt, že Sn1per byl schopen detekovat honeypot Cowrie, což samozřejmě okamžitě odhaluje skutečnost, že se jedná o nástražný systém a pravděpodobně odradí potenciálního útočníka od jakýchkoliv dalších pokusů o interakci se sensorem.

```
================================================================================•x[2022-04-21](09:44)x•
 RUNNING NUCLEI SCAN
================================================================================•x[2022-04-21](09:44)x•
[2022-04-21 09:44:16] [smb-v1-detection] [network] [low] 172.19.191.47:445
[2022-04-21 09:44:21] [cowrie-honeypot-detect] [network] [info] 172.19.191.47:22
[2022-04-21 09:44:45] [mongodb-detect] [network] [info] 172.19.191.47:27017
================================================================================•x[2022-04-21](09:46)x•
```

Obrázek 4.9: Sn1per detekuje honeypot Cowrie

4.4 Shrnutí

Po provedení experimentů je jasné, že se sensory prezentují jako velice zranitelné systémy. Problémem zůstáva fakt, že počet těchto zranitelností je příliš velký a sensor tak působí

[6] https://github.com/1N3/Sn1per [13.4.2022]

jako až příliš snadný cíl, což může pro případné útočníky být příliš podezřelé. Jelikož nasazený systém honeypotů zůstane součástí zabezpečení počítačové sítě nemocnice Jihlava i po dokončení této práce, bude nutné tento problém vyřešit. Nejjednodušším způsobem, jak toho docílit, je rozdělit honeypoty na více sensorů a provozovat několik sensorů v jedné síti. Dalším závažným problémem je schopnost automaticky odhalit honeypot Cowrie. Na základě této skutečnosti bude Cowrie s nejvyšší pravděpodobností vyřazen z využívaných honeypotů, případně umístěn na svůj vlastní oddělený sensor, aby neprozradil ostatní honeypoty.

Je nutno podotknout, že ačkoliv testy honeypotů odhalují řadu zranitelností, nemusí být tato detekce vždy zcela přesná. Ze zachycené komunikace je totiž zřejmé, že honeypoty s nízkou úrovná interakce mají omezenou sadu odpovědí a mohou tedy na nečekané dotazy vracet nesmyslnou odpověď. Honeypot Log4pot například vrací stejná data nehledě na dotazovanou URI či využitou HTTP metodu.

Pozitivním výsledkem je, že sensory jsou schopny detekovat všechny pokusy o interakci a detekovat minimálně zdrojovou adresu a port útočníka. Ačkoliv to není zdaleka dost informací pro podrobnou analýzu, je to dostatek pro upozornění na podezřelou aktivitu v síti a navázání dalšího šetření. Vzhledem k umístění se v těchto sítích počítá zejména s útoky přicházejícími z kompromitovaných zařízení v síti, nikoliv útoky z internetu. Znalost zdrojové adresy je tedy důležitou informací pro identifikaci kompromitovaného zařízení. Množství získaných informací ze sensoru bylo v rámci práce rozšířeno, jak je popsáno v následující kapitole. Celkově však systém plní svoji funkci v rámci návrhu.

Kapitola 5

Automatizace a ohlašování incidentů

Systém honeypotů byl v rámci práce úspěšně navržen, nasazen a testován. Honeypoty generují celou řadu logů pro každou interakci a odesílají je do monitoru, který je uchovává a umožňuje procházet, filtrovat a vizualizovat. Pro účely praktického využití v nemocnici ovšem nelze očekávat, že správce systému nebo správce bezpečnosti počítačové sítě nemocnice budou pravidelně ručně kontrolovat monitor a analyzovat všechna nová data. Bylo tedy zapotřebí implementovat systém automatického ohlašování zájmových dat. Nástroj Tpot v současné době tuto funkcionalitu nenabízí a bylo tedy nutné konfiguraci pro tyto účely rozšířit. Způsobů existuje celá řada a tato kapitola nastiňuje několik z nich, které byly zkoumány v rámci práce a následně detailněji popisuje metodu, která byla zvolena pro implementaci v nemocnici.

5.1 Rozšíření získávaných dat

Prvním problémem, se kterým se bylo nutno potýkat, je poměrně malé množství použitelných informací z logů honeypotů. Honeypoty totiž samy o sobě nedisponují funkcí pro automatickou analýzu a nejsou schopny určit, o jaký typ útoku či komunikace se jedná. Právě za tímto účelem byly sensory rozšířeny o nástroj Suricata[1]. Ten sleduje dění na honeypotech a obohacuje údaje získané během interakcí s honeypoty. Mimo jiné rozděluje detekované útoky do katergorií a přiřazuje jim signaturu dle typu, což jsou podstatné informace pro návrh automatického ohlašování detekovaných incidentů. Suricata rovněž přiřazuje úroveň závažnosti v rozmezí 1 - 3, kdy nižší číslo znamená vyšší závažnost detekovaného útoku.

Na obrázku 5.1 je možné vidět ukázku výstupu z nástroje Suricata při testování pomocí nástroje Sn1per. K informacím získaným z honeypotů tedy Suricata doplní kategorii a signaturu útoku, což je významná informace protože jsme podle ní schopni rychleji určit závažnost incidentu, bez nutnosti provádět vlastní předběžnou analýzu. Nástroj byl do sensorů přidán podobně jako ostatní honeypoty pomocí modifikace konfiguračního souboru pro sensor, jak je vysvětleno v kapitole 3.2.2. Využit byl obraz pro nástroj Docker poskytnutý vývojáři Tpotu, který se v základní konfiguraci plně integruje mezi ostatní. Informace získané z nástroje Suricata byly dále využity pro návrh pravidel stanovení závažností.

[1] https://suricata.io/ (15.4.2022)

t alert.action	allowed
t alert.category	Attempted Administrator Privilege Gain
# alert.gid	1
t alert.metadata.attack_target	Server
t alert.metadata.created_at	2021_07_28
t alert.metadata.deployment	Internal, Perimeter
t alert.metadata.former_category	EXPLOIT
t alert.metadata.signature_severity	Major
t alert.metadata.tag	Exploit
t alert.metadata.updated_at	2021_07_28
# alert.rev	1
# alert.severity	1
t alert.signature	ET EXPLOIT GraphQL Introspection Query Attempt

Obrázek 5.1: Ukázka logu z nástroje Suricata při testování sensoru pomocí Sn1per

5.2 Možnosti ohlašování incidentů

Jak již bylo zmíněno, ačkoliv nástroj Tpot v základu neposkytuje žádnou funkci automatického ohlašování incidentů, existuje řada možností, jak tuto funkci doplnit. Zejména díky modulárnosti nástrojů ELK stack (Kibana a Logstash), lze snadno docílit další funkcionality pomocí pluginů pro tyto nástroje. V rámci práce bylo zkoumáno několik možných řešení a po konzultaci s vedením oddělení sítí a hardware a správcem bezpečnosti byl implementován systém automatizovaného ohlašování incidentů do lokální instance GitLab, jak je popsáno níže.

5.2.1 Systém Kibana Alerting

První ze zkoumaných možností je pravděpodobně implementačně nejméně náročná, ovšem pro účely práce nevyužitelná. Jedná se o systém Kibana Alerting[2] což je doplněk nástroje Kibana, který nabízí snadnou integraci a ohlašování detekce zájmových dat přímo z grafického prostředí nástroje Kibana. Ohlašování probíhá na základě uživatelsky vytvořené sady pravidel, přičemž každé pravidlo sestává z podmínek, načasování a akcí. Celý systém je poměrně jednoduchý a intuitivní. Podmínky definují, kdy mají hodnoty ve zkoumaném záznamu (v tomto případě logu honeypotu) vyvolat akci danou tímto příslušným pravidlem, jsou definovány stejným dotazem, jaký je použit pro filtrování v nástroji Kibana.

Načasování (schedule) udává, jak často má být tato podmínka kontrolována. Pokud je podmínka pravidla vyhodnocena jako True, pak je vyvolána akce definovaná v pravidle. Kibana v základu podporuje několik akcí a využití takzvaných connectorů, což jsou rozhraní, které umožňují využití služeb třetích stran. Lze tedy například odeslat email s upozorněním, odeslat zprávu do MSTeams, Telegram či dalších chatovacích programů, aktivovat webová

[2]https://www.elastic.co/guide/en/kibana/current/alerting-getting-started.html [5.4.2022]

API a mnoho dalších. Toto lze opět velice jednoduše nastavit přímo v grafickém rozhraní Kibana.

Kibana Alerting však vyžaduje měsíčně placenou licenci pro využívání nástroje, což jej činí nepoužitelným pro účely této práce, neboť je na ni vyhrazen nulový rozpočet s vyjímkou hardwarových nákladů sensorů samotných. Tato možnost tedy zůstává pouze jako potenciální vylepšení pro budoucí rozšiřování, nicméně pro účely bakalářské práce je nutné se obrátit na jinou metodu.

5.2.2 Doplňky pro výstup z Logstash

Asi nejvhodnější možnost pro tvorbu automatizovaného reportování nabízí nástroj Logstash, který je zodpovědný za příjmání, filtrování, manipulaci a ukládání logů do Elasticsearch. Logstash definuje značnou část vytvořeného systému shromažďování logů z honeypotů a právě toho lze využít pro implementaci ohlašování incidentů. Logstash je totiž schopen předávat kopie přijatých a filtrovaných logů dalším výstupním pluginům na základě konfigurovaných podmínek. Je tedy možné sestavit sadu pravidel, na základě kterých budou konkrétní zájmová data z logů předána k dalšímu zpracování. Těchto výstupních pluginů je celá řada, ovšem pro účely této práce byly blíže zvažovány následující možnosti:

1. Výstup do elektronicé pošty

Logstash umožňuje odesílat zprávy SMTP na základě jednoduché konfigurace, kterou lze vidět na obrázku 5.2. Odesilatele, příjemce, předmět i tělo této zprávy se dá upravit přímo v Logstash konfiguraci a lze tedy odeslat pouze zajímavá data. Mimo to lze nastavit z jakého portu se má zpráva odeslat, v jaké doméně atd. Pro účely této práce však tato metoda nakonec zvolena nebyla zejména kvůli požadavku na další shromažďování a zpracování dat. Jak lze vidět z testů, honeypoty produkují obrovské množství logů, a jejich seskupování a následná tvorba varovného e-mailu se jeví jako příliš složitá pro konfiguraci přímo v nástroji Logstash.

```
...
email {
      to => 'piskj@nemji.cz'
      from => 'Tpot-monitor@nemji.cz'
      subject => 'Alert - %{[hostinfo][sensor]}'
      body => "%{[message]}"
      domain => 'nemji.cz'
      port => 25
    }
...
```

Obrázek 5.2: Ukázka konfigurace email výstupu pro Logstash

2. Výstup do souboru

Dalším možným řešením bylo vyčlenit konkrétní zájmové logy a umístit je pomocí výstupního pluginu file do odděleného úložiště na monitoru. Toto úložiště by pak pravidelně kontroloval vlastní skript, který by tato data dále analyzoval a vytvářel upozornění dle

potřeby. Tento způsob je však zbytečně prostorově náročný vzhledem k nutnosti uchovávat další kopii zájmových logů a navíc by zpracovávání těchto dat skriptem znamenalo další nároky na paměť monitoru, která už je zatížena provozem samotného ELK stacku.

Výstup přes HTTP

Asi nejvhodnějším způsobem se jeví plugin HTTP, který umožňuje odesílat vybraná data pomocí HTTP protokolu na konkrétní adresu a port. Plugin rovněž umožňuje specifikovat cestu k certifikátu a komunikovat pomocí HTTPS. Plugin umožňuje využívat všech HTTP metod a upravovat hlavičky dotazu. Tělem dotazu pak může být uživatelsky definovaná zpráva nebo celý soubor tak, jak jej Logstash zpracovává. Tato metoda byla nakonec zvolena pro implementaci ohlašování incidentů do nástroje GitLab.

5.3 Stanovení závažnosti

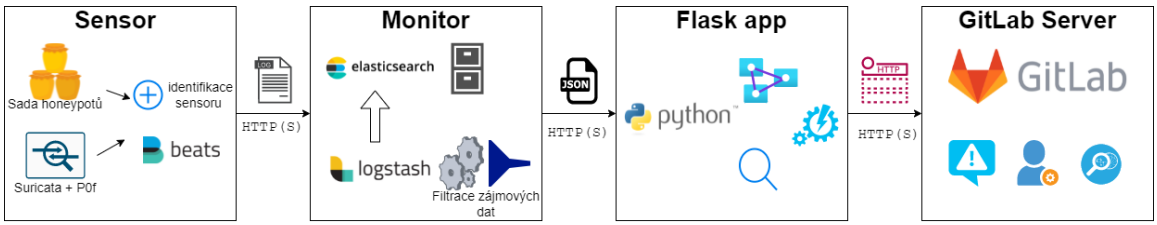
Dalším důležitým krokem pro navržení automatického ohlašování incidentů bylo stanovení úrovní závažnosti. Vzhledem k využití nástrojů pro správu incidentů v lokální GitLab instanci jakožto primárního zdroje pro reportování incidentů z honeypotového systému bylo využito pět úrovní závažností, tak jak je nabízí GitLab. Jedná se tedy o následující:

- **S5 Unknown** - Závažnost nelze určit. Jedná se o startovní bod každého nově vzniklého incidentu, ovšem je rychle nahrazen jednou z ostatních úrovní, jakmile jsou získána další data související s incidentem. Jedná se většinou o první detekci interakcí například nástrojem P0f ovšem s nedostatkem dat k určení, zda se jedná o cílenou či závadnou aktivitu.
- **S4 Low** - Nízká závažnost. Na této úrovni se nachází incidenty, při kterých je detekována pouze minimální interakce s honeypotem. Řadí se sem například pasivní skenování pouze několika málo portů, zachycený broadcast a další interakce podobného charakteru. Tato úroveň rovněž odpovídá závažnosti třetí úrovně reportované nástrojem Suricata.
- **S3 Medium** - Do střední úrovně závažnosti se řadí jakýkoliv kontakt s konkrétními honeypoy. Pokud je incident hlášený z honeypotu pak byl vyvolán pokusem o využití nabízené falešné služby a jde tedy o cílenou interakci. Tato úroveň odpovídá závažnosti druhé úrovně reportované nástrojem Suricata.
- **S2 High** - Mezi závažné incidenty se řadí interakce s medicínskými honypoty Medpot a Dicompot a detekované pokusy o využití konkrétních zranitelností, pokusy o získání zvýšených oprávnění a rovněž incidenty nižších tříd, které přesáhly dané množství záznamů. Tato úroveň odpovídá závažnosti úrovně jedna reportované nástrojem Suricata. Incidenty s touto úrovní závažnosti získávají vysokou prioritu k vyřešení v rámci činnosti oddělení správy sítí.
- **S1 Critical** - Kritickými incidenty jsou takové, které prozrazují bližší znalost nemocnice útočníkem, například díky klíčovým slovům v pokusech prolomit hesla, využití správného formátu uživatelských jmen a tak dále. Dále do této kategorie spadají útoky pocházející z pracovních stanic administrátorů a rovněž útoky cílící na obslužné porty sensoru, kdy může hrozit kompromitace systému. Na incident s kritickou závažností je okamžitě upozorněn správce systému a správce bezpečnosti počítačové sítě nemocnice.

Tento návrh byl konzultován a schválen správce bezpečnosti počítačové sítě nemocnice, nicméně se počítá s průběžnými změnami v závislosti na detekovaných incidentech.

5.4 Implementované řešení

Po konzultaci se správcem bezpečnosti počítačové sítě nemocnice a vedoucím oddělení správy sítí a hardware byl zvolen způsob ohlašování incidentů vyvolaných honeypotem do nástroje pro správu incidentů na lokálním GitLab serveru, kterou oddělení správy sítí využívá pro dokumentaci, automatizaci, přidělování úkolů a správu incidentů. Vznikl tedy systém pro detekci incidentů počínající vygenerováním logu na sensoru a končící vytvořením či aktualizací záznamu incidentu v nástroji GitLab, kterou popisuje obrázek 5.3.



Obrázek 5.3: Implementovaný systém ohlašování událostí

Sensory v jednotlivých sítích čekají na interakci od útočníka. Jakmile honeypot zaznamená interakci, vygeneruje příslušné logy, které převezme nástroj Filebeats na sensoru, připojí informace o sensoru ze kterého je log pořízen, a odešle je do centrálního monitoru, kde je dále zpracuje nástroj Logstash. Při zpracování nástrojem Logstash jsou tyto logy upraveny do zadaného formátu a projdou filtrací. Posléze jsou předány jako Elasticsearch dokument pro uchování a analýzu. Kromě Elasticsearch jsou údaje (jako například logy z konkrétních honeypotů) dále předány do výstupního pluginu HTTP. Jsou z nich extrahována zájmová data a ta jsou odeslána ve formátu json na příslušný endpoint vlastního HTTP serveru.

```
...
if [type] == "Cowrie" {
    http {
      format=>"message"
      http_method=>"post"
      url=>"http://REDACTED/cowrie"
      # Cowrie fields: timestamp, src_ip, sensor, eventid,session,message
      message=>'{"timestamp":"%{[@timestamp]}","src_ip":"%{[src_ip]}", ...
    }
  }
  ...
```

Obrázek 5.4: Ukázka konfigurace http výstupu pro honeypot Cowrie

Tento server je naprogramovaný v jazyce Python s využitím frameworku Flask[3] a běží na development serveru oddělení správy sítí v mé správě. Program shromažďuje záznamy

[3]https://github.com/pallets/flask/ [11.4.2022]

a třídí je do incidentů na základě zdrojové adresy útoku a sensoru, ze kterého pochází. Dále rozhoduje o závažnosti těchto incidentů a spouští další automatizované akce dle definovaných pravidel. Program vytváří a aktualizuje záznam o incidentu za pomoci knihovny python-gitlab[4], která vytváří a odesílá požadavky na REST API GitLab serveru.

Filtrace zájmových dat je zajištěna dvěma způsoby. Nejprve jsou data filtrována přímo na monitoru pomocí nástroje Logstash, kdy jsou na glincident (vlastí Flask server zajišťující práci s Gitlab incidenty) odeslána pouze vybrané části logů ze sensorů ve formátu JSON. Jakmile glincident tyto data přijme, vyhledá nebo vytvoří patřičný incident pro nástroj Gitlab a dále filtruje pouze nové informace k incidentu. Glincident rovněž seskupuje přijaté záznamy do logických celků, jak jsou například jednotlivé relace SSH z nástroje Cowrie. Glincident rovněž rozhoduje o závažnosti incidentu dle pravidel popsaných v tabulce 5.1.

Pravidlo	**Závažnost**	**Další akce**
Nesmyslné či nezpracovatelné logy nebo prázdný incident	S5-UNKNOWN	Alert správce systému. Jedná se o chybu.
Interakce ohlašované nástrojem P0f a Suricata (severity 3)	S4-Low	Žádné
Interakce ohlašované konkrétními honeypoty	S3-Medium	Žádné
Interakce ohlášené nástrojem Suricata (severity 2)	S3-Medium	Žádné
Interakce ohlášené medicínskými honeypoty Medpod/Dicompot	S2-High	Incidentu je přiřazena vysoká priorita pro řešení.
Interakce ohlášené nástrojem Suricata (severity 1)	S2-High	Incidentu je přiřazena vysoká priorita pro řešení.
Počet událostí v rámci jednoho incidentu přesáhl stanovenou hranici	S2-High	Incidentu je přiřazena vysoká priorita pro řešení.
Interakce s obslužnými porty honeypotu	S1-Critical	Ohlášeno správci systému a správci bezpečnosti.
Při pokusu o přihlášení byl využit login odpovídající formátu firemních účtů	S1-Critical	Ohlášeno správci systému a správci bezpečnosti.
Interakce z administrátorské stanice	S1-Critical	Ohlášeno správci systému a správci bezpečnosti.

Tabulka 5.1: Pravidla pro stanovení závažnosti incidentu

Ohlašování incidentů je již zajištěno GitLabem samotným, který na základě tagů incidentu případně konkrétní zmínky uživatele (pro kritické incidenty) upozorní patřičné správce e-mailem. Incidenty v nástroji GitLab jsou pravidelně monitorovány a řešeny pracovníky oddělení správy sítě a hardware a rovněž správcem bezpečností počítačové sítě nemocnice.

Cílem tohoto systému není nahradit nástroj Kibana, který je umístěný na monitoru, ale ohlásit výskyt incidentu a poskytnout přehled základních informací správcům. Detailní analýza incidentu, je li potřeba pak může být provedena na monitoru samotném. Díky tomuto návrhu však systém zabraňuje zahlcení administrátorů bezpočtem logů a poskytuje výchozí bod pro podrobnější analýzu problému. Tímto způsobem je tedy systém honeypotů plně integrován do bežné činnosti oddělení správy sítí a zvyšuje úroveň zabezpečení nemocnice.

[4]https://github.com/python-gitlab/python-gitlab [11.4.2022]

Kapitola 6

Závěr

Cílem této práce byl průzkum existujících honeypotů a jejich praktické využití pro detekci bezpečnostních incidentů v počítačové síti nemocnice Jihlava. Práce se dále věnovala vysvětlení principu fungování honeypotů jakožto nástražných systémů sloužících k detekci, výzkumu a automatickému upozornění na útočníka a jejich dělení dle úrovně interakce na honeypoty s nízkou, střední či vysokou úrovní. Rozdíly mezi těmito úrovněmi byly dále popsány a přiblíženy jejich výhody a nevýhody. Dále byla zkoumána dostupná řešení ze světa otevřeného software a popsány známější honeypoty dle kategorií a služeb které nabízí. Podrobněji byly zkoumány honeypotové platformy Nova, HoneyTrap, Modern Honey Network, Community Honey Network a platforma Tpot, která byla v rámci práce nasazena. Rovněž byla prozkoumána dostupná komerční řešení FortiDeceptor od společnosti FortiNet a BotSink společnosti Attivo Networks. Na základě tohoto průzkumu byl navržen a nasazen systém honeypot sensorů pokrývající tři sítě v nemocnici a do budoucna snadno rozšiřitelný do dalších sítí. Tyto sensory byly následně podrobeny sadě několika testů pro ověření správné funkčnosti a zjištění nedostatků. Výsledky testů byly diskutovány a byly navrženy změny pro budoucí provoz systému. V poslední kapitole bylo prozkoumáno několik možností automatizace varování při detekci interakce se sensory. V rámci automatizace byl navržen a implementován systém ohlašování podezřelých aktivit detekovaných sensorem do nástroje GitLab, který je interně využívaný oddělením správy sítě a hardware a správcem bezpečnosti počítačové sítě nemocnice. Součástí této integrace byla navržena sada pravidel pro stanovení závažnosti detekovaných incidentů, honeypoty byly rozšířeny o IDS nástroj Suricata za účelem získání většího množství informací z interakcí útočníka se sensorem.

Systém honeypotů navržený a implementovaný v této práci nadále zůstane trvalou součástí zabezpečení nemocnice Jihlava pod sdílenou správou mé osoby a správcem bezpečnosti počítačové sítě nemocnice. Ačkoliv bude systém vyžadovat další úpravy diskutované v rámci vyhodnocení výsledků testování honeypotů, považuji cíle této práce za splněné a pevně věřím že jsou pro nemocnici přínosem.

Literatura

[1] *About DICOM* [online]. The Medical Imaging Technology Association, a division of NEMA [cit. 2022-01-03]. Dostupné z: https://www.dicomstandard.org/about.

[2] *What is a Zero-day Attack? - Definition and Explanation* [online]. Kaspersky Lab [cit. 2011-05-02]. Dostupné z: https://www.kaspersky.com/resource-center/definitions/zero-day-exploit.

[3] *What is Elasticsearch?* [online]. Elasticsearch B.V [cit. 2011-05-02]. Dostupné z: https://www.elastic.co/what-is/elasticsearch.

[4] *GREYCORTEX Mendel Data Sheet* [online]. Greycortex s.r.o., únor 2022 [cit. 2022-04-06]. Dostupné z: https://www.greycortex.com/sites/default/files/perm/document/greycortex-mendel-data-sheet-cz-2022_2.pdf.

[5] *UPOZORNĚNÍ NA ZVÝŠENÉ RIZIKO KYBERŠPIONÁŽNÍCH ČI RANSOMWAROVÝCH ÚTOKŮ PROTI ČESKÉ REPUBLICE* [online]. Národní úřad pro kybernetickou a informační bezpečnost, 28. ledna 2022 [cit. 2011-05-02]. Dostupné z: https://www.nukib.cz/download/publikace/analyzy/Upozorneni_na_zvysene_riziko_proti_CR.pdf.

[6] Fu, X., Yu, W., Cheng, D., Tan, X., Streff, K. et al. On Recognizing Virtual Honeypots and Countermeasures. In:. Listopad 2006, s. 211 – 218. DOI: 10.1109/DASC.2006.36.

[7] Grudziecki, T., Jacewicz, P., Juszczyk, l., Kijewski, P. a Pawlinski, P. Proactive Detection of Security Incidents. European Union Agency For Cybersecurity. Listopad 2012.

[8] Müter, M., Freiling, F., Holz, T. a Matthews, J. A generic toolkit for converting web applications into high-interaction honeypots. Leden 2008.

[9] Nawrocki, M., Wählisch, M., Schmidt, T., Keil, C. a Schönfelder, J. A Survey on Honeypot Software and Data Analysis. Srpen 2016.

[10] Tsikerdekis, M., Zeadally, S., Schlesener, A. a Sklavos, N. Approaches for Preventing Honeypot Detection and Compromise. In:. říjen 2018. DOI: 10.1109/GIIS.2018.8635603.

Příloha A

Obsah přiloženého DVD

```
glincident/
    glincident.py           - Zdrojový kód pro flask aplikaci zajišťující
                              vytváření incidentů na GitLab
    requirements.txt        - Seznam závislostí pro instalaci pomocí python pip
    incident_template.txt   - Template pro incident v gitlabu

config/
    sensor/
        fields.yml          - Konfigurace pro Filebeat s přejmenováním pole input.type
        filebeat.yml        - Konfigurační soubor pro nástroj Filebeat
        sensorICT.yml       - Docker-compose soubor definující sadu honeypotů
                              pro SensorICT a SensorVLAN1
        sensorUsrB.yml      - Docker-compose soubor definující sadu honeypotů
                              pro SensorUsrB
    monitor/
        logstash.conf       - Konfigurace nástroje logstash na monitoru
        monitor.yml         - Docker-compose soubor definující sadu nástrojů
                              pro monitor
tech_zprava/
    latex.zip               - Zdrojové soubory pro tvorbu technické zprávy v LaTeXu
    technicka_zprava.pdf    - Elektronická verze technické zprávy

README.md                   - Návod na instalaci honeypotů a práci s glincident.py
```